Extraction and Coordination
in
Phrase Structure Grammar and Categorial Grammar

Glyn Verden Morrill

PhD
University of Edinburgh
1988
Declaration

I declare that this thesis has been composed by myself and that the research reported therein has been conducted by myself unless otherwise indicated.

Glyn Morrill
Edinburgh, July 25, 1988
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 assistance 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. Pure Phrase Structure Grammar 2
2. Pure Categorial Grammar 5
3. Grammar for Canonical English 9

Chapter II: Phrase Structure Grammar Extended with Metarules 19

1. Simple Non-Canonicality 19
  1.1. Extraction 20
    1.1.1. Left Extraction 20
    1.1.2. Right Extraction 31
    1.1.3. Parasitic Extraction 38
  1.2. Coordination of ‘Non-Constituents’ 42
    1.2.1. Right Node Raising 42
    1.2.2. Left Node Raising 46
    1.2.3. Across-the-Board Extraction 49
  1.3. Summary 51
2. Compound Non-Canonicality 53
  2.1. Multiple Extraction 54
    2.1.1. Multiple Independent Extraction 54
    2.1.2. Independent Extraction Plus Parasitic Extraction 60
  2.2. Multiple Extraction from Coordinate Structure 61
    2.2.1. Multiple Across-the-Board Extraction 61
    2.2.2. Left Node Raising Plus Across-the-Board Extraction 66
  2.3. Extraction of Incomplete Elements 71
3. Discussion: Categories and Compound Non-Canonicality 73
Chapter III: Categorial Grammar Extended with Rules

1. Simple Non-Canonicality
   1.1. Coordination of ‘Non-Constituents’
       1.1.1. Right Node Raising
       1.1.2. Left Node Raising
   1.2. Extraction
       1.2.1. Right Extraction
       1.2.2. Left Extraction
       1.2.3. Pied Piping, That-Relatives, and That-Less Relatives
       1.2.4. Parasitic Extraction
2. Discussion: Rules and Compound Non-Canonicality

Chapter IV: Categorial Grammar Extended with Metarules

1. Simple Non-Canonicality
   1.1. Extraction
   1.2. Coordination of ‘Non-Constituents’
2. Compound Non-Canonicality

Chapter V: Universal Grammar

1. Syntax
   1.1. Metarules and Directional Consistency
   1.2. Category Structure
   1.3. Free Word Order
   1.4. Weak Generative Capacity
2. Semantics

Chapter VI: Processing

1. Parsing Pure Categorial Grammar with Unification Semantics
   1.1. Pure Categorial Grammar and Charts
   1.2. Implementation with Unification Semantics
2. Parsing Generalised Categorial Grammars with Combinatory Logic Semantics
   2.1. Generalised Categorial Grammars and Equivalences
2.2. Implementation with Combinatory Logic Semantics 151

Chapter VII: Conclusion 155

1. Summary 155
2. Future Directions 155
   2.1. Syntax and Processing 156
   2.2. Semantics 158

Appendix A: Complexity of Categorial Grammar with Unification 163

1. Computational Complexity 163
   1.1. Some Features of Computational Complexity Theory 163
   1.2. Reduction of 3SAT to Categorial Grammar with Unification Recognition 167
2. Discussion: Semantics of Categorial Grammar with Unification 172

Appendix B: Parser Listing and Illustrative Log 174

1. Parser Listing 174
2. Illustrative Log 184

References 192
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 than 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, suggesting that canonical grammar should be approached in the first place from the point of view of categorial grammar.

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:

\[
(3) \quad \begin{align*}
  a. & \quad S \to NP \ VP \\
  b. & \quad VP \to TV \ NP
\end{align*}
\]

The interpretation of these rules is that expressions of the categories on the right hand side can be concatenated to form expressions of the categories 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).

\[
(4) \quad \begin{align*}
  a. & \quad TV \to met \\
  b. & \quad REL \to who \ VP
\end{align*}
\]

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
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

Figure 1

Figure 2
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 expression.\footnote{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 s-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.} For example:\footnote{In λ-terms application is indicated by juxtaposition and is left-associative.}

\begin{align*}
(6) & \quad \text{a. } S \rightarrow \text{NP VP} \quad \lambda x\lambda y[x \, y] \\
& \quad \text{b. } \text{VP} \rightarrow \text{TV NP} \quad \lambda x\lambda y[x \, y] \\
\end{align*}

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 \textit{Bill met Fred}, as derived earlier, will have the meaning given by the λ-term (7a) which has the reduced form (7b); \textit{met'}, \textit{Fred'}, and \textit{Bill'} denote the meanings of the corresponding words.

\begin{align*}
(7) & \quad \text{a. } \lambda z\lambda w[w \, z] \text{ Bill'} (\lambda x\lambda y[x \, y] \text{ met'} \text{ Fred'}) \\
& \quad \text{b. } \text{met'} \text{ Fred'} \text{ Bill'} \\
\end{align*}

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

\begin{align*}
(8) & \quad \text{a. } \text{PV} \rightarrow \text{TPV NP} \quad \lambda x\lambda y[x \, y] \\
& \quad \text{b. } \text{VP} \rightarrow \text{PV PP} \quad \lambda x\lambda y[x \, y] \\
& \quad \text{c. } \text{PP} \rightarrow \text{TP NP} \quad \lambda x\lambda y[x \, y] \\
\end{align*}

\begin{align*}
(9) & \quad \text{showed'} \text{ Edinburgh'} (\text{to' Sue'}) \text{ Fred'} \\
\end{align*}

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 \textit{NP} is of type \((e\rightarrow t)\rightarrow t\) and a \textit{VP} is of type \(((e\rightarrow t)\rightarrow t)\rightarrow t\) 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:

(10) a. If $X$ is a basic category
    then $X$ is a category

b. If $X$ and $Y$ are categories
    then $X/Y$ and $XY$ 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 $XY$ 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 $(SNP)/NP$ 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 $(SNP)/(SNP)NP$ whereby it combines with an object on its right, then a prepositional adverbial, $(SNP)SNP$ 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 $(SNP)/NP$ and $((SNP)$SNP$)/NP$ may be written SNP/NP and SNP(SNP)/NP respectively. Analyses of Bill met Mary and Fred showed Edinburgh to Sue are as follows:

(11) Bill met Mary
     NP SNP/NP
     S

---

3 The set defined is the smallest one satisfying the specified conditions.
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 properties encoded in the lexical information associated with words.

Accompanying the syntactic rule that an expression of category $X/Y$ (or $X\backslash Y$) 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 $X\backslash Y$) *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* (>)
       \[ X/Y: x + Y; y \Rightarrow X: x y \]
b. *Backward Application* (<)
       \[ Y: y + X/Y: x \Rightarrow X: x y \]

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)  \[ \text{met' Mary'} \]

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

(15)  \[ \text{met' Mary'} \text{' Bill'} \]

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

(16)  \[ \text{showed' Edinburgh' (to' Sue')} \text{' Fred'} \]

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)  \[ \tau(X/Y) = \tau(XY) = \tau(Y) \rightarrow \tau(X) \]

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), Szabolcsi (1983), and van Benthem (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 (XY)/Z is written Z/(YN), and that used by Lambek, according to which (XY)/Z is written (YN)/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 SNP/NP*NP might index a category of expressions which combine with two NPs 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. \( NP[3,NUM,CASE] \rightarrow DET[NUM] \ N[NUM] \)
    b. \( S[VFORM] \rightarrow NP[PER,NUM,nom] \ VP[PER,NUM,VFORM] \)

(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, prefer in he prefers that John stay may be introduced by the following rule and lexical entry:
<table>
<thead>
<tr>
<th>PSG Categories</th>
<th>CG Categories</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>S</td>
<td>sentence</td>
</tr>
<tr>
<td>VP</td>
<td>SNP</td>
<td>verb phrase</td>
</tr>
<tr>
<td>ADV</td>
<td>SNP(SNP)</td>
<td>adverbial</td>
</tr>
<tr>
<td>PP</td>
<td>SNP(SNP)</td>
<td>intransitive preposition</td>
</tr>
<tr>
<td>TP</td>
<td>SNP(SNP)/NP</td>
<td>transitive preposition</td>
</tr>
<tr>
<td>XP</td>
<td>SNP(SNP)/(S(NP))</td>
<td>control preposition</td>
</tr>
<tr>
<td>AUX</td>
<td>SNP(SNP)</td>
<td>auxiliary</td>
</tr>
<tr>
<td>XV</td>
<td>SNP(SNP)</td>
<td>control verb</td>
</tr>
<tr>
<td>TXV</td>
<td>SNP(SNP)/NP</td>
<td>transitive control verb</td>
</tr>
<tr>
<td>TV</td>
<td>SNP(NP)</td>
<td>transitive verb</td>
</tr>
<tr>
<td>TTV</td>
<td>SNP(NP)/NP</td>
<td>ditransitive verb</td>
</tr>
<tr>
<td>SV</td>
<td>SNP(SP)</td>
<td>sentential verb</td>
</tr>
<tr>
<td>TSV</td>
<td>SNP(SP/NP)</td>
<td>transitive sentential verb</td>
</tr>
<tr>
<td>TTSV</td>
<td>SNP(SP/NP/NP)</td>
<td>ditransitive sentential verb</td>
</tr>
<tr>
<td>PV</td>
<td>SNP/(S(NP))/(S(NP))</td>
<td>prepositional verb</td>
</tr>
<tr>
<td>TPV</td>
<td>SNP/(S(NP))/(S(NP))/NP</td>
<td>transitive prepositional verb</td>
</tr>
<tr>
<td>COP</td>
<td>SNP/(N/N)</td>
<td>copula</td>
</tr>
<tr>
<td>SP</td>
<td>SP</td>
<td>complementized sentence</td>
</tr>
<tr>
<td>COMP</td>
<td>SP/S</td>
<td>complementizer</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>common noun</td>
</tr>
<tr>
<td>PP</td>
<td>NN</td>
<td>intransitive preposition</td>
</tr>
<tr>
<td>REL</td>
<td>NN</td>
<td>relative clause</td>
</tr>
<tr>
<td>TP</td>
<td>NN/N</td>
<td>intransitive preposition</td>
</tr>
<tr>
<td>RELPROs</td>
<td>NN/(S(NP))</td>
<td>subject relative pronoun</td>
</tr>
<tr>
<td>RELPROo</td>
<td>NN/(S(NP))</td>
<td>object relative pronoun</td>
</tr>
<tr>
<td>AP</td>
<td>N/N</td>
<td>adjective</td>
</tr>
<tr>
<td>Af</td>
<td>N/N/(S(NP))/NP</td>
<td>tough-like adjective</td>
</tr>
<tr>
<td>SN</td>
<td>N/SP</td>
<td>sentential noun</td>
</tr>
<tr>
<td>PN</td>
<td>N/(NN)</td>
<td>prepositional noun</td>
</tr>
<tr>
<td>NP</td>
<td>NP</td>
<td>proper name, noun phrase</td>
</tr>
<tr>
<td>DET</td>
<td>NP/N</td>
<td>determiner</td>
</tr>
</tbody>
</table>

Figure 4: Common Categories

(24)  \[\text{VP[PER,NUM,VFORM]} \rightarrow \]

\[\text{SV[PER,NUM,VFORM,COMP,SUBORDVFORM]} \quad \text{SP[COMP,SUBORDVFORM]}\]

(25)  \[\text{prefers := SV[3,sg,fin,that,bse]}\]
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 $S[/fin\backslash NP[3,sg,nom]/SP[that,bse]]$, 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. $NP \rightarrow DET \quad N \quad \lambda x\lambda y[x\, y]$
    the $:= DET$
    b. the $:= NP/N$

(27) a. $S \rightarrow NP \quad VP \quad \lambda x\lambda y[y\, x]$
    left $:= VP$
    b. left $:= SNP$

Thus:

(28) $[[The\, students]\, left]$

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

(29) a. $VP \rightarrow VP \quad ADV \quad \lambda x\lambda y[y\, x]$
    quickly $:= ADV$
    b. quickly $:= SNP(SNP)$

(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. $PP \rightarrow TP \ NP \ \lambda x \lambda y[x \ y]$  
    with $:\ \rightarrow TP$  
    $VP \rightarrow VP \ PP \ \lambda x \lambda y[y \ x]$  
    b. with $:\ \rightarrow S\NP/(S\NP)/NP$

(32) The students left [with John]

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

(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. 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. $VP \rightarrow AUX \ VP$  
    will $:\ \rightarrow AUX$  
    b. will $:\ \rightarrow S\NP/(S\NP)$

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 $S[fin], S[se]$ and $S[inf]$ are written $Sfin, Sse$ and $Sinf$.

(36) \[
\begin{array}{cccc}
Sfin\NP/(Sse\NP) & Sse\NP/(Sinf\NP) & Sinf\NP/(Sse\NP) & Sse\NP \\
Sinf\NP \rightarrow f & Sinf\NP \rightarrow f & Sinf\NP \rightarrow f & Sse\NP \rightarrow f \\
Sse\NP \rightarrow f & Sse\NP \rightarrow f & Sse\NP \rightarrow f & Sfin\NP \rightarrow f \\
\end{array}
\]

4Semantically, adnominal and adverbial prepositional phrases will presumably both be of a type mapping $e \rightarrow t$ to $e \rightarrow t$. 
Unacceptable orderings such as *have will to leave cannot be derived.\(^5\)

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

\[
\begin{align*}
(37) \quad & a. \quad VP \rightarrow XV \ VP \quad \lambda x \lambda y[x \ y] \\
& \quad \text{try ::= } XV \\
& \quad b. \quad \text{try ::= } S\text{NP}/(S\text{NP})
\end{align*}
\]

(38) \quad I [tried [to leave]]

It is natural to assume that the ‘subject-control’ verb promise in Sue promised Ralph to go is subcategorized for two complements, as opposed to an infinitival sentence, because Sue, and not Ralph, is the subject of the subordinate verb phrase. It is less clear whether an ‘object-control’ verb such as want (I want John to leave) should be regarded as taking a noun phrase and a verb phrase complement, or a single sentence complement, and it is controversial to propose that believe (I believe John left) might have the former character. These verbs are mentioned here for completeness; discussion of their categorisation is postponed.

Transitive verbs take a single object; ditransitives take two:

\[
\begin{align*}
(39) \quad & a. \quad VP \rightarrow TV \ NP \quad \lambda x \lambda y[x \ y] \\
& \quad \text{referenced ::= } TV \\
& \quad b. \quad \text{referenced ::= } S\text{NP}/NP
\end{align*}
\]

(40) \quad I [referenced you]

(41) \quad a. \quad TV \rightarrow TTV \ NP \quad \lambda x \lambda y[x \ y] \\
& \quad b. \quad \text{lent } l \rightarrow S\text{NP}/NP/NP

(42) \quad I [lent John] Faust

Complementizers and verbs subcategorized for complementized sentences are characterised thus:

\[^5\text{For discussion of auxiliary ordering in phrase structure grammar and categorial grammar see e.g. Gazdar, Pullum and Sag (1982), and Bach (1983) and Carpenter (forthcoming), respectively.}\]
(43) a. \( SP \rightarrow \text{COMP} \ S \) \( \lambda x\lambda y[x \ y] \)
    that := COMP
b. that := SP/S

(44) [that [John left]]

(45) a. \( VP \rightarrow \text{SV} \ \text{SP} \) \( \lambda x\lambda y[x \ y] \)
    thinks := SV
b. thinks := \( \text{SNP} \text{/SP} \text{/SP} \)

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

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

(47) a. \( SV \rightarrow \text{TSV} \ NP \) \( \lambda x\lambda y[x \ y] \)
    told := TSV
b. told := \( \text{SNP} \text{/SP} \text{/NP} \text{/NP} \)

(48) She [told Ralph] that she went

(49) a. \( TSV \rightarrow \text{TTSV} \ NP \) \( \lambda x\lambda y[x \ y] \)
    bet := TTSV
b. bet := \( \text{SNP} \text{/SP} \text{/NP} \text{/NP} \text{/NP} \)

(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. \( VP \rightarrow \text{PV} \ PP \) \( \lambda x\lambda y[x \ y] \)
    searched, looked := PV
b. searched, looked := \( \text{SNP} \text{/SNP(SNP)} \text{(SNP)} \)
(52) a. We [searched [for Ralph]]
b. We [looked up]

(53) a. PV → TPV NP \( \lambda x \lambda y[x \ y] \)
    put, rang := TPV
b. put, rang := S\(\text{NP}(S\text{NP}(S\text{NP})))/NP\)

(54) 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:\textsuperscript{6}

(55) a. VP → COP AP \( \lambda x \lambda y[x \ y] \)
    is := COP
b. is := S\(\text{NP}(N/N))\)

(56) John [is fat]

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

(57) a. N → AP N \( \lambda x \lambda y[x \ y] \)
    fat := AP
b. fat := N/N

(58) the [fat man]

(59) a. N → N PP \( \lambda x \lambda y[y \ x] \)
    outside := PP
b. outside := NNN

(60) the [man outside]

\textsuperscript{6}The copula cannot combine with intensional adjectives:

(i) *John is alleged
(61) a. \( PP \rightarrow TP \ NP \quad \lambda x \lambda y [x \ y] \)
   from := TP
b. from := N/N/NP

(62) the man [from Edinburgh]

(63) a. \( REL \rightarrow RELPROs \ VP \quad \lambda x \lambda y [x \ y] \)
   who := RELPROs
b. who := N/N/(SNP)

(64) 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 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 ‘adjunctively’ anyway, but if noun subcategorization for complemented sentences is being hypothesized, it seems appropriate to assume noun subcategorization for prepositional phrases also.

(65) a. \( N \rightarrow SN \ SP \quad \lambda x \lambda y [x \ y] \)
   belief := SN
b. belief := N/SP

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

(67) a. \( N \rightarrow PN \ PP \quad \lambda x \lambda y [x \ y] \)
   search, picture := PN
b. search, picture := N/(N/N)

(68) 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:

\[(X \text{ Coord } X)_X\]

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 (70), but not with each other as in (71):\(^7\)

(70)  
- 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

(71)  
- 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:

(72)  
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 (73).

(73)  
- 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 (74) where ‘+’ indicates one or more repetitions.

---

\(^7\) Throughout ‘*’ and ‘?’ are used to indicate (my) acceptability judgements.
(74) Coordination

[X+ and X]_X
[(X and)+ X]_X
[X+ or X]_X
[(X or)+ X]_X

For example:

(75) 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) a. London, I liked \(e_i\)
   b. the town which, 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 think that John argued that Sue likes \(e_i\)
   b. the town which, 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) \(X \rightarrow \ldots Y \ldots\)

\[ \Rightarrow \]

\(X/Z \rightarrow \ldots Y/Z \ldots\)

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

\[1\]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.
form an expression of category $X/Z$. By way of example, application of (3) to (4a) and (5a) can yield (4b) and (5b) respectively.

\[
\begin{align*}
(4) & \quad \text{a. } S \rightarrow \text{NP, } \text{VP} \\
& \quad \text{b. } S/\text{NP} \rightarrow \text{NP, } \text{VP/NP} \\
(5) & \quad \text{a. } \text{VP} \rightarrow \text{TV, } \text{NP} \\
& \quad \text{b. } \text{VP/NP} \rightarrow \text{TV, } \text{NP/NP}
\end{align*}
\]

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 $X$:

\[
X/X \rightarrow e
\]

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).

\[
\begin{align*}
(7) & \quad X \rightarrow \ldots Y \ldots \\
& \quad \quad \Rightarrow \\
& \quad X/Y \rightarrow \ldots \ldots
\end{align*}
\]

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

---

![Figure 1](attachment:image.png)
$X/Y$. 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 course of this discussion.

