# A Logical Model of Competence and Performance in the Human Sentence Processor

**Matthew Walter Crocker**

Ph.D.
University of Edinburgh
1992

# Abstract

This work is concerned with the way in which principled theories of syntax and modular theories of mind may participate in an incremental model of human linguistic performance. Central to current linguistic theory is the distinction between *competence*, what we know about language, and *performance*, how we use that knowledge. While current theories of grammar suggest a highly modular, abstract, language universal characterisation of linguistic competence, traditional models of performance have postulated parsing strategies based on construction-oriented, phrase-structure grammars. In contrast, we construct a principled theory of performance on the basis of cross-linguistic evidence, with the aim of shedding greater light on the relationship between grammar and processing.

On the basis of Fodor's *Modularity Hypothesis* and a range of empirical evidence, we assume the existence of a distinct syntactic processor within the human sentence processor. We further hypothesize *The Principle of Incremental Comprehension*, entailing that the sentence processor strive to achieve maximal incremental comprehension as each word in a sentence is encountered. To achieve global incremental comprehension at the various levels of linguistic processing (syntactic, semantic, etc.) we therefore predict that modules operate concurrently. We then suggest that the modularity paradigm is one to be exploited whenever possible, precisely because it permits distributed processing within a particular domain, thereby improving real-time performance. This maxim of modularity, combined with the natural partitioning of syntactic information into several informationally encapsulated representation types leads us to posit four sub-modules *within* the syntactic processor: (1) phrase structure, (2) chain structure, (3) coreference, and (4) thematic structure. We then present several processing strategies which are claimed to be operative within the phrase structure module and demonstrate how the proposed architecture successfully accounts for a variety of empirical phenomena, across several languages, suggesting both the cross-linguistic application and possibly innate status of the model. We further argue that the processing strategies follow from the more general Principle of Incremental Comprehension.

To demonstrate the operation of the proposed model, and illustrate its principled basis, we employ the *Algorithm = Logic + Control* paradigm of logic programming. In particular, we present a 'deductive' implementation wherein each module is realised as a specialised meta-interpreter which constructs its own representation with respect to the relevant principles of grammar. We then demonstrate how these interpreters may be 'coroutined' so as to model the concurrency of the performance theory. Finally, we see that the implemented model successfully accounts for subsets of both English and German, without any variation of the control strategy. We take this as further support for the existence of an innate, unparameterised sentence processing mechanism, and discuss some possible implications of this conclusion.

## Acknowledgements

First and foremost, I would like to thank my supervisors Elisabet Engdahl and Robin Cooper. I must especially thank Elisabet for her constant encouragement, guidance, and support — I feel very fortunate to have worked with her. I would also like to thank my two examiners, Eric Wehrli and Ronnie Cann, for their comments and suggestions.

Without doubt, I owe a great debt to various colleagues within the Edinburgh academic community. In particular, thanks must go to Ian Lewin and Martin Pickering, for many helpful discussions during the course of this research. I would also like to thank Alan Black, Kathrin Cooper, Suresh Manandhar, and Mike Reape — all of whom have contributed valuable comments and discussion during my time in Edinburgh. Additionally, I must acknowledge all those members of the Department of Artificial Intelligence, the Centre for Cognitive Science, and the Human Communication Research Centre who have contributed to one the richest environments in which to pursue this research.

Finally, I would like to acknowledge the financial support which made my study possible: an Edinburgh University Studentship, and Overseas Research Scheme Award, and a research assistantship at the Human Communication Research Centre. And last but not least, I owe a special thanks to the generosity of MFS Limited.

# Contents

# Chapter 1

# Introduction

The study of language is both a fundamental and paradigmatic programme of research in our efforts to develop a comprehensive theory of mind. This is to say, a theory of language will not only be of central importance to the more general theory, but it will also prove to be a valuable case study in terms of the methodologies and knowledge sources which are invoked during the investigation. The programme of research is inherently interdisciplinary: In addition to all facets of formal linguistics, various aspects of psychology, computer science, and the philosophy of mind are all brought to bear in pursuing a robust theory of language.

The notion of a distinct language faculty is fundamental to this enterprise; the assumption that language is in some sense modular, or autonomous, and is not governed directly by general cognitive processes. This view has been championed by Chomsky, who argues not only for linguistic autonomy, but also for a rich innate structure as part of the human genetic endowment [Chomsky 80]. The primary motivation for this stance rests on the *poverty of stimulus* argument: how do we come to know so much, given the relative poverty of our environment. As a potential solution to this problem, Chomsky has hypothesised the existence of a *universal grammar* (UG): a system of principles which are common to all possible human languages, a characterisation of our genetically determined language faculty. As Chomsky observes, this position demands empirical inquiry, and in this regard he has identified the following fundamental concerns [Chomsky 86b, page 3]:

(1) (i) What constitutes knowledge of language?
(ii) How is knowledge of language acquired?
(iii) How is knowledge of language put to use?

The first question demands an explicit theory of grammar which characterises the *state of mind* of a person competent in a particular language. The second question is concerned with how a person comes to have this knowledge based on their linguistic experience. Finally, the third question asks how our knowledge of language enters into the task of linguistic communication: by what means do people bring their linguistic knowledge to bear during the comprehension and expression of individual utterances?

The observation of certain regularities — within and across languages — combined with seemingly systematic variation, is instrumental in upholding the notion of UG. As a result, linguistic inquiry has experienced a change in emphasis from the superficial description of linguistic behaviour, the 'external' or E-language, to study of the 'internal' I-language: that knowledge of language which constitutes the state of mind. More specifically, linguistic theory has moved from sets of rules which describe specific linguistic constructions, towards the development of a system of principles, fundamental to human language, and the identification of the parameters which determine the scope of linguistic variation. Insofar as these fundamental principles constitute a theory of universal grammar, they provide an initial state for the acquisition process. Under this view, a theory of language acquisition must characterise the way in which parameters of variation are instantiated by linguistic experience. In this regard, there may be constraints on the nature of parameters, as determined by the properties of early linguistic exposure, such as the lack of negative evidence — i.e. parameters must be determinable on the basis of positive evidence only.

The shift from descriptive linguistics has been motivated, therefore, by a desire to explain basic problems of language acquisition and in so doing shed greater light on the nature of the human language faculty in general. Crucially, however, the theory of linguistic competence need only concern itself with the *logical problem of language acquisition*, i.e. the formal properties of the acquisition process which are relevant to linguistic competence. This is to say, the theory of competence need not take into consideration the *manner* in which acquisition occurs, it must only ensure that the grammar is 'acquirable' given the constraints of linguistic experience [Wexler & Culicover 80].

The impact of this shift on linguistic theorising tells us that there is something to be gained in examining the conjunction of the concerns outlined in (1). While it is possible to isolate the issues of competence, acquisition, and performance it is still necessary to ensure compatibility of the respective theories at their interfaces. Indeed, the theory of syntax has been substantially influenced by the requirements of acquisition. This naturally leads us to pose the questions: What is the relationship between knowledge of language and its use? To what extent is one dependent on the other? What aspects of language are functionally determined? Is it the case that we can identify the *logical problem of linguistic performance* — some set of basic constraints of language processing which must be met by a model of syntax? The only way to address this issue is to begin constructing theories of performance with respect to the current syntactic theory.

In this thesis, we endeavour to construct a model of human linguistic performance which is based directly upon the principles and representations of the current *principles and parameters* model of grammar which has arisen out of the theories of *Government and Binding* [Chomsky 81b], [Chomsky 86b] and *Barriers* [Chomsky 86a] as developed by Noam Chomsky and others. The intent is to show that it is possible to construct a natural, principle-based process model and to address the more interesting theoretical issue of what properties such a model must have. Finally, we will address the issue of the innateness and universality of the sentence processing mechanism, and turn our attention to possible accounts of the relationship between parser and grammar.

## 1.1 The Competence-Performance Distinction

Before we proceed, it is worth taking some time to discuss the implications of the *a priori* distinction between linguistic competence — our knowledge of language, and performance — how we use this knowledge. Indeed, there is much confusion in the literature about whether or not such a distinction is necessary, desirable, or even meaningful. In fact, the distinction is a formal property of any processing system: inherent to any process is both a declarative semantics, a characterisation of *what* the process computes, and an operational semantics, a specification of how the declarative specification is used, thereby realising a particular algorithm. That is, the process is characterised by the conjunction of a declarative specification and some operational semantics. Crucially, however, we can vary the operational semantics to produce any number of algorithms all having the same declarative semantics. That is, all such algorithms will compute the same relation, but in different ways.

Let's consider, for example, how to construct a parser for some simple fragment of English. To begin with, we create a context-free grammar (CFG) which constitutes a declarative specification of the fragment. Given these rules of grammar, there are any number of operational semantics (roughly, parsing algorithms) which may be invoked to create an actual parser. The various parsers may construct an analysis of a sentence top-down or bottom-up, left-to-right or from the inside out, but crucially each parser has the same declarative semantics represented by our original CFG. Suppose now that you give one of these parsers to someone else (such that they can only run it, not examine the actual program code), and charge them with the task of reproducing it — i.e. constructing an equivalent program. This amounts roughly to the task with which we are faced in our endeavour to characterise the human sentence processor. In tackling this problem we could begin by trying to write an actual program which mimicked the behaviour of the original. As Pylyshyn points out, however, there is a good chance that this route will lead to the construction of ad hoc systems whose performance is only superficially similar to that of the original program [Pylyshyn 84, page 85], not a faithful model of it. As Chomsky further points out:

> " If we knew only that language consists of words, our performance models would necessarily be very primitive and of restricted interest; we could study the sequence of linguistic signs and their formal and semantic properties, but nothing else."
>
> [Chomsky 80, page 225-226]

The alternative route is to pursue richer theories of linguistic competence — a declarative specification of the linguistic rules and representations — so that we may develop more accurate and interesting process models. More generally, such theories can help to inform us of the expressive power required to describe language[1]. Thus the study of formal syntax

[1] Witness, for example, the past decade's debate concerning the context-freeness of natural languages: While the current consensus is against the context-free hypothesis, such inquiry remains useful in its aim of determining the formal and computational power which is required for expressing linguistic theory (see for example [Postal 64], [Pullum & Gazdar 82], and [Shieber 85]).

is not in any way at odds with the more general goal of modelling the human language faculty, nor is it irrelevant. While a theory of grammar does not directly yield a processing model (as discussed above), it does give us greater insight into how such a model might be constructed.

There are several respects in which the contribution of syntactic theory to a theory of parsing might be weakened. In developing a theory of competence we may use varying degrees of abstraction to describe syntactic phenomena. This has been noted above where we observed the recent shift from large systems of language specific rules to a system of abstract, cross-linguistic principles, with particular parameters being set for specific languages. While it may be interesting to describe syntactic theories at the more abstract explanatory level, it may be the case that the human sentence processing mechanism (HSPM) makes use of an equivalent, or, *covering* grammar. As Berwick and Weinberg discuss (at some length), the parser may use a grammar which is *strongly* equivalent to that of the theory, in that it generates the same structural descriptions [Berwick & Weinberg 84]. As an example, the parser might use a transformed or compiled out version of the original theory[2]. Indeed the parser might use a *weakly* equivalent grammar which yields somewhat different structural descriptions, which require a mapping function to relate them to the descriptions of our original grammar. One reason the sentence processor might use a covering grammar is to improve efficiency.

To summarise, the distinction between competence and performance is not so much necessary as it is inherent to any process, and it is most certainly meaningful given its underpinnings in formal computational semantics. But perhaps most importantly it is desirable, in that it permits us to richly characterise what the language processor does, so that we may construct more interesting, accurate and principled process models. This modularisation of the research programme allows a 'divide and conquer' strategy which seems invaluable given how little we know about the mind (see [Berwick & Weinberg 85, chapter 2], for in-depth discussion of this methodology). The *Strong Competence Hypothesis*[3] holds that the process model must make direct use of the principles of grammar (as defined in the theory): what Berwick and Weinberg call *type transparency*. As the discussion above identifies, this condition need not hold: the *Weak Competence Hypothesis* is satisfied even where the process model is based on some equivalent version of the grammar. However even if strong competence is upheld, it is important to note that there exist many possible parsing strategies which satisfy it.

In the context of the principles and parameters theory of grammar we have assumed here, I will take the strong competence hypothesis to be synonymous with the notion 'principle-based': A model which uses the principles of grammar directly in its recovery of a syntactic analysis. I.e. it does *not* use a compiled-out, transformed, or covering grammar. We may, however, go one step further: As we shall see there are a variety of performance theories which may be considered principle-based, in that they use the principles of grammar on-line, but which make decisions on the basis of 'non-linguistic' criteria, such as computational

[2] This is illustrated by the Marcus parser [Marcus 80], which computes roughly an annotated surface structure, including antecedent-trace relations. The parser does this without making explicit use of the grammar principles, but obeying them nonetheless.

[3] We use this term in the spirit of its original formulation (as the *Competence Hypothesis*) in [Bresnan 82].

efficiency or representational complexity. Such models clearly contrast with theories of processing which operate according to grammar-based strategies, thereby suggesting an even closer relationship between parser and grammar. To the extent that a theory of performance is both principle-based and incorporates strategies which are grammar-based (in this sense just described), we will say that it is 'strongly principle-based'.

## 1.2 The Universal Parser

If we assume that the UG hypothesis is essentially correct, then there are two possible scenarios concerning the origins of the human sentence processor: (i) people develop a sentence processor from scratch, or (ii) we have some innate sentence processing mechanism as part of our linguistic inheritance. The first option suggests a relatively uninteresting relationship between the grammar and processor, where the mind simply learns procedures for analysing the language being acquired, subject to basic limitations on computational resources. It also suggests that human linguistic performance might vary quite radically across languages, since processing strategies could be tailored to the surface structures of the particular language acquired. Further, this position precludes the possibility of a close parser-grammar relationship, where some aspect of UG might actually be the result of processing limitations. Finally, all the acquisition arguments invoked to motivate UG naturally extend to linguistic performance, possibly even more so: In the first place, if we assume that parameter setting is performed on the basis of the 'structure' of early linguistic input, then children must be equipped with the necessary machinery to assemble such structures on the basis of UG. In addition, children receive no cues from the linguistic data as to how such processing might be accomplished. Thus it seems that we must be innately endowed with at least some rudimentary facility for processing early linguistic data, and consequently this facility must further be capable of analysing the range of possible natural languages.

This latter position raises a number of interesting possibilities. Is the innate sentence processor just some general parsing mechanism, which becomes more specialised and efficient during and after the acquisition stage? Is it specialised for UG-possible grammars? Are any fundamental aspects of processing parameterised, say, in tandem with parameters in the grammar or does the basic processing machinery hold constant for all languages? Finally, the ultimate question pertaining to the relationship between knowledge of language and its use: are any aspects of the innate sentence processor which are independently determined — say, by constraints on efficiency or memory limitations — reflected in our linguistic competence? Simply put, to what extent, if any, is our linguistic competence determined by aspects of performance? One striking empirical prediction of this position is that, just as principles of grammar apply cross linguistically, so should any innate aspects of the sentence processing mechanism, assuming limited parameterisation of the HSPM (see [Frazier & Rayner 88] and [Mazuka & Lust 90] for discussion).

## 1.3 A Programme of Research

Ultimately, investigation of the human sentence processing mechanism will characterise the process by which our knowledge of language is brought to bear in the task of linguistic communication. Just as the desire for explanatory adequacy has influenced modern syntactic theorising, so should it be addressed by any theory of performance which takes the Universal Grammar hypothesis and the problem of language acquisition seriously. As outlined in the previous section, the notion of UG as realised by the principles and parameters approach raises interesting questions concerning the parser-grammar relationship:

(2)
1. Given the abstract, language universal nature of UG, is it the case the the HSPM uses the grammar directly, or does is some 'compiled-out' or covering grammar developed during/after acquisition?
2. Is the HSPM also innate (possibly modulo some parameterisation), or is it developed from scratch?
3. If the HSPM is both principle-based and innate, is it possible that aspects of the grammar are artefacts of the processing mechanisms limitations?

We have suggested that learnability might pose a problem for acquiring the HSPM from scratch, since at least some knowledge of how to process linguistic input will be required to boot-strap the acquisition process. In some regards this argument is even more compelling with regard to performance than competence, since children receive no obvious cues from their linguistic experience about how to parse sentences. It is also uncontroversial that an innate HSPM must also be language independent. However, even if UG and some *Universal Parser* (UP) form the initial state of the language acquisition device, there exists the possibility that maturation[4] of the language processor leads to the transformation of the grammar into some more efficiently computable form, thereby altering both parser and grammar.

In this thesis we provide a model of processing which is consistent with the position that the human sentence processor is both innate and uses the principles of grammar directly. Insofar as the innateness hypothesis is justifiable on grounds of the *poverty of stimuli*, we further suggest that this position is the theoretically stronger one, since no subsequent transformation of the innate grammar and parser is required. This in turn might be taken as an indication that the innate HSPM is sufficiently 'optimised' for efficiency (since there is no need to alter the basic innate grammar or parser). If this is so, then it is reasonable to suggest that the principles of grammar may be used in a manner which is efficient for processing, and have perhaps evolved[5] precisely along these lines, i.e. the grammatical principles have at least conformed to some requirements of the processing machinery, thus suggesting a close relationship in the evolution of the both grammar and parser. On the basis of the model we develop, we will endeavour to show that this view can be maintained.

[4] This is distinct, but possibly related to, the maturation of principles of grammar during the course of acquisition as proposed by Borer and Wexler (see [Borer & Wexler 87], [Borer & Wexler 88]).

[5] We will not diverge here into a discussion of how UG and possibly UP has come to be part of the human genetic endowment, the interested reader is referred to [Lightfoot 82].

Furthermore, the proposed model does not involve any 'parameterisation' of the HSPM, at least for the languages considered here. In chapter 7 we will consider recent proposals suggesting that parameterisation is necessary, and show that it is possible to reconcile the relevant data with the current model, thus eschewing the need to explicitly parameterise the HSPM.

On the basis of a variety of empirical evidence we propose that, not only is the syntactic processor a distinct module within the language comprehension system, but also that it consists of several sub-modules, contained wholly within the syntactic processor. In addition to providing a robust account of the empirical data, we motivate the organisation on the theoretical grounds; Fodor's *Modularity Hypothesis* and the natural encapsulation of the principle-based grammar into distinct representational types which correspond naturally to the proposed modules of the process model. On the basis of the empirical evidence, we propose that the HSPM must obey an over-arching requirement of incremental interpretation, constructing a maximal, partial interpretation of a sentence as each lexical item is encountered. This entails that each module within the syntactic processor operate concurrently and incrementally.

Within the individual modules we further identify a number of processing strategies which are based on notions defined within UG, and demonstrate that these strategies hold for a number of languages (where relevant data is available), thereby providing a grammatical basis for the operation of the sentence processor, suggesting it is strongly principle-based. Furthermore, the proposed strategies can in fact be seen as derivative from the overriding requirement of maximal incremental interpretation, and external demands on the sentence processor. We will argue that this dual nature of the strategies forges the link between grammar and processor, and also suggest that the principles of grammar satisfy a number of requirements for incremental processing.

The theory of processing we develop is founded upon the notion of modularity, and the strict requirement of incrementality entails that modules operate concurrently. Furthermore, we assume the direct use of grammatical principles by the parser — a proposal which is considered controversial by much of the community. To illustrate the viability of these proposals we construct a computational model of these central aspects of the theory. We draw upon the *parsing as deduction* hypothesis of logic programming, which permits the transparent distinction and specification of linguistic competence, modular organisation, and performance. The advantage of this approach is that we can begin to model the theory in a top-down manner, without stipulating a particular architecture[6]. In this way we can identify aspects of processing which are *not* determined by the theory, and hence subject to future refinement and revision. Thus in some sense the model defines the class of possible parsers which are consistent with the theory; what we will dub *a logical model of competence and performance*.

[6] Although we will argue that the computational model may be most naturally realised on a platform which permits distributed concurrent processing.

## 1.4 Organisation of the Thesis

In the next chapter, we undertake to summarise the current state of research in sentence processing. We will outline both the empirical evidence relevant to our discussion, and the range of theories which have arisen. Throughout, we will highlight how existing proposals contribute to the approach assumed here, we critically evaluate proposals with respect to both their empirical coverage and grammatical basis, and sketch the directions we will take. In chapter 3, we examine the principles and parameters model of grammar, and suggest a reformulation of the traditional T-model, making use of representational *types* rather than *levels*. While the revised model is essentially equivalent to the T-model, we will argue that it is a more relevant characterisation for the purposes of the process model. In chapter 4, we will motivate a particular model of sentence processing on the basis of our conclusions in chapters 2 & 3. We propose both a modular organisation of the syntactic processor, and describe the operation of the individual modules and strategies therein. In chapter 5 we discuss the computational model of the theory: First we outline the general issues in the implementation of principle-based systems, and then motivate use of the *parsing as deduction* paradigm of logic programming, and illustrate how the modularity and incrementality assumed in the theory may be naturally achieved. In chapter 6 we continue with some discussion of the individual modules. In chapter 7, we discuss the results of the present work, and situate them within the programme of research we have outlined above. The intent is to distinguish concrete results from the speculative and outline directions for future research. Chapter 8 summarises the thesis and its contribution to the study of language.

# Chapter 2

# Current Perspectives on Sentence Processing

A theory of human sentence processing is a characterisation of *how* people recover an interpretation for a given utterance. Ultimately such a theory should detail the processor's organisation, the representations employed, and the algorithms used to construct an analysis — where such a characterisation is isomorphic to that of the mind. Furthermore, we must situate the sentence processor with respect to the other cognitive systems with which it interacts; outlining the nature and degree of communication at all interfaces.

As with most scientific theorising, a primary source of information is empirical evidence: what behaviour do people exhibit when processing language? Given this data, we can make hypotheses about the strategies employed by the human sentence processor, and proceed by testing their predictive validity. Such accounts of human performance, however, can only be formulated in context: a theory of *how* we compute the analysis of an utterance must be defined in terms of *what* we are computing, and the computational resources available to perform the task. Thus, before we can begin with specific hypotheses we must address a number of general questions:

(3) (i) What knowledge sources are brought to bear by the HSPM?
(ii) What is the grammar used by the sentence processor?
(iii) What computational machinery is available to the processor?
(iv) What determines processing complexity?

The first point concerns the status of the sentence processor with respect to other aspects of the language comprehension system. Not only must we justify the existence of a distinct sentence processor, but we must also determine its informational vocabulary. The stance taken on this issue is fundamental to any theory of sentence processing, and is discussed in some detail in the following section. Indeed, the reason we have adopted the term 'sentence processor' thus far, is because of its relative neutrality: we wish to refer to that aspect of language processing which is based upon the syntactic, sentential unit, not the full range of discourse etc., without making any *a priori* claims about the domain of knowledge over which this processor presides.

Regardless of which other knowledge sources are involved in sentence processing, the lexicon and grammar — the linguistic competence of the speaker/hearer — must be of paramount importance, as they determine the basic linguistic concepts and the structural relationships which may hold between them. This is of even greater concern if we wish to claim that humans use the grammar directly, in accordance with the strong competence hypothesis. Current theories of sentence processing vary widely in terms of the grammars upon which they are based. Indeed, the success of a syntactic theory is often judged by its ability to explain aspects of linguistic performance.

Perhaps the most difficult of the above questions are the last two. We have at our disposal no articulated theory of the mind's computational architecture[1]. Is language processed serially; is the mind able to distribute tasks such that they are performed in parallel; or is there some other architecture for which we have no adequate characterisation? What are the memory limitations which constrain computation? And once we have an answer to these questions we must also determine what the relevant metrics for processing complexity are — are they determined by the complexity of representations held in memory, or the efficiency of the algorithms which recover the representations. To summarise: What is the nature of the mind's computational architecture, and what aspects of computation are easy or difficult?

In this chapter, we will first turn our attention to the hypothesis that there exists an autonomous sentence processor, since this is fundamental to the initiative as a whole. We will then have a general look at the range of available empirical evidence concerning human linguistic performance. These two sections are intended to provide us with a common starting point: the assumption of a modular sentence processor and the data which such a model must account for (or be consistent with). Following this we examine prevalent models of sentence processing, in an effort to determine their adequacy with respect to the empirical data. We will also try to distill the principled, explanatory aspects of these approaches, and identify the stipulative and unmotivated.

## 2.1 Modularity in Language Processing

As stated above, the programme of research we have undertaken is prefaced with the assumption that there exists a distinct sentence processing mechanism within the human language faculty. Indeed, the identification of a language faculty which is in some sense separate from other cognitive systems is not entirely uncontroversial. A thorough discussion on this latter point would take us too far afield, rather we will simply adopt the Fodorian position that *input systems* (the perceptual and linguistic faculties) are modules:

[1] Even if something like neural networks prove to be a good approximation of *low-level* mental architecture, there is reason to believe they are an inappropriate level at which to characterise the *high-level* symbol manipulation processes [Fodor & Pylyshyn 88]. For further discussion of *parallel distributed processing* models see [McClelland *et al* 89], [Mitchell 89b], and references cited therein.

> " ...We can abbreviate all this by the claim that input systems constitute a family of modules: domain-specific computational systems characterised by informational encapsulation, high-speed, restricted access, neural specificity and the rest."
>
> [Fodor 83, page 101]

In this section we will discuss the issue of syntactic autonomy: the hypothesis that there exists a distinct syntactic module within the language comprehension system. This has the appearance of a recursive application of Fodor's hypothesis. It remains to be seen, however, the extent to which the properties of these sub-modules are coextensive with those outlined by Fodor. Indeed, the psycholinguistic literature presents us with a broad range of positions concerning the modular status of the syntactic processor.

As we discussed in chapter 1, current linguistic theorising is motivated by the hypothesis of an innate language faculty as part of the human genetic endowment. We further discussed the existence of a corresponding sentence processing mechanism which is possibly also innate. Before we proceed, however, it is important to note that *processing* autonomy — i.e. the notion of a distinct processor for some particular input-output domain — may be quite separate from autonomous aspects of linguistic theory, which has been termed *formal autonomy* by [Crain & Steedman 85]. As a result, there are two possible positions: that of *modularity*, which holds that there exist functionally and informationally encapsulated processors for specific domains, and that of *interaction* which assumes that processes may operate across informational domains: a model which is perfectly consistent with the partitioning of these domains (e.g. syntax, semantics, etc.) at a formal level. Stated simply, the presence of modular structure in our theory of linguistic competence does not entail a corresponding modularity in the performance model, although this might be argued on the grounds of the (strong) competence hypothesis. That is, not only must the grammar be used directly, but so must the organisation of linguistic theory into informational domains (i.e. the (sub-) theories of syntax, phonology, etc.) be reflected by the process model. This is clearly the more attractive position.

Crain and Steedman illustrate the formal/processing autonomy distinction in their model which formally separates a categorial syntax from semantics, where there is a rule-to-rule mapping between the two [Crain & Steedman 85]. At a processing level, however, they suggest that the syntactic representation is never actually constructed, but rather defines the mechanism used to recover possible semantic representations[2]. In their model, Crain and Steedman assume that processing occurs incrementally and serially: as each item is encountered the syntax generates all possible interpretations, and the 'most contextually appropriate' of these is then selected by the subsequent semantic and pragmatic systems, such that only one possible analysis is entertained. As Crain and Steedman observe we may wish to call this *weak interaction*, since parsing is still driven largely by the syntax. Crucially however, non-syntactic processes are used to determine the direction the analysis will take in the face of ambiguity, *contra* the strong modular view (see [Fodor 83, Part V] for further discussion of this *weakly* modular position). The stronger form of interaction,

[2] In fact, contrary to their argument, a rule-to-rule syntax and semantics is not necessary for this type of architecture. In chapter 7, we will take up in some detail the issue of syntactic representations and their psychological reality in the context of a principle-based sentence processor.

however, would give semantic and context information equal status to the syntax in making the decision about which syntactic analysis to actually *propose* in the first instance[3]

These perspectives exemplify the range of possible permutations which may exist between informational (knowledge) and functional (how it is processed) domains: A single processor may operate across various knowledge levels. A number of processors may have intersecting knowledge domains. Or there may be some direct correlation between informational and functional domains. This latter view possesses a certain theoretical appeal which has prompted the current modular hypotheses of language. So, while the formal partitioning of linguistic phenomena does not necessarily support the hypothesis of corresponding psychologically real, informationally encapsulated processors, it does present a strong motivation for entertaining such a hypothesis. That is, the reason we have distinct theories for, say, syntax and semantics is presumably because the type of rules and representations needed for characterising these phenomena are fundamentally different. The result is a natural division in some 'informational' sense. This informational encapsulation of our linguistic competence provides sufficient conditions for invoking the modularity paradigm: domain specific systems with limited representational and informational vocabularies, which as a result can operate at high-speed.

The strongest empirical support for the modularity of the syntactic *processor* is the apparent evidence of 'stages' of linguistic processing. In particular, there is evidence to suggest distinct levels of processing for lexical information [Swinney 79], syntactic structure, and ultimate interpretation of the utterance in a semantic and pragmatic context [Frazier 79]. Indeed, there is even evidence suggesting that the syntactic processor makes little use of even lexical information (beyond major category) in making initial decisions (see [Clifton 90], [Clifton *et al* 89] and references cited therein). While we defer a discussion of this evidence until the next section, let us propose a modular organisation for the language comprehension system (LCS), as sketched in Figure 2.1. Roughly, this organisation predicts that processing is "input-driven", with each module generating its representation on the basis of its own knowledge. That is, initial decisions are made without recourse to other systems, and feedback is presumably limited.

Existing empirical evidence supports, roughly, the distinction between lexical and syntactic processing, and between syntactic processing and subsequent semantic[4] interpretation. We make no comment in this thesis as to the status of the semantic/pragmatic system, i.e. whether or not it is exists as some encapsulated system within the LCS, or is simply part of the general cognitive system (GCS), and nothing of our discussion will hinge on this.

One aspect of human sentence processing which has been used to argue against a modular model is that of incrementality. There is a wide body of empirical evidence which illustrates, quite conclusively, that the interpretation of an utterance, including pragmatic and contextual, is performed *during* during the analysis of the input (again, see [Marslen-Wilson 75] and [Marslen-Wilson & Tyler 87]). According to this argument, the modular model 'pre-

[3] In support of the 'interactionist' position, see in particular [Marslen-Wilson 75] [Marslen-Wilson & Tyler 87].

[4] We use semantics here to refer to the semantic properties of lexical items and the use of contextual and pragmatic information. It is possible that the syntactic processor generates some logical form (LF) representation, which is not to be confused with semantics in the above sense.

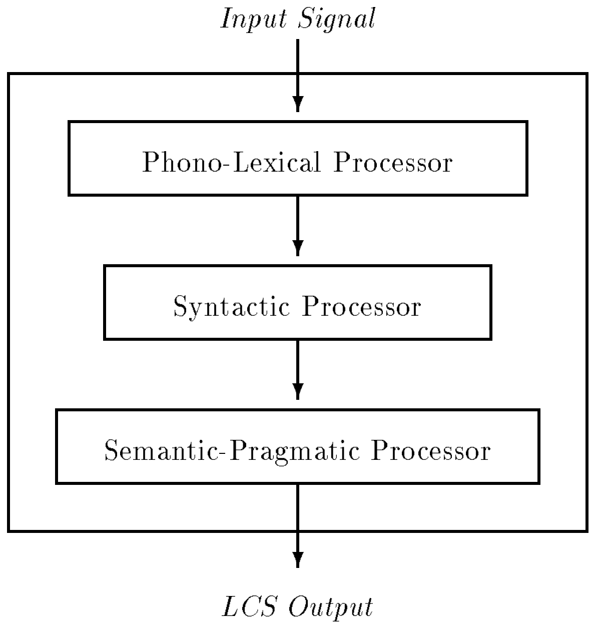



Figure 2.1: The Language Comprehension System

dicts' that we should first construct a list of words, for the entire input signal (the lexical processor), then derive a syntactic analysis for the string (the syntactic processor), and only then can the process of semantic interpretation begin. One instance of this argument is levelled in [Crain & Steedman 85], where they go on to consider a 'weaker' modular model in which the *unit of modularity* is the phrase (rather than the entire sentence).

As Stabler has demonstrated, however, the proponents of this argument have entered the *pedestrians paradox* [Stabler 89a]. Modularity itself does not make this prediction, it only requires that a module construct an output representation, based on an input, and be both functionally and informationally encapsulated. The further suggestion that the entire input (corresponding to the entire sentence) must be available before processing can begin is purely stipulative, and in no way implicit or natural to the notion of modularity. Thus the arguments Crain and Steedman invoke against 'strong' modularity can only be interpreted as falsifying the particular version of modularity they propose — a model which is not particularly natural, nor does it capture the notion of 'strength' in the Fodorian sense. We will assume here that each module begins constructing its own representation incrementally (thereby making it available to subsequent systems) as its input representation is received, such that the LCS as a whole operates incrementally. The strength (or, 'degree') of modularity for a particular processor is determined by its encapsulation: the extent to which the restricted informational and representational domain over which a module presides can be maintained.

To summarise, we will assume the existence of a distinct syntactic processor, while simultaneously seeking an account of the influence which non-syntactic systems (semantics, pragmatics, discourse, etc.) clearly exert during comprehension. In the following sections we consider the range of empirical evidence bearing on the issue of syntactic modularity and interaction, and examine the prevalent theories of syntactic processing.

## 2.2 The Nature of the Empirical Evidence

Just as a theory of syntax must account for the range of constructions which occur in language, so must a theory of processing account for the relevant behaviour exhibited when processing language. That is, to achieve descriptive adequacy, the theory must account for, or be consistent with, the various empirical phenomena.

The study of linguistic competence has an advantage in that the range of empirical evidence is, by definition, the set of utterances which are possible in language, and are thus relatively straightforward to identify. In the pursuit of a theory of linguistic performance, however, the empirical data are less concrete, and not as readily available[5]. To assess human sentence processing behaviour, a number of techniques are employed. The range of techniques vary considerably in terms of the resolution, and the degree to which they interfere with normal processing. Off-line tasks give little direct evidence of performance during processing, but have the advantage that — if properly controlled — they can be taken to reasonably reflect normal reading performance. On-line tasks seek to explore performance during comprehension, providing greater resolution but often at the expense of unnatural procedures. The following list outlines the range of some frequently used procedures:

(4) *Global reading time:* Measurements of the time taken for a subject to 'understand' an utterance (off-line).

*Self-paced reading time:* Word-by-word presentation, with measurements of the time taken for subject request the next word (on-line).

*Grammaticality judgement:* Subjects are required to identify if and when a sentence becomes ungrammatical, during processing (off-line).

*Eye movement:* The subjects eye movements are recorded during reading, giving information about regression, and gaze duration (on-line).

*N-400 anomaly:* A measurement of brain activity which indicates the presence of anomaly during comprehension (on-line).

In addition to experimental data using the above techniques, another potentially rich source of information is available from the study of aphasia — instances where patients have suffered limited brain damage, resulting in very localised loss of specific mental functions. Such evidence is particularly useful in forming hypotheses about the organisation of the language faculty, and the nature of processing [Caplan & Hildebrandt 88]. As a specific example, Linebarger argues that there exists neuropsychological evidence to support the basic modular architecture of the language faculty depicted in Figure 2.1 [Linebarger 89]. In particular,

[5] While a syntactician can use his own judgements concerning the grammaticality of sentences, the psycholinguist is generally unable to identify subtle variations in his own processing of utterances. Since sentence processing is generally an unconscious task, direct introspection during parsing is impossible.

she provides two-strands of evidence; one which indicates that patients with lexical impairment retain sophisticated syntactic processing ability, and another where patients who are incapable of comprehension can nonetheless determine the grammaticality of an utterance.