A topic may be introduced thus:

(8) \[ \text{Topic Introduction} \]
\[ S \rightarrow X \ S/X \]

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

(9) \[ [\text{London}_{NP}, [I \ \text{liked} \ e_1]_{S/NP}]_S \]

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:

(10) \[ \text{REL} \rightarrow RELPROo \ S/NP \]

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

---

\[ S/NP \]
\[ \text{NP} \]
\[ I \]
\[ \text{VP/NP} \]
\[ \text{TV} \]
\[ \text{liked} \]

Figure 2

---

\(^2\)Subject relative pronoun introduction was illustrated in chapter 1; pied piping is discussed later, in connection with categorial grammar.
As well as noun phrases, complementized sentences can topicalise; complex adjectives topicalise better than basic ones:

(11) [That John will stay], I can believe $e_i$

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

The topicalisation of a finite verb phrase in is unacceptable; that of a base form verb phrase and infinitival verb phrase is better:

(13) a. *[Will eat mushrooms], I think that John $e_i$
    b. *[Eat mushrooms], I think that John will $e_i$
    c. *[To go to London], 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], Sue arrived \( e_i \)
    b. ?[To London], 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 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 put \( e_i \) on the table
    b. [That John will stay], 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.\(^3\)

\(^3\)The formulation of the semantics here is different from the usual 'designated variable' method, which Engdahl (1986, pp24-28) notes to be technically problematic.
(18) \textit{Right Abstraction} \\
\[ \begin{align*}
X & \to Y \ Z & \phi \\
\implies \text{R} & \\
X/W & \to Y \ Z/W & \lambda x \lambda y \lambda z[\phi \ x \ (y \ z)]
\end{align*} \]

(19) \textit{Middle Abstraction} \\
\[ \begin{align*}
X & \to Y \ Z & \phi \\
\implies \text{M} & \\
X/W & \to Y/W \ Z & \lambda x \lambda y \lambda z[\phi \ x \ z \ y]
\end{align*} \]

(20) \textit{Null Rule Gap Introduction} \\
\[ \begin{align*}
X/X & \to e & \lambda x[x]
\end{align*} \]

(21) \textit{Topic Introduction} \\
\[ \begin{align*}
S & \to X \ S/X & \lambda x \lambda y[y \ x]
\end{align*} \]

(22) \textit{Relative Pronoun Introduction} \\
\[ \begin{align*}
\text{REL} & \to \text{RELPRo0} \ S/NP & \lambda x \lambda y[x \ y]
\end{align*} \]

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. By way of example of how the semantics works, the semantics of the derived rules, and the meanings of the constituents, in the analysis of \textit{London, I liked} are as follows:

(23) a. \[ \begin{align*}
S & \to \ NP \ \ VP & \lambda x \lambda y[y \ x]
\implies \text{R} & \\
S/NP & \to \ NP \ \ VP/NP & \lambda x \lambda y \lambda z[y \ z \ x]
\end{align*} \]

b. \[ \begin{align*}
\text{VP} & \to \ TV \ \ NP & \lambda x \lambda y[x \ y]
\implies \text{R} & \\
\text{VP/NP} & \to \ TV \ \ NP/NP & \lambda x \lambda y \lambda z[x \ (y \ z)]
\end{align*} \]
\[ e \Rightarrow \lambda x[x] \]
liked \[ \Rightarrow \text{liked}' \]
liked \( e \) \[ \Rightarrow \lambda x[\text{liked}' \times x] \]
I \[ \Rightarrow I' \]
I liked \( e \) \[ \Rightarrow \lambda x[\text{liked}' \times I'] \]
London \[ \Rightarrow \text{London}' \]
London I liked \( e \) \[ \Rightarrow \text{liked'} \text{ London'} I' \]

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:

\[(25)\]
\[a. \ *\text{the machine which}_i ^1 \text{ I met [the man who invented}_i ^1 \text{ NP} \]
\[b. \ *\text{the items which}_i ^1 \text{ he explained [the fact that he bought}_i ^1 \text{ NP} \]

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 \( N/\text{NP} \) is not legitimate. However although the generalisation holds by and large, I find, say (26) semi-acceptable.

\[(26)\]
\[?\text{a colleague whom John acquired [a belief that I disliked}_i ^1 \text{ NP} \]

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):

\[(27)\]
\[a. \ *\text{the child who}_i ^1 \text{ John married [a girl who dislikes}_i ^1 \text{ NP} \]
\[b. \ ?\text{the child who}_i ^1 \text{ I know [a family which is willing to adopt}_i ^1 \text{ NP} \]
\[c. \ \text{the child who}_i ^1 \text{ there is [nobody who is willing to adopt}_i ^1 \text{ NP} \]

Chung and McCloskey (1983) claim that extraction from subject relative clauses is more acceptable than extraction from object relative clauses; for example they contrast (27c) with (28) which they mark as unacceptable, though I find this particular example good.
(28) the child that, there is [no one who the authorities can persuade to accept $e_i]_{\text{NP}}$

In view of data such as (29), Chomsky (1973) forwarded the ‘subject condition’ which asserts that all subjects are islands.\(^4\)

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

Again this could be captured by stipulating that $\text{NP[nom]}/\text{NP}$ is somehow ill-formed, but again also the facts are not clear; I find (29) semi-acceptable, and (30) fine.

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

Stipulating $\text{*NP/NP}$ 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 (31) but implies that those in (32) have some exceptional structure.

(31) a. $\text{*the man who}_i$ John destroyed a book about $e_i$
   b. $\text{?the man who}_i$ I lost a picture of $e_i$

(32) 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:

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

In this case an appropriate condition might be $\text{*X/X}$.

\(^4\)The constraint also covers Ross’s sentential subject constraint according to which sentential subjects are islands:

(i) $\text{*the subject which}_i$ that John likes $e_i$ is obvious
(ii) $\text{*the pleasures which}_i$ for you to give up $e_i$ would be a pity
In general, adverbials (like adnominals) have an island character, suggesting *ADV/NP:

(34)  *a debate which, John made his vote [without attending e₁]ADV

However not all extractions are completely unacceptable:

(35)  a. ?the city which, John met Mary [in e₁]ADV
     b. ?the people who, John left the party [without meeting e₁]ADV

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

(36)  a. the papers which, John went to Paris [without reading e₁]ADV
     b. the people who, he arrived [with e₁]ADV
     c. the path which, we ran [along e₁]ADV

Here too then island constraints on extraction present a complicated picture; compare incidentally the acceptable NP extraction in (36a) with the unacceptable prepositional phrase extraction in (37) (cf. the A-over-A constraint).

(37)  *the people [to whom] he went to Paris [without speaking e₁]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.

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

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

(40)  a. You and I ought to go shopping
     b. *I and you ought to go shopping
Example (38) illustrates how apparent well-formedness diminishes with centre-embedding of relative clauses. Well-formedness judgements of (39) are susceptible to the 'heaviness' of the object noun phrase, and the ordering preference in (40) 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:

\[(41) \quad * \text{the man who I think that } e_1 \text{ left}\]

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 (42); see Figure 4.

\[(42) \quad S \rightarrow NP \ VP
\]

\[==>M \]

\[S/NP \rightarrow NP/NP \ VP\]

In some versions of Generalised Phrase Structure Grammar (GPGS, 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 achieved by separate feature percolation conventions. Then the rule in (42) would not be derived because the input would not be a rule with a lexical head.\(^5\) Note however that the gap-introduction metarule (43), as an alternative to null rule gap introduction, appears to get

\(^5\) In GPGS intransitive verbs would be introduced by a unary rule mapping an X-bar level 0 intransitive verb to the VP X-bar level.
the facts right.

(43) \[ X \rightarrow Y \ Z \]

\[ \Rightarrow \]

\[ X/Z \rightarrow Y \]

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:

(44) a. I saw [John's book]
    b. *the man who, I saw 's book

(45) a. This is John's
    b. Whose is this?
(46)  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):

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

(48)  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 a version of the left branch condition.

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:

(49)    A man e<sub>i</sub> arrived [who swims]<sub>i</sub>

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:

(50)    **Rightward Filler Introduction**
         $X \rightarrow X/Y \ Y \ \ \ \ \lambda x \lambda y\{x\ y\}$

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 non-canonical counterparts. For example (49) is analysed as shown in Figure 5; the semantics is as follows:

(51)    man $\Rightarrow \text{man'}$
        $e \Rightarrow \lambda x\{x\}$
        $\text{man} \ e \Rightarrow \lambda x\{x \ \text{man'}\}$
In addition to non-subcategorized adnominals, a noun complement can be right extrapo-
posed:

(52) A rumour \( e_i \) spread [that TTK had gone bust], \( i \)

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

(53) 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], \( i \)

\[
\begin{align*}
S & \quad S/REL \quad NP/REL \quad REL \quad VP \quad RELPROs \quad VP \\
& \quad DET \quad N/REL \quad arrived \quad who \quad swims \\
& \quad a \quad man \quad e
\end{align*}
\]

Figure 5
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 extraposited past the relative clause:

(54) 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). 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. (One possibility is that this connects with the tendency for new information to come at the end of sentence: heavy elements are relatively ‘likely’ to contain new information.) In the following the direct object is heavy shifted past the indirect object:

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

This has the analysis shown in Figure 6 where semantics is as follows:

(56)

\[
\begin{align*}
& e & \Rightarrow \lambda x [x] \\
& \text{gave} & \Rightarrow \text{gave'} \\
& \text{gave } e & \Rightarrow \lambda x [\text{gave'} x] \\
& \text{to} & \Rightarrow \text{to'} \\
& \text{John} & \Rightarrow \text{John'} \\
& \text{to John} & \Rightarrow \text{to'} \text{ John'} \\
& \text{gave } e \text{ to John} & \Rightarrow \lambda x [\text{gave'} x \,(\text{to'} \text{ John'})] \\
& \text{the most } ... & \Rightarrow \text{the-most-...'} \\
& \text{gave } e \text{ to John the most } ... & \Rightarrow \text{gave'} \text{ the-most-...'} \,(\text{to'} \text{ John'}) \\
& \text{I gave } e \text{ to John the most } ... & \Rightarrow \text{gave'} \text{ the-most-...'} \,(\text{to'} \text{ John'}) I'
\end{align*}
\]

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 is specifically a noun phrase.

7Cf. the discussion of acceptability and grammaticality earlier.
(57) a. I believe $e_i$ to be incompetent [a good number of the members of the board]

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

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:

(58) I rang $e_i$ up [the press]

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

(59) *I looked up it

Heavy shift of the indirect object of a verb like gave in its 'dative-shifted' form is less

---

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.
acceptable.\(^9\)

\[(60)\]

\(a. \) *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 conflated, and the corresponding left extraction is of considerably higher acceptability:

\[(61)\]

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:

\[(62)\]

*I bet \(e_i\) $5 [the man over there],\(i\) that we’d win

Note that the ‘landing site’ in (62) 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:

\[(63)\]

I gave \(e_i\) to John [the most recent version of the paper],\(i\) without remembering that it criticised his thesis

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

\[(64)\]

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.\(^{10}\)

\[(65)\]

\(a. \) 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:

\[(66)\]

*I wanted \(e_i\) yesterday [to go shopping],\(i\)

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

---

\(^9\)In transformational approaches, [gave NP(dir) NP(dir)], is regarded as having been derived by a rule of ‘dative-shift’ from [gave NP(dir) to NP(dir)].

\(^{10}\)On other accounts, (65) may not be regarded as non-canonical, i.e. both word orders may be regarded as basic; the assumption here is that there is a single underlying ‘normal’ order for this part of English.
Although various categories can right extract, and be introduced at various locations, not all categories can do so. Thus:

\[(67) \quad *a_{e_i} \text{ left [man from London]}_{i} \]

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 \((68)\) 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.

\[(68) \quad *[e_{NP/NP} \text{ left}_{VP\text{S/NP}} \text{ Bill}_{NP}]_{S} \]

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

Rightward Filler Introduction will 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 7; but these analyses both assign the canonical meaning. Thus we saw earlier that the meaning assigned to *Bill met* is $\lambda x[\text{met' } x \text{ Bill']}$, and since the semantics of filler introduction is to apply this to the filler, the analysis in Figure 7b will assign the canonical meaning *met' Bill' Mary*. Since Right Abstraction and

---

![Figure 7](image-url)
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:

\[(69)\]  
a. *[an argument [about a picture e₁] started [of bill’s first wife]i]

b. *I [believed that John [liked e₁] 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:

\[(70)\]  
[A number [of reports e₁] soon appeared [on the Watergate Affair]i

And Stucky’s (1987, p391) example (71) involves right extraposition of an embedded modifier.

\[(71)\]  
The names of all the painters e₁ 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:

\[(72)\]  
I have [wanted [to meet e₁] 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 (70), (73) and (74) are of very low acceptability.

\[(73)\]  
*He has believed that [Mary knows a man e₁] for many years [who smuggles]i
(74) *He has believed that [Mary knows $e_1$] for many years [a man who is widely believed to be involved in smuggling]$_1$

The collapsing of right extrapolation and heavy shift suggests that (75) should be grammatical.

(75) *A woman who knows $e_1$ arrived [a man widely believed to be involved in smuggling]$_1$

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

(76) a. *I talked about $e_1$ to Mary [all the news]$_1$
b. the news which$_1$ I talked about $e_1$ 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. Since as before, the account developed does not lead us to expect these constraints on right extraction, and in view of their 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 the fact 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:

(77) *I think that $e_1$ left [the man who you wanted to meet]$_1$

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 1981, 83) 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.\footnote{Recall however that we are regarding neither subjects nor adverbs as complete islands.}
(78) a. a paper which I filed the records without reading $e_i$
b. a paper which I filed $e_i$ without reading $e_i$

(79) a. a man whom the friends of $e_i$ envied Sue
b. a man whom the friends of $e_i$ envied $e_i$

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

Intuitions as to acceptability of parasitic constructions vary considerably. I find (80b) 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.\(^{12}\)

(81) a. $X \rightarrow \ldots Y/NP \ldots Z \ldots$

\[ \Rightarrow \]

$X \rightarrow \ldots Y/NP \ldots Z/NP \ldots$

b. $X \rightarrow \ldots Y \ldots Z/NP \ldots$

\[ \Rightarrow \]

$X \rightarrow \ldots Y/NP \ldots Z/NP \ldots$

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, (81) can also apply to the rules for Topic Introduction and the Rightward Filler Introduction, with calamitous results:

(82) a. $S \rightarrow NP$ $S/NP$

\[ \Rightarrow \]

$S \rightarrow NP/NP$ $S/NP$

b. $S \rightarrow S/NP$ $NP$

\[ \Rightarrow \]

$S \rightarrow S/NP$ $NP/NP$

According to these the following should be grammatical sentences:

\(^{12}\)Sag is able to express the rule rather more economically using immediate dominance/linear precedence factoring but this is not important here.
(83)  

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 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 (84).

(84)  

Parasitic Abstraction

\[
\begin{align*}
X & \rightarrow YZ \\
\quad & \v  \\
\Rightarrow_p & \\
X/W & \rightarrow Y/W Z/W \\
\lambda x \lambda y \lambda z [\v(x z) (y z)]
\end{align*}
\]

Then *filed without reading* is analysed as shown in Figure 8.

Parasitic extraction can be to the right as well as to the left, and is unbounded:

---

![Diagram](image_url)  

Figure 8
I filed $e_i$ without reading $e_i$ (at all) [a paper I was meant to review at once]$_i$

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 (87) appears to have a prepositional phrase-prepositional phrase parasitic reading, and Tait (1988) offers (88) as an instance of prepositional phrase-noun phrase parasitic extraction.

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

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

Consider the following:

(89) *the patient who$_i$ I showed $e_i$, $e_i$

The current account generates this as shown in Figure 9. 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.

---

![Diagram](image-url)

**Figure 9**
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, but in both cases the unacceptability of (90) remains unaccounted for (cf. Gazdar et al. 1985, p166).

\[(90)\]
\[\text{a. } ?\text{the models whom}_{1} I \text{ sent the pictures of } e_{1} \text{ to } e_{1}\]
\[\text{b. } *\text{the slave whom}_{1} I \text{ gave } e_{1} \text{ to } e_{1}\]

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 (91) are described as exhibiting ‘right node raising’ (Postal 1974 pp125-128; Bresnan 1974).

\[(91)\]  
\[\text{[I liked } e_{1} \text{ but Suzy hated } e_{1}, \text{ London}_{1}\]

Transformationally, the right-peripheral object shared by the two verbs in (91) is viewed as having been ‘raised’ out of the coordinate structure, as illustrated in Figure 10. Right node raising is not a local (i.e. clause-bound) phenomenon; in (92a) it crosses a clause boundary and in (92b) it crosses two clause boundaries; the relation between the filler and the gap is an instance of unbounded dependency.
Figure 10: Classical right node raising

(92) a. [John said that Sue likes $e_i$ and Fred said that Sue dislikes $e_j$] [newsletters full of trivia]$_i$
   b. [John said that Sue likes $e_i$ and Robert said that Liz thinks that Sue dislikes $e_j$] [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 (91) is analysed as shown in Figure 11, and the unboundedness exemplified by (92) is captured by iteration of the relevant operations, like the unboundedness of left extraction.

In addition to noun phrases, complementized sentences can be right node raised; note that in (93b) right node raising is out of a noun phrase as opposed to a sentence:

(93) 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:
(94) [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 (96) the node raising is again out of a noun phrase:

(95) 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$_i$

(96) 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 (97).

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

Right node raising of common noun phrases is also an unclear area; the node raising out of a noun phrase in (99b) is more acceptable than that out of a verb phrase in (99a).

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

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 (100) and adnominals as in (101), notwithstanding their island character as regards left extraction.

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

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

Examples (102), (103), and (104) exhibit right node raising out of common noun phrases, verb phrases, and complementized sentences respectively.

(102) a. the [paintings $e_i$ and small sketches $e_i$] [by Picasso],
   b. the [belief $e_i$ and foolish hope $e_i$] [that they would come back],
   c. the [arguments for $e_i$ and arguments against $e_i$] [the second option],
(103) a. We [will talk e_i and might argue e_i] [about some personal things]_
   b. I [read e_i and will reference e_i] [several papers]_
   c. He [believes e_i and has proposed e_i] [that we should take more direct action]_
   d. We [looked for e_i and found e_i] [a village with a good inn]_
   e. He [arrived with e_i but left without e_i] [the girl he used to date at school]_

(104) a. He claims [that he arrived e_i and that he left e_i] yesterday_
   b. He claims [that he knew e_i and that he loved e_i] Maria_
   c. She wondered [whether he said e_i or whether he implied e_i] [that they laughed]_

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:

(105) a. I met [e_i John on Monday and e_i Sue on Tuesday]
   b. He said [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 [e_i good natured on Fridays but e_i moody on Mondays]
   d. We looked [e_i for blackberries on Monday and e_i for strawberries on Tuesday]
   e. He wanted [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 (106), provided it is interpretable by the informal elliptical interpretation rule (107).

(106) \( V^2[\text{CONJ } \alpha] \rightarrow \alpha, X^{2^+} \)

where \( \alpha \in \{\text{and, but, nor, or}\} \)

(107) 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 \( [\text{CONJ } \alpha] \) 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 \( \text{either} \) in (108) makes it clear that the structure is as shown in (109); the Sag et al. account will erroneously attempt to assign a structure such as (110a) or (110b) since the material preceding the end of the coordinate structure must be analysed as a verb phrase or a sentence.

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

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

(110) 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:

(111) \( X \rightarrow Y \ldots \)

\[ \Rightarrow \]

\( XZ \rightarrow YZ \ldots \)
(112) \( X \rightarrow Y \ldots \)

\[ \Rightarrow \]

\[ X \backslash Y \rightarrow \ldots \]

Schachter and Mordechay use a different notation (the backward slash used here is deliberately suggestive of categorial grammar). The binary instance of (111) is (113), and although they introduce gaps by the metarule (112), I will continue to illustrate using the null rule gap introduction (114); the filler introduction rule for backward slashes is (115).

(113) \textit{Left Abstraction}

\[ X \rightarrow Y \ Z \quad \phi \]

\[ \Rightarrow_L \]

\[ X \backslash W \rightarrow Y \backslash W \ Z \quad \lambda x \lambda y \lambda z[\phi(x \ z) \ y] \]

(114) \textit{Null Rule Gap Introduction}

\[ XX \rightarrow e \quad \lambda x [x] \]

(115) \textit{Leftward Filler Introduction}

\[ X \rightarrow Y \ \backslash Y \quad \lambda x \lambda y[y \ x] \]

For example, (1) \textit{met John on Monday and Sue on Tuesday} is analysed as shown in Figure 12.
The following, in which a noun is left node raised from conjuncts containing a complement and an adnominal, is less acceptable for some reason:

(116) 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:

(117) 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 (118b) is less acceptable than (118a).

(118) 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:

(119) 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]

In (119) there is extraction from two non-peripheral positions. In (120) one extraction site is right-peripheral and the other is non-peripheral; in (121) both extraction sites are right-peripheral. Note that (121b) 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).
(120)  a. the book which, I [read e, and sent e, to John]
b. I [read e, and sent e, to John] [a book about horses],

(121)  a. a book which, [John wrote e, and I read e,]
b. [John wrote e, and I read e,] [a book which was later to be censored],

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 (120) 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:

(122)  a. *the man who, [John liked e, and Mary hated Fred]
b. *the man who, [John liked Fred and Mary hated e,]

(123)  the man who, [John liked e, and Mary hated e,]

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.

(124)  a. *the woman whom, I saw [e, and a picture of e,]
b. *the woman whom, I saw [a picture of e, and e,]

Also, empty node gap introduction allows an intransitive verb to analyse as S/NP, incorrectly permitting (125).

(125)  *the man who, [e, arrived and Mary met e,]

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 13.

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 this is true:

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

(127) 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:

(128) 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:

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

(130) 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$_i$

(131) 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]$_i$

(132) 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]$_i$

Bouma (1987) argues that asymmetry in the island constraints for extraction and coordination indicate that their treatment should not be collapsed, citing (133) (from Wexler and
Right Abstraction
\[ X \rightarrow Y Z \rightarrow^R \phi \]
\[ X/W \rightarrow Y Z/W \lambda x \lambda y \lambda z [\phi \ x \ (y \ z)] \]

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

Parasitic Abstraction
\[ X \rightarrow Y Z \rightarrow^P \phi \]
\[ X/NP \rightarrow Y/NP Z/NP \lambda x \lambda y \lambda z [\phi \ (x \ z) \ (y \ z)] \]

Left Abstraction
\[ X \rightarrow Y Z \rightarrow^L \phi \]
\[ X/W \rightarrow Y/W Z \lambda x \lambda y \lambda z [\phi \ (x \ z) \ y] \]

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

Leftward Filler Introduction
\[ X \rightarrow Y X/Y \lambda x \lambda y [y \ x] \]

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

Relative Pronoun Introduction
\[ \text{REL} \rightarrow \text{RELPROo} S/NP \lambda x \lambda y [x \ y] \]

Figure 13: Augmentations to PSG

Culicover 1980, p299) and (134) (from Levine 1985) which exhibit right node raising out of complex noun phrases.
(133) Mary knows a man who buys, and Bill knows a man who sells, pictures of Fred

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

However it was suggested earlier that in general left extraction from complex noun phrases may not be ungrammatical.

The current unified account of extraction and coordination is strengthened insofar as the predicted concord exists; and weakened insofar as it does not. I will regard the simplicity of the unified account as providing adequate motivation to continue this line of inquiry, and in the next section I show that the existing metarules characterise cases of compound non-canonicity with no further amendment.

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-canonicity. 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:

(135) \[ e_i \ e_j \text{ yesterday \{a copy of the newsletter\}_i \text{ to every student\}_j} \]

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:

(136) a. the people [to whom] \[ e_i \ e_j \text{ yesterday \{copies of the newsletter\}_i } \\
     b. the newsletter which \[ e_i \ e_j \text{ yesterday \{to every member in the area\}_j } \\

Consider the following applications of metarules:

(137) \[ PV \rightarrow TPV \ NP \]
     \[
     \Rightarrow_R
     \]

\[ PV/NP \rightarrow TPV \ NP/NP \]

\[ \lambda x \lambda y \lambda z [x \ (y \ z)] \]

(138) \[ VP \rightarrow PV \ PP \]
     \[
     \Rightarrow_R
     \]

\[ VP/PP \rightarrow PV \ PP/PP \]

\[ \lambda x \lambda y \lambda z [x \ (y \ z)] \]

\[ \Rightarrow_M \]

\[ VP/PP/NP \rightarrow PV/NP \ PP/PP \]

\[ \lambda x \lambda y \lambda z \lambda w [x \ z \ (y \ w)] \]

(139) \[ VP \rightarrow VP \ ADV \]
     \[
     \Rightarrow_M
     \]

\[ VP/PP \rightarrow VP/PP \ ADV \]

\[ \lambda x \lambda y \lambda z [x \ (y \ z)] \]

\[ \Rightarrow_M \]

\[ VP/PP/NP \rightarrow VP/PP/NP \ ADV \]

\[ \lambda x \lambda y \lambda z \lambda w [x \ (z \ w)] \]

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 \textit{posted yesterday a copy of the newsletter to every student} to be
analysed as shown in Figure 14, with correct assignment of the same meaning as that of the canonical ordering:

(140) posted e e e e yesterday a copy ... to every ...

⇒ yesterday’ (posted’ a-copy-...’ to-every’)

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

```
Figure 14
```

(141)  \( ?I \text{ posted } e_i e_j \text{ yesterday [to every member in the area]} \) \(_i \text{ [a copy of the newsletter]} \) \(_j \)

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:

(142)  \[ \begin{align*}
\text{VP} & \rightarrow \text{PV PP} \\
& \Rightarrow M \lambda x \lambda y [x y] \\
\text{VP/NP} & \rightarrow \text{PV/NP PP} \\
& \Rightarrow R \lambda x \lambda y \lambda z [x z y] \\
\text{VP/NP/PP} & \rightarrow \text{PV/NP PP/PP} \lambda x \lambda y \lambda z \lambda w [x w (y z)]
\end{align*} \]

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

(143)  \[ \begin{align*}
X & \rightarrow X/Y Y \\
& \Rightarrow M \lambda x \lambda y \lambda z \lambda w [x w (y z)]
\end{align*} \]

The output of (143) will combine \( VP/PP/NP \), derived as in (140), with a prepositional phrase first. Thus, if it were necessary to avoid examples like (141) 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:

(144)  \( a \text{ paper which}_i a \text{ woman } e_j \text{ presented } e_i [\text{who has been studying computational linguistics for six years}]_j \)

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

(145)  \[ \begin{align*}
S & \rightarrow \text{NP VP} \\
& \Rightarrow R \lambda x \lambda y \lambda z \lambda w [x w (y z)] \\
\text{S/NP} & \rightarrow \text{NP VP/NP} \\
& \Rightarrow M \lambda x \lambda y \lambda z \lambda w [x w (y z)] \\
\text{S/NP/REL} & \rightarrow \text{NP/REL VP/NP}
\end{align*} \]

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:
(146) *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 (145) 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 (147) in which two subject modifiers are extracted, one to the left and one to the right.\(^{13}\)

(147) 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 (147) 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 (148), or else Middle Abstraction should apply to Rightward Filler Introduction.

(148) \[ N \rightarrow N \text{ REL} \]

\[ \Rightarrow \text{M} \]

\[ N/PP \rightarrow N/PP \text{ REL} \]

\[ \Rightarrow \text{R} \]

\[ N/PP/REL \rightarrow N/PP \text{ REL/REL} \]

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 (149) of Right Abstraction to Topic Introduction enables unacceptable double topicalisation as in (150).

(149) \[ S \rightarrow X \text{ S/X} \]

\[ \Rightarrow \text{R} \]

\[ S/Y \rightarrow X \text{ S/X/Y} \]

(150) *[[the book]_{NP} [[on the table]_{PP} [I put } e \ e_j \]_{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 (151), from Baltin (1982), quite acceptable, though I find (152a) and (152b), cited as acceptable by Iwakura (1980) and Gazdar, Klein,

\(^{13}\) The pied piping in (147) is addressed later.
Pullum, and Sag (1982), less good.

\begin{enumerate}
\item It’s obvious that Mary, he can’t stand
\item a. ?Harry said that Max, Joan would never be willing to marry
   b. *The inspector explained that each part he had examined very carefully
\end{enumerate}

The unacceptability of double topicalisation might be interpreted as indicating that metarule application to füller introduction rules should be prohibited. However I find Baltin’s (p17) example (153), which involves relativisation out of an embedded topicalisation, acceptable:

\begin{enumerate}
\item He’s a man to whom liberty we could never grant
\end{enumerate}

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 (154d) less acceptable.

\begin{enumerate}
\item a. [Many divorces] are tough (for men) to get over \( e_i \)
   b. [The exams] are easy (for students) to pass \( e_i \)
   c. [Some exams] are hard (for lecturers) to persuade students to take \( e_i \)
   d. ![Some theories] are hard (for students) to believe that anyone understands \( e_i \)
\end{enumerate}

I will avoid the complication of the optionality of the subject in the complement clause.\(^{14}\)

In the case that the subject is absent, the following rule enables (154b) to be analysed as shown in Figure 15 with the semantics shown in (156):

\footnote{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 \textit{SUBJ}. Borsley (1987) presents several cases where generalisation across verb phrases and sentences is required.}
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.\footnote{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.}

\begin{enumerate}
\item a violin which \textsubscript{\textit{i}} [the sonatas],\textsubscript{\textit{i}} are hard to play \textit{e} \textsubscript{\textit{j}} on \textit{e} \textsubscript{\textit{j}}
\item *some sonatas which \textsubscript{\textit{j}} [the violin],\textsubscript{\textit{\textit{j}}} is hard to play \textit{e} \textsubscript{\textit{j}} on \textit{e} \textsubscript{\textit{j}}
\end{enumerate}

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.\footnote{Also, Dick Oehrle (personal communication) has pointed out that in the following the contrast is not so sharp:}

\begin{enumerate}
\item ?some evidence which \textsubscript{\textit{i}} [the witnesses],\textsubscript{\textit{i}} are hard to show \textit{e} \textsubscript{\textit{j}} \textit{e} \textsubscript{\textit{i}}
\item some witnesses whom \textsubscript{\textit{j}} [the evidence],\textsubscript{\textit{j}} is hard to show \textit{e} \textsubscript{\textit{j}} \textit{e} \textsubscript{\textit{j}}
\end{enumerate}

2.1.2. Independent Extraction Plus Parasitic Extraction

Compound non-canonicality can involve parasitic phenomena. Consider (159), which I regard as grammatical.

\begin{equation}
?\text{a paper which }\textsubscript{\textit{i}} \text{ he showed } \textit{e} \textsubscript{\textit{j}} \textit{e} \textsubscript{\textit{i}} \text{ before submitting } \textit{e} \textsubscript{\textit{i}} [\text{a good number of his colleagues}],\textsubscript{\textit{j}}
\end{equation}

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 (160) as shown in Figure 16.

\begin{equation}
\begin{array}{c}
VP \rightarrow VP \ ADV \\
\Rightarrow_p^p \\
VP/NP, \rightarrow VP/NP, \ ADV/NP, \\
\Rightarrow_M^p \\
VP/NP, \rightarrow VP/NP, \ ADV/NP, \\
\end{array}
\end{equation}

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.\footnote{The relevant reading is one in which \textit{showed} is a ditransitive}

\begin{equation}
\text{a picture which }\textsubscript{\textit{j}} \text{ he showed } \textit{e} \textsubscript{\textit{j}} \textit{e} \textsubscript{\textit{i}} \text{ without forewarning } \textit{e} \textsubscript{\textit{j}} [\text{the unsuspecting members of the jury}],\textsubscript{\textit{j}}
\end{equation}

This can be obtained by applying the metarules the other way round:
(162) \[ VP \rightarrow VP \text{ ADV} \quad \Rightarrow>^M \]
\[ VP/NP_i \rightarrow VP/NP_i \text{ ADV} \quad \Rightarrow>_P \]
\[ VP/NP_i/NP_j \rightarrow VP/NP_i/NP_j \text{ ADV/NP}_j \]

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):

(163)  [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:

(164)  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 (164b) it should also be possible to right extract it, and (165) is indeed acceptable.
(165) [Mary sent $e_i$ to $e_j$ or John gave $e_i$ to every student]$_j$ [a full and detailed report]$_i$

The cases are facilitated by the rules in (166) and (167) and partial analyses are shown in Figure 17 and Figure 18.

(166) 
\[
\begin{align*}
VP & \rightarrow PV \ PP \\
VP/PP & \rightarrow PV \ PP/PP \\
VP/PP/NP & \rightarrow PV/NP \ PP/PP
\end{align*}
\]

(167) 
\[
\begin{align*}
VP & \rightarrow PV \ PP \\
VP/NP & \rightarrow PV/NP \ PP \\
VP/NP/PP & \rightarrow PV/NP \ PP/PP
\end{align*}
\]

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

(168) a. [Mary promised $e_i$ and I gave $e_i$]$_j$ Ralph$_i$ [a first edition copy of Syntactic Structures]$_j$

b. a book which$_j$ [Mary promised $e_i$ and I gave $e_i$]$_j$ Ralph$_i$

c. a student whom$_i$ [Mary promised $e_i$ and I gave $e_i$]$_j$ [a first edition copy of Syntactic Structures]$_j$

---

![Diagram](image)

Figure 17
(169) 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]$_i$

c. ![Diagram]

In these examples there is double right node raising out of sentences. In (170) there is
double right node raising out of verb phrases.

(170) 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 (171) there is double right node raising from a relative clause and in (172), from
an adverbial.

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

(172) 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).

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

(174)  [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 (175a) is semi-acceptable. The right node raising of two complements and an adjunct in (175b) seems fine.

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

(176)  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:

(177)  \[\text{VP} \rightarrow \text{VP} \text{ ADV} \implies \text{M} \]
        \[\text{VP/NP}_i \rightarrow \text{VP/NP}_i \text{ ADV} \implies \text{R} \]
        \[\text{VP/NP}_i/NP_j \rightarrow \text{VP/NP}_i \text{ ADV/NP}_j \]
The corresponding left extraction is at least as acceptable:

(178)  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:

(179)  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 (180); however the ‘crossed’ right extraction (181) that this would also
facilitate is not acceptable (cf. the general problem of preposition stranding and right extraction).

\[(180)\]
\[
\begin{align*}
\text{VP} & \rightarrow \text{VP ADV} \quad \Rightarrow_R \\
\text{VP/NP}_j & \rightarrow \text{VP ADV/NP}_j \quad \Rightarrow_M \\
\text{VP/NP}_j/\text{NP}_i & \rightarrow \text{VP/NP}_i \text{ ADV/NP}_j
\end{align*}
\]

\[(181)\]
*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.

\[(182)\]
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\(_i\) to talk e\(_i\) and urged e\(_i\) to shout e\(_j\)] [the vast majority of his anxious patients] \_j

\[(183)\]
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\(_i\) to leave e\(_i\) and urged e\(_i\) to oppose e\(_j\)] [a large number of formerly active supporters] \_j

\[(184)\]
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

\[(185)\]
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 (185a) is derived thus:
(186) \[ \begin{align*}
    \text{VP} & \rightarrow \text{PV} \text{ PP} \quad \Rightarrow_M \\
    \text{VP/NP}_i & \rightarrow \text{PV/NP}_i \text{ PP} \quad \Rightarrow_R \\
    \text{VP/NP}_i/\text{NP}_j & \rightarrow \text{PV/NP}_i \text{ PP/NP}_j
\end{align*} \]

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 (187) a complement is extracted; in (188) an adjunct is.

(187) 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$_i$]

(188) ?the coach [in which]$_i$ most$_j$ [e$_j$ girls arrived e$_i$ and e$_j$ boys left e$_i$]

An example like (187a) will be generated via (189) as illustrated in Figure 19.

(189) \[ \begin{align*}
    \text{S} & \rightarrow \text{NP} \text{ VP} \quad \Rightarrow_R \\
    \text{S/PP} & \rightarrow \text{NP} \text{ VP/PP} \quad \Rightarrow_L \\
    \text{S/PPDET} & \rightarrow \text{NPDET} \text{ VP/PP}
\end{align*} \]

---

![Figure 19](image-url)
While a verb is left node raised, it is possible to also extract from the left or right complements of 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:

(190) a. a topic [about which] I lent [e_j John a book e_i and e_j Mary a paper e_j]

b. I lent [e_i John a book e_j and e_i Mary a paper e_j] [about subjacency]

(191) a motion which a survey revealed [e_j Southerners to support e_i and e_j Northerners to oppose e_i]

(192) the group which he wanted [e_i Mary to join e_j and e_j John to leave e_i]

(193) [That John left] he wants [e_i Mary to believe e_i and e_j Sue to disbelieve e_j]

(194) the countries which he smuggled [e_j alcohol to e_i and e_j drugs from e_i]

Example (190), for instance, will be generated via the rule derived in (195).

(195) 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:

(196) I met [e_i John before e_j and e_i Mary during e_j] [the second session]

Secondly, there is extraction from the left-hand element while left node raising. The examples in (197) are acceptable, but those in (198) are less good.

(197) a. I lent [e_i a book e_j to John and e_i a paper e_j to Mary] [about subjacency]

b. a topic [about which] I lent [e_j a paper e_i to John and e_j a book e_i to Mary]
(198)  a.  ?I gave, [e_i the men e_j the books and e_j the women e_j the records] [who seemed to me to most deserve them]j

   b.  ?He wanted, [e_i the men e_j to be imprisoned and e_j the women e_j to be detained] [who had publicly opposed his methods of reform]j

   c.  *I notified, [e_i the students e_j that I’d attend and e_j the staff e_j that I’d speak] [whom I’d informed earlier that I’d be out of the country]j

(197) will be generated via the rule derived in (199).

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

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:

(200)  a.  the people [to whom], i we sent, [e_i the report e_i on Monday and e_j the newsletter e_i on Tuesday]

   b.  the newsletter which, i we sent, [e_j John e_i on Monday and e_j Mary e_i on Tuesday]

   c.  [That he was Italian], i he told, [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:

(201)     a town which, i I bought, [e_i 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 20.

In the following a gap in the adverbial is extracted parasitically with the first complement of the left node raised verb; see Figure 21:

(202)     the subjects who, i we gave, [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 22:
(203) 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]

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

(205) a. *some friends_{i} e_{j} who play golf [of John]_{j} and e_{i} e_{k} who swim [of Mary]_{k}

b. *the beliefs_{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_{j} on Monday [the directors]_{j} and e_{i} e_{k} on Tuesday [the union leaders]_{k}

d. *We searched_{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_{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 23 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:

(206)  
  a.  a topic [on which] I included \( e_j \) in the package [several seminal papers \( e_i \)]
  b.  a celebrity whom several news stories appeared [on \( e_j \)].

Cases like these require metarule application to the Rightward Filler Introduction rule as shown in (207), in order that the gap information can be transmitted out of the right extracted element as in (208).

(207)  
  \[ X \rightarrow X/Y \; Y \]
  \[ \Rightarrow_R \]
  \[ X/Z \rightarrow X/Y \; Y/Z \]

(208)  
  \([\{several news stories \, e\} \, appeared \} \, S/PP \rightarrow \, 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.

---

![Diagram](image.png)

Figure 23
Consider also the following: 18

(209)  I [see each \(e_i\), \([e_j\) 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 24) 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 \(VP/N(\neg VP/N)\) which can be expanded as the empty string.

(210)  \[VP \rightarrow VP/N N \implies L\]
\[VP(\neg VP/N) \rightarrow VP/N(\neg VP/N) N\]

(211)  would require the corresponding treatments.

(211)  a.  ?He [said that \(e_i\), \([e_j\) he was Italian,\(i\) on Monday and \(e_j\) he was Spanish,\(i\) on Tuesday

b.  We [looked for \(e_i\), \([e_j\) John,\(i\) on Monday and \(e_j\) Mary,\(i\) on Tuesday]

Dowty’s (1988) example (212) is another case in which the value of a slash must bear a slash specification

---

18 The example cannot be regarded as double left node raising (i) as follows.

(i)  I see, each, \([e_i\) \(e_j\) boy on Monday and \(e_j\) \(e_j\) girl on Tuesday]

Such an analysis would require inheritance of a leftward slash \(\neg DET\) 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.
(212) [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 PT) and VP PT respectively, in another they would be S/PP NP and S(S/PP NP) respectively.

3. Discussion: 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 (213) seems to fare better than the null rule gap introduction (214) as regards constraints like the left branch condition and coordinate structure constraint.

(213) a. X → Y Z =>
    X/Z → Y

    b. X → Y Z =>
    X/Y → Z

(214) a. X/X → e

    b. XXX → 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:

(215) X → Y Z =>
    X/Z → Y Z/Z

Z/Z in (215) can be expanded as the empty string so that the material dominated by Y is analysed as X/Y. This is the effect of (213a). 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, \( S/\text{NP} \) means \( S/[\text{SLASH} \text{ NP}] \), 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 \( S/\text{PP}/\text{NP} \) means \( S/[\text{SLASH} \langle \text{NP},\text{PP} \rangle] \). 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 (216) with various features including \( \text{SUBCAT} \) and \( \text{SLASH} \).

\[
(216) \quad \text{X[SUBCAT \ldots, SLASH \ldots, \ldots]}
\]

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.\textsuperscript{19} 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

\footnotesize
\begin{figure}[h]
\centering
\begin{tikzpicture}
    \node {S[TNS PRES]}
    child {node {NP[NUM SG]}
        child {node {John}}}
    child {node {VP[TNS PRES, AGR [NUM SG]]}
        child {node {TV[TNS PRES, AGR [NUM SG]]
            child {node {loves}}
            child {node {NP[NUM SG]}
                child {node {Mary}}}}}};
\end{tikzpicture}
\caption{Features in Phrase Structure Grammar}
\end{figure}

\begin{figure}[h]
\centering
\begin{tikzpicture}
    \node {S[TNS PRES]}
    child {node {NP[NUM SG]}
        child {node {John}}}
    child {node {S[TNS PRES] NP[NUM SG]
        child {node {NP[NUM SG]}
            child {node {loves}}
            child {node {S[TNS PRES] NP[NUM SG]/NP
                child {node {NP}}
                child {node {Mary}}}}}}};
\end{tikzpicture}
\caption{Features in Categorial Grammar}
\end{figure}

\textsuperscript{19}Engdahl (1986, pp22-24, 132-37) also argues against the imposition of an upper bound.
idea of degrees of grammaticality. In this way 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 directiona
categorial grammar.

Further, the filler-introduction rules (217) have the semantics of the application primitives of categorial grammar.

(217) a. \( X \rightarrow X/Y \ Y \)

b. \( X \rightarrow Y \ X \ Y \)

If we encode complements on slashes, as in the CG of Chapter I, (217) 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'.

As a final additional motivation for the new approach, I want to consider a couple of points relating to 'feature percolation'. Within GPSG, complex feature percolation principles, the 'head feature convention' and 'control agreement principle', were invoked to characterise the distribution of head features such as TNS and agreement features such as NUM in Figure 25. However by placing agreement features on argument categories, some of the need for such conventions as the control agreement principle is removed: category

---

20 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 'I' were the gap argument operator, we could have a gap introduction rule mapping e.g. SNP/INP to SNP/INP.

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 XXX, 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 theoretically possible, empty nodes in a categorial grammar seem, empirically both unnecessary, and undesirable.
matching implements the requirement. And if slash is interpreted as an operator, placing head features on result categories achieves head feature ‘percolation’ without a head feature convention, as shown in (217). In the light of these many motivations for a move from phrase structure grammar towards categorial grammar, the next chapter considers non-canonically from the latter perspective.
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 combinatorics 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-Constiuents’

Sections 1.1.1 and 1.1.2 address right node raising and left node raising respectively.