### 2.2.1 Ambiguity in Language

The primary emphasis of psycholinguistic experimentation is on the study of ambiguity: how do people behave when presented with more than one interpretation of the input? By formulating descriptive characterisations of the decisions people make, and the strategies they use, we can begin to make hypothesis about the underlying process models which explain these strategies. Below we give a brief overview of ambiguity phenomena which are relevant to studies of sentence processing.

#### Lexical Ambiguity

There are two primary sources of ambiguity which concern syntactic processing: The first concerns ambiguity in the *input* to the syntactic system, i.e. at the lexical level, and the second concerns indeterminism in the choice of possible syntactic analysis after decisions of lexical ambiguity have been made. If we first consider lexical ambiguity, we note that a given word may exhibit both semantic and category ambiguities:

(5) a. "I robbed the *bank*."
  b. "I fished from the *bank*."
  c. "I *bank* with Lloyds."
  d. "The pilot *banked* the plane."

In (5a&b), *bank* is used as a noun, but with two distinct meanings; a semantic 'sense' ambiguity. In (5c),*bank* is a verb, but has a meaning clearly related to that in (5a), but completely unrelated to (5b). Finally, (5d) demonstrates another verbal sense of *bank*, which is quite separate from the other uses (although possibly vaguely related to that in (5b)), partially illustrating the range of possible category and sense ambiguities which can occur in the lexicon.

#### Syntactic Ambiguity

Examination of lexical ambiguity tells us something about the mechanisms for lexical access and the nature of lexical-syntactic interaction. Insight into syntax-internal processes, however, is perhaps best provided by inquiry into the range of structural ambiguities which occur. These are cases where multiple syntactic analyses can be assigned to a particular string of lexical items. In cases where this ambiguity ranges over the entire sentence, we have a global ambiguity:

(6) "I saw the girl in the garden with the binoculars."

For this sentence, there exist a number of possible interpretations, some of which are more or less felicitous. The explanation for this is that a number of syntactic analyses obtain for the string, where each analysis gives rise to a different semantic interpretation. It is also possible, if we accept that sentences are processed incrementally and left-to-right, that we might encounter local ambiguities, where there are a number of possible analyses for the current, partial input:

(7) "I knew the solution to the problem was incorrect."

In this sentence, the first ambiguity occurs as a result of *know* being ambiguous as to the category of its complement: it may take either a noun phrase or a sentence. Thus, when the noun phrase *the solution to the problem* is encountered, we have two possible analyses: we can attach it directly as the object of *knew* or we can create a sentential complement with *the solution to the problem* as the embedded subject. In contrast with (6), however, only the sentential complement analysis is sustained when the remaining words *was incorrect* are encountered. In fact, such sentences are only ambiguous if we assume *incremental* interpretation. There are a number of parsing techniques which can be adopted in such circumstances: one option is keep all possible intermediate analyses alive until disambiguating material is discovered, another is to choose one analysis and then 'backtrack' to alternatives if required, or finally, we might adopt a non-incremental approach, where no attachment of the noun phrase is performed until disambiguating information is found. In this latter case there is really no relevant sense of ambiguity, except that the parser makes an explicit decision to delay attachment since it is locally underdetermined. All of these approaches are entertained in the literature as we shall see later in this chapter.

### 2.2.2 Relative Complexity and Semantic Interaction

The range of ambiguities which occur in natural language lead us to ask the question: How do people perform in the face of local and global ambiguities? Indeed, this question underlies the lion's share of experiments on syntactic processing. Using the experimental techniques outlined above, it is possible to test human linguistic performance for preferred interpretations of ambiguous sentences, and relative increases in processing complexity. From a theoretical standpoint, this data has provided a significant amount of evidence concerning the validity of syntactic autonomy.

In §2.1 we sketched the basic organisation of a modular language processor, which has been widely adopted in one form or another, by the proponents of modularity. As we discussed, the model predicts that the various informationally encapsulated systems are basically data-driven, bottom-up, rather than being influenced by subsequent processes: so initial lexical decisions are made independent of the syntax, and initial syntactic decisions are made without recourse to semantic, real-world knowledge. In an effort to tease out the empirical evidence which would support this position, there have been a number of experiments which have attempted to illustrate that humans exhibit uniform syntactic preferences in ambiguous sentences, regardless of the plausibility of the the particular interpretation. Where this initial syntactic preference turns out to be incorrect, leading the sentence processor "down the garden path", the model predicts that there will be some increased cost in correcting this erroneous analysis.

Perhaps the strongest garden-path effect occurs in the so-called *reduced relative* construction. Consider the following, somewhat overworked, example:

(8) "The horse raced past the barn fell."

The ambiguity in this sentence occurs when *raced* is encountered, permitting two possible analyses; one as an active clause where *the horse* is the subject of *raced* such as "The horse raced past the barn", and another where *raced* begins a reduced relative clause equivalent to "The horse which was raced past the barn fell". The difficulty of interpreting the sentence in (8) seems, therefore, to arise from a preference to adopt the active clause analysis over the reduced relative. The *interactionist* explanation for this phenomenon would be that active analysis is preferred for reasons of semantic content. Crain and Steedman propose several non-syntactic rules which might account for this preference, including *The Principle of A Priori Plausibility*, a preference for an interpretation which is more plausible in terms of our knowledge of the world, and *The Principle of Referential Success*, the favouring of an analysis which succeeds in referring to an entity already established in the hearer's mental model. The latter principle might be invoked in (8), since we have no reason to believe there is more than one horse, the restrictive (reduced) relative interpretation is not motivated (see [Crain & Steedman 85] and [Altmann & Steedman 88] for further discussion). This view predicts that we can eliminate the garden path effect by manipulating the context accordingly. The *modularists*, on the other hand, hold that there is some independent syntactic preference for the active analysis, regardless of plausibility.

In a series of experiments, Crain and Steedman illustrated that controlling definiteness, plausibility, and context could significantly alter the degree of garden-pathing which occurred. In particular, they conducted a grammaticality judgement task using the following paradigm:

(9) a. "The teachers taught by the Berlitz method passed the test."
b. "The children taught by the Berlitz method passed the test."
c. "Teachers taught by the Berlitz method passed the test."
d. "Children taught by the Berlitz method passed the test."

Results sustained their predictions, showing that the (9b) & (9d) (plausible) were judged grammatical more often than their (implausible)(9a) & (9c) counterparts. Similarly, the (9c) & (9d) (indefinite) sentence were judged grammatical significantly more often than the (9a) & (9b) (definite) versions. These results support the hypothesis that the Principles of A Priori Plausibility and Referential Success, respectively, can affect the severity of garden paths. However there is an important deficiency in this experiment: The grammaticality judgement task is an *off-line* measure, which can be interpreted in two ways; either the people choose the correct analysis because of recourse to the semantic principles, or they choose the wrong (syntactically preferred) analysis but are able to recover, possibly with the the help of these semantic principles, in time to judge the sentence as grammatical[6]. One strong result of this experiment, however, is the fact that non-syntactic processes are apparently brought in to play, during the analysis of the sentence not after the fact.

The latter of these two interpretations is clearly the view that the *modularists* would choose to take. That is to say there is a purely syntactic preference[7] for the active analysis over the reduced relative, and informational encapsulation enforces this preference regardless of context or plausibility. Any semantic affects only serve to facilitate reanalysis, and cannot influence the initial path taken. This view, in contrast with the previous one, predicts that there will *always* be a garden-path effect when the ambiguous reduced relative construction is encountered. To elicit this result, however, it is necessary to invoke on-line measures which are capable of detecting increased processing load during sentence processing.

Using an eye-movement study and confirmation from a self-paced reading task, Ferreira and Clifton have demonstrated that an initial preference for the active clause analysis does obtain, even when a bias towards the reduced relative analysis is generated in a preceding context [Ferreira & Clifton 86][8]. That is, first-pass[9] gaze durations revealed an increased amount of time spent at the disambiguating region of the sentence, when the overall analysis was syntactically unpreferred, while total reading times did not reveal a syntactic preference. These two studies demonstrate how differing experimental paradigms exhibit varying degrees of sensitivity. Furthermore, both studies provide us with a strong empirical result: although there is sufficient evidence to pursue the investigation of a modular syntactic processor, an adequate theory of sentence processing must explain the way in which the semantic system contributes to the analysis of an utterance during processing.

### 2.2.3 Summary

In sections 2.1 & 2.2, we have presented theoretical arguments for the existence of a modular syntactic processor, and outlined the sorts of empirical evidence which can be brought to bear in supporting the hypothesis. While there is clear evidence that syntax makes initial decisions concerning the direction of analysis in the face of ambiguity, it is also clear that 'post-syntactic' processes are closely coupled: That is, the degree of increased complexity introduced by initial (incorrect) decisions on the part of the syntactic processor, is related in some manner to the semantic and pragmatic plausibility of the analysis.

In the following section, we review several prevailing theories of sentence processing, all of which assume a modular syntactic parser. In addition to evaluating their theoretical

---

[6] This experiment has been replicated using an on-line task in [Altmann & Steedman 88], but the results remain open to either interpretation.

[7] Or perhaps we should say, a 'non-semantic' preference, at least in the first instance. The particular nature of the strategies which have been suggested is not required for this discussion. We take up this issue in §2.3.2.

[8] There aim was to replicate the results achieved in [Rayner *et al* 83] which were conducted using only a null-context: an approach justifiably criticised in [Crain & Steedman 85].

[9] Eye-tracking studies keep track of the initial gaze duration for the regions of a sentence, as well as any subsequent gaze resulting from regression. This *first pass* gaze is often interpreted as the relevant figure reflecting initial, syntactic, processes.

status, as principled models of syntactic performance, we will also consider their success in addressing the broader empirical issues outlined above: The extent to which they are consistent with the incrementality demands placed on the (modular) syntactic processor so as to permit immediate interpretation and explain the resulting interaction effects.

## 2.3 Extant Theories of Linguistic Performance

From a purely descriptive point a view, a theory of sentence processing must satisfy two basic requirements. First, it must specify how the rules of grammar are used to construct an analysis of an utterance. Second, it must account for the empirical processing phenomena:

(10) a. How do humans manage to parse sentences so rapidly?
b. What leads to preferred readings in ambiguous sentences?
c. Why are some sentences more difficult to process than others?
d. How do humans recover from errors made during parsing?
e. What causes processor breakdown, or 'garden path' phenomena?
f. What leads to over-loading in non-ambiguous sentences?

In the discussion up to this point we have converged on a working hypothesis which asserts the autonomy of syntactic processing and incremental interpretation of input. In addition, we assume that language is processed serially: the LCS only entertains one analysis of the (partial) input at any given moment, and alternatives are the result of backtracking. It is important to note that while syntactic analyses are constructed serially, this does not contradict the possibility that modules operate concurrently (so as to achieve incremental operation).

Distinguishing serial processing (with backtracking) versus parallel is a difficult task. While the backtracking model seems to provide a natural characterisation of increased processing complexity, it is also possible to construct 'ranked parallel' architectures which can be contrived to make similar predictions (for discussion see [Gorrell 87], [Gorrell 89] and [Gibson 91]). In actual fact, there is a certain equivalence between these approaches and it seems that most theories of processing can be recast in either ranked parallel or serial architectures without significantly affecting their status.

Central to all theories of processing, regardless of the underlying computational assumptions, is some notion of complexity. To explain the relative increases in processing complexity demonstrated by people, it is reasoned that some aspect of reanalysis is particularly complex, and hence the parsing mechanism must operate so as to minimise such complexity during processing. In what follows we present three theories, each with a difference notion of the the relevant complexity metric. The first view assumes that *computational* complexity must be minimised, and argues that the HSPM therefore strives to operate deterministically, so as to avoid backtracking. The second view presumes that *representational* complexity — the complexity of the parse tree — must be minimised, and posits strategies which ensure the most minimal structural analysis is pursued during processing. The third, approach departs from the traditional 'time and space' notions of complexity just mentioned, and suggests that the syntactic processor strives to resolve certain syntactic licensing relations as soon as possible, and that it is precisely the reanalysis of these relations which leads to processing difficulty. Thus, while processing complexity is still ultimately determined by time and space (in some manner), the operation of the syntactic processor is not determined directly by the desire to minimise them. I will call this a *grammar-based* account.

### 2.3.1 Computationally Based Theories

Perhaps the most striking aspect of the human sentence processor is that of (10a), i.e. speed. For most utterances, people are able to construct an interpretation in real time, and without any conscious effort. This observation was the prime motivation for the deterministic parser developed by Marcus [Marcus 80]: The basic reasoning was that since the sentence processor is fast, it must be deterministic, i.e. form syntactic analyses only when there is sufficient grounds to guarantee it is the correct on, and thus avoid backtracking. Interestingly, a deterministic parser will *fail* if it encounters input which cannot be incorporated into the current analysis, since determinism prohibits *backtracking* to an alternative analysis. Such a model naturally predicts that failure on the part of the parser, should occur for precisely those garden path sentences which cause humans to fail. Specifically, Marcus implemented an LR(3) parser — bottom-up, left-corner, with three item look-ahead — for English.

The parser developed by Marcus, has since been refined by Berwick and Weinberg. Significantly, they propose *informational monotonicity*, rather than determinism, as the relevant underlying computational constraint on the parser. That is, once a representation is constructed, no subsequent action of the parser can destroy it. They propose an LR parser which uses a state table based on single item lookahead, bounded left context, and relevant lexical information (such as subcategorization frames) to determine the next action the parser should take. The parser acts serially, constructing only one analysis of the utterance, but cannot be considered incremental. The machinery of the parser allows lexical items or phrases to remain unstructured – in the input buffer – until necessary disambiguating information is found. Perhaps the most interesting result of their effort is that the use of bounded left context provides a functional, processing account of c-command and subjacency – two central notions in grammatical theory. This supports the possibility of the parser determining, at least in part, the principles of grammar.

The last point raises the issue of their model's principled basis. Roughly, they argue that both their and Marcus' parser construct the same structural descriptions of current transformational grammar, and also that the parser 'obeys' the principles of grammar (such as the projection principles, subjacency, etc.) even though they are not explicitly present. They suggest the parsers are therefore principle-based, insofar as they use (more or less) a compiled-out instance of the principles of grammar (parameterised for English), and not some unrelated set of rules — which might still yield the same structural descriptions — or covering grammar which would require the translation of alternative structures into those of the theory proper. The parsers are not, however, principle-based in the sense we have assumed here: I.e. they do not make *direct* use of the principles of grammar. Crucially, however, if their parsers are truly just compiled-out transformation of the original theory, then it may be possible to construct an equivalent principle-based parser, which operates similarly but accesses the individual principles on-line. However, since the parsers are designed first and foremost to operate deterministically/monotonically, and do not make decisions regarding the grammatical principles themselves, they are not grammar-based, and hence cannot be strongly principle-based.

The model proposed by Berwick and Weinberg falls short in a number of respects. The ability of the parser to leave input material unstructured is an obvious violation of incrementality. Moreover, the reliance of the parser on lexical, subcategorization information, while being quite natural for a head-initial language such as English, predicts serious problems for languages such as German, Dutch, and Japanese, where heads may follow their arguments, rendering single constituent look-ahead inadequate. For these cases the parser would have to leave vast amounts of material unstructured, or 'buffered', until the subcategorizing verb was found, running counter to a variety of empirical evidence to the contrary (see [Frazier 87a] for data and discussion). Furthermore, it is unclear how their functional account of c-command, in terms of bounded left context, would carry over into exclusively head-final languages such as Japanese.

Finally, the deterministic/monotonic models provides an inadequate account of the range of *relative* processing effects, such as slight increases in processing complexity, recoverable garden paths, and total processor breakdown (as discussed in section 2.2). Furthermore, there is no way of explaining the thematic/pragmatic effects on parsing which have been observed in numerous experiments (see [Crain & Steedman 85], [Carlson & Tanenhaus 88], [Stowe 89] and references cited therein). In sum, the Berwick and Weinberg model provides only a partial, and language specific, account of processing phenomena, and the strong assumption of determinism or monotonicity in the syntactic processor seems incompatible with any explanation of relative processing complexity.

### 2.3.2 Strategy Based Theories

Some of the best known research on sentence processing has arisen from the work of Lyn Frazier and her colleagues. The underlying structure of the theory which has emerged assumes an organisation which is similar to that presented in §2.1. Specifically, Frazier proposes a model of processing consisting of two basic modules: a *syntactic processor* which constructs a constituent structure representation, and a *thematic processor* which selects an appropriate assignment of semantic roles for the syntactic structure, on the basis of real-world knowledge [Frazier 84] [Rayner *et al* 83]. These two stages of processing provide a potential explanation for the range of relative processing affects which occur due to variations in pragmatic plausibility:

> " ...it follows automatically that a sentence will be easier to process when the frame chosen by the thematic processor is consistent with the initial syntactic analysis of the input, than in cases where the two conflict."
>
> [Frazier 84]

Frazier assumes that the thematic processor acts in parallel (concurrently) with the syntactic processor, permitting the rejection of inappropriate analyses after they are proposed. Crucially, however, Frazier maintains that the *initial* decisions concerning constituent structure are made solely by the syntactic processor without 'top-down' influence from the thematic processor. Given this organisation, Frazier has concentrated on identifying the strategies which are operative at the syntactic level. Following the work of Kimball [Kimball 73], Frazier has suggested that the syntactic processor is guided by two basic principles of *Minimal Attachment* and *Late Closure*, defined as follows [Frazier 79, page 76]:

(11) **Minimal Attachment** (MA): Attach incoming material into the phrase marker being constructed using the fewest nodes consistent with the well-formedness rules of the language.

**Late Closure** (LC): When possible, attach incoming material into the clause or phrase currently being parsed.

The model of processing is implicitly incremental: lexical items are incorporated into the current, partial analysis as they are encountered[10]. Where there is an ambiguity in the direction the analysis may take, the principles of MA and LC are consulted, and furthermore, MA has the higher priority of the two. If the analysis chosen turns out later to be incorrect — i.e. the parser has been led down the garden path — then the parser backtracks to pursue an alternative analysis. It is important to note that the notion of garden-path which Frazier adopts is very general, ranging from conscious garden paths, which are noticeably difficult to recover from, to unconscious garden-paths which can only be observed by performing experiments which are sensitive to subtle increases in complexity. This differs from the simple notion of garden-path phenomena assumed by Marcus and Berwick & Weinberg — i.e. just conscious examples. In general, we will adopt Frazier's view that a theory of sentence processing must explain the entire range of complexity increases which occur, and account for the decisions made in the face of local ambiguity during strictly incremental left-to-right processing.

In contrast with the Berwick and Weinberg model, aimed at minimising the computational complexity, Frazier's strategies are motivated by a desire to minimise the representational complexity of the syntactic analysis, at each stage of processing[11]. The tacit assumption which Frazier makes, is that the cost of representing syntactic structure is relevant enough to warrant making choices on this basis, rather than trying to minimise the computational cost of, say, backtracking.

The principles of MA and LC provide a reasonable account of the core attachment preferences in ambiguous constructions. Let's reconsider the PP attachment ambiguity mentioned in (6), which we have simplified below:

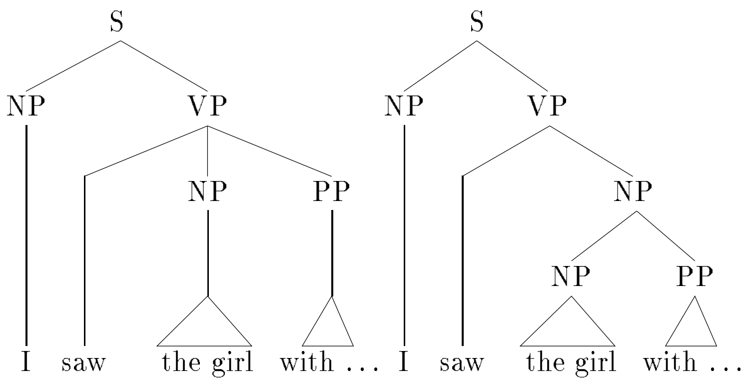
(12) Preferred VP attachment over NP adjunction.
  a. "I [$_{VP}$ saw [$_{NP}$ the girl ] [$_{PP}$ with the binoculars ]]."
  b. "I [$_{VP}$ saw [$_{NP}$ the girl [$_{PP}$ with red hair ]]]."

[10] This has recently been made explicit as the *Left-to-Right Constraint* in [Frazier & Rayner 88].

[11] It is conceivable that pursuing the minimal analysis at some intermediate point in the sentence could result in a more complex global analysis, although it is difficult to conjure such examples.

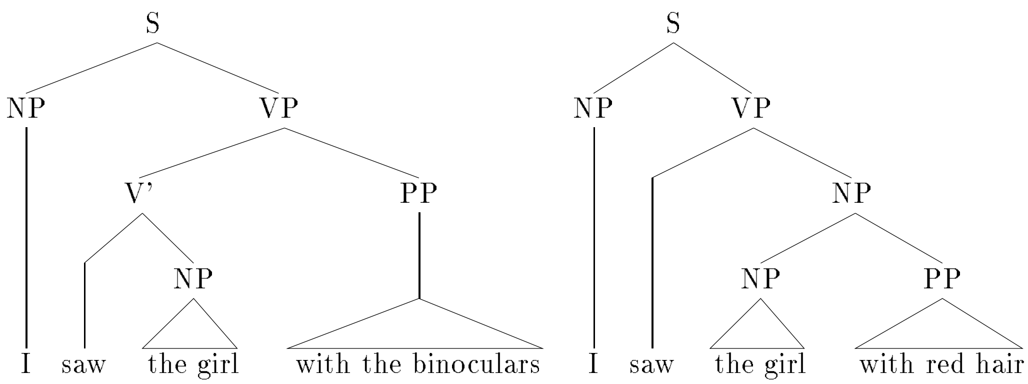
These sentences illustrate that there are two possible attachments for the *with* PP, as either a complement of the verb, or a modifier of the NP *the girl*. Given incremental processing, the attachment of the preposition *with* must be performed before its object NP is encountered. Interestingly, this suggests that even recourse to semantic knowledge, e.g. the properties of the preposition's object, would be of no use in making this particular decision. The two possible phrase structures at this point are illustrated below:

(13) a. b.




Minimal attachment dictates that the sentence processor will opt for the analysis (13a) over (13b) on that grounds that it involves postulating one less node, namely the extra NP node. The prediction is that sentences where the PP continuation is consistent with the attachment into the VP (e.g. such as *the binoculars*) will be easier to process. This preference has been demonstrated using eye-movement studies in [Rayner *et al* 83] and replicated in [Ferreira & Clifton 86] using a similar paradigm which also tested for possible contextual effects (as discussed in §2.2.2). As Abney and others have observed [Abney 89], however, this account is rather suspect in that this particular structural analysis is currently thought to be incorrect — Kayne and others standardly assume a binary branching structure throughout [Kayne 84]. This would require an extra VP branch in (13a), resulting in equal complexity to that of (13b), as shown below:

(14) a. b.



Indeed, this illustrates the sensitivity[12] of Frazier's strategies to slight variations in the syntactic analysis, a topic to which we will return at the end of this section. If we consider, however, the reduced-relative garden path (8) discussed in §2.2.2 (again based on data from Rayner *et al* and Ferreira and Clifton cited above), the minimal attachment analysis seems much less controversial:

(15) Preferred active clause over reduced relative.
  a. "[$_S$ [$_{NP}$ The horse ] [$_{VP}$ raced past the barn ]] and fell."
  b. "[$_S$ [$_{NP}$ The horse [$_{Rel}$ [$_{VP}$ raced past the barn ]]] fell ]."

Here we can see that when *raced* is encountered, the two available analyses vary widely in their syntactic complexity, regardless of the particular theoretical details. In the active analysis (15a), the verb *raced* projects to a VP which is then attached to the existing root S node. The complex-NP (15b) interpretation, however, requires the creation of the relative-clause structure (an S' and S, which we have abbreviated as Rel in (15b)), and the insertion of an adjunction site within the subject NP. Another example is possible local ambiguity of a complement as either NP or S, as observed in (7):

(16) Preferred NP vs. S complement.
  a. "The scientist knew [$_S$ [$_{NP}$ the solution to the problem ] was trivial ]."
  b. "The scientist knew [$_{NP}$ the solution to the problem ]."

MA predicts the NP *the solution to the problem* will be initially analysed as the direct object (16b), since this avoids postulating the intervening S node. This prediction is borne out by the eye-movement experiment described in [Frazier & Rayner 82]. That study also tested cases of clause boundary ambiguity illustrated by the following sentences:

(17) Preferred object attachment where possible.
  a. "While Mary was [$_{VP}$ mending [$_{NP}$ the sock ] ] [$_S$ it fell off her lap ]."
  b. "While Mary was [$_{VP}$ mending ] [$_S$ [$_{NP}$ the sock ] fell off her lap]."

There is a strong preference for attaching *the sock* as the object of mending, as in (17a), rather than as the subject of the main clause (17b), which results in a conscious garden-path. Assuming, however, that the main clause S node is available for attachment, both analyses are equally minimal. To resolve this problem, Late Closure prefers attachment to the VP, the most recent phrase considered by the parser, over the main S, which has yet to be analysed[13].

---

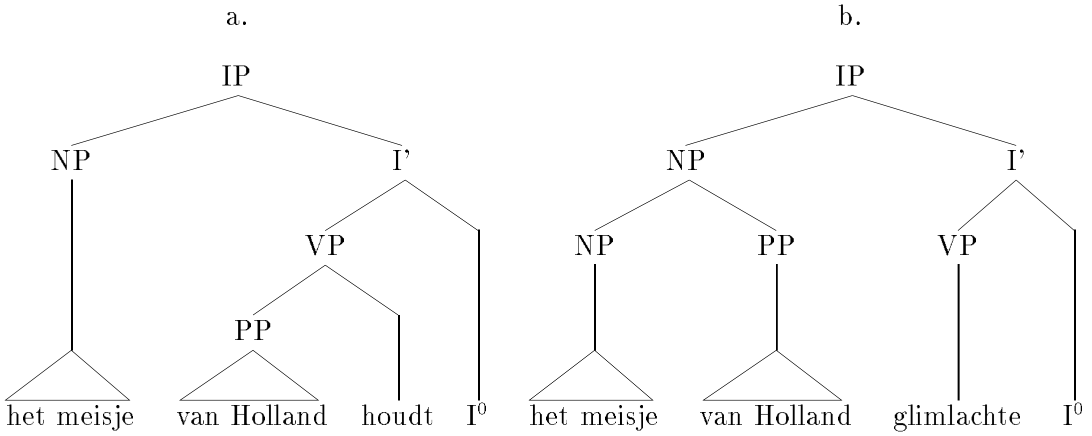
[12] Pereira notes this problem in his attempts to implement MA and LC using a shift-reduce parser with an oracle [Pereira 85].

[13] In most cases LC is used to explain the preferred 'low attachment' of constituent in multiple clause sentences. Consider, for example, *"I told you John bought the car yesterday."*. In this sentence, *yesterday* may modify either the main or embedded clause, but there is a general preference for the latter, which is accounted for by the LC strategy, since this is the clause "currently being parsed".

### Cross-Linguistic Accounts

The strategies which Frazier and her colleagues have proposed are based on a desire to minimise the representational complexity of analysis during processing. Assuming people, regardless of their mother tongue, are equipped with similar sentence processing machinery, MA and LC should hold across languages[14]. Of particular interest are languages in which head-final constructions can occur, such as Dutch, German (which have a mixture of head-final/initial constructions) and Japanese which is entirely head-final. Indeed, at a somewhat speculative level, Frazier and Rayner have argued that their proposed model holds across left- and right-branching languages, specifically English and Japanese, and cite some initial evidence supporting the use of MA in Japanese [Frazier & Rayner 88][15]. In this regard, Frazier has done a number of experiments in Dutch, to compare the preference for VP attachment over NP modifier as described for English in (12) above [Frazier 87a]:

(18) (a) "...dat $[_S$ $[_{NP}$ het meisje ] $[_{VP}$ $[_{PP}$ van Holland ] houdt ]]."
*...that the girl Holland likes*
(b) "...dat $[_S$ $[_{NP}$ het meisje $[_{PP}$ van Holland ]] glimlachte ]."
*...that the girl from Holland smiled*




Until the verb is reached, *van Holland* is ambiguous as either an NP modifier or VP object. Frazier's evidence indicates, however, that sentences of the form in (b) are more difficult to parse than (a), demonstrating a preferred attachment to the VP even though the verb has not been encountered. If we assume the VP node is available for attachment then MA explains this phenomena in the same manner as described for (12) above. Frazier also argues that MA succeeds in explaining this even if the VP node has yet to be postulated assuming that, given two equally minimal attachments, the one consistent with the current analysis (the VP attachment) is chosen over one requiring revision (the insertion of an adjoined NP

[14] See [Cuetos & Mitchell 88] for evidence suggesting that LC does not hold in Spanish. This is particularly problematic given the constituent order for the relevant constructions is similar for English and Spanish.

[15] They do, however, argue that some aspects of the model must be parameterised for individual languages. In particular, they suggest that, while in English we begin by attaching constituent into the top-most S node, this cannot be the case in Japanese. We return to a discussion of this chapter 7.

node for the NP modifier reading)[16] — although I am unfamiliar with any appeals to this particular refinement of MA [Frazier 87a, page 528], prior to the account given for this data. Furthermore, the late closure strategy should presumably prefer the attachment of the PP as the NP modifier, in the face of equally minimal attachments.

### Discussion

In most regards, the theory of sentence processing developed by Frazier and her colleagues appears descriptively successful. It has a strong foundation in that the strategies proposed are assumed to be motivated by a desire for representational parsimony. The lack of a supporting computation model, however, leaves it unclear as to what the computational properties of these strategies are: if we consider how MA might be implemented[17], it becomes apparent that in fact all possible analyses will have to be enumerated so that the minimal one may be selected (although there are probably numerous ways in which this search space might be pruned). If there are more efficient techniques for implementing MA, then they are not obvious, and should be investigated by its proponents. Other operations, such as inserting daughter nodes into existing structures (i.e. where non-binary branching structure are allowed) are assumed without discussion, even though this implies that the entire partial phrase structure is potentially available for subsequent attachment, requiring a non-trivial search of the right edge of the tree at (potentially) each stage of processing, although as observed at the end of the last section, Frazier does seem to imply that this insertion operation is at least sometimes unpreferred. Furthermore, in §2.3.2 we highlighted how the precise invocation MA and LC has been refined or modified to account for particular constructions — clearly, a computational model would ensure consistent application of these strategies.

Perhaps the largest question mark against Frazier's model is its grammatical basis. The syntactic analyses she provides for the constructions under consideration seem to be based on out-dated phrase structure grammars[18]. This issue is serious when taken in the context of a theory which makes crucial reference to the number of nodes involved in a particular analysis, and which also makes predictions based on when attachments can be made directly into existing nodes (VP attachment) or involve inserting new nodes (such as adjunction). While Frazier makes no real claims to being principle-based, she does suggest that her theory is compatible with the view that the principles of grammar are compiled out. As we have indicated, however, there remain several incompatibilities with the structural descriptions

[16] This highlights Frazier's rather traditional assumptions concerning how structures are built. There do exist well-known techniques which permit the monotonic 'insertion' of structure into a tree [Marcus *et al* 83].

[17] In Pereira's implementation, the oracle simply chooses the rule which is minimally complex [Pereira 85]. In cases where multiple rules must be invoked to attach a lexical item, more sophisticated techniques for comparing the relative complexity of candidate analyses will be required.

[18] In addition, to the lack of binary branching, the analysis presented for Dutch [Frazier 87a, pages 532-533, for example], are completely at odds with current analyses involving movement to the head and specifier positions of CP (see chapter 3). As a further example, Frazier's analysis of PP modifiers is inconsistent; they are attached as adjoined phrases for NP's (crucially introducing an extra node), but are attached directly into VP's — an analysis which was reserved for complements even before binary branching was introduced.

she assumes and those of current theories of grammar.

Related to this last concern is the fact that the strategies proposed by Frazier are ignorant of grammatical *content*; rather, they refer only to the *form* of the particular syntactic analysis. In this way, the model of sentence processing she proposes is completely insensitive to the various grammatical relations and their relative importance[19]. In no way is any desire to satisfy various licensing requirements, such as theta-role and case assignment, reflected in the operation of the parsing machinery — these grammatical concerns are simply well-formedness conditions, not 'major players', at the processing level. While this is entirely possible, it seems somewhat counter-intuitive, especially as grammars becomes more abstract. Within the principles and parameters view of grammar, structures are licensed and ruled out by independent constraints, resulting in a less direct correspondence between structural descriptions and the principles which license them. Thus representations become mere 'artefacts' of syntactic analysis rather than fundaments. Therefore to emphasise the role of the form of representations seems misguided with respect to views of syntax. Further, Frazier's view entails that the representation must actually be constructed which, as we will see in chapter 7 is not necessarily the case.

Having levelled these criticisms, we should also point out that Frazier's basic organisation seems to provide the most successful account of the data. In the model we develop here, we will adopt Frazier's view of relative garden-path effects, and the strictly incremental and serial behaviour of the sentence processor. We will also adopt roughly Frazier's notion of modularity, with distinct syntactic and semantics processes which are encapsulated (i.e. structural analyses are proposed by syntax alone) but operate concurrently, such that pragmatic and contextual interpretation is performed during parsing, and can hence explain contextual effects. Within this basic organisation, however, we propose a more grammar-based characterisation of processing preferences, which emphasises the content of grammatical relations, rather than the representations themselves. Before concluding this chapter, let us consider one such grammar-based model which has been proposed.

### 2.3.3 Grammar Based Theories

At the end of the previous section, we raised the issue of grammatical *content* and its role in the operation of the sentence processor. There is a view that syntactic relations are fundamental in determining the actions taken by the parser during processing. This assumption shifts the emphasis away from the abstract notions of computational or representational complexity which have been taken as fundamental in the two previous sections (respectively). Rather, grammar based accounts assume that the sentence processor is *driven* by the desire to construct a well-formed analysis of an utterance, and in this regard employs strategies which take into consideration syntactic relations such as licensing conditions (e.g. theta and Case assignment). This approach is theoretically attractive, in that it supports the notion of a close parser/grammar relationship, suggesting the HSPM is

[19] Although [Frazier 85] presents a metrical system for evaluating syntactic/processing complexity which assigns relative weights to different syntactic categories, suggesting some abstract link between their content and processing complexity.

strongly principle-based.[20]

Recently, it has been proposed that thematic roles, or $\theta$-roles, play a central part in sentence processing. In particular, Pritchett has adopted a bottom-up, head-driven parsing strategy which attempts to maximally satisfy the principles of UG at each point during processing [Pritchett 88], [Pritchett ar], and has proposed a principle of processing which is based directly upon the $\theta$-criterion of GB theory:[21]

(19) $\Theta$ **Attachment:** The $\theta$-criterion attempts to apply at every point during parsing given the maximal $\theta$-grid.

Roughly, this says attach constituents so as to saturate the $\theta$-roles of a given lexical items. The central assumption of this is that lexical entries are fully specified for thematic roles, and that such information is immediately accessed and used to drive parsing. In addition, Pritchett suggests the following principle, to account for the cost of reanalysis in the event that the parser makes an incorrect attachment[22]:

(20) **Re-Licensing Constraint:** Upon relicensing, a constituent must remain governed by its current licensor.