1.1.1. Right Node Raising

Consider the following:

(1) I [read e₁ and will reference eₙ] [your paper]₁

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\(\overline{VP}\)) 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)

\[
X/Y: x + Y/Z: y \Rightarrow X/Z: B x y
\]

(3) 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 \( \lambda \)-terms, the composition of \( x \) and \( y \) is \( \lambda z [x (y z)] \). A combinatory logic is an applicative system like the \( \lambda \)-calculus, but one in which functional abstraction is not expressed by \( \lambda \)-binding of variables, but by various combinator primitives of which \( B \) is one defined by

(4) \[(B x) y z \equiv x(y z)\]

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

(5) \[
\begin{align*}
(((\lambda (B) B) x) y) z & = w \\
((\lambda (B x)) y) z & = w \\
(\lambda ((y z) w) & = x
\end{align*}
\]

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) \[I x \equiv x\]

(7) \[C x y z \equiv x z y\]

Example (1) has the following analysis:

(8) \[
\begin{array}{llllllllll}
NP & S \NP/NP & S \NP/ (S \NP) & S \NP/NP & NP/N & N & \longleftarrow B & \longleftarrow NP \\
S \NP/NP & \longleftarrow C r d & S \NP/NP & \longleftarrow S \NP & \longleftarrow S
\end{array}
\]
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) \[
\begin{array}{c}
\text{will reference your paper} \\
\text{VP/VP VP/NP NP} \\
\text{VP/NP} \rightarrow \text{B} \\
\text{VP} \rightarrow \text{B}
\end{array}
\]

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) \[
\begin{array}{c}
\text{I [desire } e_i \text{ and want to search for } e_i] \text{[the meaning of life]}_i
\end{array}
\]

(11) \[
\begin{array}{c}
\text{want to search for} \\
\text{VP/VP VP/VP VP/VP VP/NP} \\
\text{VP/VP} \rightarrow \text{B} \\
\text{VP/NP} \rightarrow \text{B}
\end{array}
\]

However other instances are blocked:

(12) \[
\begin{array}{c}
\text{[I liked but Mary disliked] the second play}
\end{array}
\]

(13) \[
\begin{array}{c}
\text{I liked} \\
\text{NP S/NP/NP}
\end{array}
\]

Such cases can be captured with the help of a rule called Forward Type-Raising (see Steed-
man 1985, 1987a, and Dowty 1988, and references therein):\(^1\)

\[
(14) \quad \text{Forward Type-Raising} \ (\Rightarrow T) \\
\text{X: } x \Rightarrow Y/(\forall X) \cdot T \ x
\]

\[
(15) \quad T \ x \ y \equiv y \ x
\]

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

\[
(16) \quad \text{[I liked but Mary disliked] the second play}
\]

\[
(17) \quad \begin{array}{c}
\text{I} \\
\text{liked} \\
\text{NP} \\
\text{SNP/NP} \\
\Rightarrow T \\
S/(S/NP) \\
\Rightarrow B \\
S/NP
\end{array}
\]

In each conjunct, the subject \(NP\) is Forward Type-Raised to \(S/(SNP)\) and then Forward Composed with the transitive verb, \(SNP/NP\), 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 \(\lambda x(z \ john)\) of type \(e \Rightarrow t\) 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 \(REL\).
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) Mary said that Sue adores cheese soufflé

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

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

(21) 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

---

3A non-directional version of (20) is called *backward partial combination* in Steedman (1985, p533).
type-raising, in accord with the symmetry of left node raising and right node raising for which the respective rules account.

Consider first complement-adjunct left node raising:

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

A suitable category for the conjuncts is \(VP\langle VP/NP\rangle\): they form verb phrases once they apply to transitive verbs on their left. The adverbials are \(VP\backslash VP\), and the \(NP\) objects can be Backward Type-Raised to \(VP\langle VP/NP\rangle\), where \(Y\) in (21) is instantiated to \(VP\). Then an object, \(VP\langle VP/NP\rangle\), and an adverbial, \(VP\backslash VP\), can combine by Backward Composition to form a constituent of category \(VP\langle VP/NP\rangle\), as desired:

(23) \[
\begin{array}{c}
\text{John} \\
\text{on Monday} \\
\text{NP} \\
\text{VP\backslash VP} \\
\text{VP\langle VP/NP\rangle} \\
\text{VP\backslash (VP/NP)} \\
\end{array}
\]

The rest of the analysis is straightforward.

Next, consider the complement-complement case:

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

The first complement, \(NP\), can be backward type-raised to \(VP\langle PP\langle VP/PP/NP\rangle\rangle\) (where \(PP\) abbreviates \(SWP\langle SWP\rangle\)): 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 \(VP\langle VP/PP\rangle\), and now these can backward compose to form conjuncts of category \(VP\langle VP/PP/NP\rangle\):

(25) \[
\begin{array}{c}
a \text{ book} \\
\text{NP} \\
\text{PP} \\
\text{VP/PP\langle VP/PP/NP\rangle} \\
\text{VP\langle VP/PP\rangle} \\
\text{VP\langle VP/PP/NP\rangle} \\
\end{array}
\]

Left node raising of a determiner proceeds thus:
(26) Each boy dances and girl sings

NP/N N S\NP N S\NP

NP\(NP/N) \quad \text{<T}

NP\(NP/N) \quad \text{<T}

S\(NP/N) \quad \text{<B}

S\(NP/N) \quad \text{<B}

S\(NP/N) \quad \text{Cr d}

S\(NP/N) \quad \text{<T}

S

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

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

(28) see each boy on Monday ...

VP/N VP\(VP/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 $Y/(YX)$ still only allows $X$ and $YX$ to combine in their original order $X+YX$ 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:

(29) \[ \text{I met } e_i \text{ yesterday [an old school friend who has become a respected film critic]} \]

We seem to need to combine \( \text{met}_{\text{VP/NP}} \) and \( \text{yesterday}_{\text{VP/VP}} \) to form an expression of category \( \text{VP/NP} \) so that the result can apply forward to a rightwardly displaced heavy noun phrase. The rule (30) achieves this, so that (29) receives the analysis shown in (31).

(30) Mixed Backward Composition \( (<B_x) \)
\[
Y/Z: y + X/Y: x \Rightarrow X/Z: B x y
\]

(31)
\[
\begin{array}{cccc}
\text{met} & \text{yesterday} & \text{an old} & \ldots \\
\hline
\text{VP/NP} & \text{VP/VP} & \text{NP} & \hline
\hline
\text{VP/NP} & \text{VP/VP} & B_x & \text{NP} \\
\hline
\text{VP/NP} & \text{VP/VP} & \text{NP} & \hline
\text{VP} & \hline
\end{array}
\]

Mixed Backward Composition appears in Moortgat (1988), Morrill (1987a), and Steedman (1987a). The forward counterpart is (32).

(32) Mixed Forward Composition \( (>B_x) \)
\[
X/Y: x + Y/Z: y \Rightarrow X/Z: B x y
\]

The use of (30) as opposed to (32) corresponds to the phrase structure grammar inheritance of leftward slashes from left-hand but not right-hand daughters. The rule (32) must be strictly prohibited in English, otherwise orderings like (33) are obtained.

(33)
\[
\begin{array}{cccc}
*\text{John that left} & \ldots & \\
\hline
\text{NP} & \text{SP/S S/NP} & \hline
\text{SP/NP} & B_x & \text{NP} & \hline
SP & \hline
\end{array}
\]

(34) \text{I met } e_i \text{ John yesterday}

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

(35) \text{I gave } e_i \text{ to John [a large red box]}_i \text{ "}

In a case like this the PP complement can be backward type-raised to \( \text{VP}(\text{VP/PP}) \), 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 (36).

(36)  
\[
\begin{array}{cccccccc}
\text{gave} & \text{to} & \text{John} & \text{a} & \text{large} & \text{red} & \text{box} \\
\hline
VP/PP/NP & PP & NP \\
\hline
\hline
\hline
\hline
\text{<T} \\
\text{VP\,(VP/PP)} \\
\hline
\hline
\hline
\hline
\text{<B_x} \\
\text{VP/NP} \\
\hline
\hline
\hline
\hline
\rightarrow \\
\text{VP}
\end{array}
\]

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:

(37)  
\[
\begin{array}{cccccccc}
\text{I} & \text{rang} & \text{up} & \text{John} \\
\hline
\hline
\hline
\hline
\hline
\hline
\hline
\hline
\hline
\hline
\hline
\hline
\hline
\hline
\hline
\text{NP} & \text{S\,NP/PP/NP} & \text{PP} & \text{NP} \\
\hline
\hline
\hline
\text{<T} \\
\text{S\,NP\,(S\,NP/PP)} \\
\hline
\hline
\hline
\text{<B_x} \\
\text{S\,NP/NP} \\
\hline
\hline
\hline
\text{S\,NP} \\
\hline
\hline
\text{<} \\
\text{S}
\end{array}
\]

Consider next right extraposition:

(38)  
\[
\begin{array}{cccccccc}
\text{A man} & \text{e_i} & \text{arrived [who spoke Russian]} & \text{i} \\
\hline
\hline
\hline
\hline
\hline
\hline
\hline
\hline
\hline
\hline
\hline
\hline
\hline
\hline
\hline
\text{A noun like man_\text{N} can be forward type-raised to N/(NN) -- a noun type-raised over an adnominal such as a relative clause. A determiner a_{NP/N} can forward compose with this to give a man_{NP/(NN)} and the analysis can proceed as shown in (39). This is the account of right extraposition presented in Morrill (1987a); Moortgat (1988) gives the corresponding treatment for right extraposition in Dutch.}
\end{array}
\]
(39) a man arrived who spoke Russian

NP/N N S/NP N/N

----- >T
N/ (N/N)

----------------- >B
NP/ (N/N)

----------------- <B_x
S/ (N/N)

----------------- >
S

The meaning of type-raised \textit{man} is $T \text{ man}'$ (i.e. $\lambda x (x \text{ man}')$) and the meaning of the expression as a whole, which is the same as that of its canonical counterpart, is derived thus:

(40) a man

$\Rightarrow$ B a’ ($T \text{ man}'$)

a man arrived

$\Rightarrow$ B arrived’ (B a’ ($T \text{ man}'$))

a man arrived who speaks Russian

$\Rightarrow$

B arrived’ (B a’ ($T \text{ man}'$)) who-speaks-Russian’

(41) B arrived’ (B a’ ($T \text{ man}'$)) who-speaks-Russian’ =

arrived’ (B a’ ($T \text{ man}'$) who-speaks-Russian’) =

arrived’ (a’ ($T \text{ man}'$ 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 \{NN, SP, NP, SNP(SNP), 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).\footnote{In Ades and Steedman subjects were pre-assigned a higher type so that only composition was required.}
In (42) 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 (43).

(43)  [The beginning], John thinks that Mary likes _1

*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. 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 (43) 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:

(44)  \(\text{Topic Introduction (t)}\)

\[X: x + S/X: y \Rightarrow S: y x\]

\[X \in \{\text{NP, SP, N/N, } N\N, \text{ S/NP}/(S/NP)\}\]

Note that this will assign a topicalised sentence the same meaning as its canonical

\(^5\)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/NP}]* and *John think[s that Mary likes]_{S/NP}*.\}
counterpart. As mentioned earlier, the markedness of the intonation accompanying topicalisation 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 (44) 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 (44) 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:

(45) On Monday John arrived

\[
\begin{align*}
S \backslash NP \backslash (S \backslash NP) & \quad NP \quad S \backslash NP \\
& \quad \rightarrow T \quad \rightarrow T \\
S / (S \backslash NP) & \quad S \backslash NP / (S \backslash NP) (S \backslash NP) \\
& \quad \rightarrow B \\
S / (S \backslash NP) (S \backslash NP) & \\
& \quad \rightarrow B \\
& \quad S
\end{align*}
\]

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,VP} will combine with met_{VP,NP} by mixed backward composition to give VP/NP, from which derivation can proceed as above to yield (46).

(46) the man who_{i} I met \epsilon_i yesterday

Similarly (47) has the derivation (48).

(47) the book which_{i} I gave \epsilon_i to John
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:

(49) a. *the man who I believe that e₁ left
    b. the man who I believe e₁ left

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

(50) \textit{that left}

The combination in (50) 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 (49b):

(51) \textit{believe left}

The solution adopted here is that \textit{believe} is of the category \textit{SNP/VP/NP} whereby the subor-
dinate 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 com-
plement on its own. However, assuming that \textit{believe} is of this category, it is not necessary
to assume that it is also of category \textit{SNP/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
To-infinitival complement verbs will have the same categorization, facilitating extraction like the following:

(53) the people who\textsubscript{1} I wanted \textsubscript{e} to go

According to the above account subject gaps are like object gaps in that e.g. \textit{I voted for} and \textit{I believe won} are both \textit{VP/NP}. As Steedman (1987a, p424) points out, this is consistent with the fact that the same relative pronouns appear with both:

(54) a. the man who\textsubscript{m} \textit{I voted for} \textsubscript{e}

b. the man who\textsubscript{m} \textit{I believe} \textsubscript{e} won

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

(55) the man who\textsubscript{m} \textit{I [voted for} \textsubscript{e} and \textit{believe} \textsubscript{e} won]

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

(56) *who swims and I know

This follows because the conjuncts have distinct categories, \textit{SNP} and \textit{S/NP}.

Note that mixed backward composition is needed to achieve across-the-board extraction from a non-right-peripheral position; (57a) has the derivation shown in (58) which illustrates how the conjuncts can be analysed as the transitive verb category as required in (57b) and (57c).

(57) a. I [gave \textsubscript{e} to members and sent \textsubscript{e} to affiliates] \textit{[long and detailed reports]}\textsubscript{i}

b. I [read \textsubscript{e} and sent \textsubscript{e} to affiliates] \textit{[long and detailed reports]}\textsubscript{i}

c. I [gave \textsubscript{e} to members and published \textsubscript{e}]. \textit{[long and detailed reports]}\textsubscript{i}
(58) gave to members and sent to affiliates

\[
\begin{align*}
S\text{NP}/PP/NP & \quad PP \quad S\text{NP}/PP/NP & \quad PP \\
S\text{NP}/(S\text{NP}/PP) & \quad <T \quad S\text{NP}/(S\text{NP}/PP) & \quad <T \\
S\text{NP}/NP & \quad <B_x \quad S\text{NP}/NP & \quad <B_x \\
\text{NP}/NP & \quad \text{Crd} \\
\end{align*}
\]

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:

(59) *the man who, I saw \[e_i\] and a picture of \(e_j\]

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 (60) are forbidden because the conjuncts are of unlike category.

(60) a. *the people who, [John likes \(e_i\) and Sue likes Mary] 
    b. *the people who, [John likes Mary and Sue likes \(e_j\)]

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 a 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 (61) crosses boundaries ‘governed’ by lexical categories, and so cannot be so straightforwardly achieved by manipulation of lexical categories.

(61) 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 NP constraint or A-over-A constraint. Nor does it predict that extraction from a subject is impossible:
Extraction from adjuncts is made possible by type-raising over them as shown in (63); 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.

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 (64) can be managed straightforwardly by lexical assignment of categories NNN/(S/NP)/N and NNN/(S/NP)/N to whose.

(64)  a. the man whose book arrived
       b. the man whose book I read

Cases like (65) are more challenging.

(65)  a. the people whom we looked for ei
       b. the people [for whom] we looked ei

The pied piped relative clause in (65b) is synonymous with the relative clause in (65a). Example (65a) has the following analysis:
We looked for is assigned the meaning of the sentence abstracted over the meaning of the preposition’s object, i.e. $\lambda x[\text{looked}(\text{for x} \text{ we})]$, so that the relative clause meaning is $\text{whom}'(\lambda x[\text{looked}(\text{for x} \text{ we})])$.\(^8\) The semantics of $\text{who}(m)$ will be $\lambda x\lambda y\lambda z[x \; z \land y \; z]$ but I will continue to write $\text{who}(m)$. Szabolcsi (1987, section 4) characterises pied piping by lexically assigning a relative pronoun of ordinary category $\NNN/(S/NP)$, to an additional pied piper category $\NNN/(S/X)/(X/NP)$\(^9\). Morrill (1987b) presents a binary syntactic schema for pied piping.\(^10\) 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 (67) which combines the pied piped material $X/NP$ and a relative pronoun $\NNN/(S/NP)$.

(67) \textit{Non-Subject Pied Piping (pₐ)}

$X/NP: x + \NNN/(S/NP): y \Rightarrow \NNN/(S/X): \lambda z[y (\lambda w[z \; (x \; w)])]

X \in \{PP, NP\}$

'X' may be $VP \Rightarrow VP, \NNN$, and possibly also $NP$:

(68) a. the people [for whom]$_{\text{NNN}/(S/(VP\Rightarrow VP))}$, we looked $e_i$
    b. the subject [about which]$_{\text{NNN}/(S/(NNN))}$, we had an argument $e_i$.
    c. ?the man [a picture of whom]$_{\text{NNN}/(S/(NP))}$, John purchased $e_i$.

In (67), $x$ 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 (69) $\text{for whom}$ has meaning $\lambda z[\text{whom}'(\lambda w[z \; (\text{for w})])]$. We looked has the sentence meaning abstracted over

\(^8\)In the interests of ease of comprehension, the $\lambda$-calculus, rather than combinatory logic, will be used in discussion of semantics in this section.

\(^9\)Szabolcsi’s proposals are made primarily with reference to reflexives and reflexive pied piping ($\text{Mary likes herself, Mary talks about herself}$).

\(^10\)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.
the prepositional phrase, i.e. \( \lambda v[\text{looked } v \text{ we'}] \), so that the overall meaning is 
\( \text{whom'}(\lambda x[\text{looked } (\text{for } x \text{ we'})]) \), the same as the non-pied piped relative clause.

(69) \[
\begin{array}{llllll}
\text{for} & \text{whom} & \text{we} & \text{looked} \\
\text{PP/NP} & \text{NN} & (\text{S/NP}) & \text{NP} & \text{S/NP/PP} \\
\text{------------------} & & & & \text{T} \\
\text{NN} & (\text{S/PP}) & \text{S/} & (\text{S/NP}) & \text{------------------} & \text{B} \\
\text{------------} & & & & & \text{------------------} \\
\text{NN} & & & & & \text{------------------} \\
\end{array}
\]

Consider next

(70)  
\begin{enumerate}
\item the man whom_i a picture of e_i exists
\item the man a picture of whom exists
\end{enumerate}

Mixed composition enables (70a) to be derived as shown in (72). However the existing schema does not capture (70b) because this requires the fronted element to combine with a verb phrase \( SNP \) rather than a sentence-with-gap \( S/NP \). For subject pied piping I propose:

(71) \( \text{Subject Pied Piping (p_s)} \)
\[
\text{NP/NP: } x + \text{NN}/(\text{S/NP}): y \Rightarrow \text{NN}/(\text{S/NP}): \lambda z[y (\lambda w[z (x w)])] 
\]

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 (72) and (73) are the same.

(72) \[
\begin{array}{llllll}
\text{whom} & \text{a picture of} & \text{exists} \\
\text{------------------} & \text{------------------} & \text{------------------} \\
\text{NN} & (\text{S/NP}) & \text{NP/N} & \text{NN/PP} & \text{PP/NP} & \text{S/NP} \\
\text{---------------} & \text{------------------} & \text{------------------} & \text{------------------} & \text{------------------} & \text{------------------} \\
\text{NN} & (\text{S/NP}) & \text{NP/N} & \text{---------------} & \text{------------------} & \text{------------------} \\
\text{------------------} & \text{------------------} & \text{------------------} & \text{------------------} \\
\end{array}
\]

(73) \[
\begin{array}{llllll}
\text{a picture of} & \text{whom} & \text{exists} \\
\text{------------------} & \text{------------------} & \text{------------------} \\
\text{NP/NP} & \text{NN} & (\text{S/NP}) & \text{S/NP} \\
\text{---------------} & \text{------------------} & \text{------------------} \\
\text{NN} & (\text{S/NP}) & \text{---------------} & \text{------------------} \\
\text{------------------} & \text{------------------} & \text{------------------} \\
\end{array}
\]
The current grammar does not generate *that*-less relatives such as (74).

(74) 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 unboundedly 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 $NN$. Under the assumption of like-category coordination, the acceptability of (75) indicates that the sentence-with-gap itself belongs to the relative clause category, indicating the unary mapping of $S/NP$: $x$ to $NN$: whom $x$.

(75) 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:

(76) 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 $NN$ with semantics as before, and because it does not have the relative pronoun category, *that* cannot pied pipe. Gabrial Bes (personal communication) has pointed out that the same device appears to capture *for*-relatives such as (77) where again the relative clause consists of a complementized sentence with an NP gap.

(77) the man for you to meet

Overall then we have the following:

(78) \textit{Object That and That-Less Relatives ($r_o$)}
\begin{align*}
X/NP: x & \Rightarrow NN: \lambda y \lambda z [x \, z \, & y \, z] \\
X & \in \{ S, SP \}
\end{align*}

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 (78) enables (the man) John met and (the man) that John met to be analysed shown in (79) and (80) respectively. The normal relative clause
semantics is assigned.

\[
\begin{align*}
(79) & \quad \text{John met} \\
& \quad \text{S/NP} \\
& \quad \text{r}_o \\
& \quad \text{N/N}
\end{align*}
\]

\[
\begin{align*}
(80) & \quad \text{that John met} \\
& \quad \text{SP/S NP} \\
& \quad \text{B} \\
& \quad \text{NP} \\
& \quad \text{r}_o \\
& \quad \text{N/N}
\end{align*}
\]

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:

\[
\begin{align*}
(81) & \quad \text{a. Forward Substitution (S)} \\
& \quad X/Y/Z: x + Y/Z: y \Rightarrow X/Z: S x y \\
& \quad \text{b. Backward Substitution (S)} \\
& \quad Y/Z: y + X/Y/Z: x \Rightarrow X/Z: S x y
\end{align*}
\]

\[
(82) \quad S x y z = x z (y z)
\]

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:

\[
(83) \quad \text{the student who I warned e}_i \text{ that Mary envied e}_i
\]

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:

\[(84) \quad \text{who} \quad \text{I} \quad \text{warned} \quad \text{that Mary envied} \]

\[
\begin{array}{c}
N/N/(S/NP) \quad \text{NP} \quad S/NP/SP/NP \quad SP/NP \\
\quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \ qua
\[
\text{Forward Application} \\
X/Y: x + Y: y \Rightarrow X: x y \\
\text{Backward Application} \\
Y: y + X/Y: x \Rightarrow X: x y \\
\text{Forward Composition} \\
X/Y: x + Y/Z: y \Rightarrow X/Z: B x y \\
\text{Backward Composition} \\
Y/Z: y + X/Y: x \Rightarrow X/Z: B x y \\
\text{Mixed Backward Composition} \\
Y/Z: y + X/Y: x \Rightarrow X/Z: B x y \\
Z \in \{NNN, SP, NP, S\NP(S\NP, N/N)\} \\
\text{Forward Substitution} \\
X/Y/NP: x + Y/NP: y \Rightarrow X/NP: S x y \\
\text{Backward Substitution} \\
Y/NP: y + X/Y/NP: x \Rightarrow X/NP: S x y \\
\text{Forward Type-Raising} \\
X: x \Rightarrow Y/(Y/X): T x \\
\text{Backward Type-Raising} \\
X: x \Rightarrow Y/(Y/X): T x \\
\text{Topic Introduction} \\
X: x + S X: y \Rightarrow S: y x \\
X \in \{NP, SP, N/N, N/N, S\NP(S\NP)\} \\
\text{Non-Subject Pied Piping} \\
X/NP: x + NNN/(S/NP): y \Rightarrow NNN/(S/X): B y (C B x) \\
X \in \{PP, NP\} \\
\text{Subject Pied Piping} \\
NP/NP: x + NNN/(S/NP): y \Rightarrow NNN/(S/NP): B y (C B x) \\
\text{Object That and That-Less Relatives} \\
X/NP: x \Rightarrow NNN: S (B & x) y \\
X \in \{S, SP\} \\
\]

Figure 1: Augmentations to CG

In the existing rules, only one slash is ever inherited by the rules of combination. Consider the following:
(88) a. I posted \( e_i e_j \) yesterday [a copy of the newsletter] \( i \) [to every student] \( j \)  
b. the people [to whom] \( i \) I posted \( e_i e_j \) yesterday [copies of the newsletter] \( i \)  
c. the newsletter which \( i \) 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:

\[(89)\text{ posted yesterday}\]

\[\text{VP/PP/NP} \quad \text{VP/VP} \quad \text{VP/PP/NP}\]

Similarly:

(90) 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] \( i \) [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:

(91) Mary sent

\[\text{S/VP} \quad \text{VP/PP/NP} \quad \text{S/PP/NP}\]

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:

(92) a. Generalised Forward Composition (>\(B^n\))
\[X/Y: x + Y/Z...: y => X/Z...: B^n x y\]
b. Generalised Mixed Backward Composition (<\(B^n\))
\[Y/Z...: y + X/Y: x => X/Z...: B^n x y\]

(93) \(B^0 = B\)
\(B^n = B \cdot B^{n-1} B\)

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:
(94) **Principle of Directional Consistency** (PDC)
All syntactic combinatory rules must be consistent with the directionality of the principal function. [Steedman (1987a, p407)]

(95) **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.11

(96) 
  a. \( *X \backslash Y + Y \Rightarrow X \)  
  b. \( *X \backslash Y + Y/\backslash Z \Rightarrow X/\backslash Z \)

The PDI serves to help interpret the ellipses in (93). The corresponding arguments on mother and daughter must share directionality, thus while (97a) is included in the schema of (93a), (97b) is excluded.

(97) 
  a. \( X/\backslash Y + Y/\backslash Z/W \Rightarrow X/\backslash Z/W \)  
  b. \( *X/\backslash Y + Y/\backslash Z/W \Rightarrow X/\backslash Z/W \)

Note however that while the PDI does not exclude the instance (98) 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 (98) of \( > \B^2 \).

(98) \( X/\backslash Y + Y/\backslash Z/W \Rightarrow X/\backslash Z/W \)

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 (98). However it causes overgeneration. For example \( X/\backslash Z/W \) in (98) matches the form of a direct object Backward Type-Raised over a prepositional ditransitive verb, thus the verb is erroneously allowed to 'move left':

---

11 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.
(99) * gave John a book to Fred

So the ellipsis generalisation seems to be too strong in that it overgenerates.

But as well as being too strong, the above generalisations appear 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:

(100) a paper which_j a woman e_j presented e_j [who has been studying computational linguistics for six years].j

Here there must be inheritance from both the subject and the predicate verb phrase:

(101) a woman presented

In (102) there must be inheritance from both the verb and the adverbial.

(102) 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:
And in (104) to (107) there is inheritance of the first complement from the verb, and from the second complement.

(104) a. the people whom he [persuaded e₁ to talk e₂ and urged e₁ to shout e₂] [about their childhood oppressions]
   b. the things [about which] he [persuaded e₂ to talk e₁ and urged e₂ to shout e₁] [the vast majority of his anxious patients]

(105) a. the people who he [persuaded e₁ to leave e₂ and urged e₁ to oppose e₂] [the political party which they had always supported]
   b. an institution which he [persuaded e₂ to leave e₁ and urged e₂ to oppose e₁] [a large number of formerly active supporters]

(106) a. a scholar who I know e₁ to have argued e₂ and suspect e₁ to believe e₂] [that binding theory has explanatory power]
   b. I know e₁ to have argued e₂ and suspect e₁ to believe e₂] [that binding theory has explanatory power] [several of the workers in that research group]

(107) a. the tokens which he [handed e₁ to e₂ and took e₁ from e₂] [each acolyte]
   b. the acolytes whom he [handed e₂ to e₁ and took e₂ from e₁] [small tokens of remembrance]

These require something like (108):

(108) \( \text{VP/CMP_2/CMP_1 CMP_2/X} \)
    \( \text{VP/CMP_1/X} \)

Such cases indicate a further generalisation along the lines of (109) where there can be (possibly non-parasitic) inheritance from both daughters.

(109) a. \( Y... + X Y... \Rightarrow X... \)
   b. \( X/Y... + Y... \Rightarrow X... \)

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 (110) and (111) from Section 2.1.2 of Chapter II.

(110) a paper which he showed $e_i$, $e_i$ before submitting $e_i$ [a good number of his colleagues]$_j$

(111) a picture which he showed $e_j$, $e_j$ without forewarning $e_j$ [the unsuspecting members of the jury]$_j$

These need something like (112) and (113) respectively:

(112) \[ \text{VP/NP}_i / \text{NP}_j \hspace{1cm} \text{VP/VP/NP}_i \]
\[ \text{VP/NP}_i / \text{NP}_j \]

(113) \[ \text{VP/NP}_i / \text{NP}_j \hspace{1cm} \text{VP/VP/NP}_j \]
\[ \text{VP/NP}_i / \text{NP}_j \]

Further apparatus is needed for (114).

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

Since two modifiers of argument$_{N/(NN)}$ are extracted, it seems that it must somehow be mapped to $N/(NN)/(NN)$. In (115), (116), and (117) 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):

(115) a topic [about which]$_i$ I lent$_j$ [of which]$_j$ John a book $e_i$ and $e_j$ Mary a paper $e_j$

(116) I lent$_i$ [of which]$_j$ a book $e_j$ to John and $e_i$ a paper $e_j$ to Mary [about subjacency]$_j$

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

Examples (115) and (116) appear to require a book$_{NP/(NN)}$ 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 (117) 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:

\[(118) \quad X/Y + Y \Rightarrow X \Rightarrow X/Y + Y/Z \Rightarrow X/Z\]

And Application of Middle Abstraction to Backward Application yields Mixed Backward Composition:

\[(119) \quad Y + X Y \Rightarrow X \Rightarrow Y/Z + X Y \Rightarrow X/Z\]

Furthermore, recursive application of metarules achieves the required generalisations; for example applying Right Abstraction to the outputs of (118) and (119) provides the following:

\[(120) \quad \begin{align*}
a. \quad X/Y + Y/Z \Rightarrow X/Z \Rightarrow X/Y + Y/Z/W & \Rightarrow X/Z/W \\
b. \quad Y/Z + X Y \Rightarrow X/Z \Rightarrow Y/Z + X Y/W & \Rightarrow X/Z/W \\
\end{align*}\]

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, 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:

\[(121) \quad [X \ [Y \ Z]]_V \Rightarrow [X \ Y]_{V/Z}\]

Note that the output of (121) 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 (121) 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/(SNP). Secondly, the ‘universal grammar’ suggested by the ‘double-barreled’ 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 rulespace 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:

\[(X Y Z)_{V} \Rightarrow [X Z]_{V/Y}\]

This was adopted because the alternative (123a) would allow violation of the fixed subject constraint via (123b).

\[(X Y Z)_{V} \Rightarrow [X Z]_{V/Y}\]
\[(SP/S \ [NP \ S\{NP\}]_{SP} \Rightarrow [SP/S \ S\{NP\}]_{SP/NP}\]

However, if right node raising such as that in (124) were to be allowed, \(SP/S+NP+S\{NP\}\) would have the analysis (125) to which (122) can apply as in (126), to nevertheless violate the fixed subject constraint.

\[(?I \ think \ [that \ John \ e_{1} \ and \ that \ Mary \ e_{1}] \ [went \ to \ London]_{i}\]

\([(SP/S \ NP)_{SP/(S\{NP\})} \ S\{NP\}]_{SP}\]

\([(SP/S \ NP)_{SP/(S\{NP\})} \ S\{NP\}]_{SP} \Rightarrow [SP/S \ S\{NP\}]_{SP/NP}\]

For these reasons I have shifted from the earlier model, dubbed meta-categorial grammar (MCG) to the one presented in the next chapter which (if a name is required) might be called MCG-II.
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. \( f \) : \( X/Y + Y \Rightarrow X \)
    \[ f \times y = x y \]
    b. \( b \) : \( Y + X \Y \Rightarrow X \)
    \[ b \times x = x y \]

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) \( R \) : \( X + Y \Rightarrow Z \Rightarrow X + Y/W \Rightarrow Z/W \)
    \[ R g x y w = g x (y w) \]

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 \( Rf \):

¹Bob 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 ∈ \{NNN, 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^1: X => Y((\YYX))
r_f x y ≡ y x

**Backward Type-Raising**
r_b^2: 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 y)

Figure 1: CG Extended with Metarules

(3)  \[ R_f: X/Y + Y/Z => X/Z \]
    \[ R f x y z = x (y z) \]

So far as unary rules are concerned, one interesting possibility is to employ the PSG gap introduction metarules (4).\(^2\)

\(^2\)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.
(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:

(5)  
   X/Y + Y => X => Y => X\(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\VP => VP/NP => VP/NP => VP/NP/(VP\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\VP)/NP. Therefore I adopt the type-raising primitives (7a) along with the unary abstraction metarules (8b).

(7)  
   a.  r_f: X => Y/(\Y\X)  
   b.  r_b: X => \Y(Y/X) 

(8)  
   a.  F: X => Y => X\Z => Y\Z  
   b.  B: X => Y => X\Z => Y\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\N) to N/(N\N)/(N\N); this can be done as follows. 

3 On occasion I will refer to such derived rules as type-raising, though more strictly speaking they are generalisations of type-raising.
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 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:

\[(11) \quad \text{Rb: } Y + X\backslash Y/Z \Rightarrow X/Z\]

For example:
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) so that 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

As with the Chapter III grammar, fixed subject constraint violations such as (15) are not possible.

(15) *the man who I think that e₁ 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.