Given these definitions, let's reconsider the example from (17) above, repeated for convenience:

(21) a. "While Mary was [$_{VP}$ mending [$_{NP}$ the sock ] ] [$_S$ it fell off her lap ]."
b. "While Mary was [$_{VP}$ mending ] [$_S$ [$_{NP}$ the sock ] fell off her lap ]."

In (21a), at the point of processing *the sock*, it is attached as an object of *mending* since this locally satisfies the $\theta$-criterion assuming the maximal grid for *mending*. However should this attachment turn out to be incorrect, as for the continuation given in (21b), a garden path results and we must reanalyse *the sock* as the subject of fell, as shown in (21b). This reanalysis involves moving *the sock* out of the government domain of *mending*, and into a new government domain, i.e. of the verb *fell*. The Re-Licensing Constraint, therefore, correctly predicts a garden path effect. In contrast, let's consider again example (16), again repeated below:

---

[20] We will not digress here into discussion of the so-called *Derivational Theory of Complexity* (DTC), which when refuted was taken as concrete evidence against the psychological reality of TG. Not only has this been dealt with thoroughly elsewhere (see [Berwick & Weinberg 84]), but it also bears little relevance to current principle-based theories of grammar since it was developed in the context of a rule-based TG (the Standard Theory). We will take this point up briefly in chapter 7.

[21] This principle of grammar will be presented in chapter 3. The details are not necessary for discussion of this particular strategy.

[22] In the original formulation of his theory, Pritchett defined the $\Theta$-Reanalysis constraint, which required that a constituent remain in the $\Theta$-Domain upon reanalysis — the Re-Licensing Constraint effectively subsumes the original formulation.

(22) a. "The scientist knew [$_S$ [$_{NP}$ the solution to the problem ] was trivial ]."
b. "The scientist knew [$_{NP}$ the solution to the problem ]."

Pritchett argues that no garden path occurs in such sentences. As in Frazier's account, Pritchett suggests the NP *the solution to the problem* is originally licensed and attached as the direct object of *knew*, as in (22b). When the final VP is encountered in (22a), the NP is re-licensed as the subject of the embedded clause. Crucially, however, the NP remains governed by *knew*. This predicts the lack of a garden-path affect, contrary to the analysis of Frazier. At this point is is worth noting some crucial differences in the accounts given by Frazier and Pritchett. While Frazier is concerned with characterising the broad range of processing phenomena, from subtle preferences to conscious garden-paths, Pritchett's account is exclusively concerned with the latter. Having given the flavour of Pritchett's analysis of English[23], we now turn our attention to the issue of cross-linguistic coverage, particularly head-final languages.

### Cross-Linguistic Accounts

Just as the model proposed by Frazier is naturally predicted to apply across languages, so is that of Pritchett's, albeit for rather different reasons. Frazier suggest principles of processing which are motivated by representational parsimony; it is therefore natural to assume similar principles are operative across languages. Pritchett's model, based on the principles of universal grammar, would also be expected to apply for all languages, subject to variation only where it relates to the parameter settings for a particular language. As we have noted throughout this chapter, one of the most striking variations possible is that of word order: the position of a head with respect to its complements.

As Pritchett observes, his model of processing crucially relies on the occurrence of heads to drive parsing. To see how it operates consider again the Dutch examples introduced in (107), repeated below:

(23) (a) "...dat [$_S$ [$_{NP}$ het meisje ] [$_{VP}$ [$_{PP}$ van Holland ] houdt ]]."
*...that the girl Holland likes*
(b) "...dat [$_S$ [$_{NP}$ het meisje [$_{PP}$ van Holland ]] glimlachte ]."
*...that the girl from Holland smiled*

Frazier's evidence indicates that reading times are increased for sentences of the (b) form, relative to those of the (a) form. To account for this, Pritchett suggests that NP and PP phrases remain unattached until the verb is encountered, at which point the relevant structure is projected based on the the verb's thematic grid. Rather, inconsistent with his model, however, Pritchett attempts to explain the marginally increased complexity of the adjunct analysis on the grounds that "initial failure of the complement attachment and the

[23] For a complete exposition of his analysis of the range of conscious garden path constructions, see [Pritchett 88].

restructuring as an adjunct provide an account of the longer processing times" [Pritchett ar]. This seems odd since presumably the complement attachment never "fails" since it should never even be entertained (as the verb *glimlachte* doesn't license the attachment), and secondly, no restructuring should be required since none is initially built — the NP and PP were simply left unattached until the verb was reached. Furthermore, the garden-path is not a conscious one, so it seems curious that Pritchett would wish to account for it anyway.

Pritchett also argues for his head-driven account on the basis of Japanese syntactic structure [Pritchett ar]. Japanese is an exclusively head-final (and hence left branching) language, unlike German and Dutch which exhibit mixed positions (where only V, I and a subset of prepositions are head-final). Ignoring, for the moment, the interesting ambiguities which can arise as a result of recursively embedded clausal structures in such a left-branching language[24], Pritchett notes that even simplex clauses may be many ways ambiguous up until the final verb is encountered:

(24) a. "Rex ga John ga suki desu"
*Rex-NOM John-NOM fond-of is*
b. "John ga koibito ga sinda"
*John-NOM's lover-NOM saw*

The -ga particle, typically marks subjects, and may also mark two NPs as in the double subject construction of (24b). Similarly, however, the -ga particle may be used to mark the complement of (roughly) stative verbs, as in (24a)[25]. Thus, given two -ga marked NPs there exists an ambiguity between a double-subject and a subject-complement reading. Pritchett remarks that a strictly incremental model of parsing (such as Frazier's) will predict a preference (on the basis of MA & LC) for one of the two analyses, rather than waiting for the disambiguating verb. Regardless of which analysis Frazier's parser will prefer, the alternative is therefore predicted to be a garden-path. Pritchett denies (without experimental empirical support) that either analysis is marked in any way, and thus argues that such strictly incremental parsing accounts must be incorrect, at least for Japanese. There is a problem with this argument, however, since Pritchett seeks only to account for conscious garden paths (as we have remarked above), while Frazier has undertaken to explain more subtle processing complexity effects. So in fact, Frazier would predict an increase in complexity, for the non-preferred analysis, of similar magnitude to that for the Dutch example discussed above — i.e. a very slight effect, not a conscious garden path in Pritchett's sense.

The head-driven model of Pritchett's does, however, assume that once the subcategorizing verb is reached the structure will be built and the preceding arguments will be attached. As we alluded to above, the left branching nature of Japanese, combined with the option for arguments to scramble to pre-subject position, can introduce a range of cross-clause ambiguities with regard to attachment. This is made clearer by the following pair of sentences:

[24] We will discuss this issue shortly, but the reader is referred to [Mazuka & Lust 90] for a thorough discussion of the data.

[25] See [Pritchett ar] and references cited therein for details distinguishing the two analyses.

(25) a. "$[_S\ [_{\overline{S}}$ Bill ni Tom ga nanika o hanasita to $]_i$ John wa $\varepsilon_i$ omotte-iru]"
*[ [ Bill-DAT Tom-NOM something-ACC spoke that] John-TOP thinking ]*
*John thinks that Tom said something to Bill*

b. "$[_S$ [Bill ni]$_j$ $[_{\overline{S}}$ Tom ga nanika o hanasita to $]_i$ John wa $\varepsilon_i\ \varepsilon_j$ iwa-seta]"
*[ [Bill-DAT] [Tom-NOM something-ACC spoke that] John-TOP said-CAUSE]*
*John made Bill say Tom said something (to Bill)*

By both Frazier's and Pritchett's account, the material *Bill ni Tom ga nanika o hanasita* will be analysed as the simplex clause *Tom said something to Bill* (note, here that the indirect object *Bill ni* has scrambled to the pre-subject position). This analysis is completely consistent with the continuation in (25a), where the simplex clause is simply attached to the matrix clause as a (scrambled) sentential complement[26]. The continuation shown in (25b), however, ends with a causative verb which requires three arguments, including the dative *-ni* marked causee. This forces *Bill ni* to be reanalysed as an argument of the matrix clause, removing it from the domain of its original licensor (the embedded verb). This violates the Re-licensing constraint, and thus correctly predicts a conscious garden path effect for the sentence in (25b).

### Conflating Competence and Performance

The ability of a theory to explain processing phenomena in terms of the principles of grammar has a strong appeal. Roughly speaking we have defined such a grammar based theory of performance to be one in which decisions taken by the parser are motivated by grammatical considerations, such as licensing relations. Furthermore, in the context of current modular, language universal theories of grammar, we can say that a theory of processing which makes direct use of abstract principles lends support to the strong competence hypothesis and is 'strongly principle-based' in this respect. This contrasts with, for example, a processing model in which the principles have been compiled out into a homogeneous set of surface rules for purposes of parsing.

There is, however, a dangerous tendency for proponents of this approach to actually ascribe procedural interpretations to the declarative principles of grammar. This *grammar as parser* approach is not a rational position given the competence-performance division discussed in §1.1, which clearly separates the declarative properties of the syntactic theory from any procedural notions. Simply put, the rules of grammar have no implicit or natural realisation in a theory of processing: they are simply declarative conditions on the well-formedness of utterances. One instance of this "conflation", which has appeared in the literature in various guises, is the *Head Projection Hypothesis* :

> "If the individual principles of grammar are used directly in parsing, that is in isolation from each other, then we might expect phrasal nodes to be projected from their heads as dictated by $\overline{X}$ theory:

[26] Note, Frazier and Rayner observe that the strict top-down requirement, i.e. beginning with the top-most S, must be relaxed for Japanese, so that we can freely postulate higher S nodes is needed [Frazier & Rayner 88]. We will take up this point again in chapter 7.

> *Head Projection Hypothesis(HPH):* A phrasal node is postulated by projecting the features of its head."
>
> [Frazier 87b, page 523]

Frazier goes on to suggest that this proposal need not be correct, since it may be the case that the parser exploits other information, such as the fact that a determiner may only be specifier of an NP, to allow postulation of an NP before the head is reached — a completely reasonable position to take. Indeed, Frazier ends up abandoning the HPH on the basis of the evidence from Dutch discussed earlier in §2.3.2. What is wrong with this argument is the notion that $\overline{X}$ theory implies (let alone dictates) that a phrasal node may *only* be postulated when its head is encountered: $\overline{X}$ theory makes no such claims. The notion of 'projecting features from the head' is simply a *declarative* statement of the fact that the ultimate well-formedness of a phrase is determined by properties of the head in conjunction with the rules of $\overline{X}$. To make clear a point which should by now be obvious: a principle of grammar, as part of a theory of linguistic competence, does not and cannot imply anything of *how* it may be employed by a model of linguistic performance. As a result, a theory of processing which abides by the *Head Projection Hypothesis* is no more or less principle-based, than one which does not, other things being equal.

Despite this, Pritchett does argue that HPH must be operative in a "strongly principle-based parser", although he suggests this as a consequence of the Projection Principle: "...which holds that each level of syntactic representation is a uniform projection of the lexical properties of heads"[27] [Pritchett ar]. As a result, Pritchett adopts a head-driven control strategy, which is a perfectly acceptable approach on its own, although there is no reason to suggest it is "...as a natural consequence of the Projection Principle", or any other principle for that matter. To sum up, while the use of a head-driven strategy can be regarded as an instance of a grammar-based approach — given the distinguished status assigned to heads by $\overline{X}$-theory — it is not a necessary condition for a principle-based parser or even a strongly principle-based parser[28]. Determining the principles of grammar which are directly reflected in the parser's operation (i.e. strongly PB) is a matter for empirical enquiry, and on the grounds of incremental interpretation, the head-driven model appears to fail.

In other regards, however, Pritchett's model successfully achieves a grammar-based status and can be considered strongly principle-based. Specifically, he suggests that attachment preferences are determined on the basis of syntactic licensing obligations, and furthermore characterises the cost of reanalysis in terms of government: a fundamental structural configuration of the grammar. The fact that these decisions are made on the basis of grammatical

[27] This is perhaps even less motivated than Frazier's HPH which was based on $\overline{X}$ theory, since the Projection Principle (as developed in [Chomsky 81a]) is not concerned with the well-formedness of particular phrasal projection, but rather is intended to ensure that lexical properties of heads are honoured consistently at each level of representation (DS, SS, and LF).

[28] Recall the distinction we have made in chapter 1: A principle-based parser makes direct use of the principles of grammar on-line, while a *strongly* principle-based parser makes decisions (say, in the face of ambiguity) which are further influenced by the individual principles, i.e. some may have higher priority etc.

information, rather than purely computational or representational concerns, qualifies the system as grammar-based.

The only point of this section has been to drive home that fact the principles of a particular grammar have no *a priori* procedural interpretation, nor is one implied. While this should be clear, conflation of these issues still occurs. The suggestion that the grammar *is* the parser, is simply not well-formed. One interesting aspect of Pritchett's work, however, is that it explores the converse of this notion: the idea that certain phenomena traditionally accounted for in the grammar (such as the blocked extraction from adjuncts[29]), may simply be a consequence of the processors operation [Pritchett ar]. In this way it provides a potentially functional explanation of grammatical conditions, in the manner of Marcus [Marcus 80] and Berwick and Weinberg [Berwick & Weinberg 84].

### Discussion

Pritchett's grammar based approach has a strong attraction due to its ability to explain conscious garden-paths in terms of fundamental grammatical principles. While Pritchett does make some unfounded arguments to motivate his head-driven approach in terms of grammatical principles as discussed above, this choice of strategies is perfectly legitimate, and the strategies it employs are defined in terms of the content of syntactic relations. Crucially, however, the model of processing is non-incremental — any number of phrases of possibly arbitrary length may be left unattached until a licensing head is encountered, permitting structure to be built. While this model seems to provide a good characterisation of conscious garden paths, it is incapable of characterising the more subtle processing phenomena which seem to support the notion of strictly incremental interpretation and subsequent reanalysis. Consider for example the evidence presented in [Mitchell & Holmes 85] indicating that there is an initial tendency for people to attach the NP as object of the initial verb as in (21) above, even when the verb is obligatorily intransitive, suggesting some attachment strategy which, at least initially, operates on the basis of category information only (see also [Clifton 90] for further support of this claim).

In addition, Pritchett adopts a purely syntactic notion of thematic roles. The Re-Licensing Constraint, which is solely responsible for predicting (conscious) garden paths, provides no mechanism for incorporating pragmatic information into the reanalysis process, although it might be possible to extend the theory in this respect. That is, while the syntactic account of garden paths may be fundamentally correct, Pritchett's model gives no explanation for the variation in complexity observed in reduced-relative garden paths, as determined by semantic and pragmatic factors [Crain & Steedman 85]. Indeed the distinction between unconscious garden paths, conscious but recoverable garden paths, and unrecoverable garden paths becomes hazy when there exists biasing contextual and pragmatic interference, and Pritchett's model simply fails to address this issue in any way.

More positively, however, the idea that the sentence processor is driven by a desire to satisfy syntactic relations has a much stronger appeal than a theory such as Frazier's where such relations are really just filters on representations which have been postulated for independent

[29] This has been syntactically defined as the *Condition on Extraction Domain* (CED), to be discussed in §3.

reasons. As we mentioned earlier, this is particularly appealing in the context of principle-based grammars where syntactic representations play a less central role. Furthermore, if the purpose of the sentence processor is to recover the thematic interpretation of an utterance, then it seems reasonable that the basic operative strategies are based on this concern.

## 2.4 Conclusions

In this chapter we have explored the existing literature in an attempt to identify the various issues relevant to a theory of sentence processing and the nature of the empirical evidence which we can bring to bear in our study. We have further examined existing theoretical approaches in an effort to identify the fundamental principles and assumptions upon which they are based, and determine their descriptive and explanatory adequacy with respect to the range of empirical phenomena. We have assumed without significant argument that the language faculty is modular, roughly in the Fodorian sense. Further, we have motivated the existence of distinct lexical and syntactic systems within this faculty: the theoretical motivation following from a generalisation of Fodor's hypothesis. That is, Fodor characterises modules roughly as domain specific computational systems which are both fast and informationally encapsulated. A generalisation of this hypothesis suggests that any process which satisfies these criteria is intrinsically modular. That is to say, modularity is a paradigm to be employed whenever possible.

The evidence in support of this position is far from conclusive. There does exist evidence for stages of processing, lending support to the modularity hypothesis. In contrast, however, it is clear that the knowledge sources correlating to these stages of processing also do interact in some way. While processing seems to be *driven* by input in a manner compatible with a modular organisation, it is quite apparent that top-down information can facilitate reanalysis quickly after this analysis is constructed. This does not necessarily weaken the modular position, but it does require that we account for the nature of module interaction in a complete theory of processing.

In examining prevalent theories of processing we have identified three basic approaches. These models are distinguished by which aspect of language processing they consider to be most relevant in determining the processors operation:

(26) **Computational Efficiency:** Motivated by the desire the typical speed exhibited by the HSPM.

**Representational Parsimony:** Intended to minimise the representational complexity of the syntactic analysis.

**Grammatical Basis:** Operates so as to maximally satisfy the rules of the grammar.

It is uncontroversial to assume that all of these aspects must be taken into consideration in any theory: the question is, which is relevant for determining the actions of the sentence processor. The first two are motivated by classical theories of computational complexity:

we can formally characterise the complexity of an algorithm in terms of the number of operations it performs and the amount of memory required, i.e. time and space. To decide between these two positions we would have to decided which resource is at more of a premium in the mind, processing power or memory, and then optimise the more costly of the two. The grammar based approach suggests that the need to satisfy syntactic relations is the foremost consideration of the processor. This is not to say that such a model need be inefficient, but rather to simply suggest that computational considerations are not so restricted that they fundamentally determine processing.

One problem of the modularisation of syntactic processing, is that there has been a tendency to isolate the parsing task. Not only do many existing theories fail to account for semantic interaction, but we can further observe that all of the models of processing discussed thus far are driven exclusively by the desire to optimise some aspect of *syntactic* processing, without regard to more general task of language comprehension. To exemplify this, suppose that semantic processing of syntactic analysis occurs concurrently, as the syntactic analysis is constructed. If the syntactic processor buffers constituents (for its own purposes, as do the models proposed by [Berwick & Weinberg 84] and [Pritchett ar]), then it delays constructing the maximal partial analysis possible, and prohibits the semantic processor from constructing an incremental interpretation of the utterance. Similarly, the strategies which control Frazier's parser are purely structural, and while they operate incrementally, they are not particularly sensitive to the more general aims of comprehension. In this respect, the models presented do not consider the possibility that the operation of the syntactic processing may be determined, at least in part, by a desire to optimise the more general task of comprehension.

In the coming chapters we develop a model of sentence processing which begins roughly with the organisational assumptions of Frazier; distinct, encapsulated syntactic and semantic modules which operate concurrently. In contrast, however, we will aim to provide a set of processing strategies which are grammar-based, departing from the purely structure-oriented strategies proposed by Frazier. In this way we hope to incorporate the insights of Pritchett's highly principle-based account while providing a more thorough account of relative complexity effects. Finally, we will suggest that the syntactic processor must simultaneously satisfy the more global requirements of incremental comprehension, and that the strategies are not only based upon grammatical notions but also operate so as to meet this demand.

Before we proceed with the details of the process model, we will first take a closer look at the principles and parameters model of grammar upon which it is based. The aim here is to identify precisely the content of the syntactic analysis which must be recovered, the way in which principles of grammar constrain these analyses, and the properties of these constraints. We will suggest that current syntactic theory may be more naturally viewed as conditions on more than one type of representation, and that this in turn suggests a more articulated, modular organisation of the syntactic processor. In chapter 4 we present the model, discuss the particular strategies which are operative within each processor, and reconsider the empirical evidence. Finally in chapters 5 and 6 we explicate the computational model which makes clear the operational details of the proposed model.

# Chapter 3

# Principles, Parameters and Representations

We remarked in chapter 1 that any interesting proposal concerning human linguistic performance must make reference to the structure of language, i.e. the representational form and informational content which determines the interpretation of utterances. As Chomsky has pointed out, without reference to such structure, theories of performance will be inherently superficial [Chomsky 80] and uninteresting. Furthermore, to the extent that any proposed model of processing claims to be (at least partially) innate, it will crucially depend upon the knowledge of language determined by UG.

In this chapter, we will examine in some detail the nature of current theorising about UG, as it has developed within the principles and parameters framework. There are two basic aims here: The first is to give the reader a feel for the theory itself and how it has resulted from the move towards linguistic explanation. The second is to identify the fundamental representational and informational components of the model and their implication for models of performance. That is to say, a process model must in the first place consider *what* information and structure is entailed by the grammar, before proceeding to the details of how the analysis is constructed and the strategies involved[1].

Following a recent proposal, we adopt a model of grammar which can be considered 'orthogonal' to the traditional derivational organisation. The basic motivation for the new model is to make explicit the types of information and representations which contribute to the analysis of utterances. Most notably, we adopt from the reinterpretation of the derivational component in terms of the representational notion of Chains, thus resulting in a mono-stratal model of grammar — a proposal currently receiving much favour in the literature. In addition, we propose factoring out other aspects of syntactic representation, and identifying the principles of grammar which 'cluster' around them. Crucially, we will try to demonstrate that the proposed model of grammar is isomorphic to the derivational one in all relevant respects, and that the principles of grammar can be naturally stated

[1] That is, if we assume that the processor recovers analyses which are strongly equivalent to those of the grammar — regardless of whether or not grammatical principles are used directly (recall the discussion in §1.1).

in either system. By adopting the purely representational approach, however, we hope to illustrate that there is no *a priori* motivation to ascribe a procedural interpretation to the derivational component of the grammar, when constructing a model of performance — a misconception which has plagued several proponents and critics of the transformational approach to syntax, as we discussed in §2.3.3.

We will begin with a brief review of the research programme (section 3.1) and an overview of the transformational model (section 3.2). These two sections are intended primarily as an introduction/overview for readers less acquainted with the area, but also define the version of the theory I am assuming here, so as not to confuse readers who are familiar with alternative accounts. In section 3.3, we propose the representational approach, which replaces the derivational model. While the two models appear superficially different, we will endeavour to highlight the underlying equivalence of the two systems. We will also provide a relatively formal characterisation of the representational system, and discuss some potentially relevant computational properties.

## 3.1 Explanation in Universal Grammar

In chapter 1 we observed that, while Chomsky has separated the issues of linguistic competence and acquisition, the two must meet some interface conditions — put simply, the grammar must be learnable. Indeed, the desire to explain problems of language acquisition has influenced modern syntactic theory since as early as *Aspects of the Theory of Syntax* [Chomsky 65]. In particular, the theory has sought to account for the ability of children to achieve a highly sophisticated knowledge of language in the face of linguistic experience which is considered to be both degenerate and deficient, along the lines outlined by Hornstein and Lightfoot:

(27) (i) Children hear speech which does not consist uniformly of complete grammatical sentences, but also utterances with pauses, incomplete statements, slips of the tongue, etc.

(ii) Despite being presented with finite data, children become able to deal with an infinite range of utterances.

(iii) People attain knowledge of the structure of language, despite the absence of such data. That is, people are able to make judgements concerning complex/rare sentences, ambiguity relations, and grammaticality using knowledge which is not available as primary linguistic data (PLD) to the child.

[Hornstein & Lightfoot 81]

The essential problem is to explain how we come to have such a rich and highly structured knowledge of language, despite it being underdetermined by the data which we encounter during the acquisition process, i.e. despite the poverty of stimuli outlined in (27). To address this problem, it has been proposed that people are endowed with some *a priori* knowledge of language; a set of 'genetically encoded' principles which form the basis of an innate language faculty [Lightfoot 82]. In this context, the acquisition process can be considered a parameter setting operation: The child begins at the initial state $S_i$ with an uninstantiated set of principles. The parameters then become set via some *Language Acquisition Device* (LAD) which interprets the data presented to the child (for further discussion see [Chomsky 81b]). After sufficient information and experience, all the parameters are set and the child has a knowledge of the core grammar[2] (i.e. the final state, $S_f$).

While traditional efforts in syntactic theory concerned themselves primarily with descriptive accounts for particular languages, the innateness hypothesis introduces further demands. We must identify both the innate principles of grammar, and the parameters which are acquired on the basis of linguistic experience. Furthermore, there are some fundamental constraints on the nature of both principles and parameters: Firstly, it is uncontroversial that a child has the ability to acquire any natural language to which she is exposed, entailing that the principles of grammar be sufficiently abstract and universal to account for all attainable languages. Crucially however, principles must be sufficiently rich to explain acquisition given the poverty of stimuli. That is to say, we must minimise the degree and complexity of parameter setting, such that acquisition is possible despite the deficiencies outlined in (27). Interestingly, however, this tension between the rich and the abstract has played a fundamental role in the development of an explanatory theory of universal grammar.

## 3.2 The Transformational Model

The early transformational model, as proposed in *Syntactic Structures* [Chomsky 57], presupposes two levels of syntactic representation: *deep-structure* (or, D-structure) and *surface-structure* (or, S-structure)[3]. The motivation for this model was to provide a level of representation for the "canonical" representation of the semantically relevant grammatical functions (D-structure), and a level which permitted various surface permutations of the base constituent structure via transformational rules (i.e. S-structures). Consider the following ubiquitous example of the passivisation rule:

(28) a. Mozart [$\text{TNS}_{past}$] compose the requiem.
b. The requiem was composed by Mozart.
c.

| | X | NP | AUX | V | NP | Y | by | Z |
|---|---|---|---|---|---|---|---|---|
| SD: | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| SC: | 1 | 5 | 3+be | 4+en/ed | $\phi$ | 6 | 7+2 | 8 |

This rule states that the simple, active clause *Mozart composed the requiem*(28a) may be passivised as in (28b) via the rule template given in (28c). The rule defines a structural change (SC), permuting the original structure description (SD) by moving the direct object into the former subject position, moving the subject into an indirect object *by* phrase,

[2] By 'core grammar' we simply wish to abstract away from the idiosyncratic and idiomatic aspects of language (the 'periphery') which are not relevant to initial language learning based on UG.

[3] In fact, the notions of 'deep' and 'D' structure are not coextensive, and the same can be said for 'surface' and 'S' structure. Indeed, the abbreviated notation was adopted by Chomsky explicitly to avoid confusion in the literature. For our purposes however, we may take them to be similar.

and adding a *be* auxiliary (with appropriate tense). In this way, we capture the intuition that both active and passive instantiations of a clause share a common underlying structure. Using numerous rules of this sort, it was possible to describe various transformations including heavy-NP-shift, subject-raising, and WH-movement.

While the transformational mechanism provided a powerful device for factoring apart issues of canonical constituent structure from numerous possible surface realizations, the rules involved where richly articulated, rather vast in number, and highly language specific in nature. These properties of the grammar, however, bring us no closer to explaining acquisition, since the rules cannot be universal and innate, and their complexity and number present a problem for acquisition from linguistic experience, as discussed above. This observation signalled a rather radical shift in the direction of research, away from systems of rules such as those outlined above, to systems of principles of language universal status.

Perhaps the most significant contribution of this shift from rules to principles was the replacement of construction-specific transformational rules (such as (28) above) with the single operation Move-$\alpha$ [4]. Furthermore, the revised model of grammar incorporated a two 'interpretive' levels of representation: the LF (Logical Form) component, and the PF (Phonetic Form) component. Both are derived from S-structure, resulting in the so-called *T-Model*, as illustrated in Figure 3.1.

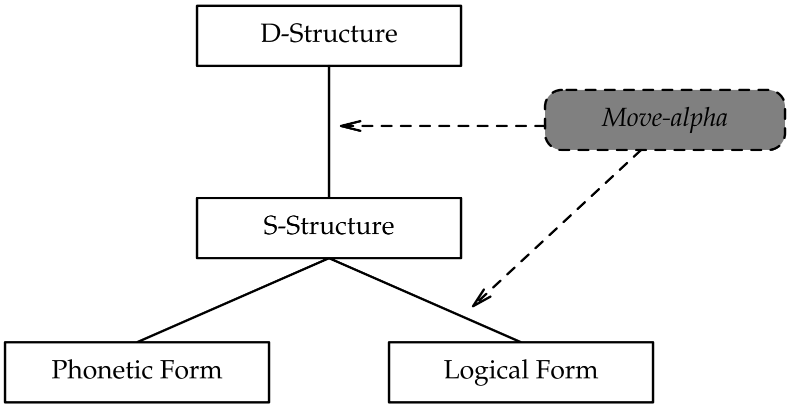



Figure 3.1: The T-Model of Grammar

Within this model, the generation of D-structures is performed by the *Base Component*, which determines well-formed constituent structures on the basis of the selectional requirements of lexical items (transitivity, subcategorization, etc.) and the rules of phrase structure. The *Transformational Component* maps such a D-structure representation into a corresponding S-structure via the application of Move-$\alpha$. Alone, the transformational rule Move-$\alpha$ — 'move anything, anywhere' — will severely overgenerate possible S-structures (and, indeed, LF which is another syntactic level of representation). This in turn has led

[4] This move did not take place in one fell, swoop, but rather took place in stages, first collapsing 'classes' of movement rules into single rules, such as NP-movement (passive, raising, etc.) and WH-movement (long-distance movement of WH-constituents). For a thorough, chronological account of the stages leading up to the present model, the reader is referred to [vanRiemsdijk & Williams 86].

to the search for independent principles of grammar which will conspire to rule out ill-formed structures resulting from movement. As we alluded to above, these principles are proposed as innate universals, with possible parameters of variation, thereby simplifying the acquisition process.

The principles of GB theory interact so as to impose well-formedness conditions at the various syntactic levels of representation (i.e. D-structure, S-structure, and LF). These subsystems of principles, as outlined in [Chomsky 82] are:

(29) a. $\overline{X}$-theory
b. $\theta$-theory
c. Case theory
d. Binding theory
e. Bounding theory
f. Control theory
g. Government theory

In the remainder of this section we will give a brief introduction to the various syntactic subsystems, emphasising the way in which they constrain, and indeed force, the application of Move-$\alpha$.

### 3.2.1 $\overline{X}$-theory and Lexical Selection

We noted above that the base component is concerned with the generation of D-structures, on the basis of phrase structure rules, lexical insertion, and the selectional properties of lexical items. Traditionally, phrase structure rules took the following form:

| | | | | | |
|---|---|---|---|---|---|
| (30) | a. | S | $\rightarrow$ | NP Aux VP | |
| | b. | NP | $\rightarrow$ | Det N (PP) | |
| | c. | VP | $\rightarrow$ | V NP PP | *(put, give)* |
| | d. | VP | $\rightarrow$ | V NP | *(hit, kiss)* |
| | e. | VP | $\rightarrow$ | V | *(sleep, sneeze)* |
| | f. | PP | $\rightarrow$ | P NP | *(in, on)* |
| | g. | PP | $\rightarrow$ | P | *(away)* |

Where multiple productions are possible for a given phrase (as for the VP rules (30c) - (30e) above), the appropriate one is determined by the *subcategorization frame* of the head of the phrase (example candidates are given after the relevant rules above). The subcategorization frame specifies the precise set of complements which are permitted for a particular 'head'. So the lexical entry for *put* might include the following subcategorization information:

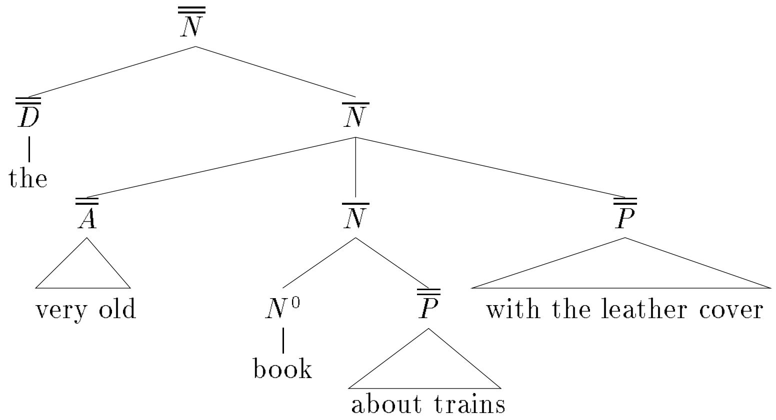
(31) *put:* < ___ NP PP >

Given that the lexicon encodes information concerning the categorial selection (c-selection) requirements (roughly, the arguments) of particular lexical items, the set of rules in 30 seems rather redundant. We might, for example replace the three rules (30c) - (30e), by a single rule: VP $\rightarrow$ V Complements , where *Complements* is determined directly by the subcategorization frame for the phrasal head $V$. Indeed, such patterns occur systematically across phrasal categories, as illustrated by the PP rules (30f) & (30g). The observation of such redundancies in the phrase structure system motivated formulation of $\overline{X}$-theory ; a generalised schema for the representation of phrase structure:

(32) a. $\overline{\overline{X}} \rightarrow$ Specifier $\overline{X}$
b. $\overline{X} \rightarrow$ Adjuncts $\overline{X}$ Adjuncts
c. $\overline{X} \rightarrow X^0$ Complements
d. $X^0 \rightarrow$ Lexeme

The 'bar levels' exist to distinguish the hierarchical status of satellites: Complements, as discussed above, are taken to be sisters of the head $X$ (written $X^0$ here, to be explicit). Modifiers (or, 'adjuncts'), occur at the next higher level, sister to $\overline{X}$, while specifiers appear directly under the maximal projection $\overline{\overline{X}}$ (or, XP). The final rule (d) above simply allows lexical insertion, where X may be any lexical category[5], N, V, A or P. As an example of $\overline{X}$ phrase structure, consider the following NP:

(33)




In this structure, the noun *book* takes a determiner phrase as its specifier[6], is modified on the left by the AP *old* and on the right by the PP *with the leather cover*, and licenses an *about*-PP as it's complement. In addition to determining the phrase structure for lexical categories, the theory has more recently been extended to handle the $S$ and $\overline{S}$ phrases, which were previously handled by the following exceptional rules:

(34) a. S $\rightarrow$ NP Aux VP
b. $\overline{S} \rightarrow$ Comp S

[5] The lexical categories are not "arbitrary". Rather, they can be considered abbreviations for the more fundamental *substantive* [+/− N] and *predicative* [+/− V] features. The correspondence is as follows: N = [+N -V], V = [-N +V], A = [+N +V], and P = [-N -V].

[6] Abney has proposed an alternative structure of DP and NP [Abney 87], which we will not go into here.