(16) the man who I met e₁’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
```

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
     ---------------
  VP/NP   ---------------f
     ---------------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
     ---------------rf
     ---------------rf
     ---------------rf
     ---------------rf
  N/(N\N)
  NP/(N\N)
  S/(N\N)
     ---------------f
     ---------------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 \rightarrow Z \rightarrow X/W + Y/W \rightarrow Z/W**
(21)  
a. Pf: \(X/Y/Z + Y/Z \Rightarrow X/Z\)  
b. Pb: \(Y/Z + X/Y/Z \Rightarrow X/Z\)

Also as in Chapter III, extraction like (22) is not possible.

(22)  
*the man who I showed \(e_1 e_1\)

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)  
\(L: X + Y \Rightarrow Z \Rightarrow X/W + Y \Rightarrow Z/W\)

(24)  
\(Lb: Y/Z + X/Y \Rightarrow X/Z\)

(25)  
\(\begin{array}{c}
\text{John} \\
\text{on Monday} \\
\hline
\text{NP} \\
\text{VP}\backslash VP \\
\hline
\text{VP}\backslash (VP/NP) \\
\text{VP}
\end{array}\)

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) \( Fr_b: X/Y \Rightarrow Z(Z/X)/Y \)

(28) a book to John

\[
\begin{array}{c}
NP/N \\
N \\
PP \\
\end{array}
\]

\[
\begin{array}{c}
Fr_b \\
VP/PP/(VP/PP/NP)/N \\
f \\
VP/PP/(VP/PP/NP) \\
f \\
VP/(VP/PP/NP)
\end{array}
\]

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-Canonicity

Double heavy shift past an adverbial arises through recursion of Middle Abstraction on itself:

(29) \( M(Mb): Y/Z/W + XY \Rightarrow X/Z/W \)

(30) posted yesterday a copy of the newsletter to every student

\[
\begin{array}{c}
VP/PP/NP \\
VP/VP \\
NP \\
PP \\
\end{array}
\]

\[
\begin{array}{c}
-M(Mb) \\
VP/PP/NP \\
f \\
VP/PP \\
f \\
VP
\end{array}
\]

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) \( R(Rf): X/Y + Y/Z/W \Rightarrow X/Z/W \)
(33) has given
VP/VP VP/PP/NP
------------------------R(Rf)
VP/PP/NP

(34) [Mary sent e₁ e₇ or John gave e₁ e₇] [a full report] to every student

(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 [Mary sent e₁ e₇ or John gave e₁ e₇] to every student

(38) Mary sent or John gave to every student

If one of two elements extracted is an adjunct, generalised type-raising is required:

(39) [I saw e₁ e₇ but Mary missed e₁ e₇] Dallas, yesterday

(40) I saw
NP S/NP/NP
------------------------Fr₁
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 Fr₁ is also required in the case of (41) where both a complement and an adjunct are extracted:

(41) a woman [about whom] an argument e₁ e₇ started [which went on all night]
The meaning assigned by the analysis is expressed by the combinator logic term (43a) which is equivalent to the λ-term (43b).

\[(43)\quad f(p_m \text{ about' whom'}) (M f (M (M b) (R (R f) \text{ an' (F r}_f \text{ argument'}) \text{ started'})) \text{ which...}')\]

\[\text{b. whom'} (\lambda y(\text{started'} (\text{an' (which...'} (\text{argument'} (\text{about'} y))))))\]

Cases of simultaneous independent extraction and parasitic extraction are characterised in a manner basically equivalent to that in the phrase structure grammar:

\[(44)\quad \text{a paper which, he showed } e_j e_i \text{ before submitting } e_i \text{ [a good number of his colleagues]}_j\]

\[(45)\quad M(P_b): \ Y/Z/W + X Y/Z \Rightarrow X/Z/W\]

\[(46)\quad \text{showed before submitting}\]

\[\text{VP/NP} / \text{NP}_j \quad \text{VP/VN} / \text{NP}_i\]

\[-M(P_b)\]

\[(47)\quad \text{a picture which, he showed } e_j e_i \text{ without forewarning } e_j \text{ [the unsuspecting members of the jury]}_j\]

\[(48)\quad P(M_b): \ Y/Z/W + X Y/W \Rightarrow X/Z/W\]

\[(49)\quad \text{showed without forewarning}\]

\[\text{VP/NP} / \text{NP}_j \quad \text{VP/VN} / \text{NP}_j\]

\[-P(M_b)\]
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] a woman whom I’ve always thought of as my Aunt]_i

(51) \[
\begin{array}{c}
\text{met} \\
\text{during} \\
\text{VP/NP}_i \\
\text{VP/VP/NP}_j \\
\text{VP/NP}_j/\text{NP}_j
\end{array}
\]

(52) the tokens which \(i\) he [handed \(e_i\) to \(e_j\) and took \(e_i\) from \(e_j\) [each acolyte]_j

(53) \[
\begin{array}{c}
\text{handed} \\
\text{to} \\
\text{VP/PP/NP}_i \\
\text{PP/NP}_j \\
\text{VP/NP}_j/\text{NP}_j
\end{array}
\]

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) \[
\begin{array}{c}
\text{is} \\
\text{hard} \\
\text{to} \\
\text{understand} \\
\text{S/NP/}(\text{N/N}) \\
\text{N/N/}(\text{S/NP/NP}) \\
\text{S/NP/}(\text{S/NP}) \\
\text{S/NP/NP} \\
\text{S/NP/NP} \\
\text{f} \\
\text{N/N} \\
\text{S/NP}
\end{array}
\]

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 \textsubscript{j} [the sonatas] \textsubscript{i} are hard to play \textsubscript{e} \textsubscript{i} on \textsubscript{e} \textsubscript{j}  
b. *the sonatas which \textsubscript{i} [the violin] \textsubscript{j} is hard to play \textsubscript{e} \textsubscript{i} on \textsubscript{e} \textsubscript{j}  

(57)  a. *some evidence which \textsubscript{j} [the witnesses] \textsubscript{i} are hard to show \textsubscript{e} \textsubscript{i} \textsubscript{e} \textsubscript{j}  
b. some witnesses whom \textsubscript{i} [the evidence] \textsubscript{j} is hard to show \textsubscript{e} \textsubscript{i} \textsubscript{e} \textsubscript{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 \textsubscript{i} the sonatas are hard to play \textsubscript{e} \textsubscript{i} on  
       \textsubscript{e} \textsubscript{j}  

(59)  which \textsubscript{i} the violin is hard to play \textsubscript{e} \textsubscript{i} on  

(60)  which \textsubscript{i} the witnesses are hard to show  

(61)  whom \textsubscript{i} the evidence is hard to show  

Example (62) exemplifies left node raising with across-the-board extraction from the second element.

(62)  a topic [about which] \textsubscript{i} I lent \textsubscript{j} [e \textsubscript{j} John a book e \textsubscript{i} and e \textsubscript{j} Mary a paper e \textsubscript{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)  
\[
\begin{array}{c}
\text{John} \\
\text{NP} \\
\text{VP/NP/(VP/NP/NP)} \\
\text{VP/(N/N)/(VP/NP/NP)} \\
\end{array}
\quad \begin{array}{c}
a \; \text{book} \\
\text{NP/N} \\
\text{VP/NP/(VP/NP/NP)} \\
\text{VP/(N/N)/(VP/NP/NP)} \\
\end{array}
\]

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)  
\[
\begin{array}{c}
a \\
\text{NP/N} \\
\text{VP/PP/(VP/PP/NP)/N} \\
\text{VP/PP/(VP/PP/NP)/(N/N)} \\
\text{VP/(VP/PP/NP)/(N/N)} \\
\end{array}
\quad \begin{array}{c}
\text{book} \\
\text{N/(N/N)} \\
\text{VP/PP/(VP/PP/NP)/(N/N)} \\
\end{array}
\quad \begin{array}{c}
to \; \text{John} \\
\text{PP} \\
\end{array}
\]

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)  
\[
\begin{array}{c}
a \\
\text{NP/N} \\
\text{VP/VP/NP} \\
\text{VP/(VP/NP)/NP} \\
\end{array}
\quad \begin{array}{c}
ticket \; to \; not \; wanting \; to \; visit \\
\text{N/NP} \\
\text{VP/VP/NP} \\
\text{VP/(VP/NP)/NP} \\
\end{array}
\]

The rules also enable first complement-adjunct left node raising of a verb seeking two complements:

(68)  
\[
\begin{array}{c}
\text{the people [to whom]_i we sent_j [e_j the report e_i on Monday and e_j the newsletter e_i on Tuesday]}
\end{array}
\]
(69)  the report on Monday

\[ \text{VP/PP/(VP/PP/NP)} \quad \text{VP/Vp} \]
\[ \text{-L(Mb)} \quad \text{VP/PP/(VP/PP/NP)} \]

However, the grammar characterises as ungrammatical commutation of verb dependents with left node raising, thus:

(70)  He believes \( [e_i e_j \text{ to be misguided } \text{[the GB-ers]}_j \text{, and } e_i e_k \text{ to be mistaken } \text{[the GPSG-ers]}_k] \)

(71)  believes \( [\text{to be misguided the GB-ers} \ldots] \)

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’. 
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. 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). 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/W + Y/W => Z/W
R g x y w ≡ g x (y w)

Forward Middle Abstraction
Mₐ: X + Y => Z ==> X/W + Y/W => Z/W
Mₐ g x y w ≡ g (x w) y

Forward Parasitic Abstraction
Pₐ: X + Y => Z ==> X/W + Y/W => Z/W
Pₐ g x y w ≡ g (x w) (y w)

Forward Type-Raising
rₐ: X => Y(Y/X)
rₐ x y ≡ y x

Forward Abstraction
F: X => Y => X/Z => Y/Z
F g x z ≡ g (x z)

Backward Application
b: Y + XY => 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
Mₜ: X + Y => Z ==> X + Y/W => Z/W
Mₜ g x y w ≡ g x (y w)

Backward Parasitic Abstraction
Pₜ: X + Y => Z ==> X/W + Y/W => Z/W
Pₜ g x y w ≡ g (x w) (y w)

Backward Type-Raising
rₜ: X => Y\(Y/X\)
rₜ x y ≡ y x

Backward Abstraction
B: X => Y => X\(X/Z\) => Y/Z
B g x y ≡ 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 Direction 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 SNP/NP applies to form SNP, 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](image)
(3) \[ X/Y + Y/Z \Rightarrow X/Z \]

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 \([SLASH \ldots]\), \([SUBCAT \ldots]\), and all the other top-level attribute-value pairs.

(4) \[ X[SUBCAT <C\ldots>, SLASH \ldots, \ldots] \]

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) \[ SNP/NP \]
\[ [TNS PRES] \]

Then the result of application is again not a unit, but the result of associating \(SNP\) 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 '∧':

\[
(6) \quad \text{Liisa nukkui} \\
\text{Lisa-nom slept-3sg} \\
'\text{Lisa slept}'
\]

\[
(7) \quad \text{Liisa nukkui} \\
\text{NP\_s} \quad \text{Sfin\_NP\_s} \\
\text{Sfin} \\
\text{Sfin}
\]

\[
(8) \quad \text{Nukkui Liisa} \\
\text{slept-3sg Lisa-nom} \\
'\text{Lisa did sleep}'
\]

\[
(9) \quad \text{Nukkui Liisa} \\
\text{Sfin\_NP\_s} \quad \text{NP\_s} \\
\text{Sfin} \\
\text{Sfin}
\]

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):

\[^{3}\] I am grateful to Kristina Jokinen for discussion relating to this section; all errors are of course my own responsibility.
(10) \[ M_f: X + Y \Rightarrow Z \Rightarrow X/W + Y \Rightarrow Z/W \]

(11) \[ M_b: X + Y \Rightarrow Z \Rightarrow X + Y \backslash W \Rightarrow Z \backslash W \]

(12) \[
\begin{array}{c}
\text{rakasti} \quad \text{Jussi} \quad \text{Liisaa} \\
\text{Sfin} \quad \text{NP_o} \quad \text{NP} \quad \text{NP_o} \\
\text{Sfin/} \quad \text{NP_o} \\
\text{Sfin} \quad \text{f} \\
\end{array}
\]

(13) \[
\begin{array}{c}
\text{Liisaa} \quad \text{Jussi} \quad \text{rakasti} \\
\text{NP_o} \quad \text{NP_s} \quad \text{Sfin} \quad \text{NP_s} \quad \text{NP_o} \\
\text{Sfin/} \quad \text{NP_o} \\
\text{Sfin} \quad \text{b} \\
\end{array}
\]

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  
Liri-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) \[
\begin{array}{c}
\text{Liisa} \quad \text{ei} \quad \text{ole} \quad \text{nukkunut} \\
\text{NP_s} \quad \text{Sfin} \quad \text{NP_s} \quad \text{Sfin} \quad \text{NP_s} \\
\text{Sneg} \quad \text{NP_s} \quad \text{Spcl} \quad \text{NP_s} \quad \text{Spcl} \quad \text{NP_s} \quad \text{f} \\
\text{Sfin/} \quad \text{NP_s} \\
\text{Sfin} \quad \text{b} \\
\end{array}
\]
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 nälissä tennistä

Not I have intended-pcp start-inf3 play-inf3 these-in tennis-ptv

'I have not intended to start to play tennis in these (clothes)'

Assuming that an adverbial like *nälissä has a category SING, 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 nälissä ole aikonut ruveta pelaamaan

Not I tennis in-these have intended start play

(20) a. nälissä ole aikonut ruveta pelaamaan

Not I tennis in-these have intended start play

b. En minä tennistä nälissä...
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). It remains to be seen how far free ordering and extraction can be equated, and what means of distinction might be invoked.

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.

\[
\begin{align*}
(21) & \quad a := A \\
& \quad b := S\sigma/\sigma S, S\sigma/\sigma C \\
& \quad c := C \\
(22) & \quad L: X + Y \Rightarrow Z \Rightarrow XW + Y \Rightarrow Z/W \\
& \quad M: X + Y \Rightarrow Z \Rightarrow X/W + Y \Rightarrow Z/W
\end{align*}
\]

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).

\[
(23) \quad A^n S\sigma/\sigma C S\sigma/\sigma C S\sigma/\sigma C S\sigma/\sigma C^n
\]

\[
(24) \quad a \quad a \quad a \quad b \quad b \quad c \quad c \quad c
\]

\[
\begin{align*}
A & \quad A \quad A \quad S\sigma/\sigma C \quad S\sigma/\sigma C \quad S\sigma/\sigma C \quad S\sigma/\sigma C \quad S\sigma/\sigma C \quad S\sigma/\sigma C \\
& \quad L(M(Lb)) \\
& \quad S\sigma/\sigma C \quad S\sigma/\sigma C \quad S\sigma/\sigma C \quad S\sigma/\sigma C \\
& \quad -M(Lb) \\
& \quad S\sigma/\sigma C \quad S\sigma/\sigma C \quad S\sigma/\sigma C
\end{align*}
\]

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 'e'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 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 0-criterion of Government-Binding theory, and the completeness and coherence conditions of Lexical-Functional Grammar. Although the discussion becomes quite technical it is intended to be suggestive

---

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 Kit Fine, Ewan Klein, Marcus Kracht, and Barry Richards.
rather than logically irrefutable, and it included partly to illustrate some interesting connections between type systems and implicational logics.\textsuperscript{5}

Consider the following ungrammatical sentences:

\begin{equation}
\text{(25)} \quad *\text{John says}
\end{equation}

\begin{equation}
\text{(26)} \quad *\text{John laughed Mary}
\end{equation}

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. I will show how a principle of compositionality can rule out such ungrammaticality, independent of a theory of syntax.

In a general form, the principle of compositionality (see e.g., Janssen 1983 chapter I; Partee 1984) can be stated:

\begin{equation}
\text{(27)} \quad \textit{Compositionality}
\end{equation}

The meaning of an expression is a 'function' of the meanings of its parts.

Since the meanings of the 'parts' will themselves be functions of the meanings of their parts, the principle entails that the meaning of an expression is ultimately a function of the meanings of its basic parts or words, this function being essentially the composition of the functions successively applying to parts. I will propose a specific constraint on compositionality by delimiting this function, and show how the principle ensures effects like those of Lexical-Functional Grammar's completeness and coherence conditions, and Government-Binding's $\theta$-criterion. This is achieved 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, it is shown that typehood is equivalent to theoremhood, and by proving non-validity, it is demonstrated that certain kinds of ill-formed sentences could never be generated by grammars respecting what I will call $\lambda$I-compositionality.\textsuperscript{6}

\textsuperscript{5}Cf. van Benthem (1987)

\textsuperscript{6}The discussion is abstracted over what 'meanings' actually are, other than that they can be regarded as set-theoretic objects and functions.
Classes of functions will be identified by reference to the pure typed \(\lambda\)-calculus, and combinatory logic. "Pure" means that there are no constants; functions of "type" \(A \rightarrow B\) map from objects of type \(A\) into objects of type \(B\). Given a non-empty set \(\Delta\) of basic types, a set of types is defined which is the smallest set satisfying the following conditions:

(28) 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 \(\Delta\) includes \(NP\), \(S\), and \(SP\), the types of the meanings of noun phrases, sentences, and complemented sentences. 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; for example this last formula may be written \(NP \rightarrow NP \rightarrow (NP \rightarrow S) \rightarrow S\). Given an infinite set \(\text{Var}_A\) of variables for each type \(A\) the set of \(\lambda K\)-terms is defined by:

(29) a. If \(v_A \in \text{Var}_A\)
then \(v_A\) is a \(\lambda K\)-term of type \(A\)

b. If \(\phi\) is a \(\lambda K\)-term of type \(A \rightarrow B\) and \(\psi\) is a \(\lambda K\)-term of type \(A\)
then \(\phi \psi\) is a \(\lambda K\)-term of type \(B\)

c. If \(v_A \in \text{Var}_A\) and \(\phi\) is a \(\lambda K\)-term of type \(B\)
then \(\lambda v_A \phi\) is a \(\lambda K\)-term of type \(A \rightarrow B\)

A \(\lambda K\)-term without any free variables is said to be closed. Assuming the standard functional interpretation, we will call the functions definable by closed \(\lambda K\)-terms the \(\lambda K\)-functions. Then one version of the principle of compositionality is:

(30) \(\lambda K\)-Compositionality

The meaning of an expression is a \(\lambda K\)-function of the meanings of its words.

The \(\lambda K\)-functions are closed under permutation, i.e. if there is a function mapping certain arguments into certain results, then there are functions mapping all permutations of those arguments into the same result: the different functions are defined by terms in which the \(\lambda\)-bindings appear in different orders. I 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 of types $A_1, A_2, \ldots$ can form an expression of type $A_0$ \(\lambda K\)-compositionally only if $A_1 \rightarrow A_2 \rightarrow \ldots \rightarrow A_0$ is a \(\lambda K\)-type, i.e. the type of some \(\lambda K\)-function: if it were not, the meaning of the expression could not be a \(\lambda K\)-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 \(\lambda K\)-type. If there were no \(\lambda K\)-function of this type then there could be no \(\lambda K\)-function mapping meanings of type NP and SP $\rightarrow$ NP $\rightarrow$ S into meanings of type S. 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 \(\lambda K\)-type.

In order to determine whether a type is a \(\lambda K\)-type, we take advantage of the fact that a function is definable by a closed \(\lambda K\)-term if and only if it is definable by a combinatory logic term as follows:\(^8\)

\[ (31) \]

\begin{enumerate}
\item If A, B, and C are types,
\[
I_{A \to A} (\equiv \lambda x_A [x_A])
\]
is a CL-term of type $A \to A$

\[
B_{(A \to B) \to (C \to A) \to C \to B} (\equiv \lambda x_A \to B \lambda y_C \to A \lambda z_C [x_A \to B (y_C \to A z_C)])
\]
is a CL-term of type $(A \to B) \to (C \to A) \to C \to B$

\[
C_{(A \to B \to C) \to B \to A \to C} (\equiv \lambda x_A \to B \to C \lambda y_B \to A \lambda z_B [x_A \to B \to C z_B y_B])
\]
is a CL-term of type $(A \to B \to C) \to B \to A \to C$

\[
W_{(A \to A \to B) \to A \to B} (\equiv \lambda x_A \to A \to B \lambda y_A \to A \to B y_A y_A)
\]
is a CL-term of type $(A \to A \to B) \to A \to B$

\[
K_{A \to B \to A} (\equiv \lambda x_A \lambda y_B [x_A])
\]
is a CL-term of type $A \to B \to A$

\item If $\phi$ is a CL-term of type $A \to B$ and $\psi$ is a CL-term of type $A$

then $\phi \psi$ is a CL-term of type $B$
\end{enumerate}

Thus the \(\lambda K\)-types are those types which are derivable from the axiom schemata (32a), corresponding to the combinators in (31a), and the modus ponens rule (32b), corresponding to the application in (31b).

\(^8\)See Curry and Feys (1958) or Barendregt (1981) for proofs of the equivalence. Often, combinatory logic definitions are given using the substitution combinator S; the above formulation is more convenient for our purposes.
(32) a. \( A \to A \)
    \( (A \to B) \to (C \to A) \to C \to B \)
    \( (A \to B) \to C \to B \to A \to C \)
    \( (A \to A \to B) \to A \to B \)
    \( A \to B \to A \)

b. \( A \to B, A \vdash B \)

Viewing ‘\( \to \)’ as implication, (32) provides an axiomatisation of the implicational intuitionistic logic which Anderson and Belnap (1975) call \( H_{\rightarrow} \). So we know that a type \( A \) is a \( \lambda K \)-type if and only if, regarded as an implicational formula, it is a theorem of \( H_{\rightarrow} \). For example, where the types are assumed to be as indicated, the string in (33a) could be generated \( \lambda K \)-compositionally as a sentence only if (33b) is a theorem of \( H_{\rightarrow} \).

(33) a. \(*John says*

b. \( NP \to (SP \to NP \to S) \to S \)

I will prove that (33b) is not a theorem of \( H_{\rightarrow} \) by deriving a counter-model.

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 \( M = (L, \cup, \bot, \nu) \) where \( L \) is a set, \( \bot \) is a member of \( L \), \( \cup \) (join) is a binary operator over the set \( L \) satisfying (34,) and \( \nu \) is a valuation function mapping from \( L \) into subsets of \( P \), satisfying (35).

\[
(34) \quad \begin{align*}
\bot \cup i &= i \\
i \cup i &= i \\
i \cup j &= j \cup i \\
i \cup (j \cup k) &= (i \cup j) \cup k 
\end{align*}
\]

\[
(35) \quad \textit{Hereditary Condition}
\]

For every \( p \in P \) and all \( i, j \in L \), if \( p \in \nu(i) \) then \( p \in \nu(i \cup j) \).

Thus \( (L, \cup) \) is a join semi-lattice with least element \( \bot \). Intuitively, the hereditary condition requires that all propositions true at some index are true at the least upper bound of that index with any other one. For a model \( M \) we define a satisfaction relation \( \models_M \) between indices and formulae by:

---

9The arguments are conditional on the given assignment of the meanings of words to types, but I believe these assignments are uncontroversial.
(36) a. For every $p \in P$ and $i \in L$, $i \models_M p$ if and only if $p \in v(i)$
b. For all formulae $\phi, \psi$, every index $i \in L$, $i \models_M \phi \rightarrow \psi$ if and only if for
   every $j \in L$, $j \models_M \phi$ only if $i \cup j \models_M \psi$

That is, a model satisfies a basic formula $p$ at index $i$ if and only if $p \in v(i)$, and a model
fails to satisfy an implicational formula at index $i$ if and only if there exists some index $j$
such that the antecedent is satisfied at $j$ but the consequent is not satisfied at the least upper
bound of $i$ and $j$. A formula $\phi$ is valid with respect to a model $M$ if and only if $\bot \models_M \phi$.
A formula is valid if and only if it is valid in every model. The formula (33b) is not valid
because $(\bot, \cup, \bot, v)$ where $v(\bot) = \{NP\}$ is a counter-model, that is a model which does
not satisfy it. A proof that this is a counter-model runs as follows. We are required to show (37).

(37) $\bot \not\models NP \rightarrow (SP \rightarrow NP \rightarrow S) \rightarrow S$

This would be the case if (38) holds.

(38) $\bot \models NP$ and $\bot \not\models (SP \rightarrow NP \rightarrow S) \rightarrow S$

The left-hand conjunct is true, so it remains to show that the right-hand conjunct is true.
This would be so if (39) holds.

(39) $\bot \models SP \rightarrow NP \rightarrow S$ and $\bot \not\models S$

The right-hand conjunct is true. We would have shown the left-hand one to be true if we
could show (40).

(40) it's not the case that both $\bot \models SP$ and $\bot \not\models NP \rightarrow S$

I.e.

(41) either $\bot \not\models SP$ or $\bot \models NP \rightarrow S$

The first of these is true, and thus so is the whole disjunction. Hence (33b) is not a
theorem of $H \rightarrow$ and not the type of any $\lambda K$-function. So assuming the given assignment of
word meanings to types, (33a) could not be generated by any $\lambda K$-compositional grammar.

Within Lexical-Functional Grammar (LFG), sentences like (33a) in which an argument
is missing are excluded by the completeness condition, and within Government-Binding
(GB), they are excluded by the $\theta$-criterion:10

\footnote{More precisely, they are excluded by the $\theta$-criterion in conjunction with the projection principle, but the details are
not important.}
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]

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 (44a) which contain a redundant argument. In LFG this is done by the coherence condition (45).

a. *John laughed Mary
   b. NP→(NP→S)→NP→S

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 (44a) 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→(NP→S)→NP→S:

\[ \lambda x_{NP} \lambda y_{NP→S} \lambda z_{NP} [y_{NP→S} x_{NP}] \]

What is distinctive about this function is that it engenders vacuous abstraction: \( z_{NP} \) does not appear in the body of the λK-term. It would be odd for universal grammar to afford such abstraction because its significance would be that the meanings of some words do not con-
tribute to the meanings of the expressions in which they appear. This suggests a refined principle of compositionality making reference to just the \(\lambda I\)-functions: the functions definable by closed \(\lambda K\)-terms without vacuous abstraction:

\[
(47) \quad \lambda I\text{-Compositionality}
\]

The meaning of an expression is a \(\lambda I\)-function of the meanings of its parts.

The combinator corresponding to vacuous abstraction is \(K\). The functions definable by combinatory logic terms as indicated earlier, except without \(K\), are the \(\lambda I\)-functions. The axioms corresponding to these remaining combinators, with *modus ponens*, are those of the implicational relevance logic which Anderson and Belnap call \(R_{\rightarrow}\).

We can prove that (44b) is not a theorem of \(R_{\rightarrow}\), and hence prove that sentences like (44a) cannot be generated \(\lambda I\)-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. Intuitively, since a theorem is valid in every possible model, relaxing the hereditary condition expands the set of possible models and results in a stiffer requirement for validity. Thus the theorems of \(R_{\rightarrow}\) form a subset of those of \(H_{\rightarrow}\). A counter-model for (44b) is represented by the following diagram:

\[
(48)
\]

Each node corresponds to an index and its associated image under \(v\); the join of any two indices is the lowest index that dominates them both. To prove that (48) is a counter-model we need to show that (49) holds.

\[
(49) \quad \bot \not\Rightarrow \text{NP} \rightarrow (\text{NP} \rightarrow \text{S}) \rightarrow \text{NP} \rightarrow \text{S}
\]

We would have shown this if we could show that for some index, the antecedent is

---

11. I would argue that the apparent nil contribution of *it* and *there*, as in *it seems that John left*, is in the 'lexical' semantics as opposed to the 'syntactic' semantics.
12. Again, see Curry and Feys (1958) or Barendregt (1981).
13. See especially their axiomatisation \(R_{\rightarrow}\) of \(R_{\rightarrow}\) on p88.
satisfied at that index while the consequent is not satisfied at the least upper bound of that index and $\bot$. In particular we would have a proof if we could show (50).

$$
(50) \quad j \models NP \text{ and } j \cup \bot \not\models (NP \rightarrow S) \rightarrow NP \rightarrow S
$$

The left-hand conjunct is true; the right-hand conjunct would be shown to be true if we could show for some index that while the antecedent is satisfied at that index, the consequent is not satisfied at the least upper bound of that index with $j \cup \bot (= j)$:

$$
(51) \quad k \models NP \rightarrow S \text{ and } k \cup j \not\models NP \rightarrow S
$$

The left-hand conjunct 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 $1$ satisfies NP, the least upper bound of this index and $k \cup j$ does not satisfy $S$. This completes the proof.

To summarise, $\lambda$I-compositionality captures missing-word and redundant-word ungrammaticality. $\lambda$I-compositionality is consistent with all the proposals for the semantics of rules and metarules made earlier in this thesis. Also, within GPSG there is a Semantic Interpretation Principle (SIP) which determines the semantics of derivation steps in terms of types associated with the participating categories (see Klein and Sag 1985; Gazdar et al. 1985). $\lambda$I-compositionality is a corollary of the SIP. The hypothesis being advocated here is that the $\lambda$I-functions exhaust the functions available as the semantics of syntactic rules in natural grammar. An interesting question is whether compositionality can be narrowed down to involve a smaller class than the $\lambda$I-functions.

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 \( w_1 w_2 w_3 \ldots \) 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 \( \phi \), and that corresponding the right input is \( \psi \), 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).

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. 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:

---

If there is no lexical entry then, obviously, fail. The algorithm generalises straightforwardly to handle lexical ambiguity.
Then the next word is read, and $\text{edge}(3, \text{SNP}(\text{SNP}), 4)$ is added. This has three edges leading into it, two of which it can apply back to. Reducing $\text{edge}(2, \text{SNP}, 3)$ and $\text{edge}(3, \text{SNP}(\text{SNP}), 4)$ gives $\text{edge}(2, \text{SNP}, 4)$ which reduces with $\text{edge}(1, \text{SNP}(\text{SNP}), 2)$ to give $\text{edge}(1, \text{SNP}, 4)$ which in turn reduces with $\text{edge}(0, \text{NP}, 1)$ to give $\text{edge}(0, S, 4)$. Reducing $\text{edge}(1, \text{SNP}, 3)$ and $\text{edge}(3, \text{SNP}(\text{SNP}), 4)$ gives $\text{edge}(1, \text{SNP}, 4)$ which also reduces with the first word’s edge to give $\text{edge}(0, S, 4)$:

(4) \begin{align*}
&\text{edge}(0, \text{NP}, 1) \\
&\text{edge}(1, \text{SNP}(\text{SNP}), 2) \\
&\text{edge}(2, \text{SNP}, 3) \\
&\text{edge}(1, \text{SNP}, 3) \\
&\text{edge}(0, S, 3) \\
&\text{edge}(3, \text{SNP}(\text{SNP}), 4) \\
&\text{edge}(2, \text{SNP}, 4) \\
&\text{edge}(1, \text{SNP}, 4) \\
&\text{edge}(0, S, 4) \\
&\text{edge}(1, \text{SNP}, 4) \\
&\text{edge}(0, S, 4)
\end{align*}

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:

\[
\begin{align*}
5. & \quad \text{likes} \quad := \ s(\text{fin}, \text{like}(X,Y)) \text{np}(\text{nom}, Y) / \text{np}(\text{acc}, X) \\
6. & \quad \text{John} \quad := \ \text{np}(C, \text{john}) \\
   & \quad \text{Mary} \quad := \ \text{np}(C, \text{mary}) \\
\end{align*}
\]

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 ‘\(^\cdot\)\(^\cdot\)’, the operator ‘::’ separating semantic representations and basic category symbols, and the lexical assignment operator ‘:=’.

\[
\begin{align*}
7. & \quad :- \ \text{op}(400, \text{xfy}, \cdot) \\
   & \quad :- \ \text{op}(400, \text{yfx}, \cdot) \\
   & \quad :- \ \text{op}(200, \text{yfx}, \cdot) \\
   & \quad :- \ \text{op}(300, \text{xfy}, ::) \\
   & \quad :- \ \text{op}(500, \text{xfy}, ::) \\
\end{align*}
\]

The top-level procedure \(prs(+Str)\) parses anew the string \(Str\). It clears the chart, using
likes := \[
\begin{array}{c}
\text{cat} & s \\
vform & \text{fin} \\
\text{sem} & \begin{array}{c}
\text{pred} \\
\text{subj} \\
\text{obj}
\end{array} & \begin{array}{c}
\text{likes} \\
X \\
Y
\end{array}
\end{array}
\]
\[
\begin{array}{c}
\text{cat} & \text{np} \\
\text{case nom}
\end{array}
\]
\[
\begin{array}{c}
\text{cat} & \text{np} \\
\text{case acc}
\end{array}
\]

john := \[
\begin{array}{c}
\text{cat} \\
\text{pred}
\end{array} & \text{np} & \text{john}
\]

mary := \[
\begin{array}{c}
\text{cat} \\
\text{pred}
\end{array} & \text{np} & \text{mary}
\]

Figure 1

John likes Mary := \[
\begin{array}{c}
\text{cat} & s \\
vform & \text{fin} \\
\text{sem} & \begin{array}{c}
\text{pred} \\
\text{subj} \\
\text{obj}
\end{array} & \begin{array}{c}
\text{likes} \\
\begin{array}{c}
\text{cat} & \text{np} \\
\text{case acc}
\end{array} \\
\begin{array}{c}
\text{sem} \\
\text{mary}
\end{array} \\
\begin{array}{c}
\text{cat} & \text{np} \\
\text{case nom}
\end{array} \\
\begin{array}{c}
\text{sem} \\
\text{john}
\end{array}
\end{array}
\end{array}
\]

Figure 2

retractall(?Cls), and calls prsI(+Str,+Pos) instantiating the current position to zero:

(8) \[\text{prs(}\text{String}) :-
\]
\[\text{retractall(edge(\ldots,\ldots))},
\]
\[\text{prsI(}\text{String},0\text{)}.
\]

The procedure prsI(+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) \text{prsl}([\text{WordString}],\text{Pos}) :\text{-}
\begin{align*}
\text{Word} & \text{ := } \text{LexCat}, \\
\text{incorp}(\text{Pos},\text{LexCat},s(\text{Pos})); \\
\text{prsl}(\text{String},s(\text{Pos})).
\end{align*}

Sample lexical entries are as follows. Note how the sentential scope of say \textit{will} is achieved by predicating \textit{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 \textit{will} with the verb’s subject by sharing variables. This amounts to a unification implementation of functional composition.

(10)
\begin{align*}
\text{john} & \text{ := } j:\text{np}. \\
\text{mary} & \text{ := } m:\text{np}. \\
\text{bill} & \text{ := } b:\text{np}. \\
\text{you} & \text{ := } \text{you}:\text{np}. \\
\text{we} & \text{ := } \text{we}:\text{np}. \\
\text{leave} & \text{ := } \text{leave}’X:s\text{\textit{X}}:\text{np}. \\
\text{eating} & \text{ := } \text{eating}’X:s\text{\textit{X}}:\text{np}. \\
\text{like} & \text{ := } \text{like}’X’Y:s\text{\textit{Y}}:\text{np}/X:\text{np}. \\
\text{today} & \text{ := } \text{today}’X:s\text{\textit{Y}}:\text{np}/(X:s\text{\textit{Y}}:\text{np}). \\
\text{tomorrow} & \text{ := } \text{tomorrow}’X:s\text{\textit{Y}}:\text{np}/(X:s\text{\textit{Y}}:\text{np}). \\
\text{will} & \text{ := } \text{will}’X:s\text{\textit{Y}}:\text{np}/(X:s\text{\textit{Y}}:\text{np}). \\
\text{to} & \text{ := } \text{to}’X:s\text{\textit{Y}}:\text{np}/(X:s\text{\textit{Y}}:\text{np}). \\
\text{seem} & \text{ := } \text{seem}’X:s\text{\textit{Y}}:\text{np}/(X:s\text{\textit{Y}}:\text{np}). \\
\text{try} & \text{ := } \text{try}’X’Y:s\text{\textit{Y}}:\text{np}/(X:s\text{\textit{Y}}:\text{np}). \\
\text{expect} & \text{ := } \text{expect}’X’Y:s\text{\textit{Y}}:\text{np}/(X:s\text{\textit{Z}}:\text{np})/Z:np. \\
\text{persuade} & \text{ := } \text{persuade}’X’Y’Z:s\text{\textit{Z}}:np/(Y:s\text{\textit{X}}:\text{np})/X:np. \\
\text{promise} & \text{ := } \text{promise}’X’Y’Z:s\text{\textit{Z}}:np/(Y:s\text{\textit{Z}}:\text{np})/X:np. \\
\text{that} & \text{ := } \text{that}’X:sp/X:s. \\
\text{think} & \text{ := } \text{think}’X’Y:s\text{\textit{Y}}:np/X:sp. \\
\text{with} & \text{ := } \text{with}’X’Y:s\text{\textit{Z}}:np/(Y:s\text{\textit{Z}}:np)/X:np. \\
\text{while} & \text{ := } \text{while}’X’Y:s\text{\textit{Z}}:np/(Y:s\text{\textit{Z}}:np)/(X:s\text{\textit{Z}}:np).
\end{align*}

The heart of the implementation is \textit{incorp}(+MPos,+RCat,+RPos) which incorporates into the chart between positions \textit{MPos} and \textit{RPos} an edge of category \textit{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,XY,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. Test clauses are shown in Figure 3. By way of illustration, a call of 
prst([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')):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

---

2I 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 unification 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).

\begin{align}
\text{(13)} & \quad \begin{array}{ll}
\text{a.} & \left[ \left[ R \left[ \left[ f X / Y \ Y / Z \right] X / Z \right] Z \right] X \right] X \\
& f (R \ f \ x \ y) \ z = x (y \ z) \\
\text{b.} & \left[ \left[ f X / Y \ Y / Z \ Z \right] Y \right] X \\
& f x (f \ y \ z) = x (y \ z)
\end{array}
\end{align}

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, naïve 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).

\begin{align}
\text{(14) ~ Forward Composition} \\
X / Y + Y / Z \Rightarrow X / Z
\end{align}
(15) \textit{Forward-Predictive Forward Functional Composition}

\[ X/(Y/Z) + Y/W \Rightarrow X/(W/Z) \]

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) \[ VP/PP + PP/NP \Rightarrow VP/NP \]

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 (1986) 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) \begin{align*}
\text{John} & \quad \text{and} \quad \text{Mary} & \text{left} \\
A/(A/j:np) & \quad B/(B/m:np) & \text{left } Z:s/(Z:np)
\end{align*}

The analysis requires predicating \textit{left} of both \textit{j} and \textit{m}; however the single variable \textit{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.

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:

\[
\begin{align*}
[f \psi X Y]_{V/Z} Z]_V &= [f \psi X Y]_{V/Z} Z]_V \\
[b \psi X Y]_W \lor W]_V &= [b \psi X Y]_W \lor W]_V \\
[ \phi [f X/Y Y]_X Z]_V &= [ \phi [f X/Y Y]_X Z]_V \\
[ \phi [b Y X/Y]_X Z]_V &= [ \phi [b Y X/Y]_X Z]_V
\end{align*}
\]

Next, suppose there is the following equivalence:

\[
[ \phi \psi X Y]_A Z]_V \equiv [\phi' \psi X Y]_B Z]_V
\]

If the left-hand path exists, then so too must the one in (20) where $Z$ becomes $Z/W$ and $R\phi$ rather than $\phi$ applies, so that $W$ is right-abstracted onto the root category.

\[
[R\phi \psi X Y]_A Z/W]_V/W
\]

There must also exist a derivation likewise related to the right hand side of (19):

\[
[R\phi' \psi X Y]_B Z/W]_V/W
\]

And since the derivations in (19) were equivalent, i.e. (22a) holds, it must also be the case that (22b) holds.

\[
\begin{align*}
\text{a. } & \phi (\psi x y) z = \phi' (\psi' x y) z \\
\text{b. } & R \phi (\psi x y) z = R \phi' (\psi' x y) z \text{ since} \\
& R \phi (\psi x y) z w = \phi (\psi x y) (z w) = \\
& R \phi' (\psi' x y) z w = \phi' (\psi' x y) (z w)
\end{align*}
\]

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)  
\[
\begin{align*}
&a. \ [M_\phi [R_\psi X \ Y/W]_{A/W} \ Z]_{V/W} \\
&b. \ [M_\phi' [R_\psi' X \ Y/W]_{B/W} \ Z]_{V/W}
\end{align*}
\]

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 \( \pi_1 \equiv \psi_2 \rightarrow \pi_3 \equiv \psi_4, \pi_5 \equiv \psi_6, \ldots \) indicates that from the antecedent equivalence, the consequent equivalences can be inferred.

(24)  
\[
\begin{align*}
&[\phi [\psi X Y]_{A} \ Z]_{V} \\
&[R_\phi [\psi X Y]_{A} Z/W]_{V/W} \\
&[M_\phi [R_\psi X Y/W]_{A/W} Z]_{V/W} \\
&[M_\phi [M_\psi X/W Y]_{A/W} Z]_{V/W} \\
&[P_\phi [R_\psi X Y/W]_{A/W} Z/W]_{V/W} \\
&[P_\phi [M_\psi X/W Y]_{A/W} Z/W]_{V/W} \\
&[M_\phi [P_\psi X/W Y]_{A/W} Z]_{V/W} \\
&[L_\phi [L_\psi X/W Y]_{A/W} Z]_{V/W} \\
&= [\psi X Y]_{B} Z]_{V} \rightarrow \\
&= [R_\psi [\psi X Y]_{B} Z/W]_{V/W}, \\
&= [M_\psi [R_\psi X Y/W]_{B/W} Z]_{V/W}, \\
&= [M_\psi [M_\psi X/W Y]_{B/W} Z]_{V/W}, \\
&= [P_\psi [R_\psi X Y/W]_{B/W} Z/W]_{V/W}, \\
&= [P_\psi [M_\psi X/W Y]_{B/W} Z/W]_{V/W}, \\
&= [M_\psi [P_\psi X/W Y]_{B/W} Z]_{V/W}, \\
&= [L_\psi [L_\psi X/W Y]_{B/W} Z]_{V/W}
\end{align*}
\]

For the equivalence rules given so far, \( W \) was inherited ‘outwardly’, (outmost 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)  
\[
[R_\psi X Y/Z]_{V/Z} Z]_{V} \equiv [\psi X Y/Z]_{B} Z]_{V}
\]

There is the path (26) closely related to the left hand side of (25).

(26)  
\[
[R_\psi X/Y/Z]_{V/W/Z} Z]_{V/W}
\]

But the argument inherited inwardly could have been inherited outwardly as in (27), related to the right hand side of (25).

(27)  
\[
[M_\psi X/W Y/Z]_{B/W} Z]_{V/W}
\]

And from (28a) we can infer (28b).
(28) a. \( f (R \, \psi \times y) \, z = \phi' (\psi' \times y) \, z \) i.e.
\( \psi \times (y \, z) = \phi' (\psi' \times y) \, z \)

b. \( f (R \, (M \, \psi) \times y) \, z = M \phi (M \, \psi \times y) \, z \) since
\( f (R \, (M \, \psi) \times y) \, z \, w = R \, (M \, \psi) \times y \times z \, w = M \, \psi \times (y \, z) \, w = \psi \times (w \, y) \, z \)
\( M \phi' (M \, \psi' \times y) \, z \, w = \phi' (M \, \psi' \times y \, w) \, z = \phi' (\psi' \times (y \, w)) \, z \)

In all, for this kind of equivalence we have:

\[
\begin{align*}
[f \cdot R' X \cdot Y/Z]_{V/Z} & \equiv [\phi' \cdot \psi \cdot X \cdot Y/Z]_B \cdot Z]_V \rightarrow \\
[f \cdot R(M \psi) \cdot Y/Z/W]_{V/W/Z} & \equiv [M_{\phi'} \cdot M_{\psi'} \cdot X/W \cdot Y/Z]_B \cdot W \cdot Z]_V/W, \\
[f \cdot R(L \psi) \cdot X/W \cdot Y/Z/W]_{V/W/Z} & \equiv [L_{\phi'} \cdot L_{\psi'} \cdot X/W \cdot Y/Z]_B \cdot W \cdot Z]_V/W.
\end{align*}
\]

\[
\begin{align*}
[f \cdot M \psi \cdot X/Z \cdot Y]_{V/Z} & \equiv [\phi' \cdot \psi \cdot X/Z \cdot Y]_B \cdot Z]_V \rightarrow \\
[f \cdot M(R \psi) \cdot X/Z \cdot Y/W]_{V/W/Z} & \equiv [M_{\phi'} \cdot M_{\psi'} \cdot X/Z \cdot Y/W]_B \cdot W \cdot Z]_V/W.
\end{align*}
\]

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).

\[
\begin{align*}
[f \cdot V/A \cdot \psi \cdot Y \cdot Z]_A & \equiv [f \cdot V/A \cdot \psi \cdot Y \cdot Z]_A \cdot V \\
[f \cdot _b \cdot X \cdot \psi \cdot Y \cdot Z]_V & \equiv [f \cdot _b \cdot X \cdot \psi \cdot Y \cdot Z]_V \cdot V \\
[f \cdot \phi \cdot X \cdot [f \cdot Y/Z \cdot Z]_V]_V & \equiv [f \cdot \phi \cdot X \cdot [f \cdot Y/Z \cdot Z]_V]_V \\
[f \cdot \phi \cdot X \cdot [f \cdot _b \cdot Y \cdot Z \cdot Y]_Z]_V & \equiv [f \cdot \phi \cdot X \cdot [f \cdot _b \cdot Y \cdot Z \cdot Y]_Z]_V.
\end{align*}
\]

\[
\begin{align*}
[f \cdot \phi \cdot X \cdot \psi \cdot Y \cdot Z]_A & \\
[f \cdot R_{\phi} \cdot X \cdot \psi \cdot Y \cdot Z/W]_{A/W} & \equiv [f \cdot R_{\phi} \cdot X \cdot \psi \cdot Y \cdot Z/W]_{A/W} \cdot V/W \\
[f \cdot R_{\phi} \cdot X \cdot [M_{\psi} \cdot Y/W \cdot Z]_{A/W} & \equiv [f \cdot R_{\phi} \cdot X \cdot [M_{\psi} \cdot Y/W \cdot Z]_{A/W} \cdot V/W \\
[f \cdot M_{\phi} \cdot X/W \cdot [\psi \cdot Y \cdot Z]_A & \equiv [f \cdot M_{\phi} \cdot X/W \cdot [\psi \cdot Y \cdot Z]_A \cdot V/W \\
[f \cdot P_{\psi} \cdot X/W \cdot Y/W \cdot Z/W] & \equiv [f \cdot P_{\psi} \cdot X/W \cdot Y/W \cdot Z/W]_{A/W} \cdot V/W \\
[f \cdot P_{\phi} \cdot X/W \cdot [M_{\psi} \cdot Y/W \cdot Z/W] & \equiv [f \cdot P_{\phi} \cdot X/W \cdot [M_{\psi} \cdot Y/W \cdot Z/W]_{A/W} \cdot V/W \\
[f \cdot L_{\phi} \cdot X/W \cdot [\psi \cdot Y \cdot Z]_A & \equiv [f \cdot L_{\phi} \cdot X/W \cdot [\psi \cdot Y \cdot Z]_A \cdot V/W \\
[f \cdot \phi \cdot X \cdot \psi \cdot Y \cdot Z]_B & \equiv [f \cdot \phi \cdot X \cdot \psi \cdot Y \cdot Z]_B \cdot V \\
[f \cdot R_{\phi} \cdot X \cdot \psi \cdot Y \cdot Z/W] & \equiv [f \cdot R_{\phi} \cdot X \cdot \psi \cdot Y \cdot Z/W]_{B/W} \cdot V/W \\
[f \cdot R_{\phi} \cdot X \cdot [M_{\psi} \cdot Y/W \cdot Z] & \equiv [f \cdot R_{\phi} \cdot X \cdot [M_{\psi} \cdot Y/W \cdot Z]_B \cdot W] \cdot V/W \\
[f \cdot P_{\psi} \cdot X/W \cdot Y/W \cdot Z/W] & \equiv [f \cdot P_{\psi} \cdot X/W \cdot Y/W \cdot Z/W]_{B/W} \cdot V/W \\
[f \cdot L_{\phi} \cdot X/W \cdot [\psi \cdot Y \cdot Z] & \equiv [f \cdot L_{\phi} \cdot X/W \cdot [\psi \cdot Y \cdot Z]_B \cdot V/W \\
[f \cdot \phi \cdot X \cdot \psi \cdot Y \cdot Z]_B & \equiv [f \cdot \phi \cdot X \cdot \psi \cdot Y \cdot Z]_B \cdot V \\
[f \cdot R_{\phi} \cdot X \cdot \psi \cdot Y \cdot Z/W] & \equiv [f \cdot R_{\phi} \cdot X \cdot \psi \cdot Y \cdot Z/W]_{B/W} \cdot V/W.
\end{align*}
\]

In addition there is the inward equivalence rule:

\[
\begin{align*}
[f \cdot _b \cdot X \cdot L_{\psi} \cdot Y/X \cdot Z]_{V/X} & \equiv [f \cdot _b \cdot X \cdot L_{\psi} \cdot Y/X \cdot Z]_{V/X} \cdot V \\
[f \cdot _b \cdot X \cdot L(R \psi) \cdot Y/X \cdot Z/W]_{V/W/X} & \equiv [f \cdot _b \cdot X \cdot L(R \psi) \cdot Y/X \cdot Z/W]_{V/W/X} \cdot V/W \\
[f \cdot \phi \cdot X \cdot \psi \cdot Y \cdot Z]_B & \equiv [f \cdot \phi \cdot X \cdot \psi \cdot Y \cdot Z]_B \cdot V \\
[f \cdot R_{\phi} \cdot X \cdot \psi \cdot Y \cdot Z/W] & \equiv [f \cdot R_{\phi} \cdot X \cdot \psi \cdot Y \cdot Z/W]_{B/W} \cdot V/W.
\end{align*}
\]
Finally, there are equivalences between left- and right-branching paths. There are the following axiomatic equivalences since (34) holds:

\[(33)\]
\[a. \quad [\rho \, [R_{\psi} \, X \, Y / Z]_{V / Z} \, Z]_{V} \equiv [\psi \, X \, [\rho \, Y / Z \, Z]_{V}]_{V} \]
\[b. \quad [\phi \, [b \, X \, Y / X]_{Y} \, Z]_{V} \equiv [b \, X \, [\phi \, Y / X \, Z]_{V / X}]_{V} \]