The generalised analysis takes Infl and Comp as *non-lexical* categories, which are subject to the rules of $\overline{X}$-theory in the same way as lexical categories [Chomsky 86a]. That is, take Infl (= I) to be the head of IP (= S) and Comp (= C) to be the head of CP (= $\overline{S}$). Under this analysis, it seems natural to let the subject NP be a specifier to IP, and assume that VP is a complement which is inherently selected for by I. Additionally, C selects for an IP complement:

(35) a. $\overline{\overline{C}} \rightarrow \overline{\overline{X}} \quad \overline{C}$
b. $\overline{C} \rightarrow C^0 \quad \overline{\overline{I}}$
c. $\overline{\overline{I}} \rightarrow \overline{\overline{N}} \quad \overline{I}$
d. $\overline{I} \rightarrow I^0 \quad \overline{\overline{V}}$

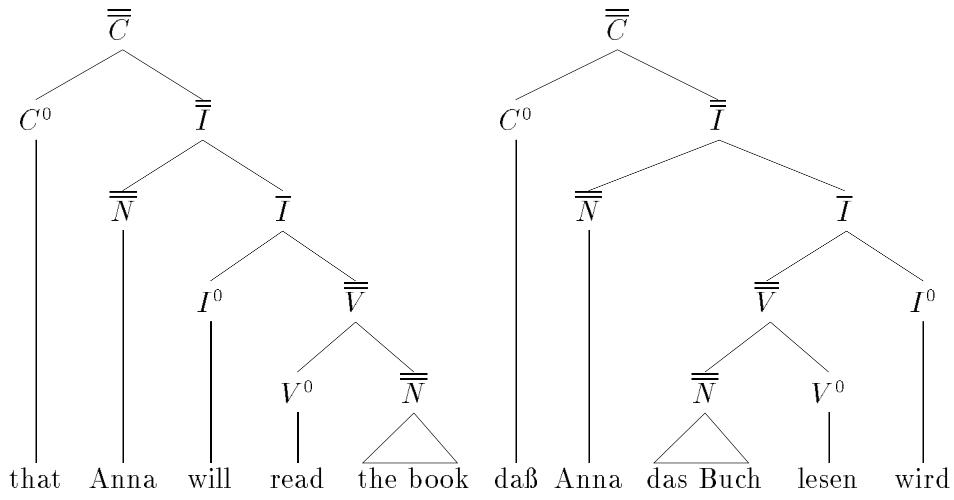
As we will see in the next section, the [Spec,CP] position, occupied by $\overline{\overline{X}}$ — while not typically filled at D-structure, provides a useful landing site for moved operators (roughly, WH-elements). While the rules of (32) (including (35)), are sufficient for describing the constituent structure of English, other languages yield differing orderings of satellites with respect to their heads. We will therefore assume that the rules in (32) determine immediate dominance and sister-hood relations only, with linear precedence of satellites with respect to the head (or its projection) being parameterised for the individual categories of languages. It is worth pointing out that linear precedence may not be fixed within a language. While complements always follows their heads in English, for example, some languages, such as German and Dutch are mixed: PP, NP, and CP are head-initial while VP and IP are head-final. Consider the following (embedded)[7] sentence structures for equivalent English and German clauses:

(36) a. "...that Anna will read the book."
b. "...daß Anna das Buch lesen wird."
*...that Anna the book read will*

The respective constituent structures for these examples are illustrated in the tree diagrams below:

[7] We illustrate the embedded sentence since it better represents the canonical structure of a clause. There is good reason to suggest that the verb-second order which occurs in German root clauses is the result of several movement operations.

(37) a. b.




The fact that the status of satellite positions (as either specifiers, complements, or modifiers) is determinable on the basis of the level of its sister and mother, suggests that we might collapse the rules in (32) even further. Indeed, if we combine the fact the linear precedence is parameterised with the recent assumption of binary branching (see [Kayne 84]) we can state $\overline{X}$-theory simply as follows:

(38) a. $\mathrm{X}^i \rightarrow (\mathrm{YP})\ \mathrm{X}^j$
b. $\mathrm{X}^0 \rightarrow$ Lexeme

where $i \leq 2$, $j \leq i$, and $0 \leq i, j$

We use the superscript integer here to denote bar-level. As above we assume that complements are sister to $\mathrm{X}^0$, modifiers to $\mathrm{X}^1$, and specifiers occupy the highest position under $X^2$ (= XP)[8]. Furthermore, we may require unary branches to reach the $\mathrm{X}^0$ level where lexical insertion is possible (hence the optionality of the YP satellite)[9].

The subcategorization frame of a lexical item, as we have discussed above, determines the constituents which may or must occur[10] as complements of that verb. Furthermore, all complements are taken to be 'arguments' of the head: i.e. constituents which are semantically selected (s-selected) as participants in the relation denoted by the verb (or other head). In

[8] Note these rules slightly overgenerate since they permit arbitrary sisters to $\overline{\overline{X}}$ — this may indeed be correct, since several current analysis assume the base generation of 'adjuncts' in such positions. We will return do a discussion of related matter in §3.2.2 and §3.3.2.

[9] Note this does present some problem in distinguishing the highest modifier from a specifier (if no specifier is present), since it too will occupy the highest position under XP. In section §3.3.2, we will suggest that the bar-levels be replaced with a feature system making the satellite positions completely unambiguous, and eliminating the need for vacuous unary branches.

[10] That is, some verbs, such as *send* s-select for a DESTINATION complement, as in *Alan sent the book to the publisher* but which may be absent as in *Alan sent the letter* (although arguably there is some understood, if possibly underdetermined, DESTINATION).

addition to the complements, a verb may also license an *external* argument in the subject position [Spec,IP][11]. The interface between the constituent structure and the semantic or *thematic* properties of lexical items is governed by $\theta$-theory, which is concerned with the assignment of thematic-, or $\theta$-roles to the arguments of a particular head. The set of roles which may or must be assigned is encoded in a thematic 'grid' (or, $\theta$-grid) associated with the particular relation. We might therefore extend the lexical entry given in (31) above to include the $\theta$-grid:

(39) *put:* < __ NP PP >
{ AGENT(EXT), PATIENT, LOCATION }

Now, in the sentence

(40) "Ian put the book on the desk."

the verb *put* has both NP and PP complements, thus honouring its subcategorization frame. In addition, *Ian* occupies the subject position — as an external argument to the verb. Each of these argument position must in turn receive a $\theta$-role from the $\theta$-grid for the governing verb. The overriding well-formedness condition imposed by $\theta$-theory concerns the 'coherence and completeness'[12] of $\theta$-roles assignment:

(41) **$\theta$-Criterion:** (i) each argument bears one and only one $\theta$-role, and
(ii) each $\theta$-role is assigned to one and only one argument.

The aim of the $\theta$-criterion is to ensure that every argument receives exactly one thematic interpretation, and make certain that no required participants are absent. It is important to note that while the head might not assign a $\theta$-role to the external argument position, all internal arguments are necessarily $\theta$-marked. This potentially undermines the need for a subcategorization frame in the first place (see [Lasnik & Uriagereka 88, page 4] for some discussion). As Chomsky points out, specifying that *hit* takes a PATIENT role appears to sufficiently determine the fact that it takes an NP complement [Chomsky 86b, page 86-92]. We might therefore suggest that all PATIENT roles are realised as NPs. This mapping may be simply defined by some 'Canonical Structural Realization' function which maps semantic categories into possible syntactic categories, i.e. CSR(PATIENT) = NP. Other roles, such as AGENT and GOAL may be similarly realised as NP's[13], while LOCATION, SOURCE, and DESTINATION roles are typically realized as appropriate PP constituents.

[11] There is an analysis which proposes base generation of the subject within the VP (possibly the [Spec,VP] position), followed by movement to [Spec,IP] [Manzini 88]. This has the advantage that no extra mechanism is required for dealing with the assignment of the verbs external role to a position outside the VP. We will take up discussion of this point in §3.3.4.

[12] This is in fact the name of the *Lexical-Functional Grammar* (LFG) principle which is roughly equivalent to the $\theta$-criterion [Bresnan & Kaplan 82].

[13] Although the CSR mechanism seems sensitive to the internal or external position of the argument since, while an AGENT in subject position is realised as an NP, it becomes a *by*-PP when internalized by the passivisation (unless this is the result of some transformation).

Interestingly, some roles may have multiple possible mappings in constituent structure. The verb *believe*, for example, s-selects for a PROPOSITION, which can apparently be realized by a clause (CP or IP) or an NP (which has an appropriate propositional interpretation):

(42) a. "I believe [$_{CP}$ that Quayle is a genius.]"
b. "I believe [$_{IP}$ Mulroney to be statesman.]"
c. "I believe [$_{NP}$ the statement about Thatcher.]"

Therefore the single $\theta$-role specified for the complement of *believe* sufficiently determines the three instantiations given above, if we assume CSR(PROPOSITION) = NP or CSR(PROPOSITION) = Clause (CP or IP). Note this eliminates the need for three different subcategorization frames, significantly reducing the amount of specific information required for each lexical entry. Clearly, this proposal may lead to overgeneration — not all verbs which permit clausal propositional complements also permit bare NP complements. The verb *seems*, for example, permits only a CP complement, not NP or IP. We will see in §3.2.3 that we can derive an account of such data in terms of Case theory, which interacts with thematic assignment to explain a number of interesting surface phenomena. For further discussion the reader is referred to [Chomsky 86b].

### 3.2.2 Types of Movement

At the beginning of this section, we pointed to the replacement of individual transformational rules with the rule Move-$\alpha$. We noted then that such a rule, if unconstrained, will severely overgenerate. Before proceeding to a discussion of the principles which indirectly control Move-$\alpha$, let's consider some fundamental restrictions on movement which have been proposed. One overarching principle which constrains the mapping between levels of representation is the *Projection Principle*, which may be stated as follows:

(43) **The Projection Principle:**
(i) Subcategorizable positions are $\theta$-marked by the governing head, at each syntactic level of representation (i.e. D-structure, S-structure, and LF).
(ii) $\theta$-marked positions must be represented categorially at each syntactic level of representation.

The Projection Principle operates so as to maintain the thematic interpretation of a sentence across the various levels of syntactic representation: i.e. the $\theta$-assignments determined at D-structure are maintained across the various levels of syntactic representation. The requirement that $\theta$-marked positions be represented at all levels further demands that the positions vacated by moved constituents be replaced with a some sort of 'empty category', ear-marking their vacated position. A further requirement, as stipulated by the *Extended Projection Principle*, demands that every clause have a subject. This is done to account for two related phenomena; the required insertion of a pleonastic *it* or *there* element[14]

[14] These are taken to be 'dummy' lexical NP's with no semantic content; requiring Case but no $\theta$-role. Interestingly, these elements have different distributional properties, but we will not consider them here. Note, these should not be confused with their referential counterparts which typically act as pronouns.

in non-$\theta$-marked subject position, and the availability of this position for raising of lower constituents:

(44) a. "*It* seems John likes Mary."
b. "$John_i$ seems $\varepsilon_i$ to like Mary."

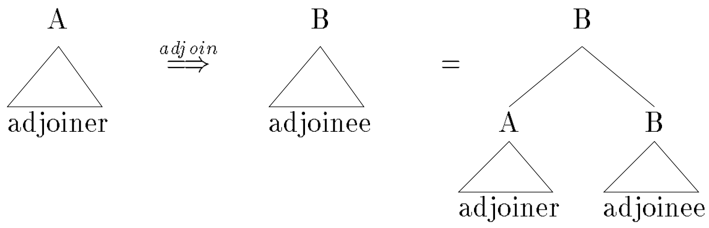
Following [Chomsky 86a, page 4] we will assume two fundamental types of movement; substitution and adjunction. Substitution — movement into an existing, base-generated position — is taken to have the following properties:

(45) a. There is no movement to a complement position.
b. Only $X^0$ can move to a head position.
c. Only $\overline{\overline{X}}$ can move to a specifier position.
d. Only $X^0$ and $\overline{\overline{X}}$ are visible to Move-$\alpha$.

In the previous subsection, we stated that heads necessarily assign a $\theta$-role to each of their complements at D-structure, thus we can derive (45a) above from the fact that a constituent moving into a complement position would necessarily be $\theta$-marked by the verb which selected for that position (regardless of syntactic level, by the Projection Principle). This would violate the $\theta$-criterion on at least two counts: Firstly, the verb would be assigning a particular $\theta$-role twice (i.e. to the constituent which vacated the position, and the constituent which moved into it), and second, the element which moved into the position will have been previously $\theta$-marked at its own base generated position, and hence would receive two $\theta$-roles.

If we accept, based on the Structure Preservation Hypothesis [Emonds 76], that maximal projections may only occupy satellite positions, and heads may only occupy $X^0$ positions (45b), then the corollary of (45a) is that only (non-$\theta$-marked) specifier positions are possible landing sites for maximal projections (45c) (as illustrated by (44) above). Finally, we stipulate that only maximal and minimal projections are visible for movement.

The adjunction transformation involves the creation of a new landing site, rather than movement into an existing one. Following [Lasnik & Uriagereka 88, page 5] we can illustrate adjunction as follows:

(46)



Following the spirit of conditions (45b) - (45d) above, we assume that the level of A equals that of B, and that it further be either maximal or minimal. That is to say, adjunction may only be head-to-head or phrase-to-phrase. Standard examples from English include the rightward adjunction (often called extraposition) of heavy constituents to VP[15], as follows:

(47) "Ian $[_{VP}$ $[_{VP}$ put $\varepsilon_i$ on my shelf $]$ $[_{NP_i}$ the book he had borrowed for weeks $]$ $]$"

Another example of adjunction, known as 'scrambling', occurs in languages with considerably less restricted constituent order, such as German. The scrambling transformation is taken to account for the preposing of objects within a clause — adjoining to either VP or IP — as illustrated by the following examples:

(48) a. "...daß der Junge gestern mit mir gespielt hat"
*...that the boy yesterday with me played has*
b. "...daß $[_{IP}$ der Junge $[_{VP}$ $[_{PP_i}$ mit mir$]$ $[_{VP}$ gestern $\varepsilon_i$ gespielt hat $]$ $]$"
c. "...daß $[_{IP}$ $[_{PP_i}$ mit mir$]$ $[_{IP}$ der Junge $[_{VP}$ gestern $\varepsilon_i$ gespielt hat $]$ $]$ $]$"

We have highlighted here the basic mechanism of Move-$\alpha$: The two types of transformation are substitution and adjunction. Elements visible for purposes of movement are restricted to maximal and minimal projections. And finally, the overarching Projection Principle entails that moved constituents leave a trace which is taken to be $\theta$-marked at all levels of syntactic representation, thus preserving the D-structure interpretation. In §3.2.5 we will turn our attention to the principles of *Bounding Theory*, which further constrain the application of Move-$\alpha$ in certain contexts.

Rather interestingly, however, we have discussed above how the Projection Principle interacts with another principle, the $\theta$-criterion, to constrain movement in a less direct manner, blocking movement to object position. This is an example *par excellence* of how independent principles may conspire to explain various restrictions on surface configurations. We will now turn our attention to another sub-theory of the grammar, Case theory, which has significant effect in both requiring and restricting movement.

### 3.2.3 Case Theory

The notion of Case is traditionally associated with particular grammatical functions: subjects are typically *nominative*, direct objects *accusative*, indirect objects *dative*, etc. The degree to which Case is morphologically realised is largely language dependent: In English is it primarily restricted to pronouns and possessive NP's as in the following example:

(49) "$He_{nom}$ said $John's_{poss}$ friend kissed $her_{acc}$."

Other languages such as German retain a relatively rich morphological Case system in which the nominative, accusative, dative, and genitive forms of most NP's are distinguished

---

[15] Whether or not extraposition involves adjunction to VP, or IP is a matter of some debate in the literature. See [Rochemont & Culicover 90] for arguments on this point.

(although not always uniquely). The minimal assumption of Case theory, however, is that NPs are assigned *abstract* Case, with the specifics of morphological reality being determined for individual languages[16].

More specifically, Case theory requires that every NP receive (abstract) Case, where an NP receives Case by virtue of occupying a position to which Case is assigned. A position is assigned Case if it is governed by a *Case-assigner*. For English the Case assigning categories are generally taken to be V, P, and I. The ability for V and P to assign Case must additionally be specified in the lexicon. If a lexical item can assign Case, it is said to be *transitive*, otherwise *intransitive*. In addition, I is a Case-assigner just in case it has the feature [+TNS][17]. The fundamental principle of Case theory, the Case Filter, is stated as follows:

(50) **Case Filter:** *NP, where NP is phonetically realized and has no Case.

Given the Case assigners outlined above, consider the distribution of subject NPs in tensed and untensed clauses:

(51) a. * "It appears [$_{IP}$ John to like Mary ]."
b. "John$_i$ appears [$_{IP}$ $e_i$ to like Mary ]."
c. "It appears [$_{CP}$ that John likes Mary ]."
d. * "John appears [$_{CP}$ (that) $e_i$ likes Mary ]."

As predicted the (51a) is ungrammatical, presumably because John does not receive Case from the untensed head of Infl. Furthermore, if we move John to the subject position of the tensed root clause, as in (51b), the sentence becomes grammatical. Note this movement does not violate the $\theta$-criterion, since *appears* does not $\theta$-mark its subject (similar to the *seems* example in (44) above) so *John* only receives a $\theta$-role from *likes*. Interestingly, however, in sentences where the NP *is* Case-marked in a lower tensed clause (51c), raising is blocked, as indicated by the ungrammaticality of (51d). This suggests that, in fact, NP must receives Case *exactly* once. Chomsky has suggested that this requirement can be re-cast in terms of *Chains* (see [Chomsky 81a], Chapter6) — a meta-theoretic representation of the 'history of movement' for a particular phrase. If we assume that a Chain represents the set of all positions occupied by a constituent en route from D-structure to LF, then the Case Filter may be re-formulated to require that every Chain be assigned Case exactly once[18]. In this way we provide a natural account for the well-formedness of (51b&c), where the Chain for John is assigned Case once (in the higher and lower clauses respectively), and the

---

16 Note, it is may be that the degree of morphological Case is dependent upon other aspects of syntax, since it seems that languages with greater freedom of constituent order, such as Latin and Finnish, have richer Case, while languages with relatively impoverished Case, such as English and Dutch have relatively fixed constituent order. We will not, however, consider this issue here.

17 Another point of view is to consider AGR a Case-assigner, not I[+TNS], since AGR is present (or, *rich*) only for I[+TNS].

18 So that we may generalise this formulation of the Case Filter, we assume that NPs *in situ* form a Chain of unit length (at least at S-Structure), which must similarly be assigned Case exactly once.

ungrammaticality of (51a&d), in which the Chain receives either no Case or receives Case twice.

There are, however, instances when the subject of an untensed clause does not need to move. Consider,

(52) a. "I believe [$_{IP}$ John to like Mary ]."
b. "I believe [$_{CP}$ that John likes Mary ]."

As we would expect, (52b) is grammatical, just as (51c). How then can we explain the grammaticality of (52a)? The solution adopted here is to assume that government[19] may take place across the maximal projection IP (=S). The recent position on this is to assume that if $\overline{\overline{X}}$ is governed, so is its specifier, $SPEC, XP$ and its head, $X^0$ [Chomsky 86a]. Therefore, since *believe* governs the IP , the subject, $SPEC, IP$, is governed and thus can be Case-marked. This permits the matrix verb to Case-mark the subject of the embedded clause, rendering (52a) grammatical. The difference then between the verbs *believe* and *appear* is therefore accounted for in terms of their specific lexical properties: we assume that *believe* is transitive, permitting it to assign Case, while *appear* is not. This process is generally referred to as *Exceptional Case Marking* (ECM).

Corroborating support for this analysis can be found if we reconsider the possible complements licensed by *believe* versus *seem* or *appear*. Recalling our discussion at the end of 3.2.1, we remarked that *believe* takes a PROPOSITION complement, which may be realised as either NP, IP, or CP. If we assume that *appear* (and *seem*) similarly license a PROPOSITION complement, then we might naturally expect the similar categorial range of possible complements:

(53) a. "It appears [$_{CP}$ that Quayle is a genius.]"
b. * "It appears [$_{IP}$ Mulroney to be statesman.]"
c. "Mulroney$_i$ appears [$_{IP}$ $\varepsilon_i$ to be statesman.]"
d. * "It appears [$_{NP}$ the tabloid press.]"

If we assume as above, however, that *appear* is intransitive, then we can rule out both (53b&d) on the grounds of a Case Filter violation: *Mulroney* is neither Case-marked by the embedded *to* nor by appears (via ECM), and *the tabloid press* is cannot receive Case from *appear* either[20]. Example (53c) simply reiterates that the IP complement is grammatical if the embedded subject raise to the Case-marked subject position of the root clause, as discussed above.

[19] The notion of government is discussed in in greater detail in the next section. For present purposes we just assume that all heads govern their complements.

[20] Note, it does seem difficult to construct an example of appropriate propositional content for the verbs *appear* and *seems*, so this might be ruled out on pragmatic grounds.

### 3.2.4 Command Relations and Government

Central to GB theory is the notion that many of its principles prescribe certain *locality* constraints on the relations between items in the tree. That is, a given principle defines a *domain* which is local with respect to the constituent that principle concerns. The most pervasive, and indeed fundamental, of these structural domains is that of *government*. Government is itself defined in terms of the less restrictive notion of *m-command*. We take these to be defined as follows[21]:

(54) **m-command:** $\alpha$ *m-commands* $\beta$ iff every maximal projection dominating $\alpha$ dominates $\beta$.

We may now define government as follows:

(55) **government:** $\alpha$ *governs* $\beta$ iff
(i) $\alpha$ m-commands $\beta$, and
(ii) $\alpha$ is a head (i.e. $X^0$), and
(iii) $\beta$ m-commands $\alpha$

The theory of Government plays a central role in determining the distribution of empty categories. Specifically there are two types of empty categories: *traces*, which occupy the positions which have been vacated by movement, and *PRO*, which is a phonetically null pronominal anaphor. The basic well-formedness condition for empty categories is known as the *Empty Category Principle* (ECP), which states that all traces must be *properly governed*. This might be extended, to include the observation that PRO must be ungoverned. This *Extended ECP* [Chomsky 82] can be stated simply as follows:

(56) **Extended ECP:** If $\alpha$ is an empty category, then
(i) $\alpha$ is trace iff it is *properly governed*
(ii) $\alpha$ is PRO iff it is ungoverned.

where,

(57) **Proper Government:** $\alpha$ *properly governs* $\beta$ iff
$\alpha$ governs $\beta$, ($\alpha$ a lexical $X^0$) or
$\alpha$ locally $\overline{A}$-binds $\beta$

Where the notion of *locally $\overline{A}$-bound*, entails that the empty category (henceforth, *ec*) be bound by an $\overline{A}$ position which is "not too far away".

In general then, if an empty category is governed, it is a *trace*, and if not it is *PRO*. We may also assume that *PRO* may only appear in subject position, since the object position is necessarily governed by its subcategorizing head. To see how the ECP applies, consider the following sentences:

[21] These definitions have been adapted from a variety of sources in the current literature.

(58) a. "Why$_i$ does Martin want [ PRO to be an academic ] $\varepsilon_i$?"
b. "Who$_i$ does Martin want [ $\varepsilon_{i'}$ [ $\varepsilon_i$ to be a train spotter ] ] ?"

In (58a) we see that the embedded subject position is ungoverned, and therefore must be occupied by PRO. In (58b) however, we see that the subject gap $\varepsilon_i$ is locally bound by $\varepsilon_{i'}$, which is in the [SPEC,CP] position (an $\overline{A}$-position). The subject *ec* of (58b) is therefore properly governed, and must be a trace.

### 3.2.5 Bounding Theory

The previous sections have outlined how principles constrain possible landing sites for movement. These constraints are imposed primarily by the $\theta$-criterion, the Projection Principle, and Case theory. Bounding theory constrains movement directly by prohibiting a constituent from being moved "too far" by a single application of Move-$\alpha$. Additionally, the theory imposes certain *island* constraints, where islands are taken to be domains out of which no constituent can be moved. In early work on transformational grammar, constraints on movement where stated as individual rules. Classic examples of such constraints were those proposed in [Ross 67]:

(59) **Complex NP Constraint (CNPC):** No element in an S dominated by an NP, may be extracted from that NP:

* "Who$_i$ do [$_S$ you like [$_{NP}$ the book that John gave to $\varepsilon_i$ ] ]."

(60) **Wh-Island Constraint (WhIC):** No element contained in an indirect question, $\overline{S}$, may be moved out of that $\overline{S}$:

* "What$_j$ do [$_S$ you wonder [$_{\overline{S}}$ who$_i$ [$_S$ $\varepsilon_i$ bought $\varepsilon_j$ ] ] ]."

(61) **Sentential Subject Condition (SSC):** No element may be extracted from an S, if that S is a (sentential) subject:

* "Who$_i$ did [$_S$ [$_{NP}$ [$_S$ that she dated $\varepsilon_i$ ] ] bother you ]."

We can summarize these constraints by requiring that $\phi$ not be related to some trace $\psi$ in the following contexts:

(62) CNPC: $\ldots\phi\ldots[_S\ldots[_{NP}\ldots\psi\ldots$
WhIC: $\ldots\phi\ldots[_S\ldots[_S\ldots\psi\ldots$
SSC: $\ldots\phi\ldots[_S\ldots[_{NP}\ldots[_S\ldots\psi\ldots$

In an attempt to capture these constraints in a principled fashion, Chomsky introduced the Subjacency Condition [Chomsky 73], a formulation of which is stated below:

(63) **Subjacency:** A singular application of Move-$\alpha$ may not cross more than one *bounding node*. That is, $\phi$ may not be related to $\psi$ in the following context:

$\ldots\phi\ldots[_\alpha\ldots[_\beta\ldots\psi\ldots]\ldots]\ldots\phi\ldots$
where $\alpha$ and $\beta$ are bounding nodes.

Under this notion of Subjacency, the bounding nodes for English were generally taken to be NP and S. As we can see, these choices for bounding nodes account for all the island constraints in (62). The choice of bounding nodes was however shown to be subject to parametric variation across languages, such as Italian where evidence suggested the bounding nodes were NP and $\overline{S}$ [Rizzi 82].

The Subjacency condition is not sufficient however, to account for all possible island effects. Consider for example the following sentences:

(64) a. "Who$_i$ did you read [ a book about $\varepsilon_i$ ] ?"
b. * "What$_i$ did you read [ a book under $\varepsilon_i$ ] ?"

While (64a) is perfectly grammatical, (64b) is not, with the intended reading: *What was the book that you read resting under ?*. To account for this phenomena, Huang proposed the *Condition on Extraction Domain* [Huang 82], which can be stated as follows:

(65) **Condition on Extraction Domain (CED):** A phrase $\alpha$ may be extracted out of a domain $\beta$ only if $\beta$ is properly governed.

The formulation of proper government adopted was assumed to exclude adjuncts. If we take *book* to optionally select for an "about" PP, then extraction out of the PP is possible, as in (64a) (since the PP will be properly governed). Extraction of the locative PP adjunct in (64a) is ruled ungrammatical by the CED principle, since the PP is not properly governed[22].

In an attempt to account for both the CED and Subjacency with one principle, Chomsky has proposed the concept of *barriers* [Chomsky 86a]. The work has been an attempt to recast the principles of government and bounding in terms of this more general notion. We restrict our discussion here to the account of Subjacency developed by Lasnik and Saito, which is a revision of that suggested by Chomsky. We take the definition of barrier to be as follows [Lasnik & Saito 89]:

(66) **barrier:** $\alpha$ is a *barrier* for $\beta$ iff:
(i) $\alpha$ is a maximal projection,
(ii) $\alpha$ is not *L-marked*, and
(iii) $\alpha$ dominates $\beta$.

[22] Note, this requires that the government domain be determined by the first (i.e. lowest) $X^2$ node. In the present system however, we assume m-command to be determined by the "collective" set of $X^2$ nodes (i.e. the highest node of the phrase).

where,

(67) **L-marking:** $\alpha$ is *L-marked* by $\beta$ iff $\beta$ is directly $\theta$-marked by $\alpha$.

A constituent is taken to be directly $\theta$-marked by $\alpha$, if it is a complement of $\alpha$, thus excluding subjects. In addition, we follow Lasnik and Saito in assuming that VP can be L-marked by I, but that IP cannot be L-marked by C. With these notions defined, we can now restate Subjacency as follows:

(68) **Subjacency:** $\beta$ is *Subjacent* to $\alpha$ if for every $\gamma$ a barrier for $\beta$, the maximal projection immediately dominating $\gamma$ dominates $\alpha$.

To see how Subjacency now applies consider the following sentences taken from the examples above[23]:

(69) a. * Who$_i$ do [$_\gamma$ you like the book [$_\gamma$ OP$_j$ that [$_\gamma$ John gave $\varepsilon_i$ $\varepsilon_j$ ] ] ] ?
b. * What$_i$ do [$_\gamma$ you wonder who$_j$ [$_\gamma$ $\varepsilon_j$ bought $\varepsilon_i$ ] ] ?
c. * Who$_i$ did [$_\gamma$ [$_\gamma$ that she dated $\varepsilon_i$ ] bother you ] ?
d. Who$_i$ did [$_\gamma$ you read a book about $\varepsilon_i$ ] ] ?
e. * What$_i$ did [$_\gamma$ you read a book [$_\gamma$ under $\varepsilon_i$ ] ] ?

The $\gamma$ symbol has been used to indicate those maximal projections which are barriers. That is, those phrases which are not L-marked. In each case, our new formulation of Subjacency accounts for the phenomena of each of the other approaches outlined above. Notice that in (69d), *book* is considered to subcategorize (optionally) for an *about* PP. The PP is therefore L-marked, and not a barrier. In (69e) however, the *under* PP is simply an adjunct, and therefore not L-marked, making a barrier. In this case, the antecedent for $t$ appears outside the first $\overline{\overline{X}}$ dominating $\gamma$, thus violating the Subjacency condition.

## 3.3 Representations: Types *vs.* Levels

In the previous section, we have illustrated how the theory of grammar relies upon the interaction of a set of 'heterogeneous' principles to constrain possible sentence structures. By heterogeneous we simply mean that the principles are concerned with rather independent aspects of syntactic well-formedness: the $\theta$-criterion is concerned only with the well-formedness of thematic assignments, Case theory with the distribution of (phonological) NPs, the ECP with empty categories, and the Projection Principle with the uniformity of interpretation across levels of representation, to highlight some examples. Taken individually, the principles are relatively abstract and unconstraining, but given a suitable interface between relevant aspects of a complete syntactic analysis they comprise a very rich system of

[23] We follow linguistic convention in using *OP* to represent an *empty operator* – a phonetically null wh-pronoun.

constraints. As an example, recall the discussion in §3.2.2, which explained the prohibition of movement into complement positions: In addition to the basic principles of $\overline{X}$-theory and $\theta$-theory, we defined an 'interface' between the two, requiring that all complements (at least of lexical categories) be $\theta$-positions (with the additional possibility of an external $\theta$-position, i.e. the subject). When combined with the Projection Principle, we showed that movement to an object position would entail a $\theta$-criterion violation, thus blocking a particular configuration on the basis of quite distinct structural and thematic conditions.

The Projection Principle is instrumental in preserving the base generated structure of the sentence at all syntactic levels of representation, by forcing the insertion of a *trace* in vacated positions, and using a mechanism of coindexation between the moved element and its trace to identify the antecedent-trace relation. Indeed, in the discussion of Case theory (§3.2.3) we introduced the notion of Chains as a sort of 'meta-level' representation of the history of movement for a particular element. Roughly, we assumed that a Chain for a particular element consists of the set of traces, presumably in some chronological order, which record the path of movement. The availability of this information at S-structure effectively subsumes that of D-structure: i.e. it is a trivial matter to read off the D-structure from this 'annotated' S-structure. This has prompted a number of researchers to assumed a 'collapsed' organisation, without D-structure, where movement is simply represented by the annotation of Chain information at S-structure (see [Koster 87] and [Rizzi 86] for examples of the purely 'representational' approach, also see discussion in [Kolb & Thiersch 90])[24].

In the following subsections we will construct a particular instance of such representational model, which decomposes the single S-structure *level* of representation into a number of different representational *types*, each of which contributes to an individual aspect of a complete syntactic analysis.

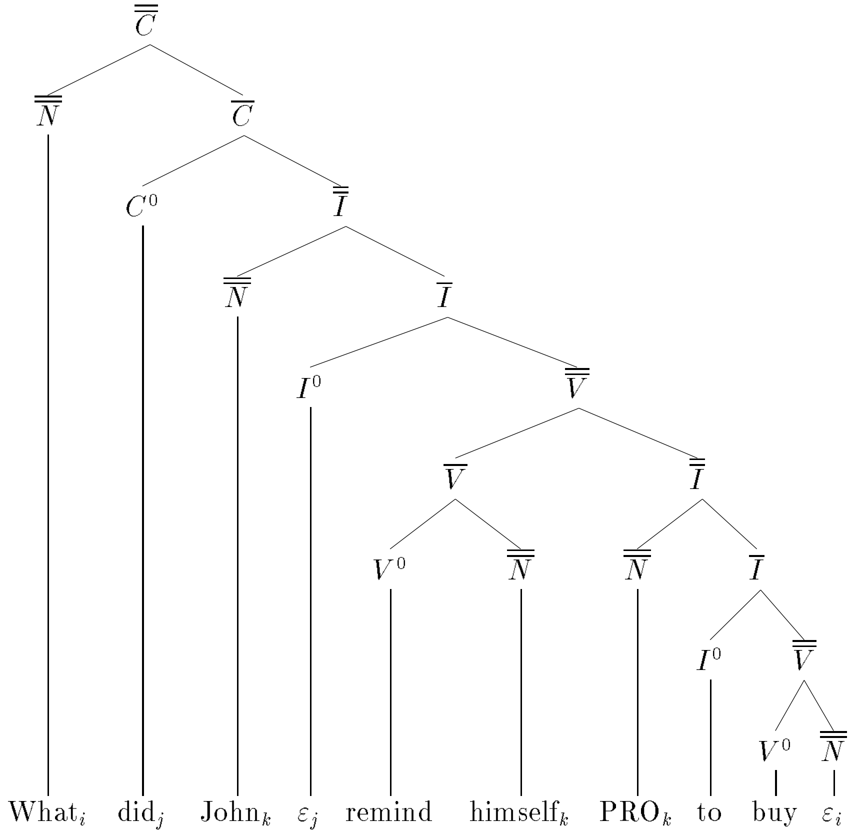
### 3.3.1 A Representational Model

The move towards a non-derivational model of grammar is, as we have pointed out above, accomplished at the the expense of introducing a richer representational component. That is, we simply replace the transformational component, which maps between successive levels of representation, with a Chain structure representation which encodes all the relevant information of the transformational component, thereby eliminating the need to reconstruct each of the levels resulting from transformational operations. Thus Chains simply provide representational means of expressing 'movement'. The result is an annotated surface structure, which encodes a variety of different kinds of information at one level of representation. Indeed, a careful look at the representational mechanisms employed by the current theory suggests that a number of different types of representations are involved in the characterisation of syntactic structure. Consider the analysis of the sentence *What did John remind*

[24] It is also possible that S-structure is subsumed by LF, but this would make it difficult to see what representation people construct on the basis of, roughly, PF input. Furthermore, recent proposals have suggested that some deletion of structure (such as traces) may be possible at LF, thereby undermining any subsumption of S- or D-structure. Johnson demonstrates that it is possible to compute an LF for some PF input without explicitly constructing an S-structure, but S-structures are still used implicitly, we take up this issue in greater detail in chapter 7.

*himself to buy?*, illustrated as follows:

(70)




In the above analysis of the sentence's structure, a number of different syntactic relations are implied. The most obvious of these is the constituent structure for the utterance. In addition we have used subscripts to indicate elements of the structure which are related, but not within the basic domain of constituency. For example, we have coindexed the 'moved' constituents *What* and *did* with their respective traces, thus creating the two Chains: $(What_i, \varepsilon_i)$ and $(did_j, \varepsilon_j)$. We have also used coindexation to denote the coreference relation between *John* and *himself*, and also between *himself* and the PRO subject of the embedded clause[25]. Finally, there is the implicit assignment of thematic relations, for the various verbs and their arguments: the verb *remind* takes an AGENT: *John*, a PATIENT: *himself* (i.e. *John*), and a PROPOSITION which consists of the embedded $\theta$-structure for the verb *buy*, which has an AGENT: PRO (again = *John*) and a GOAL: *what*.

To summarise, we will suggest that a complete syntactic analysis is composed of the four following representation types, each listed with the relevant principles of grammar[26]:

[25] We assume that *himself* is the controller of PRO, since *remind* is an object control verb. That is, if we replace the reflexive with, say, *Mary*, then it is clearly the object which controls the embedded subject.

[26] This is only intended as an intuitive grouping of principles with representation. As we see occasionally during our discussion, the relevant representation for defining a particular principle is not always obvious.

(71) Modules & Principles

| | | |
|---|---|---|
| a. | Phrase structure (PS) | $\overline{X}$-theory, Move-$\alpha$ |
| b. | Chain structure (ChS) | Bounding theory, Case Filter |
| c. | Thematic structure (TS) | $\theta$-theory |
| d. | Coindexation (CiS) | Binding and Control theory |

In the above example, we used coindexation to denote both the members of a Chain, and also coreferential elements, thereby 'overloading' the coindexation mechanism to some extent. In the present model this is no longer the case, since the explicit representation of Chains eliminates the need to coindex the members: That is, membership of the Chain is sufficient to denote the relationship, so that we may reserve coindexation for the representation of coreference alone. This does diverge from the traditional model which attempts to treat the coindexation of antecedent-trace relations and coreference in a uniform manner. Ultimately, this tack may require us to break apart the binding theory into those aspects which are relevant to either Chain formation or coreference. However in the present discussion we will not consider coreference in any detail from either a theoretical or processing perspective.

A further advantage of this non-derivational model, is that we avoid problems concerning 'intermediate' levels of representation. As Kolb and Thiersch point out, principles are not just stated as DS, SS, and LF, but may also be enforced at intermediate levels in the derivation [Kolb & Thiersch 90]. The Case Filter, while traditionally enforced at S-structure, cannot always be, since Wh-elements may be assigned Case *en route* from their D-structure to S-structure position. Consider the follow S-structure:

(72) "Who$_i$ does Mary think [$_{IP}$ $\varepsilon_{i'}$ was hired $\varepsilon_i$ ] ?"

In this sentence, the NP *Who* receives no Case at either its D-structure (object of *hired*) or S-structure ( [Spec,CP] of root clause) position, but rather when it occupies the intermediate subject position ($\varepsilon_i\prime$) of the embedded passive clause. To enforce the Case filter without using Chains entails that we reconstruct each intermediate level of structure so that the Case-marked position may be identified. Chains, however, can be understood as providing a 'complete' representation of a constituent's syntactic interpretation — retaining all the relevant information from the derivational model, but in a much more parsimonious manner. For this reason the Case assignment and the Case Filter (originally given in (50)) may be more naturally stated as follows [Chomsky 81a, page 334][27]:

(73) **Case Assignment:** The Chain C = ($\alpha_1$, ..., $\alpha_n$) has the Case K if and only if for some $i$, $\alpha_i$ occupies a position assigned K by $\beta$.

(74) **Case Filter:** Every lexical NP is an element of a Chain with Case.

[27] For further discussion, the reader is also referred to [Chomsky 86b] and [Rizzi 86].

This is just one example of how the explicit use of Chains provides a natural means for expressing a particular grammatical principle. In the remainder of the chapter we will review the principles discussed previously, addressing the issue of how they fit into the representational model.

In general, a particular representation can be broken down into two fundamental components: 1) *units of information*, i.e. the 'nodes' or feature-bundles which are fundamental to the representation, and 2) *units of structure*, the minimal structural 'schema' for relating 'nodes' with each other. With these two notions defined, the representation can be viewed as some recursive instance of its particular schema over a collection of nodes. This description of representations also provides a very strong notion of 'locality' in syntactic analysis. That is, we may take the schema of a particular representation type as the determiner of locality for that representation. Given that the notion of locality has strongly influenced the hypotheses about grammatical principles, the present model provides a clear notion of what that means for particular types of syntactic information. We will therefore highlight this issue throughout our exposition of the various representational systems proposed here, and attempt to provide a strictly local definition of grammatical principles whenever possible. Furthermore, we suggest that the locality determined by the representation, if adhered to by the principles, has significant advantages for the process model discussed in the following chapters.

In the following subsections, we will discuss each of the representational systems involved in the proposed syntactic model. We endeavour to provide a relatively formal definition of the representational structures in terms of *nodes*, *schemas*, and recursive *representations* outlined above. The aim will then be to illustrate how the principles of grammar may be stated naturally within this framework, and to highlight the interface relations among the various representations which ensure well-formedness of the overall analysis.

### 3.3.2 Phrase Structure

The representation of phrase structure (PS), as determined principally by $\overline{X}$-theory, encodes the local, sister-hood relations and defines constituency. The bar-level notation is used to distinguish the status of satellites, i.e. complements as sister to $X^0$, modifiers to $\overline{X}$, and a unique specifier position under $\overline{\overline{X}}$. The basic constraint imposed by $\overline{X}$-theory then, is the hierarchical and endocentric nature of constituent structure, as defined in (38) earlier, repeated below for convenience:

(75) (a) $\mathrm{X}^i \rightarrow \mathrm{(YP)}\ \ \mathrm{X}^j$
(b) $\mathrm{X}^0 \rightarrow$ Lexeme

where $i \leq 2$, $j \leq i$, and $0 \leq i, j$

In addition to the structures permitted by $\overline{X}$-theory, we must also permit the structures derived by Move-$\alpha$. Given that substitution is simply movement to an existing, base generated position, the rules above will suffice. Adjunction, on the other hand, and the insertion of traces, requires the addition of the following rule:

(76) (a) $X^i \rightarrow Y^i \quad X^i$
(b) $X^i \rightarrow \varepsilon$
where $i = 0$ *or* 2

This simply, permits the adjunction of a maximal or minimal $Y$ projection to an $X$ projection of similar level, and provides a rule for inserting traces ($\varepsilon$) in the position vacated by a moved constituent (again, maximal or minimal). Curiously, the phrase adjunction case (where $i = 2$) is already permitted by the rules in (75), potentially conflating the structures of base-generated adjunction (if this is indeed possible) and movement adjunction[28]. We further noted above that the Specifier position is not obviously distinguishable from a modifier on the basis of its $\overline{X}$ configuration alone.

### A Feature-Based $\overline{X}$-theory

To help eliminate the ambiguity in encoding the status of satellites, we adopt a more articulated, feature-based[29] version of $\overline{X}$-theory, wherein the ability of an X-projection to license a particular satellite (i.e. complement of specifier, since modifiers are not selected by the head) is made explicit. We take the representation of an X-projection is taken to have the following form:

$$(77)\ \mathrm{X}\!:\!\begin{bmatrix} \text{Max:} & \mathrm{M}_m \\ \text{Spec:} & \mathrm{S}_m \\ \text{Cmpl:} & \mathrm{C}_m \end{bmatrix}$$

Using this notation, we assume that X represents the category of the phrase, and that the features refer to the licensing properties of each phrasal projection. That is, Spec and Cmpl are binary $\pm$ features whose value indicates the ability of the projection to license a specific sister type. If a node X has the feature Cmpl: +, then its sister is a complement, if it has the feature Spec: + (and Cmpl: $-$), its sister is a specifier. Roughly, these features indicate the status of satellites which must be licensed by the head. The Max feature simply indicates whether a node is a maximal projection (a phrase) or not[30]. The feature values of a projected mother node are partially determined by the features of the daughter, maintaining the standard hierarchical conditions; most importantly the fact that complement positions must be saturated below the specifier. We may now rewrite the $\overline{X}$ rules shown in (32) as follows:

---

[28] And there are crucial differences, since base generated adjuncts presumably enter into some form of semantic relation (predication or modification) with the phrase they are adjoined to, while adjunction by movement creates a position which is not semantically interpreted in that position, and hence must bind a trace in its D-structure position.

[29] This idea is not particularly new. See [Kolb & Thiersch 90], [Muysken 83], and [Cann 88] for similar approaches.

[30] We might also include a Min feature to make explicit the 'zero-level' projection, but in fact this information turns out to be of little use.

(78) (a) $X{:}\begin{bmatrix}\text{Max: } + \\ \text{Spec: } - \\ \text{Cmpl: } -\end{bmatrix} \longrightarrow Y^{max},\ X{:}\begin{bmatrix}\text{Max: } - \\ \text{Spec: } + \\ \text{Cmpl: } -\end{bmatrix}$

(b) $X{:}\begin{bmatrix}\text{Max: M} \\ \text{Spec: S} \\ \text{Cmpl: } -\end{bmatrix} \longrightarrow Y^{max},\ X{:}\begin{bmatrix}\text{Max: } - \\ \text{Spec: S} \\ \text{Cmpl: } -\end{bmatrix}$

(c) $X{:}\begin{bmatrix}\text{Max: M} \\ \text{Spec: S} \\ \text{Cmpl: C}\end{bmatrix} \longrightarrow Y^{max},\ X{:}\begin{bmatrix}\text{Max: } - \\ \text{Spec: S} \\ \text{Cmpl: } +\end{bmatrix}$

(d) $X{:}\begin{bmatrix}\text{Max: M} \\ \text{Spec: S} \\ \text{Cmpl: C}\end{bmatrix} \longrightarrow \text{Lexeme}$

Each of these rules corresponds roughly to the similarly lettered rule in (32), repeated below in (79), and the linear precedence of satellites with respect to their sister X-nodes is taken to be parameterised for each category and language[31]. Where values are variables they are assumed to unify in a traditional manner. The value for Max is left uninstantiated in rules (78b-d), but note a further condition on instantiating Max to + is that the Cmpl and Spec features be saturated (i.e. $-$). $Y^{max}$ is simply used to abbreviate a phrase (i.e. Max: +) of category Y.

(79) a. $\overline{\overline{X}} \rightarrow$ (Specifier) $\overline{X}$
b. $\overline{X} \rightarrow$ (Adjuncts) $\overline{X}$ (Adjuncts)
c. $\overline{X} \rightarrow X^0$ (Complements)
d. $X^0 \rightarrow$ Lexeme

The are clearly a number of superficial differences in the rules of (78) and (79): Since we assume linear precedence is parameterised at each $\overline{X}$ level, there is no longer a need to specify that satellites (such as Adjuncts) may occur on the left and right. Furthermore, we must explicitly specify the optionality of satellites in (79), since there is no mechanism to 'by-pass' a particular level. Thus each $\overline{X}$ level must be represented even if it has no satellites. Using the feature-based approach, however, we need not generate vacuous projections, rather we only generate projections which have satellites (note, the lexeme — which is itself not a projection — has no satellites).

In a similar manner, we can state the rules for phrasal adjunction and trace insertion as follows:

(80) (a) $X{:}\begin{bmatrix}\text{Max: } + \\ \text{Spec: } - \\ \text{Cmpl: } -\end{bmatrix} \longrightarrow Y^{max},\ X{:}\begin{bmatrix}\text{Max: } + \\ \text{Spec: } - \\ \text{Cmpl: } -\end{bmatrix}$

(b) $X{:}\begin{bmatrix}\text{Max: M} \\ \text{Spec: S} \\ \text{Cmpl: C}\end{bmatrix} \longrightarrow \varepsilon$

[31] Because linear precedence is not determined by the rules, we need not specify the possibility of adjuncts on the left or right, as in (79b), nor do we concern ourselves with the over-specified ordering (Spec-initial and Compl-final) of the rules in (79).

It is important to note, here, that this version of $\overline{X}$-theory doesn't incorporate selectional information of lexical items directly — it simply determines the nature of possible constituent configurations in general. We do not assume, for example, that if the head is a di-transitive verb it permits two complements sister to Cmpl +, before Cmpl is completely saturated. Rather, we assume the free postulation of (possible) constituent structure which must satisfy in turn satisfy $\theta$-theory, as discussed at the beginning of this section. This requires that all complements (of lexical heads) be $\theta$-marked. This assumption is no different from the original version of $\overline{X}$-theory presented earlier, but it helps disarm procedural interpretations of the theory which argue that $\overline{X}$-theory relies on specific lexical information from the head thereby entailing a bottom-up, head-driven parsing strategy (such as Frazier's *Head Projection Hypothesis*, and Pritchett's bottom-up assumption discussed at the end of §2.3.3. To summarise, $\overline{X}$-theory does not inherently use lexical information, it simply defines possible structural configurations which in turn must satisfy independent well-formedness conditions, such as $\theta$-theory and Case theory, which do make crucial use lexical information.

#### The Representation of Phrase Structure

Given the rules above, we can see that possible phrase structures are limited to some combination of binary (non-terminal) and unary (terminal) branches. As discussed above, we can characterise the representational framework in terms of *nodes* and *schemas*. Consider the following:

(81)

<u>Phrase Structure Schema</u>

| | |
|---|---|
| Node: | NT-Node: {Cat,Level,ID,Ftrs}<br>T-Node: {Cat,Phon,ID,Ftrs} |
| Schema: | Branch$_{Binary}$: NT-Node/[ NT-Node$_L$ , NT-Node$_R$ ]<br>Branch$_{Unary}$: NT-Node/T-Node |
| Representation: | Tree: NT-Node/[ Tree$_L$ , Tree$_R$ ] (where,<br>NT-Node/[ root(Tree$_L$) , root(Tree$_R$) ] is a branch)<br>Tree: NT-Node/T-Node |

We allow two type of nodes: 1) non-terminals (NT-Nodes), which are the nodes projected by $\overline{X}$-theory, consisting of category, bar level (via the use of features described in (78) above), a unique ID to distinguish a node from similar nodes in the tree, and the features projected from the head, and 2) terminals (T-Nodes), which are either lexical items or empty categories, which lack bar level, but possess phonological features (for empty categories, the phonology is instantiated by the appropriate [P($\pm$),A($\pm$)] pair, depending on the particular pronominal and anaphoric properties). The schema defines the unit of structure, using the '/' to represent immediate dominance, and square brackets '[...]' to encode sister-hood and linear precedence. Using this notation we define the two possible types of branches, binary and unary, where the latter is applicable just in case the daughter is a terminal node. The full PS representation (or Tree) is defined by allowing non-terminal daughters to dominate a recursive instance of the schema (the 'root' function returns the root node of a sub-tree, for purposes of verifying the branch relation).

In addition to providing a useful recursive definition of the phrase structure representations, we can define the principles of grammar in terms of the schematic units of the relevant representations. As we have seen, $\overline{X}$-theory is naturally defined in terms of binary branches. Additionally, operations such as Case marking[32] and L-marking (the assignment of a feature to a phrase by a lexical item, for purposes of determining barrier-hood) occur strictly under sisterhood. We will see in chapter 5 that this strictly local application of principles is central to incremental interpretation.

### 3.3.3 Chains

At the beginning of this section, we motivated the use of Chains as a fundamental aspect of syntactic representation, removing the need for the multiple levels of representation inherent in the derivational model. Chains determine the well-formedness of 'moved' constituents and the resulting traces present in the phrase structure representation. In addition, we assume that all arguments form a Chain, even if it is of unit length (i.e. *in situ*). We assume further that all traces must be a member of exactly one well-formed Chain, a condition which is possibly reducible to the ECP [Chomsky 86a]. Other principles of grammar relevant to Chain structure include the Case filter (as discussed at the beginning of this section), and of course, bounding theory (i.e. Subjacency).

#### The Representation of Chains

Just as phrase structure is defined in terms of branches, we can define a Chain as a sequence of links. More specifically, each position contained by the chain is a *node*, which represents its category and level (a phrase or a head), the status of that position (either $A$ or $\overline{A}$), its ID (as inherited from its PS node), and relevant features (such as L-marking, Case, and $\theta$). If we adhere to a similar representational paradigm as is used above, we can define Chains in the following manner:

(82)

| | |
|---|---|
| <u>Chain Schema</u> | |
| Node: | C-Node: {Cat,Level,Pos,ID,Ftrs} |
| Schema: | Link: $<$ C-Node$_i$ $\infty$ C-Node$_j$ $>$ |
| Representation: | Chain: [ C-Node \| Chain ] (where, $<$ C-Node $\infty$ head(Chain) $>$ holds ) |
| | Chain: [ ] |

[32] We assume here that assignment of Case to the indirect object, and subject positions (by TNS) takes place under sister-hood, and that the Case features to be assigned percolate up the phrase. We will not address here, however, the mechanism for exceptional Case marking.

If we let '$\infty$' denote the linking of two *C-Nodes*, then we can define a *Chain* to be an ordered list of C-Nodes, such that successive C-Nodes satisfy the link relation. In the above definition we have used the '|' operator and list notation in the standard Prolog sense. The 'head' function returns the first C-Node in a (sub) Chain (possibly [ ]), for purposes of satisfying the link relation. Furthermore, $<$ C-Node $\infty$ [ ] $>$ is a well-formed link denoting the tail, Deep-Structure position, of a Chain. Indeed, if this is the only link in the Chain we refer to it as a 'unit' Chain, representing an unmoved element. Finally we should make it clear that the Chain structure for the entire utterance consists of the set of all Chains involved in that utterance — each of which must be complete and well-formed.

### Conditions on Chains and Links

As we discussed above, a central criterion for each representation is that we be able to state relevant principles and constraints locally, in terms of the schematic unit. This clearly holds for the Subjacency condition:

(83) $<$ C-Node$_i$ $\infty$ C-Node$_j$ $>$ $\Longrightarrow$ subjacent(C-Node$_i$,C-Node$_j$)

That is, subjacency is a condition that need only hold between C-Nodes which are linked[33], and is therefore local with respect to the links of a chain. Similarly, antecedent government — a central component of the Empty Category Principle (ECP) — is precisely a link relation[34].

There exist, however, several Chain conditions which are not obviously local. These involve the Case filter and $\theta$-Criterion, to the extent that they are stated with respect to Chains. The former stipulates that each NP Chain receive Case at exactly one position, while the latter entails that each argument Chain receive exactly one $\theta$-role. At first consideration it seems that such constraints are global (i.e. hold over an entire chain) in nature. Crucially, however, there are additional constraints on which positions may receive Case and $\theta$-roles. Case assignment is restricted to the 'highest' A-position in a chain. Consider first the constraint that there is no movement from an $\overline{A}$-position to an $A$-position:

(84) * $<$ C-Node$_A$ $\infty$ C-Node$_{\overline{A}}$ $>$

Given this, the only eligible Case recipient is the head of an A-Chain, or the A-position node in the $<$ C-Node$_{\overline{A}}$ $\infty$ C-Node$_A$ $>$ link of and $\overline{A}$-Chain. It is therefore possible to enforce both of these conditions on locally identifiable links of a Chain. To summarise, we can state the conditions for unique Case and $\theta$-marking as follows:

---

[33] Although it is generally assumed that non-local computations over phrase structure are required, this is not necessarily the case. We return to his point briefly in §6.2.

[34] A full exposition of these principles is not required for this discussion. Indeed a number of existing proposals could be adopted. In particular, the present system draws on the work of [Chomsky 86a], [Chomsky 86b], and [Lasnik & Saito 89].

(85) In an argument (NP) Chain,
i) $<$ C-Node$_{\overline{A}}$ $\infty$ C-Node$_A$ $> \Longrightarrow$ case-marked(C-Node$_A$) or,
ii) C-Node$_A$ = head(Chain) $\Longrightarrow$ case-marked(C-Node$_A$)

As $\theta$-role assignment occurs at D-Structure (or, thematic structure here), the tail of an argument Chain necessarily occupies a $\theta$-position. As we observed above, the tail is uniquely identifiable as the $<$ C-Node$_\theta$ $\infty$ [ ] $>$ link in a Chain, so this constraint may be stated as the following link condition:

(86) In an argument Chain,
$<$ C-Node$_\theta$ $\infty$ [ ] $> \Longrightarrow$ theta-position(C-Node$_\theta$)

We assume here that $\theta$-positions are structurally unambiguous, namely the specifier and complement positions of lexical constituents. In this way we need not make direct use of thematic information, since the structural position sufficiently determines a position as being $\theta$-marked. To achieve this, we assume that $\theta$-marked Subjects are base generated in the [Spec,VP] position [Manzini 88]. Raising of Subject into [Spec,IP] then results from the need for it to be assigned Case. This avoids the previous ambiguity concerning the $\theta$-marked status of the [Spec,IP] position — we simply assume it is *not* a $\theta$-position.

To summarise, we have broken down the representation of a Chain into a sequence of links, and further suggest that links are the relevant notion of locality in the representation of 'moved' constituents. We have followed this by establishing that the core principles which are generally considered to be defined on chains, can indeed be formulated as local conditions on links.

### 3.3.4 Thematic Structure

In the proposed model of grammar, we take thematic structure to be a pure representation of the thematic interpretation of a sentence. Thematic structure plays a similar role to D-structure in the original model, but with a much less articulated structure. Indeed, in some respects thematic structure resembles *functional*, or F-structures of LFG [Bresnan & Kaplan 82]. The similarity is, however, largely superficial, since the representation we propose is a purely thematic — making no reference to the notion of 'grammatical functions' — and remains subject to the relevant conditions of the principles and parameters theory[35]. The purpose of thematic structure is to coordinate the relevant structural information from the three other representations (PS, Chains, and coreference) with the thematic properties of lexical items to yield a thematic interpretation — the interface of syntax to the 'conceptual' system [Chomsky 88].

[35] Recalling that LFG was originally proposed as a 'psychologically real transformational grammar', it is not surprising that there are some similarities. The present model, however, retains all the underlying principles of grammar, modifying only the representational framework in which they are expressed — indeed, the representations are not so much modified as they are 'more articulated'. We return to this issue briefly at the end of the chapter.

For each 'relation' (a head with semantic content) we project a $\theta$-grid, which contains all the arguments for that relation. We further assume that modifiers/adjuncts which 'predicate' the relation are included in the grid but do not achieve $\theta$-marked status, and we will exclude them from our present discussion. To be more precise, consider the following definition of thematic structure:

(87) <u>Theta Schema</u>

| | |
|---|---|
| Node: | $\Theta$-Node: $\begin{bmatrix} \text{rel:} & \mathit{Rel} \\ \text{cat:} & C \\ \text{grid:} & \Theta\text{-Grid} \end{bmatrix}$ |
| Schema: | $\Theta$-Relation: $\begin{bmatrix} \text{role:} & \mathit{Role} \\ \text{pos:} & P \\ \text{arg:} & \Theta\text{-Node} \end{bmatrix}$ |
| Representation: | $\Theta$-Grid: $\left\{ \Theta\text{-Relation}_1 , \ldots, \Theta\text{-Relation}_n \right\}$ |

Each lexical head[36] projects a $\Theta$-Node, where the semantic properties of the head determines the relation *Rel*. Also associated with the $\Theta$-Node is the syntactic category of the relation (for reasons we shall discuss momentarily) and a $\Theta$-Grid; the set of arguments licensed by the set of $\theta$-roles for the relation. We suggest that each argument of the verb, is itself a $\Theta$-Node which when assigned a $\theta$-role forms a $\Theta$-Relation. In addition we indicate the structural position $P$ of the argument ('internal' for complements and 'external' for subject, i.e. [Spec,VP]). Thus the $\Theta$-Grid is a set of $\Theta$-Relations licensed by a particular relation. The well formedness of a $\Theta$-Node is determined simply by the $\theta$-criterion, which can be restated as follows:

(88) **$\theta$-Criterion:** For every $\Theta$-Node,
- (i) all $\theta$-roles of the relation must be associated with exactly one $\Theta$-Relation in the $\Theta$-Grid.
- (ii) all $\Theta$-Relations in the $\Theta$-Grid must receive exactly one $\theta$-role.

The well-formedness of individual $\Theta$-Relations is determined on the basis of the Canonical Structural Realisation (CSR) properties of the language, as discussed above in §3.2.1. That is the triple consisting of the $\Theta$-Relation's *Role*, its position $P$, and the syntactic category $C$ of its argument $\Theta$-Node.

The interface between thematic structure and the other representations, particularly PS and Chains, is instrumental in ensuring the well-formedness of the complete analysis. In earlier discussion, we highlighted the fact that all complement positions (of lexical heads) must be $\theta$-marked, and I will assume here that specifier positions must be similarly licensed (we refer to these positions generally as $\theta$-positions). The interpretation of phrase structure by the thematic system, therefore, simply project a $\theta$-node for each lexical head (i.e. the relation,

[36] We leave aside the issue of whether or not the TNS content of INFL (or TnsP) should head a $\Theta$-Node, or simply contribute to the semantic relation of the verb.

or 'pred' in LFG) and incorporates the specifier and complement satellites into the $\theta$-grid. The $\theta$-criterion is then the major well-formedness condition on thematic structures, since it requires that each member of the grid be assigned a $\theta$-role, and that each (obligatory) $\theta$-role associated with the relation be assigned.

In the event that a $\theta$-position is occupied by a trace, the thematic system simply assigns the antecedent for the trace (as determined by the Chain structure) to that position in the grid. Note, a major condition on Chain formation is the tail and only the tail of an argument Chain may occupy a $\theta$-position, ensuring that no moved argument will be associated with multiple $\theta$-roles, and similarly that it will receive exactly one.

### 3.3.5 Coindexation

As mentioned at the beginning of this section, we have chosen not to address the issue of coreference (as represented by coindexation). We simply assume that the coindexing device is reserved for purposes of coreference alone, on the view that coreference relations are substantively different from the representation of antecedent-trace relations, as realised by Chains.

There are theoretical issues here which must ultimately be addressed: The Binding theory of the standard principles and parameters model conflates the notions of coreference and antecedent-trace relations under the single representational device of coindexation. As a result, the Binding theory attempts to characterise these aspects of syntactic structure within the one sub-theory. The present organisation would entail splitting the theory, so as to identify the relevant conditions for each representation type (i.e. chains and coindexation).

## 3.4 Summary and Discussion

In this chapter we have provided an overview of the transformational, principle and parameters paradigm, and proposed an alternative, orthogonal model which is identifies the different types of information present in syntactic analysis and their representations. For the most part this exercise can be seen as simply 'decomposing' the rather overloaded structural description traditionally employed, into the less complex representations which fundamentally comprise these structures. The result of this clearly, has minimal impact on the theory itself — it simply provides a more explicitly articulated representational framework. There exists, however, one potentially significant departure: We have replaced the derivational component with a representational equivalent[37] of Chains. Crucially, however, we have shown how the standard conditions on derivations can be naturally stated as conditions on links of Chains.

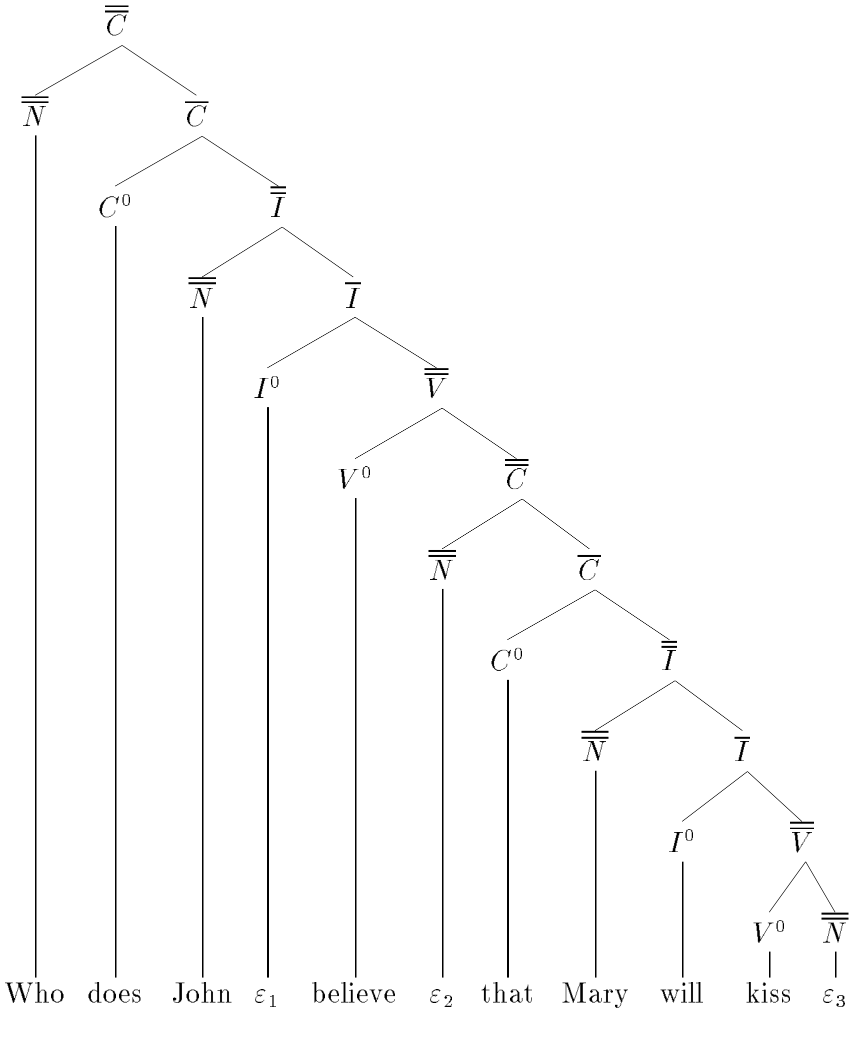
[37] It is possible that the derivational model may have more expressive power than the mono-stratal, Chain-based one. This issue is currently a matter of debate in the field, and is beyond the scope of our discussion. Chomsky has continued to argue in favour of a truly derivational theory [Chomsky 86a] [Chomsky 88] while others have argued against [Rizzi 86] [Koster 87]. The arguments are not only very subtle, but also highly theory-internal, and hence not compelling enough to deter the model proposed here.

Having spent some time discussing each of the representation types, let us now consider an entire example analysis, similar to the *What did John remind himself to buy* sentence discussed above. However, since we are not concerned with coreference, let us examine the following sentence, which is also more interesting in that it involves a longer Chain:

(89) "Who does John believe Mary will kiss."

For this utterance, the phrase-structure representation is as follows:

(90)




Notice that the coindexation of antecedent-trace relations are gone[38], as is any underlying representation of thematic assignment. This is a pure representation of constituent structure as determined by the relevant subset of principles.

The Chain structure representation must, however, meet certain interface conditions with phrase-structure, if the global analysis is to be well-formed. Specifically, each moved element

[38] The subscripts shown here only serve to identify instances of constituents which are similar, so that we may subsequently make explicit reference to them individually.

must head a Chain, and each trace must be a member of a well-formed (and complete) Chain. In our example above both the NP *Who* and the modal verb *does* occupy non-canonical positions. Thus the two Chains may be represented as follows:

(91) $\left[\ \{\text{V,Min},\overline{A},does\},\ \{\text{V,Min,A},\varepsilon_1\}\ \right]$

$\left[\ \{\text{N,Max},\overline{A},Who\},\ \{\text{N,Max},\overline{A},\varepsilon_2\},\ \{\text{N,Max,A},\varepsilon_3\}\ \right]$

In the above representation we have replaced the use of a numeric ID with the use of the terminal itself, and omitted any feature information for readability (recall the representation of C-Nodes given in (82))[39]. Considering the Chain for *Who* in more detail, we can express each link individually:

(92) a. $<\{\text{N,Max},\overline{A},Who\} \propto \{\text{N,Max},\overline{A},\varepsilon_2\}>$
b. $<\{\text{N,Max},\overline{A},\varepsilon_2\} \propto \{\text{N,Max,A},\varepsilon_3\}>$
c. $<\{\text{N,Max,A},\varepsilon_3\} \propto [\,]>$

None of the links violates the $A$-to-$\overline{A}$ movement constraint (84). The second link (92b) must and does satisfy the Case Filter (85i), and, finally, the tail of the Chain (92c) occupies a $\theta$-position, thereby fulfilling the relevant aspect of the $\theta$-criterion (86).

The recovery of the two above representations now provides us with sufficient information to recover the complete thematic structure of the utterance. Each lexical head projects a $\Theta$-Node representing its relation 'relation'. Additionally, the $\Theta$-Node for any constituent which is in $\theta$-position, forms a $\Theta$-Relation with the role assigned to that position, and this $\Theta$-Relation is added to the $\Theta$-Grid of the (governing) role-assign relation:

[39] Note that we indicate the position occupied by *does* do be an $\overline{A}$ position. While this notion is traditionally used for describing positions occupied by maximal projections only, there have been proposals to adopt symmetric characterisations of head-positions [Borer 84]. For purposes of discussion here, however, nothing hinges on this.

(93)

$$\left[\begin{array}{ll}
\text{rel:} & \textit{believe} \\
\text{cat:} & \text{v} \\
\text{grid:} & \left\{\begin{array}{l}
\left[\begin{array}{ll} \text{role:} & \text{AGENT} \\ \text{pos:} & \text{ext} \\ \text{arg:} & \left[\begin{array}{ll}\text{rel:} & \textit{John} \\ \text{cat:} & \text{n}\end{array}\right]\end{array}\right] \\
\left[\begin{array}{ll} \text{role:} & \text{PROPOSITION} \\ \text{pos:} & \text{int} \\ \text{arg:} & \left[\begin{array}{ll} \text{rel:} & \textit{kiss} \\ \text{cat:} & \text{v} \\ \text{grid:} & \left\{\begin{array}{l}
\left[\begin{array}{ll} \text{role:} & \text{AGENT} \\ \text{pos:} & \text{ext} \\ \text{arg:} & \left[\begin{array}{ll}\text{rel:} & \textit{Mary} \\ \text{cat:} & \text{n}\end{array}\right]\end{array}\right] \\
\left[\begin{array}{ll} \text{role:} & \text{THEME} \\ \text{pos:} & \text{int} \\ \text{arg:} & \left[\begin{array}{ll}\text{rel:} & \textit{Who} \\ \text{cat:} & \text{n}\end{array}\right]\end{array}\right]
\end{array}\right\}\end{array}\right]\end{array}\right]
\end{array}\right\}
\end{array}\right]$$

Where $\theta$-positions are occupied by traces, Chain structure is recruited to unify the thematic structure of the antecedent, with that of its original $\theta$-position (as mediated by the trace). Thus when the direct object trace of *kiss* is mapped into the $\Theta$-Grid for that relation, the $\Theta$-Node for the head of the Chain for that trace (the noun relation *Who*), is unified as the argument of that $\Theta$-Relation.

It is interesting to note that the proposed model bears some resemblance to the model of Lexical-Functional Grammar [Bresnan & Kaplan 82]. That theory was proposed in the context of overwhelming evidence against the 'psychological reality' of transformations, and sought to express the relevant information via two representation types. The result is that much of LFG's expressive power lies in the mapping functions from C-structure (roughly our phrase structure) and F-structure (similar to our thematic structure). In the present 'reinterpretation' of the derivational model, we do not do away with transformational component, but rather we encode it representationally as Chains. The result is a transparent mapping between the different representation types, which further enables us to define well-formedness via the original, language universal, principles of grammar. Thus the present model does not rely on complex mapping functions as a fundamental mechanism for describing particular syntactic phenomena. Rather, the mappings are simple and defined *a priori* within the representational framework (and are hence language universal) — i.e. we don't propose new mappings to account for particular constructions in particular languages.

As we pointed out at the beginning of this chapter, however, the purely representational characterisation of syntactic analysis may be fruitful in developing a theory of performance: It explicitly identifies the types of information which must be recovered by the syntactic processor, and further removes any *a priori* temptation to assign procedural interpretations to the non-representational mechanisms (i.e. transformations) which are used by the theory of competence. In sketching this revised representational model, we have further emphasised the notion of locality: the idea that principles of grammar may be formulated as constraints on local units of structure (for whatever representation they are defined on). To the extent that constraints may be applied locally, incremental interpretation is facilitated, a issue we shall discuss further in chapters 6 and 7. While we have not provided a complete reconstruction of the principles and parameters approach here, we have endeavoured to show that theories of this type can be formulated within the present framework.