\[(34)\]
\[a. \quad f (R \, \psi \times y \, z) = R \, \psi \times y \, z = \psi \times (y \, z) =
\quad \psi \times (f \, y \, z) = \psi \times (y \, z) \]
\[b. \quad \phi (b \times y \, z) = \phi (y \times x) =
\quad b \times (L \, \phi \, y \, z) = L \, \phi \, y \, z \times x = \phi (y \, x) \, z \]

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.

\[(35)\]
\[\phi \, [\psi \, X \, Y]_{A} \, Z]_{V} \quad \equiv \quad [\phi \, X \, [\psi \, Y \, Z]_{B}]_{V} \rightarrow \]
\[\equiv \quad [R_{\phi} \, X \, [R_{\psi} \, Y \, Z / W]_{B / W}]_{V / W} \]
\[\equiv \quad [M_{\phi} \, [R_{\psi} \, X \, Y / W]_{A / W} \, Z]_{V / W} \]
\[\equiv \quad [M_{\phi} \, [M_{\psi} \, X / W \, Y]_{A / W} \, Z]_{V / W} \]
\[\equiv \quad [P_{\phi} \, [R_{\psi} \, X \, Y / W]_{A / W} \, Z / W]_{V / W} \]
\[\equiv \quad [P_{\phi} \, [M_{\psi} \, X / W \, Y]_{A / W} \, Z / W]_{V / W} \]
\[\equiv \quad [M_{\phi} \, [P_{\psi} \, X / W \, Y / W]_{A / W} \, Z]_{V / W} \]
\[\equiv \quad [P_{\phi} \, [L_{\psi} \, X / W \, Y]_{A / W} \, Z / W]_{V / W} \]

Computer enumeration of instances suggests that this axiomatisation of equivalence is complete. The parsing strategy to be described will not be incorrect if the axiomatisation is incomplete, however if some equivalences are missed, the strategy will not be optimal.

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 \(\pi_l\) and \(\pi_r\) stand for left- and right-branching local paths, and \(\#(\pi)\) is the number of metarule applications in \(\pi\), the ordering is defined by:
\[(36) \quad \pi'_1 < \pi_1 \text{ iff } #(\pi'_1) \text{ is less than } #(\pi_1)\]
\[\pi'_r < \pi_r \text{ iff } #(\pi'_r) \text{ is less than } #(\pi_r)\]
\[\pi_1 < \pi_r \text{ iff } #(\pi_1) \text{ is less than or equal to } #(\pi_r)\]

Then the parser will apply rules subject to the condition that they are not \textit{un nec}:

\[(37) \quad \text{un nec}(\pi) \text{ iff } \exists \pi' \text{ such that } \pi' = \pi \text{ and } \pi' < \pi\]

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:

\[(38) \quad \text{john} \quad := j \quad : \text{np.}\]
\[\text{will} \quad := \text{will} \quad : \text{\$np/(\$np).}\]
\[\text{leave} \quad := \text{leave} \quad : \text{\$np.}\]
\[\text{tomorrow} \quad := \text{tomorrow} \quad : \text{\$np}\text{/(\$np)}\]

The main procedure in the parser is \textit{prs1(+String,+Pos)} which parses \textit{String} from the current chart, and position \textit{Pos}:

\[(39) \quad \text{prs1([Word|String],Pos) :-}\]
\[
\text{Word} := \text{T:LexCat,}\]
\[
\text{incorp(Pos,LexCat,s(Pos),T);}\]
\[
\text{prs1(String,s(Pos)).}\]

The procedure \textit{incorp(+MPos,+RCat,+RPos,RTrans)} incorporates into the chart between positions \textit{MPos} and \textit{RPos} an edge of category \textit{RCat} and translation \textit{RTrans}:
(40) 
\[
\text{incorp}(\text{MPos}, \text{RCat}, \text{RPos}, \text{RTrans}) :- \\
\text{assert}(\text{edge}(\text{MPos}, \text{RCat}, \text{RPos}, \text{RTrans})), !, \\
\text{edge}(\text{LPos}, \text{LCat}, \text{MPos}, \text{LTrans}), \\
\text{incorp2}(\text{LPos}, \text{LCat}, \text{RCat}, \text{RPos}, \text{LTrans}, \text{RTrans}).
\]

The procedure \textit{incorp2}(+\text{LPos}, +\text{LCat}, +\text{RCat}, +\text{RPos}, +\text{LTrans}, +\text{RTrans}) tries to reduce edges of category \textit{LCat} and \textit{RCat} and translation \textit{LTrans} and \textit{RTrans}, between \textit{LPos} and \textit{RPos}, and causes precipitation of consequential edges. In the case that \textit{LCat} is a coordinator, a left-hand conjunct is sought.

(41) 
\[
\text{incorp2}(\text{LPos}, \text{LCat}, \text{RCat}, \text{RPos}, \text{LTrans}, \text{RTrans}) :- \\
\text{rule}(\text{LCat}, \text{RCat}, \text{MCat}, \text{G}), \\
\langle \neg \neg \text{unnec}_l(G, \text{LTrans}), \\
\neg \neg \text{unnec}_r(G, \text{RTrans}), \\
\text{incorp}(\text{LPos}, \text{MCat}, \text{RPos}, \text{G}'\text{LTrans}'\text{RTrans}).
\]

\[
\text{incorp2}(\text{Pos,crd,Cat}, \text{RPos}, \text{C}, \text{RTrans}) :- \\
\text{edge}(\text{LPos}, \text{Cat}, \text{Pos}, \text{LTrans}), \\
\text{coord}(\text{Cat}, \text{S}), \\
\text{incorp}(\text{LPos}, \text{Cat}, \text{RPos}, \text{S}'\text{LTrans}'\text{C}'\text{RTrans}).
\]

The coordination is limited to categories resulting in \textit{S}; (for these cases the simple semantics of Partee and Rooth 1983 can be employed):

(42) 
\[
\text{coord}(s, c). \\
\text{coord}(X/-, 'A'f'S) :- \\
\text{coord}(X, S). \\
\text{coord}(X/-, 'A'b'S) :- \\
\text{coord}(X, S).
\]
(43) 
rule(X/Y,Y,X,I).
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,rv
n,snp\snp\snp\snp]),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 unnc_r/2 and unnc_l/2, and those defining equivalence and counting metarule applications, are trivial but tedious and are omitted 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.\footnote{Perhaps that is no bad sign.} 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 paradigm, 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 consequent 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 emerge from the results of the thesis. Section 1 centres discussion around syntactic complexity, and Section 2, meaning representation.
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:

(1) A review of a book \( e_i \) just came out [which Chomsky wrote] \( e_i \)

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) \[
\begin{array}{c}
\text{a review of a book} \\
\text{NP/N N N\textbackslash N/N NP/N N} \\
\text{r_f} \\
\text{N/ (N/N)} \\
\text{f} \\
\text{N/N} \\
\text{-Mb} \\
\text{Rf} \\
\text{NP/ (N/N)}
\end{array}
\]

(3) \[
\begin{array}{c}
\text{a review of a book} \\
\text{NP/N N N\textbackslash N/N NP/N N} \\
\text{r_f} \\
\text{N/ (N/N)} \\
\text{Rf} \\
\text{NP/ (N/N)} \\
\text{-Rf} \\
\text{N\textbackslash N/ (N\textbackslash N)} \\
\text{-Rf} \\
\text{N/ (N/N)} \\
\text{-Rb} \\
\text{NP/ (N/N)} \\
\text{Rf}
\end{array}
\]

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] \( e_i \) I saw the girl who swims \( e_i \)

In the dominant reading the fronted adverbial modifies the "seeing", in the subordinate one,}

\[3\text{I am grateful to Elisabet Engdahl for drawing my attention to this example in connection with the present discussion.}\]
it modifies the "swimming". Accordingly, analysis of the clause in the former case is less expensive than in the latter case:

(5) I saw the girl who swims

NP    S\NP/NP    NP/N    N    N/N\N/(S\NP)    S\NP
       -----------Fr_f
       S\NP/ADV/NP  N\N
       --------------b
       N
       --------------f
       NP
       --------------f
       S\NP/ADV
       --------------Rb
       S/ADV

(6) I saw the girl who swims

NP    S\NP/NP    NP/N    N    N/N\N/(S\NP)    S\NP
       -----------Rb
       S/NP
       --------------Rf
       N/(S\NP)
       --------------Rf
       S/N
       --------------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 combinatorial logic. The feature which I want to emphasise here is that syntactic structures are isomorphic to terms of a directionally typed combinatorial logic. Thus under the simple applicative analysis (7), John met Mary is to have the combinatorial logic translation (8).

(7) \[ b \, John_{NP} \, [f \, met_{S\NP/NP} \, Mary_{NP}]_{S\NP} \]

(8) b John' (f met' Mary')

I will illustrate the way in which the binary grammar for English induces induces a combinatorial 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:

\[(9)\]
\[
\begin{align*}
&\text{a. } f \text{ is a combinator of type } X/Y + Y \Rightarrow X \\
&\text{b. } b \text{ is a combinator of type } Y + X\neg Y \Rightarrow X \\
&\text{c. } \text{If } \phi \text{ is a combinator of type } X + Y \Rightarrow Z, \text{ then} \\
&\quad \text{R}_\phi \text{ is a combinator of type } X + Y/W \Rightarrow Z/W \\
&\quad \text{L}_\phi \text{ is a combinator of type } X\neg W + Y \Rightarrow Z\neg W \\
&\quad \text{M}_\phi \text{ is a combinator of type } X/W + Y \Rightarrow Z/W \\
&\quad \text{P}_\phi \text{ is a combinator of type } X/W + Y/W \Rightarrow Z/W
\end{align*}
\]

Then the language of combinatory logic (CL) terms is defined thus:

\[(10)\]
\[
\begin{align*}
&\text{a. } \text{If } \alpha \text{ is a constant of category } X, \text{ then} \\
&\quad \alpha \text{ is a CL-term of category} \\
&\text{b. } \text{If } \phi \text{ is a combinator of type } X + Y \Rightarrow Z, \text{ and} \\
&\quad \alpha \text{ is a CL-term of category } X, \text{ and} \\
&\quad \beta \text{ is a CL-term of category } Y, \text{ then} \\
&\quad \phi \alpha \beta \text{ is a CL-term of category } Z
\end{align*}
\]

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).

\[(11)\]
\[
\begin{align*}
&\text{a. } \text{b John'} \ (f \text{ loves'} \ (f \text{ some'} \ \text{woman'})) \\
&\text{b. } f \ (R \text{ b John'} \ \text{loves'}) \ (f \text{ some'} \ \text{woman'})
\end{align*}
\]

\[(12)\]

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 committing, 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 Bentham (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)    \begin{align*}
    \text{John} & := j_e : \text{NP} \\
    \text{loves} & := \lambda x \lambda y \text{[LOVE}(y,x)\text{]}_{e \rightarrow e \rightarrow X} : \text{SNP/NP} \\
    \text{someone} & := \lambda P \exists x [P \text{x}]_{(e \rightarrow \lambda \rightarrow \lambda)} : \text{NP}
\end{align*}

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)    \begin{align*}
    \alpha_{e \rightarrow A \rightarrow \lambda} & \rightarrow \lambda \forall \lambda Y [V (\lambda a (\alpha \rightarrow a \rightarrow Y))]_{(e \rightarrow \lambda \rightarrow \lambda) \rightarrow A \rightarrow \lambda}
\end{align*}

The rule assigns a second type to loves’ thus:
(18) \[ \text{loves}' = \]
\[ \lambda x \lambda y [\text{LOVE}(y,x)]_{e \to e \to \text{t}} \to \]
\[ \lambda V \lambda Y [V (\lambda a [\lambda x \lambda y [\text{LOVE}(y,x)] a Y)]_{((e \to \text{t}) \to e \to \text{t}) \to e \to \text{t}} = \]
\[ \lambda V \lambda Y [V (\lambda a [\text{LOVE}(Y,a)])_{((e \to \text{t}) \to e \to \text{t}) \to e \to \text{t}} \]

So \textit{loves}' \textit{someone}' \textit{John}' can evaluate as follows:

(19) \[ \text{someone}' = \]
\[ \lambda P \exists x [P \, x] \]
\[ \text{loves}' \textit{someone}' = \]
\[ \lambda Y \exists x [\text{LOVE}(Y,x)] \]
\[ \text{loves}' \textit{someone}' \textit{John}' = \]
\[ \exists x [\text{LOVE}(j,x)] \]

For \textit{everyone} \textit{loves} \textit{someone} the function-argument structure is (20).

(20) \[ \text{loves}' \textit{someone}' \textit{everyone}' \]

Another type-shifting rule, such as (21), is required to assign \textit{loves}' a type capable of applying to the subject.

(21) \[ \text{For SNP/NP}, \]
\[ \alpha_{B \to e \to \text{t}} \to \lambda X \lambda U [U (\lambda b [\alpha \, X \, b])]_{B \to ((e \to \text{t}) \to e \to \text{t}) \to \text{t}} \]

Thus one reading is obtained as follows:

(22) \[ \text{loves}' = \]
\[ \lambda x \lambda y [\text{LOVE}(y,x)] \to \]
\[ \lambda V \lambda Y [V (\lambda a [\text{LOVE}(Y,a)])] \to \]
\[ \lambda X \lambda U [U (\lambda b [\lambda V \lambda Y [V (\lambda a [\text{LOVE}(Y,a)])] X \, b])] = \]
\[ \lambda X \lambda U [U (\lambda b [X (\lambda a [\text{LOVE}(b,a)])])] \]
\[ \text{loves}' \textit{someone}' = \]
\[ \lambda U [U (\lambda b \exists x [\text{LOVE}(b,x)])] \]
\[ \textit{everyone}' = \]
\[ \lambda P \forall y [P \, y] \]
\[ \text{loves}' \textit{someone}' \textit{everyone}' = \]
\[ \forall y \exists x [\text{LOVE}(y,x)] \]

The wide scope for object reading is obtained by applying the type shifting rules the other way around:
(23) \[
\text{loves'} = \\
\lambda x \lambda y [\text{LOVE}(y, x)] \rightarrow \\
\lambda x \lambda U [U (\lambda b [\text{LOVE}(b, x)])] \rightarrow \\
\lambda V \lambda Y [V (\lambda a [Y (\lambda b [\text{LOVE}(b, a)])])] \\
\text{loves' someone'} = \\
\lambda Y \exists x [Y (\lambda b [\text{LOVE}(b, x)])] \\
\text{loves' someone' everyone'} = \\
\exists x \forall y [\text{LOVE}(y, x)]
\]

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

---

\(^1\)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.
input. Abstracting over inputs of the same size, we can speak of that function which gives 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 \( c \cdot g(n) \); 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 \(<\text{current-state}, \text{current-symbol}> \) pairs into \(<\text{symbol-written}, \text{direction-moving}, \text{new-state}> \) 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 it 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 \(<\text{current-state, current-symbol}>\) pairs into \(<\text{symbol-written, direction-moved, new-state}>\) 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 \(NP\), 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.

\[
(x \lor y) \land (\neg x \lor y \lor \neg z \lor z) \land z
\]

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 \( f(lx) \) states and \( f(lxl) \) tape cells. A number of Boolean variables not larger than a polynomial function of \( lx \) 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:

(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 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 \( X/Y, X\backslash Y \) are categories.

The interpretation of the categories is provided by the following rules:

(4) \( X/Y + Y \Rightarrow X \)

\( Y + X\backslash Y \Rightarrow X \)

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 \( \{ V, \&, x, y, z, ..., \neg x, \neg y, \neg z, ... \} \) of

---

\(^3\) 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).
basic expressions, and the following lexical assignments:⁴

(5) \[ \begin{align*}
V & := S/L/L/L \\
& \quad \& := SS/S \\
x, y, z & := L \\
\neg x, \neg y, \neg z & := L
\end{align*} \]

For example \( V \neg x \ y \ z \ & V \ w \neg w \ y \) 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) \[ \begin{align*}
\alpha/\beta + \beta' & \Rightarrow \alpha' \\
\beta' + \alpha/\beta & \Rightarrow \beta'
\end{align*} \]

This means the same as before except that \( \alpha' \) is \( \alpha \) after \( \beta' \) has been unified into \( \beta \). For example unifying (8) into the structure right of the slash in (9) leaves the structure left of the slash (10).

(8) \[
\begin{bmatrix}
\text{CAT} & S \\
G & \begin{bmatrix}
x & 0 \\
z & 1
\end{bmatrix}
\end{bmatrix}
\]

⁴As before a left-associativity convention is assumed here so that, e.g. \( S/L/L/L \) is understood to be structured \( ((S/L)/L)/L \).

⁵Note 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.
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:
\[(11) \quad V :=
\begin{align*}
\text{[CAT S, G X]} & \text{[CAT L, TVAL 0, G X]} \text{[CAT L, TVAL 0, G X]} \text{[CAT L, TVAL 1, G X]}, \\
\text{[CAT S, G X]} & \text{[CAT L, TVAL 0, G X]} \text{[CAT L, TVAL 1, G X]} \text{[CAT L, TVAL 0, G X]}, \\
\text{[CAT S, G X]} & \text{[CAT L, TVAL 0, G X]} \text{[CAT L, TVAL 1, G X]} \text{[CAT L, TVAL 1, G X]}, \\
\text{[CAT S, G X]} & \text{[CAT L, TVAL 1, G X]} \text{[CAT L, TVAL 0, G X]} \text{[CAT L, TVAL 0, G X]}, \\
\text{[CAT S, G X]} & \text{[CAT L, TVAL 1, G X]} \text{[CAT L, TVAL 0, G X]} \text{[CAT L, TVAL 1, G X]}, \\
\text{[CAT S, G X]} & \text{[CAT L, TVAL 1, G X]} \text{[CAT L, TVAL 1, G X]} \text{[CAT L, TVAL 1, G X]}
\end{align*}
\]

Note that at least one literal must be true, and that all value assignments are unified. The value assignments are also unified in:

\[(12) \quad \& := \text{[CAT S, G X]} \text{[CAT S, G X]} \text{[CAT S, G X]}
\]

Finally for each positive literal \(v\) and negative literal \(\neg v\) in the problem instance, the following lexical entries are added:

\[(13) \quad v := \text{[CAT L, TVAL 0, G [V 0]]}, \text{[CAT L, TVAL 1, G [V 1]]}
\]
\[\neg v := \text{[CAT L, TVAL 1, G [V 0]]}, \text{[CAT L, TVAL 0, G [V 1]]}\]

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 \(A, B, C, \ldots\) of each variable occurring in the 3SAT instance, say in alphabetic order. Then the lexical categories will be
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 \(\lambda\)-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).\(^6\)

\[
X/Y: \ x \ + \ Y: \ y \Rightarrow \ X: \ x \ y
\]

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).

\[
(18) \quad \alpha/\beta + \beta' \Rightarrow \alpha'
\]

where \(\alpha'\) is the result of applying to \(\alpha\) the most general unifier of \(\beta\) and \(\beta'\)

Here, \(\alpha\) and \(\beta\) are expressions of the sign language. To keep the association with (17) the meaning of \(\alpha'\) must be the result of applying that of \(\alpha/\beta\) to that of \(\beta'\), so that the sign language must have a set-theoretic semantics where the following holds:

\[
(19) \quad \mu\text{-Reduction}
\]

\[
[[\alpha/\beta \ \beta']] = [[\alpha']]
\]

where \(\alpha'\) is the result of applying to \(\alpha\) the most general unifier of \(\beta\) and \(\beta'\)

Thus the ‘/’ in the sign language is a functional abstraction operator rather like the ‘\(\lambda\)’ in the lambda-calculus. In particular, note that \(\beta\)-reduction is a special case of \(\mu\)-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 term.\(^7\)

\[
(20) \quad \beta\text{-Reduction}
\]

\[
[[\lambda\alpha \ \beta]] = [[\alpha']] \quad \text{where} \quad \alpha' \text{ is the result of applying to } \alpha \text{ the substitution } \{v=\beta\}
\]

The reason why \(\beta\)-reduction is valid in the lambda-calculus is that a variable which, by the semantics of \(\lambda\)-abstraction and application is assigned \(\{[\beta]\}\), is replaced by \(\beta\) which, of course, denotes the same value. Since the semantics is strictly compositional, the operation

\(^6\)I gloss over slash directionality here.

\(^7\)The complications of scope are mentioned below.
preserves meaning.

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 $\beta$-reduction. Either the $\beta$-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 $\lambda$-abstraction holds, it should still be anomalous to instantiate abstractor variables. For example $X/X/X$ should not $\mu$-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 metavarules, 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 metavarule CG for English; uses
% path equivalence check

:- op(400,xfy,./).
:- op(400,xfy,\).

:- op(300,xfy,\).
:- op(500,xfy,:=).
:- op(400,xfy,equ).
:- op(450,xfy,;).

% 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\XY,X,b).

rule(X,Y/Z,V/Z,’R’’G) :-
  rule(X,Y,V,G).

rule(XZ,Y,VZ,’L’’G) :-
  rule(X,Y,V,G).

rule(X/Z,Y,V/Z,’M’’G) :-
  member(Z,[np,sp,n\n,n\p,\n\p(\n\p),n/n]),
  rule(X,Y,V,G).

rule(X/np,Y/np,V/np,’P’’G) :-
  rule(X,Y,V,G).

member(X,[XI\_]).
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 \( r(L''G1,L''H1) \) ::
\( l(G,H) \) equ \( r(G1,H1) \).
\( r(L''G1,L''H1) \) 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 \( r(M''G1,M''H1) \) ::
\( l(G,H) \) equ \( r(G1,H1) \).
\( r(M''G1,M''H1) \) 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,[_\{\{\}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([\{\{'GIL\}',\{'G1\}\{L1\}]),
    less_than_or_eq([\{'GIL\}',\{'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 : \$np\$(\$np\$/np.
and := & : crd.
bankrupt := bankrupt : n/n.
before := before : \$np\$(\$np\$/np.
bill := b : np.
company := company : n.
damaged := damaged : \$np\$/np.
dearly := dearly : \$np\$(\$np\$.
dislike := dislike : \$np\$/np.
dog := dog : n.
downstairs := downstairs : \$np\$(\$np\$.
filed := filed : \$np\$/np.
give := give : \$np\$/np\$/np.
i := i : np.
inside := inside : n/n.
john := j : np.
large := large : n/n.
laughs := laughs : \$np.
leave := leave : \$np.
left := left : \$np.
like := like : \$np\$/np.
long := long : n/n.
loves := loves : \$np\$/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(h'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([TTTs],[RTs|RTTs]) :-
    reduce(T,RT),
    reduce_list(TTs,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)'y' (b'you'leave)
y' (leave'we)' (leave'you)
String 12: [the,book,which,we,show,and,you,give,john]

np
\(f^n 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^n 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' (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'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
R'if' (R'f'the' (R'b'paper'which))' (R' (R'b)'you' (M' (R'b)'filed' (R'if'without'reading)))
R'if' (R'f'the' (R'b'paper'which))' (R' (R'b)'you' (M' (R'b)'filed' (R'if'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
R'if' (R'f' (R'f'the' (R'b'paper'which)))' (R' (R'b)'you' (R' (M'b)'filed'without)')'reading
R'if' (R'f'the' (R'b'paper'which))' (R' (R'b)'you' (R' (M'b)'filed'without)')'reading

np
R'if' (R'f'the' (R'b'paper'which))' (R' (M'f') (R' (R'b)'you' (M' (R'b)'filed'without)')'reading
R'if' (R'f'the' (R'b'paper'which))' (R' (M'f') (R' (R'b)'you' (M' (R'b)'filed'without)')'reading

np
np
String 24: [mary,thinks,that,theman,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'leave')you))' (the'man)))'m

Words: 11    Readings: 4    Time: 1.88 seconds

String 25: [theman,inside,thinks,that,thewoman,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))

Words: 11    Readings: 3    Time: 0.59 seconds

String 26: [that,man,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'Iwas'bankrupt'))
spread' (a' (rumour' (that' (was'bankrupt'))))
Words: 7 Readings: 1 Time: 0.21 seconds

String 28: [the, rumour, damaged, this, company, that, john, was, bankrupt]

\( s \) (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'))) )))

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))))))

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))))))
References


University Press: New Haven, Conn.