# Chapter 4

# A Principle-Based Theory of Performance

The development of a process model — a theory of linguistic performance — is a characterisation of how knowledge of language is used. In the preceding chapters we have tried to lay a foundation for pursuing such a process model. In particular we have motivated a modular architecture, containing a distinct syntactic processor, the purpose of which is to relate sound to meaning. We suggest that phonological, syntactic, and semantic processes act concurrently and incrementally. Crucially, however, they are autonomous, input-driven, informationally encapsulated systems with a restricted bandwidth of communication: The fact that they operate incrementally, i.e. construct partial outputs for partial inputs, and in parallel, in no way detracts from their modular status.

This model is supported on theoretical and empirical grounds: The natural encapsulation of the theory provides sufficient conditions for invoking the modular paradigm. Furthermore, processing evidence suggests that there is support for the notion of modules making initial decisions autonomously, while it is also clear that the interaction of modules — incrementally, during processing — can significantly affect relative processing complexity.

Recalling our examination of some current processing theories (in chapter 2), we identified three basic schools of thought. Two are motivated by a desire to minimise the computational or representation complexity of the parsing task. That is, the parsers construct an analysis of an utterance in accordance with the grammar, but the basic operation is determined independently by the desire to minimise complexity. The third approach we examined was the grammar-based model, which assumes that the operation of the parser is based on the desire to satisfy grammatical relations, suggesting a closer and more cooperative relationship between grammar and parser. While there has been a tendency for proponents of the grammar-based approach to conflate parser and grammar, possibly motivating some aspects of processing incorrectly, the approach has a fundamental appeal. That is, the grammar-based account defines strategies in terms of the *content* of grammatical representations, rather than their *form*. Crucially, however, we must make certain that such a theory does not make unfounded claims about particular processing strategies being natural consequences of the syntactic theory (as discussed in §1.1 and §2.3.3).

In this chapter we present a model of processing which constructs syntactic analyses directly from the principles of grammar — not some derived, 'compiled out' grammar, or covering grammar — and which can therefore be said to be *principle-based*. The process model meets the general requirement of incrementality while maintaining a strictly modular architecture. Further, we will show that preferences observed in processing ambiguous sentences can be explained in terms of strategies which are defined in terms of syntactic content, rather than form, suggesting an even closer relationship between parser and grammar. Indeed, we will suggest that the strategies are defined in terms of UG, predicting their application across languages, and we will endeavour to demonstrate that this is the case for several languages where relevant data is available. And finally, we will show that, although it may be possible to construct numerous principle-based processing models which are consistent with the empirical data, they must share certain fundamental properties with the performance model presented here if the grammar is held to be psychologically real.

If we can successfully establish that the core of the human sentence processing mechanism is language invariant, then there exist two possible interpretations. Firstly, we might just assume that the strategies are acquired and that the only reason for similarities across language is that we all begin with only UG and hence develop a uniform processing system. However given that the surface configurations of languages vary so widely — head-final versus head-initial, the degree of movement, etc — it seems unlikely that we would all acquire such similar strategies. Rather, we might seek to explain the similarities of cross-linguistic performance by hypothesizing the existence of an innate sentence processing mechanism, in conjunction with UG and the language acquisition device. If this hypothesis can be sustained, then even more questions concerning the nature of the grammar-parser relationship are raised:

(94)
1. Has the innate sentence processor evolved as some 'most efficient' parser for UG?
2. Is the sentence processor constrained by independent restrictions on memory and processing resources?
3. To what extent has the performance mechanism influenced UG?

We will take up some of these issues at the end of the chapter and later in chapter 7, reflecting on the theory we have presented and existing ideas concerning these questions.

In what follows, we begin with a discussion of our general assumptions concerning the HSPM's operation: The demands placed upon the sentence processor — such as incremental interpretation — and a corresponding metric for processing complexity. We then take up the issue of the process model's grammatical basis, which leads us to propose a model of syntactic processing which is itself modular. The modules correspond directly to those representational/informational domains which we identified in the previous chapter. We then consider each of the sub-syntactic modules[1], reconsidering the basic range of empirical data. We demonstrate that the data may be naturally explained in terms of several strategies when taken in the context of the modular organisation, and our general assumptions of incrementality. These strategies are further defined with respect to fundamental

[1] Again, we will not discuss in any detail the issue of coreference, which we leave to future inquiry.

grammatical notions, suggesting a strongly principle-based account may be sustained, while achieving a higher degree of 'resolution' in accounting for relative processing effects.

## 4.1 The Foundations of the Processing Model

As we have observed up to this point, existing processing models are founded upon some notion of syntactic "complexity". It has traditionally been assumed that increases in processing complexity incurred during parsing are the result of some increased load on the syntactic processor. As a result, theories have stipulated their relevant metrics for assessing complexity as either representational parsimony, as in Frazier's *Minimal Attachment* strategy [Frazier 79], or computational efficiency as in the deterministic parsers of Marcus [Marcus 80] and Berwick and Weinberg [Berwick & Weinberg 84].

One common aspect of both approaches is that they are based upon the isolation of the parsing task — even where some vague interaction with semantic processes is suggested. That is, they are isolated in the sense that the syntactic processing strategies are insensitive to the more general comprehension task. The parsing models proposed are motivated purely by a desire to minimise syntactic complexity, be it representational or computational. While it does seem reasonable to assume that both syntactic representation and computation are relevant to processing complexity, it is possible that they may be overshadowed by the complexity of the more general comprehension task. If this is so, then it is theoretically possible that relatively inefficient representations or algorithms may be justified within the syntactic processor, should they reduce the complexity of overall comprehension. In other words, simple time and space complexity considerations *within* the syntactic processor, might not be a paramount importance, and thus possibly sacrificed for the greater good (of global comprehension, that is).

Given this possibility, the model we propose does not assume either computational or representational parsimony to be fundamental. Rather, we assume that the sentence processor strives to optimise local comprehension and integration of the partial interpretation into the current context. That is, decisions about the current syntactic analysis are made incrementally (for each input item) on the basis of principles which are intended to maximise the overall interpretation. The basic philosophy behind this position is that the syntactic processor's primary objective is to provide a maximal, partial interpretation as input is received, such that it may be quickly integrated into the current context so as to meet the real-time demands of comprehension. As such, syntactic analysis will proceed, first and foremost, in a manner which will satisfy this objective. We will dub this the *Principle of Incremental Comprehension*, defined tentatively as follows:

(95) **Principle of Incremental Comprehension** (PIC): The sentence processor operates in such a way as to maximise comprehension of the sentence at each stage of processing.

At first glance this sounds no different from any other standard requirement for incremental interpretation. As we will see throughout this chapter, however, there is a subtle difference.

While the traditional incremental interpretation[2] requirement is subsumed by PIC — i.e. each lexical item must be incorporated into the current partial syntactic analysis as it is encountered — there is an additional requirement that any structure which can be built, must be. As we will see, this is especially relevant for our characterisation of both attachment preferences and gap-filling. The latter phenomena, in particular, is not considered by traditional definitions of incremental interpretation.

As we have pointed out, incremental syntactic processing does not conflict with the strict assumptions of modularity we assume here. To clarify, consider the hypothetical organisation sketched in Figure 4.1. This is a schematic or 'declarative' view of the modules involved and their informational dependencies. Crucially, it does *not* entail the strictly serial operation of the modules. Indeed, the PIC demands that each module apply maximally to any input, thereby constructing a maximal, partial 'LCS Output' for a given partial 'Input Signal'. This is illustrated by the operation shown for the partial input of *"John saw ..."* in Figure 4.1[3]. If we assume the unit of communication between the Phono-Lexical Processor and the Syntactic Processor is one lexical item, then the output of the Syntactic Processor is produced incrementally on a word-by-word basis for interpretation by the Semantic-Pragmatic processor. The result is a 'coroutined' operation, in which all modules operate in parallel, subject to their input dependencies (e.g. the syntactic processor must wait on lexical input from the phono-lexical processor, etc.). The net result is that post-syntactic processes can incrementally interpret the incoming material and evaluate it very quickly with respect to context and pragmatics, as empirical evidence suggests [Stowe 89], [Crain & Steedman 85].

It is important to note, however, that while the PIC does not conflict with the modular architecture, it does presume that the constituent modules are sensitive to the more general task. That is, in addition to operating incrementally, the PIC also requires that — in the face of ambiguity during processing — the syntactic processor will opt for the analysis which leads to a 'maximal interpretation'. This point will be made clearer with respect to particular examples throughout the chapter.

## 4.2 The Nature of Processing Complexity

In the last section, we argued that the LCS — and hence all modules contained therein — must operate in such a way as to maximise incremental interpretation. While this maxim provides some criteria for evaluating proposals about how modules operate, it says little about the nature of processing complexity. However, to the extent that incremental comprehension is a fundamental priority of the LCS's operation, it seems natural to suggest that increases in processing complexity will occur precisely when the (partial) interpretation we have constructed turn out to be wrong, requiring reanalysis.

---

[2] Reference some of the standard definitions, such as Frazier and Steedman.

[3] In Figure 4.1, we demonstrate using the tree as a syntactic representation, and a predicate-argument structure for LCS output. These are intended purely for intuitive exposition. Later we suggest the output of the syntactic processor is in fact a thematic representation, not a tree, and we make no claim here about the precise nature of LCS output.

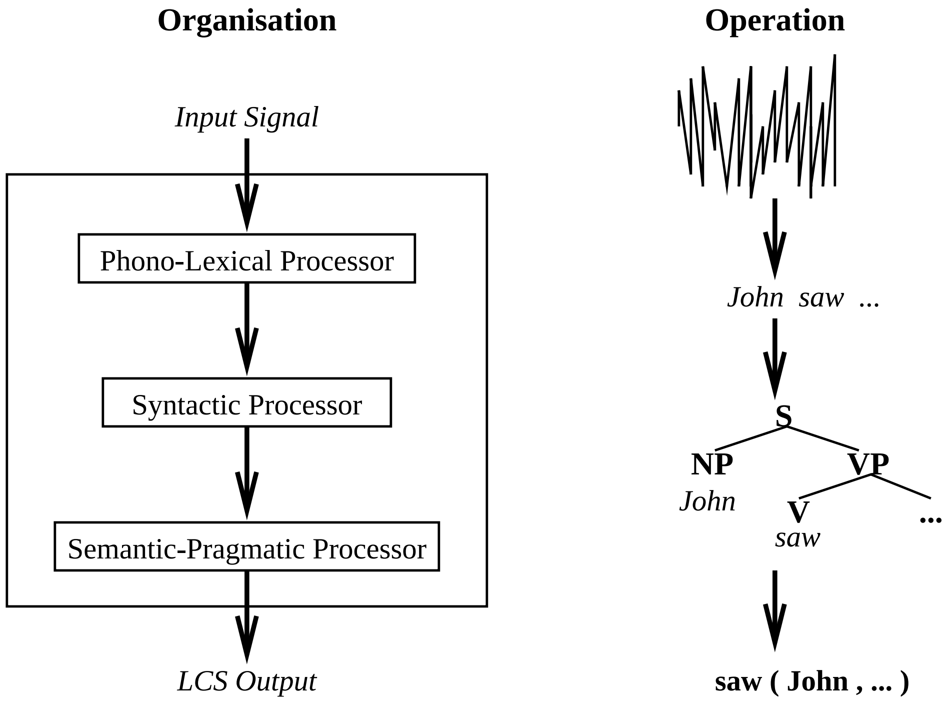



Figure 4.1: The Language Comprehension System (LCS)

We have assumed throughout that semantic interpretation occurs in parallel with syntactic analysis, incrementally. I will further suggest that — if the (partial) interpretation is plausible and consistent which the current context — we *commit* to that interpretation. We can formulate this notion as follows:

(96) **Commitment Principle** (CP): Successful integration of lexical items into the current context results in commitment to that analysis. Implausible interpretations do not.

Assuming the CP, I will suggest that the degree of complexity resulting from reanalysis depends on the extent to which we were committed to the interpretation: The greater the commitment, the less salient the choice point for reanalysis, and hence the greater the difficulty in recovering the correct interpretation. This explains, for example, the relative difference in processing complexity observed for the following reduced-relative examples [Crain & Steedman 85]:

(97) a. "The mailman delivered the junk mail threw it away."
b. "The housewife delivered the junk mail threw it away."

For sentence pairs such as these, Crain and Steedman demonstrated that the strength of the garden-path effect was reduced when the active-clause reading was made less plausible (as in (97b), above). In the present context we account for this as follows: (i) when *delivered* is reached, the active analysis is initially adopted by the syntactic processor (on the basis of strategies we propose later), (ii) this partial subject-verb interpretation is evaluated by subsequent semantic/pragmatic processes — the (97a) example results in strong commitment, while (97b) does not, (iii) when *threw* is encountered[4] and reanalysis is required, the heavy commitment in (97a) leads to a stronger garden-path effect than in (97b). Note however, because reanalysis is always required for such reduced relatives (since the active reading is initially preferred by the syntax regardless of plausibility), some increase in complexity will always be present. Recalling the discussion in §2.2.2 and §2.3.2 we observe that this account provides a robust characterisation of both initial syntactic preferences and subsequent semantic assistance in explaining relative processing effects.

It is possible, however, that the Commitment Principle is simply a descriptive characterisation of a more fundamental property of modular systems. The above example remains a conscious garden path regardless of semantic implausibility for the syntactically preferred analysis. Crucially, reanalysis of the syntactic structure leads to significant reassessment of thematic relations (in Pritchett's sense, as discussed in §2.3.3), thus syntactic reanalysis entails significant reanalysis by the subsequent semantic/pragmatic model. I will claim that it is this 'snowballing' effect that leads to the fundamental garden path effect — note that the final LCS input will also be significantly revised, entailing reanalysis by the central cognitive system (in Fodor's sense of the term). However, because the post-syntactic systems are less 'committed' to implausible analyses, the cost of reanalysis may be reduced for examples such as (97b).

This view predicts that, for utterances where syntactic reanalysis does *not* have a significant effect on post-syntactic modules, the resulting increase in complexity will be of a much lesser degree. That is, the reanalysis process will have been 'contained'. This is demonstrated by the rather more subtle PP attachment effects, as discussed for examples like (12), repeated below. Such reanalyses do not involve significant (thematic) reanalysis, assuming Pritchett's definitions.

(98) Preferred VP attachment over NP adjunction.
- a. "I [$_{VP}$ saw [$_{NP}$ the girl ] [$_{PP}$ with the binoculars ]]."
- b. "I [$_{VP}$ saw [$_{NP}$ the girl [$_{PP}$ with red hair ]]]."

The characterisation of complexity is natural given Fodor's suggestions concerning the properties of modules [Fodor 83]. Assuming modules are inherently fast due to their informational encapsulation — which is presumably a prime motive for their existence — then it seems reasonable that reanalysis within a module will be relatively easy, introducing a minimum increase in complexity. If, however, the reanalysis alters the output of the module in a manner which entails significant reanalysis by subsequent modules, then this knock-on effect will significantly increase the overall complexity of reanalysis. We will elaborate upon this details of this proposal during our exposition of the processing model, and in chapter 7, we will clarify our overall position more precisely.

[4] We leave open the possibility that, where the initial Subject-verb interpretation is particularly anomalous, this may force reanalysis *before* the ungrammaticality is discovered.

## 4.3 Modularity in the Syntactic Processor

Fundamental to this thesis is the proposal that the performance model be directly compatible with the modular, language universal, principle-based theory of current transformational grammar. Consistent with our aim of maximising the interpretation of an utterance, the principles and parameters paradigm takes D-structure — the canonical representation of a sentence's thematic interpretation — to be the interface to the lexicon [Chomsky 88]. In addition, many of the principles of grammar are aimed precisely at preserving that interpretation at each syntactic level of representation: the $\theta$-criterion and Projection Principle ensure unique $\theta$-role assignment across each level of representation, Case theory can be viewed as a 'visibility' condition on lexical noun phrases [Chomsky 86b], and within the Barriers framework even extraction conditions are reducible to constraints based on $\theta$-marking. This is to say, the principles of grammar are not 'artefactual'; rather they are based on the desire to recover a coherent interpretation of a sentence.

In the previous chapter, we constructed a model which can be considered "orthogonal" to the standard T-model. Specifically, the system we proposed characterises syntactic analysis in terms of several *types* of representations rather than *levels*. This reinterpretation is not intended as an alternative the T-model, but rather as an equivalent system which more suitably identifies the types of information which are relevant for purposes of processing. Our task here is to construct a model of performance which makes direct use of the grammatical principles to recover the various informational structure in accordance with the PIC.

We posit four modules in the syntactic processor, each affiliated with a 'representational' or 'informational' aspect of the grammar. Indeed, we will suggest that these 'types' determine the informational and representational domains for several modules *within* the syntactic processor. These are outlined below in conjunction with the grammatical subsystems to which they are related:

(99) Modules & Principles

| | | |
|---|---|---|
| (a) | Phrase structure (PS) | $\overline{X}$-theory, Move-$\alpha$ |
| (b) | Chains (ChS) | Bounding theory, Case Filter |
| (c) | Thematic structure (TS) | $\theta$-theory |
| (d) | Coindexation (CiS) | Binding and Control theory |

Before we proceed to the empirical evidence which we will argue supports such an organisation, let us first consider the theoretical motivation. Recalling the discussion in §2.1, we suggested that the reason for having distinct theories of syntax, semantics, etc. was precisely because characterisation of these phenomena demanded some specialised vocabulary and system of representations. Therefore, if the modularity of mind is a desirable paradigm for mental architecture, the encapsulation of these theories provides sufficient conditions for invoking a modular organisation which permits individual systems to operate concurrently and at high speed. We can apply precisely the same argument concerning the architecture of the syntactic processor: the heterogeneous nature of the theory of competence can be characterised as suggested in (179), by a number of different information types which cooperate to determine the syntactic analysis of an utterance. For each information type there is a natural 'clustering' of grammatical principles which determine the well-formedness of a particular representation type.

In recent work, Lyn Frazier has also considered the possibility of modularity within the syntactic processor [Frazier 90]. While Frazier has long distinguished phrase structure and thematic structure, it is important to note that Frazier assumes the use of 'real world', pragmatic knowledge in constructing a thematic structure, while we assume it to be a purely syntactic representation which is subsequently interpreted by semantic and pragmatic systems. In her more recent proposals, Frazier proposes two additional modules centered upon the notion of c-command. The first considers antecedent-trace relations, similar to our Chain module, and is strictly encapsulated within the syntax. The second module concerns coreference relationships, and is only 'pseudo-encapsulated', in a manner similar to her thematic structure.

The majority of this chapter will be concerned with developing a process model for each of the proposed modules, so as to account for the relevant empirical evidence. We will however exclude from our discussion the coindexation/coreference module, so as to reduce slightly the scope of our discussion. Beforehand, however, let us briefly consider the operation of the syntactic processor as a whole.

### 4.3.1 Incrementality in the Syntactic Processor

In Figure 4.2, we illustrate one possible instance of the organisation within the syntactic processor. We assume that the phrase structure module drives processing based on lexical input, and that the thematic structure is the relevant input to the 'conceptual' semantic system mentioned above[5]. Just as the PIC applies to the main modules of the LCS as discussed above, it also entails that each module within the syntactic processor be coroutined so as to apply maximally for a partial input.

To illustrate the concurrent, incremental processing of the modules, consider the operation shown in Figure 4.3 for the partial input *'What did John put ... '*, we can recover a partial phrase structure representation, including the *trace* of the auxiliary element *did*[6]. In addition, we can recover the chain linking *did* to its deep structure, Infl position, and also initiate a chain for *What* in the [Spec,CP] position. Finally, we can construct the $\theta$-grid for the relation *put* including the saturated agent role *John*. Note, we might also go one step further and postulate a trace as the direct object of *put* so as to complete the chain for *What*, but this action might prove incorrect if the sentence turns out to be *'What did John put the book on?'*.

[5] Note that we are excluding, for the moment, the LF interface. In this thesis we will restrict discussion to those aspects of the syntactic processor which relate lexical input (roughly PF) to thematic structure (roughly DS) and its interface with the conceptual, 'semantic-pragmatic' processor. We do, however, acknowledge the necessity of a module which can recover a representation of scope.

[6] We assume here a head movement analysis, where the head of Infl moves to the head of Comp, to account for Subject-Aux inversion.

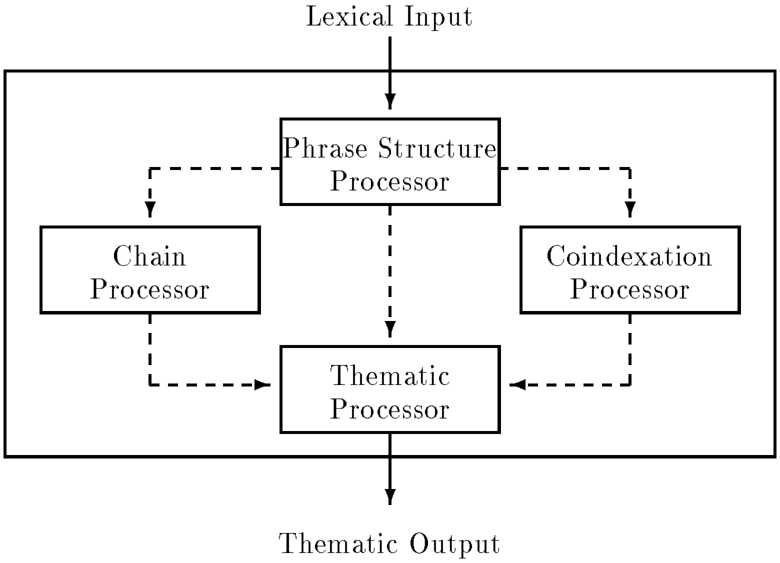



Figure 4.2: The Syntactic Processor

### 4.3.2 The Nature of Modularity

The nature of modularity *within* the Syntactic Processor is a topic of much recent interest, and may turn out to be different from that of more general modules (in Fodor's sense). In particular, we suggest that each of the processors consists of a specialised algorithm for a particular representation, over a relevant subset of the grammar (i.e. the axioms of the grammar pertaining to the representation being recovered). In addition, processors might be allowed access to the representations recovered by other processors, and some specialised knowledge about how to interpret external representations.

One crucial aspect of any modular theory is the nature and degree of communication between modules. We might wish, for example, to permit the phrase structure processor some limited access to chain information for purposes of positing traces. We could restrict this access, however, by only allowing the phrase structure processor to see the 'external' features of a chain, such as category and level (phrase vs. head) information about the head of the chain. Alternatively, we might adopt a stronger stance, and prohibit any access to chain information by the phrase structure module, increasing the degree of informational encapsulation. In this thesis we assume this stronger position, where modules may only look at their true 'input', typically just one representation type. The exception is the thematic module, which is responsible for coordinating the output representations of the other processors into a single representation for consumption by subsequent semantic and pragmatic systems.

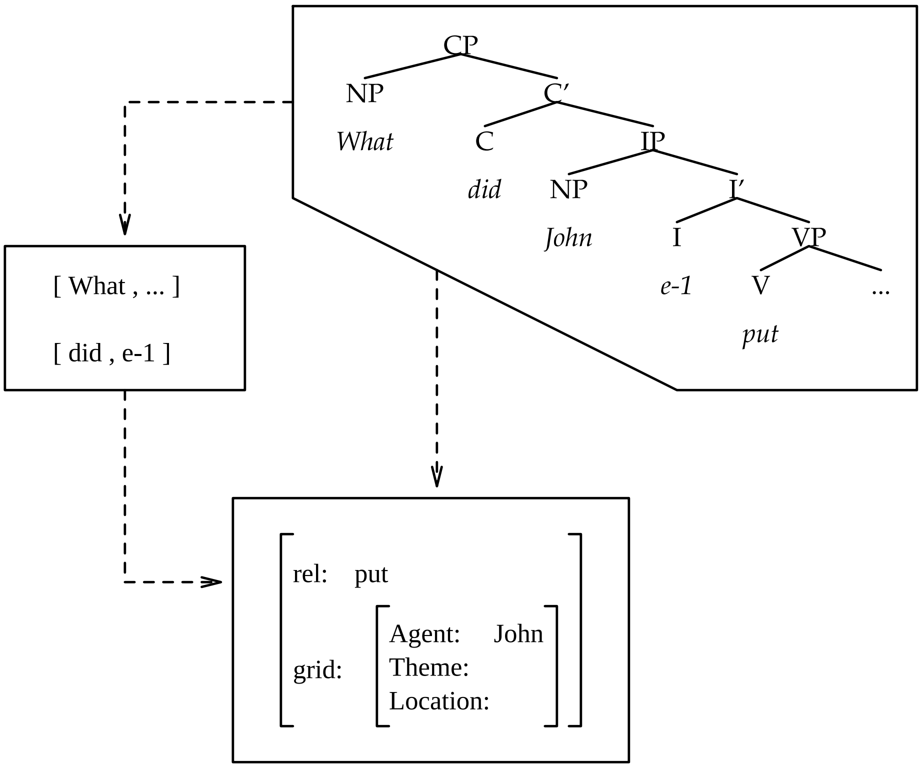



Figure 4.3: The Operation of the Syntactic Processor

## 4.4 The Phrase Structure Module

Following the organisation sketched in Figure 4.2, we will assume that the phrase structure (PS) module is responsible for assembling a constituent structure representation on the basis of lexical input. As we have suggested, a prime motivation for such a processor is that, on the basis of minimal syntactic and lexical information, it will be able to quickly construct the sisterhood and constituency relations for elements of the the string as they are encountered. This representation then serves as the basis for subsequent recovery of long distance dependencies, coreference, argument structure.

Specifically, we will assume that the PS module is concerned primarily with the principles of phrase structure, namely $\overline{X}$ theory, and the structures licensed by the transformational rule Move-$\alpha$ (i.e substitution and adjunction). In addition, we will assume the basic sisterhood relations such as Case- and L-marking may also be determined with respect to the PS representation. Furthermore, we will assume that minimal lexical information is used, such as major category and possibly rudimentary transitivity[7]. information (for Case assignment).

[7] We use the term *transitivity* to refer exclusively to the Case marking properties of lexical items, not their subcategorization frames.

The empirical data bearing upon this model is therefore concerned with two issues: First the evidence regarding structural preferences in the face of ambiguity — i.e. attachment preferences. And secondly, the data concerning the kind of information that is recruited in determining such preferences. In this section we will begin with a discussion of the latter, in an effort to provide empirical support for the assumption of minimal lexical information which was made above. We will then take up the issue of attachment preferences, examining the traditional range of English data, and hypothesize the existence of two basic attachment strategies which are grammar-based and involve minimal lexical information.

### 4.4.1 The Use of Lexical Information in the PS Module

The modular model of syntactic processing we have proposed crucially distinguishes, among other things, phrase structure from thematic structure. Having outlined a theoretical motivation for this organisation (in chapter 3), it now falls upon us to provide empirical support. In particular we must support the claim that initial constituent structure hypotheses are made on the basis of minimal lexical information — most notably, without thematic information. We should also make clear that we do not assume the existence of subcategorization information either. Recalling the discussion in §3.2.1, we adopted the recent claims by Chomsky that the canonical structural realisation of constituents is determined by a mapping function from $\theta$-roles to phrasal categories. In the context of the present model, we assume the CSR function controls the mapping from phrase structure into thematic structure, by-passing the need for explicit use of subcategorization.

While the details of the organisation we have proposed and its theoretical basis are new, the hypothesis that minimal lexical information is used during the initial structure building is not. Indeed this is precisely the position which Frazier and her colleagues maintain. One compelling piece of evidence in this regard is Frazier's evidence from Dutch, discussed in §2.3.2. The data demonstrates a preference for the transitive analysis of [NP PP V] strings (rather than the intransitive, PP-as-modifier reading) regardless of the verb, suggesting that the transitive analysis is constructed for the NP and PP, without waiting for the verb's lexical information. This implies that, at least for languages where the verb occurs at the end of the sentence, the sentence processor is prepared to make attachment decisions before the verb is reached.

The evidence from Dutch is unfortunately not sufficient to support the phrase structure/thematic structure distinction we have proposed. Rather it might simply fall out from the demand for incremental interpretation combined with the local absence of the licensing verb. Note, though, that this position is a rather unattractive one, since it implies that head-final languages will involve more 'guess-work' (in the absence of early verbal information), while head-initial languages will make such information available before arguments are encountered. Mitchell, however, has conducted experiments in English which bear directly on this issue [Mitchell 89a]. Consider the following pair of sentences:

(100) a. "After the child visited the doctor prescribed a course of injections."
b. "After the child sneezed the doctor prescribed a course of injections."

We observed in §2.3.2 that sentences similar to (100a) are garden paths (recall (17)), due to the analysis of the optionally transitive verb (here, *visited*) as transitive, consistent with Minimal Attachment. Crucially, however, Mitchell, tested processing of sentences as in (100b) where the verb is strictly intransitive. Interestingly, he found that processing of such sentences up to the word *doctor* was longer for 100b-type sentences than for (100a)-type (for details see [Mitchell 89a]). He interprets these results as evidence that *the doctor* is initially attached as direct object — without regard to the verb's thematic information — but that the lexical entry is quickly checked forcing the NP to be unattached (for further discussion see [Mitchell 89b]). This interpretation of the English data, combined with the evidence from Dutch lends strong support for the architecture we have proposed. It should be noted, however, that there is a wide body of evidence bearing on this issue, a thorough discussion of which would take us too far afield. The interested reader is referred to [Clifton 90] and references cited therein.

### 4.4.2 Attachment Preferences in English

To begin our discussion, let's reconsider the classic case of PP attachment preference from 12, repeated below for convenience:

(101) Preferred VP attachment over NP adjunction.
  a. "I [$_{VP}$ saw [$_{NP}$ the girl ] [$_{PP}$ with the binoculars ]]."
  b. "I [$_{VP}$ saw [$_{NP}$ the girl [$_{PP}$ with red hair ]]]."

The established preference in such sentences is for attachment of the *with-PP* into the VP (with an INSTRUMENT role), rather than as a modifier of the Object NP. Frazier's MA strategy accounts for this on the grounds that the VP attachment reading involves the postulation of fewer nodes and is hence preferred (as discussed in §2.3.2). In addition to hinging on a rather non-standard constituent structure, Frazier's account also misses the rather more intuitive explanation that we adopt the argument vs. modifier reading precisely because the argument reading is independently preferred. Pritchett accounts for this in terms of his Θ-Attachment strategy, and Abney's 'Licensing-Structure Parser' accounts for the preference in a similar manner.

While the accounts given by Pritchett and Abney share a strong appeal in their grammar-based nature, they are incompatible with the model presented here since they entail the use of the verb's thematic information to make initial attachment decisions. This is to say, they fail to maintain the distinction between phrase-structure and thematic structure which we have motivated independently, above.

We can, however, incorporate Pritchett's insights within the present model, to achieve broader coverage. We can, for instance, still assume that Pritchett's basic principle of Θ-Reanalysis applies within the thematic processor, possibly influenced by certain contextual information. Now we need only revise the attachment strategy which is operative in the phrase-structure module. While we cannot make explicit reference to thematic information, we can make reference to the purely structural notion of A-positions – those positions which potentially receive a $\theta$-role. Specifically, we suggest an A-Attachment strategy defined initially as follows:

(102) **A-Attachment** (AA): Attach incoming material, in accordance with $\overline{X}$ theory, so as to occupy (potential) A-positions.

where, the definition of A-position is roughly as defined in [Chomsky 81a, page 47] (see also [Sells 85, pages 44-45] for some discussion):

(103) **A-Position** Those configurational positions which are potentially assigned a $\theta$-role or Case, namely, the complement positions of lexical constituents, the external argument [Spec,VP] and the subject position [Spec,IP].

This allows us to maintain the spirit of Pritchett's $\Theta$-Attachment strategy, while avoiding specific reference to thematic information which may not be available. In this way, A-Attachment can still be considered compatible with the PIC (5) – it merely teases apart structural and thematic stages of processing, while still resulting in optimal thematic interpretation. Crucially, however, A-Attachment still refers to the *content* of the syntactic position, rather than its *form*. This contrasts with construction based principles such as Minimal Attachment. We will also see in the next section how the distinction of these two processors provides an explanation of relative processing effects.

Having hypothesized the A-Attachment strategy, let us reconsider the core set of attachment preferences introduced in §2.3.2. We begin by repeating the data from (15) to (17) below:

(104) Preferred active clause over reduced relative.
a. "$[_S$ $[_{NP}$ The horse ] $[_{VP}$ raced past the barn ]] and fell."
b. "$[_S$ $[_{NP}$ The horse $[_{Rel}$ $[_{VP}$ raced past the barn ]]] fell ]."

(105) Preferred object attachment where possible.
a. "While Mary was $[_{VP}$ mending ] $[_S$ $[_{NP}$ the sock ] fell off her lap ]."
b. "While Mary was $[_{VP}$ mending $[_{NP}$ the sock ] ] $[_S$ it fell off her lap ]."

AA predicts the preferred VP attachment in (101), since the verbal projection licenses an A-position complement (assuming the assignment of an optional 'Instrument' role), while attachment to the NP would be as a modifying phrase (an $\overline{A}$-position). In (104), the active analysis is preferred over the reduced relative, since it permits the subject A-position to be licensed locally – i.e. it receives a $\theta$-role from the verb. Finally, (105) is accounted for if we assume (quite reasonably) that attachment is preferred into the existing A-position – the object position – over the yet to be licensed subject position. Implicit in this analysis is a preference within AA to attach elements into existing, licensed A-positions over potential ones. Roughly, this can be considered to account for a subset of phenomena previously explained by Late Closure.

If we return to the example in (16), repeated below, we see that A-Attachment makes similar predictions to MA.

(106) Preferred NP vs. S complement.
   a. "The scientist knew [$_S$ [$_{NP}$ the solution to the problem ] was trivial ]."
   b. "The scientist knew [$_{NP}$ the solution to the problem ]."

Here, the embedded subject will be initially attached as the direct object. This predicts the slight increase in processing complexity for such sentences as observed by Frazier and Rayner, but also predicts the lack of a full garden path if we assume that something similar to the Θ-Reanalysis constraint is operative within the thematic processor. That is, we suggest that reanalysis within the phrase structure module is relatively inexpensive in those cases where it does not involve costly thematic reanalysis, in the sense of the Commitment Principle outlined above, and possibly some version of Pritchett's strategies (see [Pritchett 88] and [Carlson & Tanenhaus 88] for further discussion on such processes). It is precisely these relative processing effects which Pritchett's model alone is incapable of predicting in his 'single processor' organisation. Rather, we suggest that such effects can be accounted for by the interaction of several distinct processing modules. That is, the cost of reanalysis may be more or less, depending on the module in which it occurs, and whether or not it violates the Commitment Principle (we discuss this proposal further in §7.1.2).

### 4.4.3 Processing Head-Final Languages

In the previous section, we have illustrated how the A-Attachment strategy provides a natural account of the relevant attachment preferences in English. The formulation of the A-Attachment strategy emphasises the preference for a particular syntactic configuration on the basis of it's status in the theory (i.e. the preference for an argument versus modifier interpretation) rather than use of metrics based on artefactual notions such as the representational complexity of a particular construction. Furthermore, we have motivated an architecture in which the phrase structure processor, and hence the A-Attachment strategy, make minimal use of specific lexical (subcategorization or thematic) information. The obvious test for this model is to see how well it can explain the preferences observed cross-linguistically. In particular we will examine two verb-final languages, Dutch and German, to see if the proposed theory can be maintained.

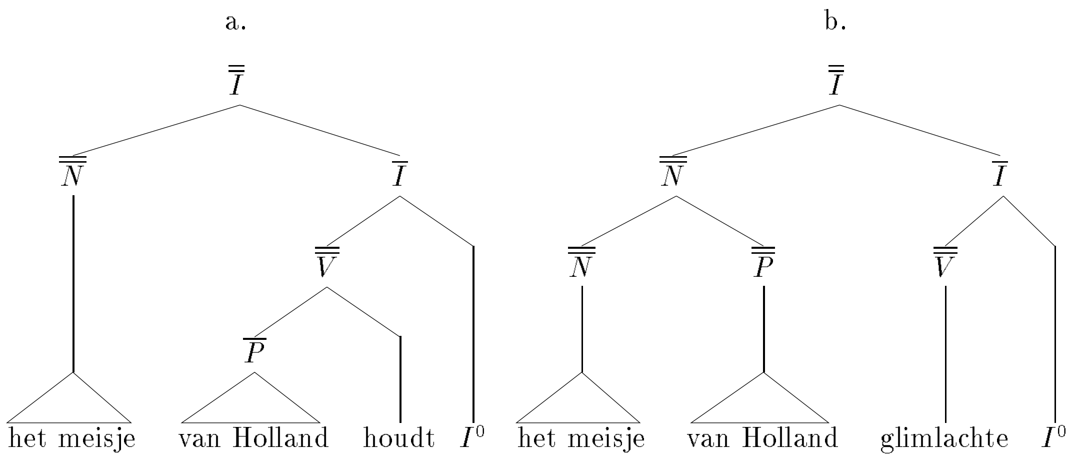
#### Evidence from Dutch

We pointed out earlier that one advantage of A-Attachment is that it doesn't rely crucially on lexical information; a quality which seems necessary in providing a robust characterisation of the English data discussed above. An interesting test, then, is to examine processing of verb-final languages where such specific lexical information, namely the $\theta$-grid for the final verb, is simply unavailable until late in the analysis. In a recent experiment conducted by Frazier, it was shown that there exists a preferred attachment of mittelfeld[8] constituents

[8] The *mittelfeld* refers to that part of a verb-final clause which precedes the verb. Typically, the mittelfeld is a sequence containing the subject and objects of the verb, possibly in some scrambled order, resulting in numerous possible local attachments.

to the VP, even though the verb has not been processed [Frazier 87a]. This is made clearer by the following example:

(107) (a) "...dat [$_S$ [$_{NP}$ het meisje ] [$_{VP}$ [$_{PP}$ van Holland ] houdt ]]."
*...that the girl Holland likes*
(b) "...dat [$_S$ [$_{NP}$ het meisje [$_{PP}$ van Holland ]] glimlachte ]."
*...that the girl from Holland smiled*




In (107a), *van Holland* is interpreted as the complement of *houdt*, while in (107b) it is attached as a subject modifier. That is, there is a temporary attachment ambiguity until the verb is reached. A reading time experiment indicated a clear preference for those sentences where the verb was consistent with the VP object analysis as in (107a). This finding is consistent with MA and also AA, if we assume 1) the existence of the VP node for purposes of attachment (even though the verb has not been encountered)[9], and 2) that the processor regards VP's as likely licensors of A-positions (i.e. likely to take complements). While the latter seems a reasonable assumption on purely linguistic grounds (since the class of obligatorily intransitive verbs is rather small), the former is reasonable if we assume the parser predicts 'obligatory structure'; namely the IP complement of CP (dat) and the VP complement of IP which are both 'functionally' selected, i.e. without regard to lexical information.

The experiments conducted by Frazier clearly indicate a preference for the attachment of locally ambiguous constituents in the mittelfeld to the final verb, even though the verb itself (and its $\theta$-grid) has not been processed. This data presents a further problem for Pritchett's theory, since the processor is clearly making attachment decisions without regard to thematic information in such verb-final Dutch clauses. Indeed, this evidence seems to diminish the cross-linguistic status of any processing model which takes lexical thematic or subcategorization information as fundamental in driving the parser. Such models include the deterministic models [Marcus 80] and [Berwick & Weinberg 84], and Abney's Licensing-Structure parser [Abney 89], all of which crucially rely upon subcategorization information.

[9] It is important to note that we define an A-position purely in terms of its structural position, e.g. the complement or subject positions. We do not require the existence of licensing head to postulate an A-position, we simply require that when the head is parsed, it must in fact license that position.

It is interesting to note, however, that the above account entails a degree of top-down prediction on the part of the phrase structure processor: In particular we have assumed that functionally selected structure is hypothesised automatically by the processor since it is obligatorily present. This contradicts the bottom-up restriction which has been attributed to principle-based parsers by both Frazier and Pritchett, but as we discussed earlier in §2.3.3 this criticism is not well-founded, and indeed is untenable given that both Comp and Infl are non-lexical categories and as such often have no lexical head.

### Evidence from German

Interestingly, the account we have outlined makes certain predictions for garden path phenomena in verb final languages such as Dutch and German. Before introducing such evidence, we should note that German permits VP complements to 'scramble' within a clause, resulting in a high degree of free constituent order in the mittelfeld. Roughly, scrambling permits VP complements to adjoin to VP or IP nodes (see discussion in §3.2.2), thereby allowing objects to appear in pre-adverbial or pre-subject positions. This is illustrated by the following set of equivalent sentences:

(108) a. "...daß der Junge gestern mit mir gespielt hat"
*...that the boy yesterday with me played has*
b. "...daß der Junge <u>mit mir</u>$_i$ gestern t$_i$ gespielt hat"
c. "...daß <u>mit mir</u>$_i$ der Junge gestern t$_i$ gespielt hat"

In these sentences we see that the PP object is free to move from its canonical position in (16a) to the VP-adjoined position (16b) or the IP-adjoined position (16c). Since the processing model we have outlined is motivated by a desire to recover Deep Structure – the pure thematic interpretation – we will assume that where A-Attachment is unsuccessful, DS-Attachment is the default:

(109) **DS-Attachment** (DSA): When an A-position is unavailable for attachment, prefer attachment of incoming material into its canonical, Deep Structure position.

That is, when attachment to a non A-position (known as an $\overline{A}$ position) is necessary, we will prefer attachment to a base generated $\overline{A}$ position, such as a modifier position, over an $\overline{A}$ position resulting from movement. This is further supported by the model's organisation, since DS-Attachment does not involve the chain-module.

Consider sentences of the form [NP PP Adv Verb], as in (16b) above, but where the PP may be plausibly ambiguous as either modifier or object, and the verb is optionally transitive[10]:

(110) "...daß der Junge mit dem Hund einige Zeit gespielt hatte"
*...the boy with the dog some time played had*

[10] The German data presented in this section is drawn from work by Lisa Breidt and Elisabet Engdahl [Breidt 89].

By the attachment preferences outlined above, we predict that the PP will first be attached as VP object, but that once the adverbial phrase is encountered the PP will be reanalysed as an NP modifier, rather than as a scrambled constituent. Thus the preferred reading is *The boy with the dog had played for sometime*. This preference is confirmed by the garden path which results in the following pair of clauses:

(111) "Der Nachbar erzählte, daß der Junge mit der Katze einige Zeit gespielt hatte."
*The neighbour said that the boy with the cat for some time played had.*
"Daraufhin hatte er angefangen sie zu quälen."
*Then had he started her to torture.*

In the first sentence, the modifier interpretation is assigned as predicted and the intransitive form of the verb is adopted. The second sentence, however, forces the PP as argument reading, requiring a thematic reanalysis in the previous clause, yielding the garden path effect.

It also appears possible to induce mild garden path effects within the clause. Specifically, if the modifier analysis is adopted due to high plausibility or a long adverbial phrase separating the constituent from the verb, a commitment to that analysis is seemingly induced. When the verb turns out to be transitive, a garden path can result. Consider the following:

(112) a. "...daß der Nachbar mit dem großen Hund verzweifelt gerungen hat."
*...that the neighbour with the big dog desperately fought*
b. "...daß der Löwe aus dem Zirkus zwischen 5 und 6 Uhr ausgebrochen ist."
*...that the Lion from the circus between 5 and 6 o'clock escaped*
c. "...daß der Entdecker von Amerika erst in 18 Jahrhundert erfahren hat."
*...that the discoverer of America first in the 18th century learned of*
*...that the discoverer learned of America first in the 18th century*

In sentences (112a) & (112b), the modifier analysis is preferred due to DS-Attachment, and then adopted due to plausibility and (perhaps) length of intervening material. When the transitive verb is discovered, however, this must be reanalysed thereby increasing complexity. It is interesting to note that Pritchett's model does not predict this complexity increase, since he would adopt the PP as argument analysis so as to satisfy the maximal $\theta$-grid when the verb is encountered. Indeed, it is difficult to see how Pritchett would process any head final language incrementally. Observe, however, in (112c) *Entdecker* does take a PP complement which it $\theta$-marks. Here, Pritchett correctly predicts a garden path due to a violation of the $\Theta$-Reanalysis Constraint, and indeed, the garden path effect does seem stronger in this example.

**Preliminary Evidence from Japanese**

In §2.3.3 we discussed the issue of parsing Japanese, a purely head-final language, and noted Pritchett's criticism of strictly incremental parsing models, such as Frazier's and indeed the model we have argued for here. In particular, Pritchett argues for a strictly bottom-up, head-driven model, where constituents are not built until their heads are encountered [Pritchett ar]. In a head-final language such as Japanese, this predicts that there may be a significant delay in the structuring of initial argument constituents, but Pritchett argues this is consistent with the data and necessary to maintain a principle-based (which he equates with head-driven) parser. Mazuka and Lust have recently argued that the HSPM is 'organised' so as to prioritise bottom-up strategies in left branching languages (such as Japanese), and top-down strategies for right branching languages (such as English)[11]. These two views challenge the assumptions we have made here concerning strictly incremental interpretation, since this requires the hypothesis of an initial S-node into which material may be structured. As an example, let's consider the following sentence taken from [Mazuka & Lust 90]:

(113) "John ga Mary o mita otoko ni atta"
*John-NOM Mary-ACC saw man-DAT met*
*John met a man who saw Mary*

A-Attachment predicts that *John ga* will be attach as subject, and *Mary o* as object, of an S node. Once the verb *mita* is reached, this remains consistent with the existing structure. The following NP *otoko ni*, however, forces reanalysis of *Mary o mita* as a relative clause. This in turn means that *John ga* must be restructured as a subject of the higher S clause (or, alternatively, *Mary o mita* may be restructured as a lower relative clause). We therefore predict at least some minor increase in complexity for such sentences, due to the cost of phrase structure reanalysis.

Mazuka and Lust argue that a top-down parsing strategy is inappropriate for parsing such structures, since *John ga Mary o mita* will be analysed as simplex S, requiring reanalysis when *otoko ni* is encountered. While Mazuka and Lust feel that there is no strong garden path effect, this is again based on intuitions alone. Indeed, Mazuka and Lust try to motivate a predominantly bottom-up parsing strategy for Japanese on the grounds that such a parser will be more 'efficient' for left-branching languages. If we consider, however, Pritchett's head-driven model (which is an explicit example of such a purely bottom-up parser), we see that once *mita* is encountered, the simplex clause *John ga Mary o mita* (or, *John saw Mary*) will be constructed, just as the top-down parser does. This implies that the same degree of backtracking will result in both top-down and bottom-up parsers, contrary to the claim of Mazuka and Lust.

In the present model, we have assumed a mixture of top-down and bottom-up processing. In particular, we suggest an essentially data-driven approach, but also suggest that constituents required by functional projections may be predicted in a more top-down manner. The spirit is simply this: make maximal use of all available information — that which is projected by lexical items and predicted by the grammar — while remaining principle-based, i.e. no compilation of principles and subcategorization information into phrase structure rules is

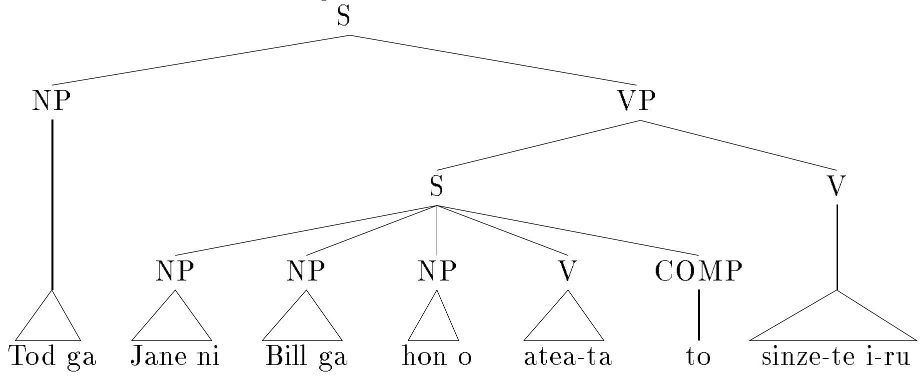
[11] As a result they argue left branching structures are difficult to process for English speakers, and right branching structures are difficult for Japanese. It is not clear, however, from their discussion, that this fact is indeed predicted by their theory (see [Hasegawa 90]), nor that it is true empirically (see [Frazier & Rayner 88]).

assumed. As we remarked above, we also assume the hypothesis of an initial S node, the purpose of this is to provide an initial syntactic structure into which initial material can be incrementally attached. Crucially, however, there is no need to stipulate that this is the top-most S node, as Frazier and Rayner point out [Frazier & Rayner 88]. We take it as an irrelevant artefact of the grammar as to whether or not the initially hypothesised S node will end up as the top S node (as in English), or the most embedded (as is possible in Japanese). The postulation of higher structure into which the initial S is attached is a purely monotonic operation, requiring no reanalysis. As an example, recall the unproblematic (25a), repeated below:

(114) "$[_S\ [_{\overline{S}}$ Bill ni Tom ga nanika o hanasita to $]_i$ John wa $\varepsilon_i$ omotte-iru]"
*[ [ Bill-DAT Tom-NOM something-ACC spoke that] John-TOP thinking ]*
*John thinks that Tom said something to Bill*

We attach *Bill ni Tom ga nanika o hanasita* into the initial S. Once the subsequent material is encountered, we simply hypothesize a higher S, and attach the initial S as a sentential complement — no restructuring of the initially postulated S is required. Turning our attention back to (113) above, however we noted that, the initial S node had to be reanalysed as a subject *John ga* and a relative clause *Mary o mita* involved in a higher S, thereby predicting some degree of increased processing complexity. We remarked earlier, however, that while structures initially built by the phrase structure module constitute a preferred analysis, the reanalysis of constituent structure does not necessarily lead to a conscious garden path effect. This is perhaps best demonstrated by the Dutch data in (107) above: The preference to attach the mittelfeld PP as an argument of the verb, rather than a modifier of the subject, results only in a minor complexity increase when processing the latter. A similar example in Japanese is considered by Ueda (cited as Ueda (1984) in [Frazier & Rayner 88]):

(115) "Tod ga Jane ni Bill ga hon o atea-ta to sinzi-te i-ru"
*Tod-NOM Jane-DAT Bill-NOM book-ACC gave that believing is*
*Tod believes that Bill gave a book to Jane*




Here, Ueda argues that both *Tod ga* and *Jane ni* are preferentially attached into the initial S node, consistent with A-Attachment (and Minimal Attachment, for that matter), even though this results in an ungrammatical continuation. If this preference can be obtained experimentally, then it would provide strong support for the position that the initial NP's are not left unattached — as suggested by the non-incremental parsers proposed by Pritchett, and by Mazuka and Lust (*op.cit.*). To our knowledge, however, appropriate experimental studies (i.e. sufficiently sensitive tasks, rather than the intuitions of scientists) have yet to be conducted for Japanese. Thus an informed comparison of the two basic positions — incremental *versus* strict bottom up — remains a topic for future inquiry.

### 4.4.4 Summary

In this section we have developed a detailed account of the phrase structure processor which satisfies the informational and functional encapsulation requirements which we proposed in our modular model of the syntactic processor. In particular we have shown how two simple strategies — A-Attachment (AA) and D-Structure Attachment (DSA) — can explain a variety of attachment preferences in English, Dutch, and German. Furthermore, these strategies are based purely on the configurational notions of A-position, D-structure position (i.e. base-generated or canonical positions), and positions which are landing sites for moved elements. In this way they are based on the *content* of the constituent structures as determined by the principles of $\overline{X}$-theory and Move-$\alpha$. Crucially, however, these strategies do not use specific lexical information. In sum, the strategies of AA and DSA are based on the principles of grammar, and make use of an appropriately restricted informational vocabulary consistent with the desired encapsulation.

From a more general perspective, these strategies seem to complement our overriding *Principle of Incremental Comprehension*. That is, while operating in the absence of specific lexical information, they function so as to maximise local comprehension: AA will lead to the maximal saturation of thematic roles, as argued for by Pritchett, and DSA expresses the preference to interpret constituents locally (in their base-generated position) permitting immediate interpretation, rather than constructing a Chain which will delay the interpretation.

On the basis of Frazier's evidence from Dutch, we conclude that the processor operates in a partially top-down manner, predicting all functionally selected structure (for C and I specifically). Contrary to the position of Frazier, however, this continues to be completely principle-based, since such structure need not (and often cannot) be projected from lexical material. Furthermore, this structure can be built 'on-line', based on the principles and parameters of the grammar, and does not entail use of a 'compiled out' set of phrase structure rules. Indeed, we suggest that the prediction of functional structure derives from the PIC, on the grounds that any structure that *can* be predicted by the grammar, *must* be. That is, if we are to construct a *maximal*, partial representation of utterance at each stage of processing — as entailed by the PIC — then all functionally selected structure must be built, since it is both obligatory and not determined by incoming lexical material.

Finally, we consider the recent claim that, while (partially) top-down 'prediction' of structure is natural in right branching languages, it is inappropriate for left branching languages, predicting them to be more difficult to parse. We point out, however, that examples cited as problematic for top-down parsers (113) are similarly difficult for bottom-up approaches. Furthermore, Ueda argues that there is a preference for the incremental structuring of arguments into an initially hypothesized S node, consistent with the assumptions (and data) for English and Dutch. Since the position we adopt is stronger — maintaining both (strict) incremental interpretation and involving no parameterisation of the parsing machinery — we argue it to be the preferred model in the absence of contradictory evidence.

## 4.5 The Thematic Module

The thematic processor is responsible for correlating the other representation types into a single structure which is the pure representation of an utterances thematic interpretation. This process involves constructing $\Theta$-Nodes for each lexical head, as discussed in §3.3.4, and filling their $\Theta$-Grid on the basis of the $\theta$-positions which are occupied in phrase structure and — if a position is occupied by a trace — Chain structure is used to fill the grid with the relevant moved element. Finally we assume that some mechanism, such as re-entrancy, is used to indicate coreference in the thematic structure. Since the thematic processor simply interprets the three other representations into a single thematic structure, there is in fact little scope for ambiguity. While Pritchett proposes a $\Theta$-Attachment strategy [Pritchett 88], we account for similar ambiguities at the phrase structure level, via A-Attachment.

In our motivation of A-Attachment (in §4.4.2), we pointed forward to the fact that we could naturally incorporate Pritchett's insights concerning the nature of thematic reanalysis in the thematic module [Pritchett ar]. Recall the Re-Licensing Constraint given in (20) and repeated below:

(116) **Re-Licensing Constraint:** Upon relicensing, a constituent must remain governed by its current licensor.

In fact, Pritchett's original formulation of this strategy — the $\Theta$-Reanalysis Constraint — was formulated strictly in terms of thematic assignment, as follows:

(117) $\Theta$ **Reanalysis Constraint:** Syntactic reanalysis which interprets a $\theta$-marked constituent as outside its current $\theta$-domain is costly.

(118) $\Theta$ **Domain:** $\alpha$ is in the $\theta$-domain of $\beta$ iff $\alpha$ receives the $\gamma$ $\theta$-role from $\beta$ or $\alpha$ is dominated by a constituent which receives the $\gamma$ $\theta$-role from $\beta$.

It is straightforward to reinterpret this account in the present model, since the thematic structure for a particular governor (i.e. head), as represented precisely by it's $\Theta$-Node, is a pure representation of the heads $\Theta$-Domain. If reanalysis forces a member of a particular $\Theta$-Grid to be reinterpreted within the $\Theta$-Grid of a difference $\Theta$-Node, then reanalysis is inherently costly. Here, we will assume (117) & (118) without further comment, we will, however reconsider this issue again in chapter 7, §7.1.2. Note that this predicts reanalysis within the $\Theta$-Grid is relatively unproblematic, as has been argued in [Carlson & Tanenhaus 88].

We have assumed in the present model that thematic structure is the relevant output of the syntactic processor to subsequent semantic and pragmatic systems. The issue then arises as to what phenomena should be accounted for within the syntactic module, as part of the thematic processor, and which is the result of post syntactic processing. Recently, for example, Gibson has characterised a range of processing 'overload' phenomena, such as centre-embedded sentences, in terms of thematic assignment properties at various stages in processing [Gibson 91]. In particular, he suggests that there is an increase in memory load for both constituents yet to receive a $\theta$-role, and heads seeking to assign $\theta$-roles. When the combined complexity reaches a particular threshold, the processor breaks down — too many thematic relationships are unresolved. Assuming his analysis is essentially correct, it is unclear precisely where this problem occurs. It may be straightforward for the thematic processor to represent such structures, but problematic for subsequent semantic and pragmatic systems. We leave this issue as a task for future research, particularly since we have not concerned ourselves with such memory overload phenomena in this thesis.

## 4.6 The Chain Module: Recovering Antecedent-Trace Relations

One aspect of psycholinguistic research which has been particularly influenced by the transformational model of grammar is the account of long-distance dependencies which result from movement[12]. In contrast with other syntactic theories, transformational grammar interprets moved elements by forming a "Chain" which links the displaced the element to a *trace* in its original position. It is therefore essential that a principle-based model of sentence processing accounts for the way in which Chains are recovered by the human sentence processor. In this section we will examine existing characterisations of the so-called "gap-filling" process[13], and see how these strategies may be explained in the context of the modular architecture we have hypothesized here. In addition, we will consider some recent evidence which challenges the traditional *trace-based* approach entirely [Pickering & Barry 91]. Specifically, we will show that naive assumptions concerning when traces are recovered must be abandoned in favour of a more "active" strategy which we will motivate, on independent grounds, in terms of the PIC.

Of all the gap-filling strategies which have been proposed, perhaps the most successful from a descriptive point of view is the *Active Filler Strategy* which has been proposed by Frazier and her colleagues, which can be most simply[14] defined as follows [Frazier & Clifton 89]:

---

[12] It is important to note that we will not consider here long-distance dependencies between antecedents and lexical pronouns or PRO, since they are coreferential relations between independently licensed (i.e. $\theta$-marked) constituents, and are not the result of movement. Discussion of the coreference mechanism is not considered in this thesis.

[13] A complete assessment of the various strategies which have been proposed would not be particularly fruitful here and the reader is referred to [Clifton & Frazier 89] for a thorough discussion of various proposals. We will concerns ourselves primarily with the *Active Filler Hypothesis* which Clifton and Frazier defend, and refer to others only where they are directly relevant.

[14] For a more detailed discussion, the reader is referred to [Clifton & Frazier 89], where Clifton and Frazier restrict the definition to apply only in cases where the filler occurs in [Spec,CP]. This is partly due to

(119) **Active Filler Strategy** (AFS): When a filler has been identified, rank the possibility of assigning it to a gap above all other options.

To illustrate the strategy, Clifton and Frazier present the globally ambiguous example given in (120) below in which there is a strong preference for the (120c) interpretation:

(120) a. "Who did Fred tell Mary left the country."
b. "Who$_i$ did Fred tell Mary $\varepsilon_i$ left the country."
c. "Who$_i$ did Fred tell $\varepsilon_i$ Mary left the country."

While this is consistent with the AFS, it is important to note that there are alternative explanations for this data. Note, that the traditional analysis suggested for the (120b) is as follows:

(121) "Who$_i$ did Fred tell Mary $[_{CP}\ \varepsilon_i\ [_{IP}\ \varepsilon_i$ left the country ] ]."

This is to say, interpreting the filler as the subject of the embedded clause involves postulating an intermediate *trace* in [Spec,CP] of the embedded sentence, thereby increasing the complexity of the Chain. Therefore, while the examples are consistent with the AFS, they are equally well explained by a strategy which prefers the Chain of shorter length. Furthermore, the AFS — a reformulation of what has been traditionally called the *Gap as a First Resort* principle — must address the intuitive data that the following sentence (122) is *not* difficult to process, despite the existence of several potential gap sites before the correct trace is discovered:

(122) "Who$_i$ ($\varepsilon$) did you want ($\varepsilon$) Mother to make ($\varepsilon$) a cake for $\varepsilon_i$."

Indeed, precisely this data led to the proposal of the *Gap as a Second Resort Principle* (GASP) — roughly, posit a gap only if no lexical attachment is possible [Fodor 78] (i.e. similar to the AFS, but using one word lookahead). These data are intuitive, however, and as Clifton and Frazier point out, it may simply be the case that revising the association between a filler and its gap is not particularly costly, especially when the correcting material occurs immediately. Further, they point to the experiments conducted by [Crain & Fodor 85] and [Stowe 86], on sentences of the type in (123) which, demonstrate significantly larger reading times for *us* in (123b) than in (123a) or (123c). This suggest that the AFS is operative, assigning the filler to the gap after *bring*, and that some expense results from revising this analysis.

(123) a. "My brother wanted to know who$_i$ $\varepsilon_i$ will bring us home at Christmas."
b. "My brother wanted to know who$_i$ Ruth will bring (* $\varepsilon_i$) us home to $\varepsilon_i$ at Christmas."
c. "My brother wanted to know if Ruth will bring us home to Mom at Christmas."

---

their conflation of *trace* and *PRO*.

In another recent study, Frazier and Clifton investigated gap filling in multiple clause extractions — i.e. where the movement involves successive cyclic extraction through [Spec,CP] of the embedded clause to [Spec,CP] of the matrix sentence [Frazier & Clifton 89]. This directly addresses the point raised concerning (120) above, as to whether or not the AFS applies across clause boundaries. In particular, they conducted a self-paced reading study for the following sentence types:

(124) a. "Who$_i$ did the housekeeper from Germany urge the guests to consider $\varepsilon_i$ ?"

b. "Who$_i$ did the housekeeper say $[_{CP}\ \varepsilon_i\ [_{IP}$ she urged the guests to consider $\varepsilon_i]\ ]$?"

In both the one and two clause examples, they observed an increased reading time for the lexical NP *the guests* when compared with the relevant declarative control sentences. This suggests that a conflict arises from the initial preference for the gap (as predicted by AFS) when it is proven wrong by the subsequent lexical material[15].

### 4.6.1 Processing Chains in a Modular Model

So far in this section, we have provided evidence favouring the descriptive characterisation of gap-filling which is provided by the Active Filler Strategy. Before we consider any further evidence concerning the processing of Chains, we will consider the implications of the AFS given the architecture for the syntactic processor we have proposed. Indeed, at first glance the AFS seems to present a problem for the modular, informationally encapsulated model. If we maintain that traces are represented in phrase structure, then the postulation of traces is the responsibility of the phrase structure processor (i.e. they cannot be posited directly by the Chain processor). This presents us with two options: Either the PS processor accesses the current Chain representation to decide whether or not to postulate a gap, or the PS processor makes the decision to postulate a gap without any Chain information.

Clearly, the first route has significant consequences for the modular status of the system. That is, while we maintain functional encapsulation, the degree of informational encapsulation is significantly diminished[16]. The second option seems odd, in that the parser is likely to make a large number of false starts, positing gaps when they are not even possible, let alone required (note, the PS processor could even posit gaps when there were no antecedents). Indeed, the only way to ensure that we will capture the same data described by the AFS is to require that the parser postulate traces whenever possible, and backtrack if they cannot be sustained. In this way, the Chain processor acts as a filtering mechanism

[15] Interestingly, there was also a substantial increase in reading time for the *say she* segment in the non-declarative sentences, suggesting that the AFS tries to fill either the direct object position of *say* or the subject of the embedded clause with the gap once *say* is reached but then encounters *she*. This side result is, however, not discussed by the Frazier and Clifton despite its support for the AFS.

[16] Note we might still restrict the amount of Chain information available to the PS processor — it may only need to know the category of any unresolved Chains, etc. But this remains a relatively unattractive position to take.

on possible traces. Despite the apparent inefficiency of this latter position, let us consider it in greater depth before retreating to the less modular stance. We may tentatively define the trace postulation strategy as follows:

(125) **Active Trace Strategy** (ATS): When constructing the phrase structure analysis, initially try to posit a trace in any potentially vacated position, i.e. as trace of $X^{max}$ or $X^{min}$ where the category of X is visible for phrase or head movement, respectively.

The concurrent, incremental operation of the modules means that once a trace is postulated in the PS representations, the Chain module will process it. Crucially, this evaluation takes place immediately — presumably before further input is processed. In §3.3.3 we assumed that a fundamental condition on well-formedness, was that every trace must be a member of exactly one well-formed chain. It is therefore clear that impossible traces will be identified immediately, when they cannot be appended to an existing Chain[17]. We assume that this forces immediate backtracking into the PS processor with negligible cost (essentially unobservable, at least by current experimental paradigms) since no incorrect Chain is ever constructed and no subsequent lexical material has been processed. In fact there is reason to suspect that this approach is no more expensive than a filler-driven model (except perhaps by some constant amount), an issue we will take up in chapter 6. Given this, the ATS makes essentially the same empirical predictions as the AFS but does not entail the use of chain information for the initial postulation of gaps, consistent with the encapsulated architecture presented here.

To summarise, the operation of the ATS predicts the following behaviour:

1. Posit traces wherever possible, restricted only on the basis of phrase structure information.
2. If a trace cannot be sustained (i.e. appended to a well-formed Chain), then backtrack, removing the trace and continue — negligible cost.
3. If trace is locally well-formed, but lexical information disconfirms immediately (in thematic structure), then there is some observable but minimal cost.
4. If the trace is locally well-formed, and only disconfirmed much later, then we commit and reanalysis is potentially more costly (subject to more general constraints on thematic re-interpretation, etc.).

The ATS has been motivated on the basis of evidence from English, combined with our criteria of informational encapsulation. We will argue below that the strategy also accounts for a variety of evidence from Dutch. While we have argued that the ATS is not particularly inefficient, it would seem inappropriate for such languages. At present, unfortunately, we are unaware of any evidence which bears on this issue.

[17] We will not discuss here the mechanisms for dealing with rightward moved constituents, where the trace occurs prior to its filler.

### 4.6.2 Against the use of Lexical Preference

One argument which has been levelled in the literature is that the postulation of gap is guided by lexical information [Ford *et al* 82]. Indeed, the examples which we have considered up to this point have all involved gaps which occur after transitive or preferred transitive verbs. This presents a challenge to both the AFS as proposed by Frazier (and her colleagues) and the ATS proposed here, both of which propose the resolution of filler-gap dependencies on the basis on minimal lexical information. This is due in large part to the fact that both accounts assume minimal lexical information in guiding the initial phrase structure analysis. This is particularly the case for the model proposed here, since the ATS is essentially a strategy implemented within the phrase structure module, and as naturally restricted to the informational vocabulary defined for that processor.

In their article concerning the operation of the AFS in two clause sentences (discussed above), Frazier and Clifton address precisely this issue. They used materials similar to those in (124) but replace the embedded verb with one which is preferred intransitive and vary the ultimate position of the gap, as in the following:

(126) a. "What$_i$ did you think the man whispered $\varepsilon_i$ to his fiancee during the movie last night ?"

b. "What$_i$ did you think the man whispered to his fiancee about $\varepsilon_i$ during the movie last night ?"

Both the AFS and ATS predict that the trace should be posited after *whispered*, resulting in increased complexity when the actual gap occurs later as in (126b). The results proved consistent with the AFS/ATS prediction, identifying a significantly longer reading time for the segment after *whispered* when it forced the late gap reading (the effect was observed in both single and double clause sentences).

#### Evidence from Dutch

As we noted in our discussion of the phrase structure processor, any theory of processing which is guided by lexical information will be confounded by head-final languages, unless decisions are delayed (see §4.4.3). To address this point, Frazier has conducted a number of experiments testing the nature of the gap-filling process in Dutch, a verb-final language. The first of these concerned preferences occurring in the processing relative clauses [Frazier 87a]. To test the preference (if any) between subject and object relatives, Frazier performed a self-paced reading study using unambiguous (determined by number agreement) and ambiguous examples (where the relative pronoun may be interpreted as either the subject or object of the relative clause) of the following sort:

(127) a. "Jan houdt niet van de Amerikaanse die de Nederlander wil uitnodigen.(Amb.)"
*John liked not the American who the Dutchperson wants to invite.*
*John liked not the American who wants to invite the Dutchperson.*
b. "Karl hielp de mijnwerkers die de boswachter vonden.(Unamb.)"
*Karl helped the mineworkers who found-PL the forester.*
c. "Karl hielp de mijnwerkers die de boswachter vond.(Unamb)"
*Karl helped the mineworkers who the forester found-SG.*

Examining both the reading time for the unambiguous sentences (as in (127b) & (127c)) and the answers to questions for the ambiguous sentences (127a) illustrated preference for the subject-relative reading (as in (127b)) in both respects (although the reading time advantage didn't quite reach significance, see [Frazier 87a] for discussion). In a more recent article, Frazier and Flores d'Arcais examined the processing of Dutch root clause where one of the arguments of the verb is raised into the [Spec,CP] position [Frazier & d'Arcais 89]. These studies revealed a similar preference to interpret the topicalised constituent as the subject, lending further support to both the AFS and ATS strategies.

### 4.6.3 Processing Gaps in the $2^{nd}$ Dimension

A fundamental difference between transformational model of grammar and alternative syntactic theories such as GPSG, LFG, and Categorial Grammar is the notion of movement and the resulting traces left in vacated positions. While all syntactic theories provide some mechanism for treating long distance dependencies (such as the use of 'slash' features in GSPG, C-Structure to F-Structure mapping functions in LFG, etc.), TG is unique in its explicit representation of canonical position via traces left by movement. While the use of traces is a perfectly reasonable mechanism in the construction of syntactic theory, it poses an interesting question to psycholinguists: Are traces 'psychologically real'? That is, are traces actually included in a person's mental representation of an utterance, or are they merely some formal mechanism for describing the well-formedness in a theory of syntactic competence?

It is possible that the traces employed in the principles and parameters approach might be equivalently represented by some alternative use of features which do not involve such an explicit structural representation. In the present model, however, we have taken traces to be real, at least in some sense, since they participate crucially in the formation of Chains. A more subtle and fundamental issue concerns the definition of psychological reality itself: what does it mean for a theory of grammar to be psychologically real? We will defer discussion of these issues until chapter 7, but will consider here some recent evidence which challenges traditional assumptions about the manner in which traces are processed. Specifically, we will argue that, if a trace-based account is to be maintained, the postulation of traces must obey an even more radical version of the ATS, but that the revised strategy is completely compatible with — and possibly derived from — the PIC.

In a recent article, Pickering and Barry contrast the trace-based mechanism of transformational grammar with a 'dependency-grammar' account, wherein a filler is associated directly with its subcategorizer (not mediated via a gap) [Pickering & Barry 91]. This contrast is illustrated by the following pair:

(128) a. "[Which man]$_i$ do you think Mary loves $\varepsilon_i$ ?"
b. "[Which man]$_i$ do you think Mary loves$_i$ ?"

The transformational model of grammar which we have assumed throughout, posits a gap in the object position of *loves* as in (128a), while a dependency grammar[18] account assumes that the filler is directly associated with *loves* as in (128b).

This alternative account of unbounded dependencies is potentially interesting from a psycholinguistic perspective, since it predicts that the sentence processor need not delay resolving dependency until a gap is posited, but rather that this association may be formed immediately upon encountering the subcategorizer. Unfortunately, however, for sentences as in (128), where the gap and subcategorizer are adjacent, it is difficult to formulate any testable empirical difference between the two accounts. Indeed, in the majority of evidence discussed above, the gap has immediately followed its subcategorizer. Pickering and Barry observe that, while there is significant evidence supporting the psychological reality of processing unbounded dependencies — such as the reactivation of the filler at the subcategorized position [Swinney *et al* 88], [Stowe 86],[Crain & Fodor 85] (see also [Bever & McElree 88], [McElree & Bever 89]) — the evidence is equally consistent with the alternative grammatical account discussed above. They point out that similar re-interpretation of the so-called 'filled-gap' effect (as discussed with respect to (123) above)[Stowe 86] is also possible.

This stalemate demands that we investigate sentences where the subcategorizing element and the position of the proposed trace are separated by intervening material. The obvious choice then is examine the extraction of objects which do not immediately follow the verb, as in the following:

(129) a. "[In which tin]$_i$ did you put the cake $\varepsilon_i$ ?"
b. "[In which tin]$_i$ did you put$_i$ the cake ?"

In this example, the trace-based account prohibits the resolution of the filler-gap relation until the end of the sentence, while the dependency account permits this to be resolved immediately upon encountering *put*. Thus the gap-free model appears to permit a greater degree of incremental interpretation, which seems to be intuitively borne out if we lengthen the direct object:

(130) "[In which tin]$_i$ did you put$_i$ the cake that your little sister's friend baked for you ?"

It seems that we are quite capable of recovering the fact that *In which tin* is the indirect object of *put*, long before we finish processing the direct object NP. Indeed, if we do not

[18] The particular account they propose is formulated within the general framework of categorial grammar, but might be similarly treated in, say, GPSG.

pied-pipe the preposition, forcing the interpretation of the indirect object to be delayed until the preposition is reached (since it is the subcategorizer of the NP), then the sentence becomes rather unwieldly:

(131) "[Which tin]$_i$ did you put the cake that your little sister's friend baked for you in$_i$ ?"

Furthermore, if we consider examples with multiple long-distance dependencies, the effect becomes even clearer:

(132) a. "John found the saucer [on which]$_i$ Mary put$^i$ the cup [into which]$_j$ I poured$^j$ the tea $\varepsilon_j$ $\varepsilon_i$."

b. "John found the saucer [which]$_i$ Mary put the cup [which]$_j$ I poured the tea into$^j$ $\varepsilon_j$ on$^i$ $\varepsilon_i$."

In the above, we show both the trace analysis, and the dependency analysis (using superscripts on the subcategorizers). Pickering and Barry suggest that while the trace-based account involves nested filler-gap relations for both sentences, the dependency account results in a nested filler-subcategorizer pattern for (132b) only, and a disjoint pattern in (132a). In this way, the difficulty of (132b) can be taken to result from the nested dependency pattern, an account which is given further support by the centre-embedded constructions which are similarly difficult:

(133) a. "I saw the farmer who$_i$ $\varepsilon_i$ owned$^i$ the dog which$_j$ $\varepsilon_j$ chased$^j$ the cat."

b. "The cat which$_i$ the dog which$_j$ the farmer owned$^j$ $\varepsilon_j$ chased$^i$ $\varepsilon_i$ fled."

Note, both the trace- and dependency-based accounts yield a similar pattern of dependencies for these sentences, with the nested pattern being inherently difficult to process. Thus if we accept that nested dependencies give rise to processing difficulty, then the trace-based account has no obvious explanation for difference in (132), since both examples would involve positing the trace at the end of the sentence yielding a nested filler-gap pattern. The dependency account, on the other hand successfully accounts for all the relevant examples.

### Positing Traces: A Revised Account

The arguments of Pickering and Barry, outlined above, present a serious challenge to the traditional views of 'gap-filling', and must be addressed in the model presented here. Crucially, however, traditional models of gap-filling tacitly assume that empty categories may only be posited once the relevant position in the string has been reached. In this way they are treated much as lexical items, despite the fact that they lack inherent phonological content[19]. An alternative is to assume that empty categories do not constitute part of the

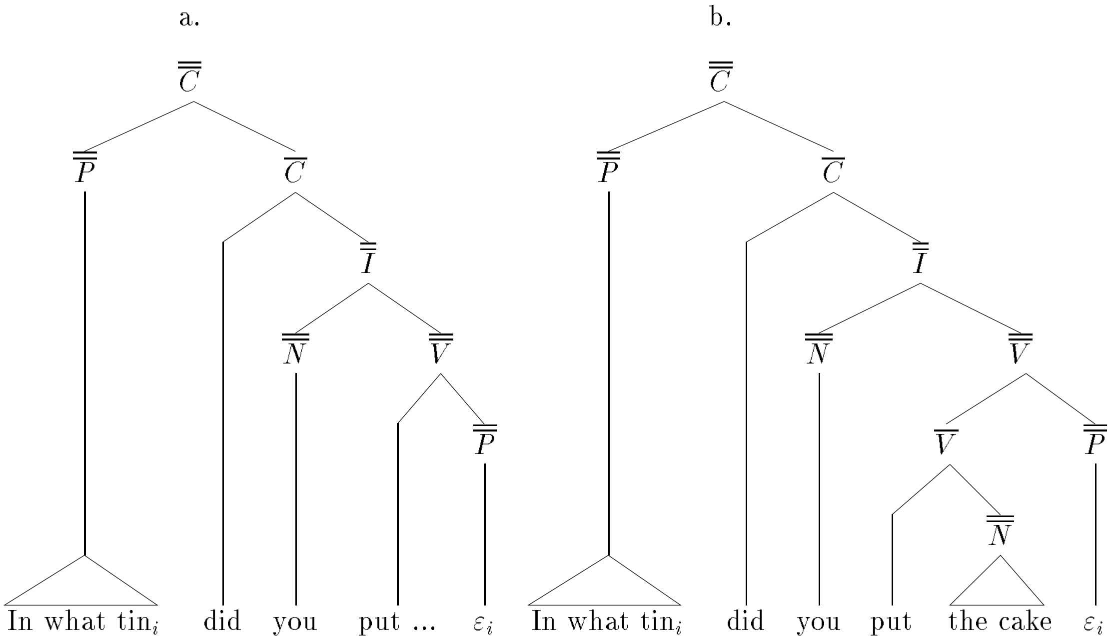
[19] This is not to deny that the existence of a trace may influence PF, as in the ubiquitous *wanna*-contraction. As we will see below, the proposed account remains completely compatible with current explanations of such examples.

PF 'yield' of a syntactic analysis, but rather are simply another aspect of the syntactic structure of an utterance, with a status similar to the branches of the phrase structure tree. If we adopt this view then there is no reason to delay positing a trace once the relevant attachment site exists in the constituent structure. Let's reconsider the following sentence, assuming the use of traces:

(134) "[In what tin]$_i$ did you put the biscuits $\varepsilon_i$ ?"

If we follow the above suggestion, then — once the verb *put* and its VP projection are incorporated into the structure — there is nothing to delay the postulation of the PP-trace as a complement:

(135)




Once this structure is built we can proceed to parse the remaining lexical material (i.e. *the cake*) — whether it precedes the trace (i.e. intervenes in the structure) or follows it, is of no concern so long as the independent principles of grammar are upheld.

If this proposal seems controversial, it is because our existing perception of gap-filling has been shaped by the use of '1-dimensional' characterisations of syntactic analyses; the string (e.g. (134) above) which constitutes the yield of a syntax tree (e.g. (135b) above) — which is itself a '2-dimensional' structure. While the 1-D characterisation may often be a suitable abbreviation for representing syntactic analyses, we have demonstrated that it has the potential to misguide our intuitions about processing, which is inherently concerned with the recovery of the 2-D structure. That is, empty categories should not be considered *a priori* part of the PF yield, and may be processed before that position is encountered.

To recap, our crucial proposal is simply to assume that empty categories are not part of the PF yield of an utterance, and thus there is no need to wait until they are 'encountered'.

Indeed to suggest this would be a category error. This is *not* to say that empty categories are not psychologically real: They are real in the same sense that constituent structures are real. Indeed, empty categories may 'influence' PF without being an explicit aspect of that level of representation. So just as various theories link prosody with constituent structure (see for example [Selkirk 84] and [Steedman 90]), we might similarly have constraints which block contraction (as in *wanna*-contraction) on the basis of an intervening WH-trace in the structure.

The result is what might be descriptively called a *hyper*-Active Trace Strategy. That is, the Active Trace Strategy (ATS) proposed above now operates such that it posits a trace in any potentially vacated position, regardless of where that position is in the string yield — thus a trace may be postulated even sooner than was dictated by the AFS of Frazier discussed earlier. In fact there is no need to redefine the ATS, it is simply less 'inhibited' given our revised interpretation of the status of empty categories. Indeed, just as we assumed the active prediction of (functionally selected) constituent structure was derived from the PIC (recall §4.4.3), we can similarly assume that PIC forces the postulation of traces in a similarly 'active' manner, so as to ensure that antecedent-trace relationships are resolved incrementally, at the earliest possible moment.

Further evidence which bears upon this concerns the ambiguous attachment of adverbials in multiple clause sentences. There is a consensus in the literature for something equivalent to the Late Closure strategy of Frazier (given earlier in (11)), preferring attachment into the most recently parsed (and hence 'lower') constituent[20]. Consider the following examples:

(136) a. "You told me (that) your mother's friend quit her job [last week]."
b. "I told you (that) your mother's friend quit her job [because it seemed a good idea at the time]."
c. "When$_i$ did you tell me (that) your mother's friend quit her job $\varepsilon_i$ ?"
d. "Why$_i$ did you tell me (that) your mother's friend quit her job $\varepsilon_i$ ?"

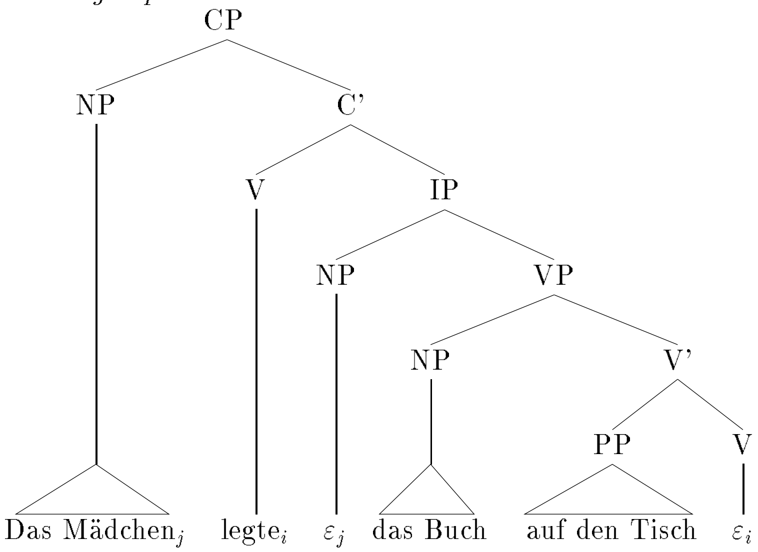
In the (136a&b) there seems a clear preference for interpreting the adverbial phrase (in square brackets) as modifying the lower, embedded clause, as predicted by Late Closure. If we assume that the postulation of gaps obey similar attachment strategies, then Late Closure would similarly predict the lower interpretation of the wh-extracted adverbial in (136c&d). The revised model proposed here, however, will prefer the hypothesis of a trace as soon as the higher verb is parsed (assuming that the adverbial is base-generated as a modifier of either VP or IP), which — if sustainable grammatically — predicts a preference for interpreting the wh-extracted adverbial as modifying the root clause. This preference seems clearly borne out by the above example — only if the higher interpretation is implausible, does the low-attachment reading appear to obtain.

Finally, let us consider some additional support for our approach which indicates that the ATS applies for traces resulting from both $X^{max}$ and $X^{min}$ movement. In particular, we have assumed that the canonical structure of German and Dutch is V and I final. In both languages, however, the highest verb (either auxiliary or main) raises to the beginning of

[20] See for example [Kimball 73], or more recently [Gibson 91] and references therein.

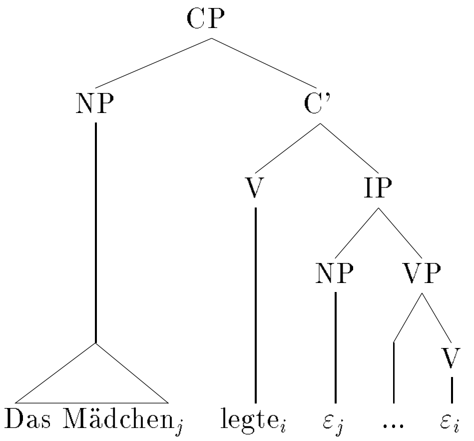
the sentence (to $C^0$) followed by topicalisation of some phrase to the [Spec,CP] position; the so-called verb-second, or V-2, phenomena. To illustrate, this considering the following example:

(137) "Das Mädchen legte das Buch auf den Tisch."
*The girl put the book on the table.*




Note, both the English and German sentence have similar word order, but the structure illustrates that in German, the D-structure position of the verb occurs at the end of the sentence. If we assumed the traditional gap-filling strategies, then German and Dutch hearers would be forced to delay the use of the verbs selectional information until the end of the sentence, after all the complements have been parsed. This contrasts with Stowe's evidence for English which seems to demonstrate the incremental use of thematic information during processing [Stowe 89]. Thus the traditional assumptions about gap-filling imply that — for sentences with virtually identical word order — English hearers have the advantage using thematic information incrementally, while German hearers do not. Given the account motivated above, however, we can achieve an equivalent degree of incrementality for both language types. Consider the structure for the following partial input:

(138) "Das Mädchen legte ..."




Given the prediction of functional constituent structure argued for earlier, we can posit both the subject trace (in [Spec,IP])[21] and the trace for the head of the VP, since the necessary structure exists for the postulation of these traces. This in turn permits the relevant thematic structure to be recovered, allowing the selectional properties of the verb to be consulted immediately. Thus if the sentence fragment was *Das Buch legte* ..., we could quickly determine that the topicalised NP was not a plausible agentive subject, and reanalyse.

### 4.6.4 Summary

In this section we have discussed the process of resolving antecedent-trace relations. We have reviewed the empirical evidence which is descriptively characterised by Frazier's *Active Filler Strategy*; the preference for filling a gap rather than attaching lexical material, with little initial regard for specific lexical information. Given, our assumptions here concerning the strict modularity of the phrase structure and Chain processors, we hypothesised the operation of an *Active Trace Strategy* (in the PS module) which continually attempts to posit traces which are sustained only if they can be incorporated into a well-formed Chain structure.

On the basis of the recent, and rather convincing, arguments of Pickering and Barry, we further suggest a slight revision[22] to the treatment of empty categories (here we have discussed traces in particular). We have suggested that they are not part of the PF yield for a syntactic structure, and are manifested in the syntax alone. Given this, the notion of 'encountering' a gap is simply not well-formed, just as we do not 'encounter' branches of constituent structure (or labelled bracketings). To the extent that this is the case, there is no reason to delay the postulation of traces until the relevant position in the string yield is

[21] For simplicity, we have shown the subject in [Spec,IP], but there is no reason we could not also postulate a trace in [Spec,VP] — assuming the subject is based generated in this position — and construct the three element for the topicalised NP.

[22] In fact this is not a revision to the theory *per se* — since the theory does not take any particular position on this point. Rather we are simply making our assumptions concerning the status of empty categories explicit.

reached. On the contrary, the trace can be attached as soon as an appropriate position in the syntactic structure exists.

## 4.7 Summary

In this chapter we have developed a modular theory of performance based on the informational domains determined by the grammar. We have assumed that the syntactic processor, and hence each of the constituent modules is governed by the overarching PIC. That is, each module must operate concurrently, incrementally constructing its output representation as input is received. Each module is, however, strictly encapsulated, and concerned exclusively with its own representation type, and subset of grammatical principles.

Within the modules we have suggested that a number of strategies are operative. In general, the ATS demands the constant and early postulation of traces so as to resolve chains as quickly as possible. In addition, the strategies AA and DSA determine preferences for attaching constituents in the face of ambiguity. We have, however, argued that all of the strategies are derived from the PIC, since each is aimed at optimising comprehension during incremental processing.

# Chapter 5

# A Logical Model of Computation

In chapters 3 and 4, we have presented a model of sentence processing and its grammatical basis. We have suggested that the human syntactic processor consists of four modules. Each module is responsible for recovering its own representation with respect to its input. We have further outlined some aspects of the model's computational behaviour: At a relatively high level, we propose that the modules operate concurrently and incrementally, with each constructing a maximal output representation for the current partial input, as input is encountered. In addition to each module satisfying the requirement of incrementality, we have also hypothesised several strategies, located primarily within the phrase structure module (since this is where the greatest amount of ambiguity occurs).

We have, however, said little about the nature of the specific algorithms employed by the individual processors. That is, the computational 'behaviour' outlined in the theory is satisfiable by a number of implementations, and potentially on a variety of architectures. This is natural given the methodological stance we have motivated here: Taking into account our poor understanding of the mind's processing and memory architecture, we have sought to develop a theory of sentence processing which is based on current syntactic and psycholinguistic theorising, both of which are supported empirically, without making arbitrary assumptions about the specific underlying computational architecture.

To construct a computational model of the theory which is highly specific and optimised for a particular architecture would therefore be a non-sequitur in a our research programme, since such an implementation will have greater 'resolution' than the theory itself. Rather, our aim here is to illustrate that the underlying computational assumptions of the theory may indeed be naturally implemented. To be specific, there are three fundamental aspects of the theory which may seem controversial and would benefit from an explicit computational realisation:

(139)
1. The combined assumptions of modularity and incrementality.
2. The tractability of encapsulation.
3. The direct use of grammatical principles, on-line.

The first point entails that the model reflect the modular organisation and concurrent, incremental operation present in our theory of performance. The second point concerns the nature of module interaction during processing: We shall demonstrate that seemingly counter-intuitive strategies such as the ATS, and the minimal use of lexical information prior to the construction of thematic structure, are not particularly inefficient. Furthermore, the benefits of modularity can be argued to significantly out-weigh any minor increases in the complexity of individual modules. Finally, we implement a fragment of the principles and parameters theory, for both English and German, and discuss how the locality of grammatical conditions is relevant to incremental operation of the constituent processors. In sum, the model provides a means for illustrating the basic computational behaviour we have proposed: we do *not* describe here a parser which incorporates the full syntactic coverage and range of parsing strategies proposed in the theory.

In addition to developing a model which distinguishes those aspects of computation which are determined by the theory from those which are not, a further aim is to identify the class of possible process models which are consistent with the theory. Beyond this, however, the modules can be considered 'black boxes' which might be realised by a variety of algorithms and vary depending on the ultimate platform they are implemented on. So rather than providing a computational 'existence proof' of the theory we have proposed, our intent is to construct a computational model which contributes to the *sufficiency* of the theory: "the ability of the theory to explain how actual instances of behaviours are generated" [Pylyshyn 84, page 75], by providing a formal mechanism to construct such 'explanations'. This is particularly important given the highly modular and principle-based theory we have developed: While the component parts of the theory are simple, in and of themselves, they interact in potentially complicated ways to predict and explain instances of human sentence processing behaviour. Indeed it is largely the process of implementing the theory which is of value — entailing the explicit formalisation of often tacit assumptions in the theory — rather than the end product. Although once constructed, the computational model is a valuable tool for generating behaviours predicted by the theory, essential for both revision and refinement.

A related benefit of the computational model, is that it helps to illuminate the computational properties of the theory: To what extent can the organisational and procedural aspects of the performance theory be naturally implemented? Our aim in this respect is thus twofold: In addition to providing an explicit, formal mechanism which realises fundamental aspects of the theory, we will also demonstrate that the proposed theory can be implemented naturally. In particular, we will show that the modular organisation and incremental operation may be transparently realised, and that the implementation is both efficient and behaves in the expected manner.

The discussion of the computational model is the concern of both this chapter and the next[1]. Here we include some general discussion of principle-based parsing and the use of the logic programming paradigm which forms the foundation of the computational model we propose. We further discuss the 'higher-level' aspects of the model's implementation including the specification of the model's modular organisation and the control mechanism

[1] Earlier versions of the material presented in this chapter and the next have appeared in [Crocker 91b] and [Crocker 91c].

which is used to 'co-routine' the modules. In chapter 6, we elaborate on the implementation of the modules themselves; the manner in which they make use of the grammatical principles and how the proposed strategies may be implemented.

## 5.1 Principle-Based Parsing

Throughout this thesis, we have assumed a process model which uses the principles of grammar directly. Thus before we proceed it makes sense to consider some relevant contributions of computational linguistic research to the construction of such principle-based systems. It is interesting to note that the shift in modern syntactic theory from large systems of homogeneous, construction-specific rules to a set a abstract, heterogeneous, language universal principles and parameters has witnessed a lag of almost a decade in computational linguistics. That is, it was not until about 1984 that the first 'GB-based' parsers began to emerge, expanding to a handful by 1988, and even now the community of researchers remains a small minority.

The basic reasons for this are two-fold: First, principle-based syntactic theories such as GB theory are typically complex, unstable, and — by computational standards — informal. Thus there is no clear cut specification of what a GB grammar is. Secondly, there has been a reluctance to abandon the traditional parsing technology, which is rendered largely inadequate by the modular, heterogeneous and abstract nature of the theory. There are, however, a number of arguments in favour of developing principle-based systems. The most obvious is that it allows the exploitation of 'state of the art' syntactic theorising. In this way we might also contribute to the formalisation of syntactic theory as it develops (see [Stabler 89b] in particular). From an 'engineering' perspective, there are other potential advantages: Principle-based systems are much more compact and may well be easier to maintain than construction-based systems. While the interaction of principles is typically more complex, the principles themselves are relatively simple and prohibit the 'yet another special rule' approach which is rife in the development of construction-based systems. Furthermore, given the typically large numbers of rules involved in the latter, there is no notion of an underlying theory which can be used to justify particular rules, while principles must remain consistent with the overall theory of grammar. A further appeal is that principle-based systems may be designed to apply cross-linguistically, sharing the fundamental grammar and parsing machinery, while construction based systems are inherently language specific.

While a review of specific principle-based systems would take us rather too far afield, it is relevant to highlight the basic techniques which have been developed. The main issues which have been addressed by existing systems are:

(140)
- Preserving the modular status of the theory.
- Separating the universal principles from language specific aspects (parameters and lexicon) to maintain the cross-linguistic status of the principles.
- Separate structure building from structure verification.
- Achieving efficiency and broad coverage.

The majority of systems use some combination of a parser, to build possible constituent structures on the basis of $\overline{X}$-theory, in conjunction with a set of procedures which annotate the phrase structures and ensure their well-formedness with respect to the other principles of grammar (the earliest such system is [Wehrli 83], [Wehrli 88]). That is, there might be a module which performs Case assignment and enforces the Case Filter, a module to assign $\theta$-role and verify the $\theta$-criterion, etc. This approach was taken by Sharp in his bilingual systems for English and Spanish, which built a candidate S-structure for the sentence, followed by application of the various grammatical principles [Sharp 85] [Sharp 86]. If the structure violated any principle, then the system backtracks into the parser, to find any alternative analysis, and proceeded in this manner until a suitable structure was found. While successfully dealing with reasonable fragments of both English and Spanish, the system suffered from obvious efficiency problems: if the parser chooses an unsustainable structure early on, it would potentially not be identified until the whole sentence had been parsed, and would thus require significant backtracking to repair.

The next generation of systems adopted a strategy of 'inter-leaving' the building of structure with the application of principles, so as to identify erroneous structures early, and localize backtracking. Such systems include Dorr's UNITRAN parser for English and Spanish, which used a set of phrase structure compiled-out on the basis of $\overline{X}$-theory and language specific parameters [Dorr 87], and Crocker's system for English and German which incrementally constructed phrase structure and Chains, using the rules of $\overline{X}$-theory and language specific parameters on-line [Crocker 88]. Both of these systems, and that of Sharp's, were applied to syntactic, bi-directional translation between their respective languages, stressing the cross-linguistic application of such approaches (see [Crocker 91a] for further discussion). More recently, Fong has experimented with the use of more dynamic techniques for controlling the application of principles. In particular he uses 'types' to determine the applicability of particular principles at any point in processing[2].

Despite the efforts to reflect the modular and cross-linguistic status there remains a tension between principle-based grammars and parsers. The shift away from rules has led to a view of principles as 'well-formedness' conditions which lack the natural procedural interpretation which was traditionally assigned to phrase structure rules, i.e. as rules which 'build' structures. Now, even the 'rules' of $\overline{X}$-theory are more naturally considered as conditions on phrase structure, rather than rules for building such structure. Indeed, it is theoretically incorrect to use $\overline{X}$-theory directly to build candidate S-structures, since $\overline{X}$-theory is typically assumed to apply at D-structure.

In an effort to construct more faithful and transparent realisations of principle-based systems, there has been recent interest in so-called 'deductive' methods of parsing. This tack separates the axiomatisation of grammatical principles — a purely declarative specification — from the procedures which use these axioms to 'prove' derivations of syntactic analyses. This approach has a number of methodological advantages, especially for the task at hand, where we wish to consider the range of possible ways syntactic knowledge might be brought

[2] Fong uses a similar organisation to the systems discussed above; i.e. an $\overline{X}$ structure builder in tandem with the application of constraints. His approach is not unlike the use of 'constraint requests' used by [Crocker 88], to determine what constraints 'remained to be satisfied'/'are relevant' in a particular unit of structure.

to bear in linguistic performance.

## 5.2 A Logical Model of Performance

In the same manner that psychological theory has made an explicit distinction between competence (what we know) and performance (how we use that knowledge), logic programming separates the declarative specification of the 'relation' to be computed from its execution (how to compute it). A program specification consists of a set of axioms from which solution(s) can be proved as derived theorems. Within this paradigm, the nature of computation is determined by the inferencing strategy employed by the theorem prover. As a result, it is possible to change the manner in which a program is executed without altering the declarative specification of the program. The two fundamental techniques for manipulating execution are:

(141)
- **Control:** Altering the procedure for searching the proof space. This is typically achieved by changing the inferencing strategy used by the theorem prover, or by implementing 'meta-interpreters' on top of the underlying inference engine.
- **Transformation:** Translation of the program axiomatisation into an equivalent one. By altering the program axiomatisation we indirectly affect the path pursued by the theorem prover through the solution space.

A central advantage of the logic programming paradigm is that it permits the declarative specification of *what* is to be computed, quite distinct from *how* computation is to take place[3]. One benefit of this is that we can independently vary the declarative specification (the logic) or the inferencing strategy (the control)[4]. However, while the declarative and procedural aspects are distinct, the relationship between them is transparent: The inference engine simply determines how the axioms are applied during the proof of a solution. Furthermore, assuming the theorem provers are both *sound* (don't prove incorrect theorems) and *complete* (are able to prove all theorems), there is no sense in which declarative knowledge can be 'hidden' within the procedural strategy. That is, if a particular theorem is proved or refuted, we know this result follows from the axiomatisation.

### 5.2.1 Parsing as Deduction

The exploitation of the distinct logical and procedural components of logic programming is naturally applied to parsing; the so called *Parsing as Deduction* hypothesis. In particular it has been shown that meta-interpreters or program transformations can be used to affect

[3] This is not to deny that in particular logic programming environments, such as Prolog, programs are 'specified' with implicit knowledge of how the particular inference engine will interpret them. Crucially, however, it is *possible* to keep the two distinct; first constructing a declarative specification, and then proceeding to determine appropriate control strategies or transformations which yield efficient computation.

[4] Variation in the specification may indirectly affect how computation takes place, just a transformation of the specification will, but the basic control mechanism remains unchanged.

the manner in which a logic grammar is parsed [Pereira & Warren 83]. That is, for a given logic grammar we can derive a variety of parsers such as LL(k), LR(k), and Earley, among others. One problem, however, is that not all parsing strategies are suited to all grammars. A recursive descent parser, for example, may not terminate for a grammar with left recursive rules[5].

Recently, there has been an attempt to extend the PAD hypothesis beyond its application to simple phrase structure logic grammars. In particular, Johnson has developed a prototype parser for a fragment of a GB grammar [Johnson 89]. The system consists of a declarative specification of the GB model, which incorporates the various principles of grammar and multiple levels of representation. Johnson then illustrates how the fold/unfold transformation, when applied to various components of the grammar, can be used to render more or less efficient implementations. Johnson also demonstrates how goal *freezing*, an alternative Prolog control strategy, can be used to increase efficiency by effectively coroutining the recovery of the various levels of representation, allowing all principles to be applied as soon as possible, at all levels of representation.

The deductive approach to parsing is theoretically attractive, but unsurprisingly inherits a number of problems with automated deduction in general. Real automated theorem provers are, at least in the general case, incomplete. That is, they cannot *a priori* be guaranteed to return all (or any) solutions to a given request. One instance of this is the left-recursion example noted above: a perfectly legitimate grammar rule will cause the Prolog inference engine to pursue an infinite path and never halt. While it is possible to solve or at least detect some of these problems, especially for parsing algorithms and grammars formalisms which are well understood[6], it is certainly not possible in the general case. We can imagine that a true, deductive implementation of GB would present a problem. Unlike traditional, homogeneous phrase structure grammars, GB makes use of abstract, modular principles, each of which may be relevant to only a particular type or level of representation. This modular, heterogeneous organisation therefore makes the task of deriving some single, specialised interpreter with adequate coverage and efficiency, a very difficult one. In the following section, we endeavour to illustrate the nature of the difference between construction-based and principle-based grammars, and the implications for automatic syntactic analysis using the deductive approach.

### 5.2.2 Deductive Parsing: Rules vs. Principles

The traditional characterisation of language in terms of construction-based systems (i.e. systems which use individual rules to describe units of structure, as exemplified by a context-free grammar) has significantly influenced our views about parsing. In particular, there is a tendency for parsers to use the rules of grammar to 'generate' instances of structure. This is possible because of the homogeneous nature of construction-based grammars, where a

---

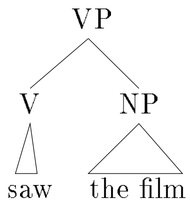
[5] For a thorough exposition see [Pereira & Shieber 87]. Note, in some cases it is possible to determine the relevant properties of a phrase structure grammar so that an inadequate strategy is not employed. See [Abramson *et al* 88] for demonstration of this point.

[6] As another example, many bottom-up algorithms cannot be guaranteed to succeed if there are empty productions in the grammar.

single rule is sufficient to license a particular unit of syntactic structure (i.e. the branching structure which corresponds to that rule).

Crucially, however, this property of 'rule-to-structure' correspondence does not obtain in principle-based grammars, where particular units of structure must satisfy the collection of relevant principles. Furthermore, any given principles might only be concerned with a particular aspect of that structure. Consider, for example, the following VP structure:

(142)




In a typical phrase structure grammar, this structure (not including the subtrees of V and NP) is licensed by the single rule, $VP \rightarrow V\ NP$. In a principle-based grammar, the structure is only well-formed if it satisfies a number of principles: $\overline{X}$-theory licenses the basic structural configuration (i.e. the complement NP as sister to the $V^{min}$ projection, and dominated by VP), the $\theta$-criterion is satisfied since the NP both requires a $\theta$-role and occupies a $\theta$-marked position, and finally, the NP must satisfy the Case filter (and does receive Case from the transitive verb). Given that the principles are each concerned with only a particular aspect of the syntactic structure, none are particularly appropriate for generating the structure. In sum, principles are more naturally viewed as constraints on syntactic structures, rather than generators of them.

The approach adopted in most existing principle-based parsers is to treat the rules of $\overline{X}$-theory as structure generators, and then apply the principles as constraints on these structures (this is the basic technique employed by [Sharp 85] [Dorr 87] [Crocker 88] and [Fong 91] to name a few). This technique has a number of potential disadvantages, however: In the first place, as we noted earlier, $\overline{X}$-theory is a principle of D-structure in most current instantiations of the theory, and hence is not sufficient for generating the set of possible S-structure configurations[7]. Secondly, there has been an increase in support for the abolition of $\overline{X}$ 'rules' *per se*, favouring feature-based constraints derived from more fundamental properties of lexical items (see [Speas 90], [Cann 88], [Muysken 83] for example).

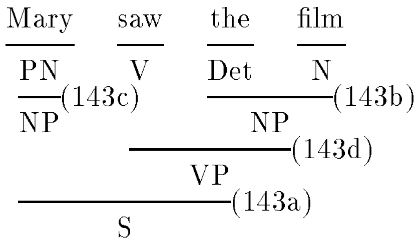
The fundamental difference between rule- versus principle-based systems — i.e. rules as 'generators' versus principles as 'constraints' — is made more precise when cast in terms of deductive parsing. To begin, consider the following (horn clause) axiomatisation of a simple context-free grammar:

[7] That is, surface structures may be composed of both the configurations licensed by $\overline{X}$-theory and those which result from transformations, such as adjunction.

| | | | | |
|---|---|---|---|---|
| (143) | (a) | S | ← | NP ∧ VP |
| | (b) | NP | ← | Det ∧ N |
| | (c) | NP | ← | PN |
| | (d) | VP | ← | V ∧ NP |
| | (e) | PN | ← | "Mary" |
| | (f) | Det | ← | "the" |
| | (g) | N | ← | "film" |
| | (h) | V | ← | "saw" |

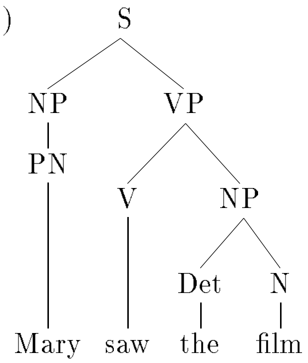
Now, we can illustrate the derivation or 'proof' of a sentence S for the string *Mary saw the film*, as follows:

(144)




Borrowing a notation widely used in the Categorial Grammar literature (as introduced in [Ades & Steedman 82]), this derivation illustrates a complete proof of the theorem: S ← *Mary saw the film*, derived from the axioms given in (143)[8]. Furthermore, the derivation transparently represents the (inverted) phrase structure for the sentence, where derivation steps are translated into branches connecting the consequence (below the derivation line) to it premises (above the derivation line) — this follows from the fact that the rules/axioms directly characterise the well-formed units of structure, as discussed above:

(145)




[8] In fact, the reader may have noticed that linear order is implicit in this system. That is, NP ← Det ∧ N, implies that Det is adjacent to and precedes N. Furthermore, any theorem (i.e. derived result) or premise (i.e. lexical item) may only be used once in the course of the derivation. This may be defined naturally within the general framework of a linear logic. For our purely illustrative purposes, however, a thorough and formal exposition of the assumed logical framework would take us too far afield.

Note, it is possible to construct our derivation such that we record the proof/constituent structure as we go along. This is accomplished by writing the derived consequence X as $[_X$ Prem1 ...PremN], where each premise in turn is the structure of the sub-proof:

(146)

$$\dfrac{\dfrac{\dfrac{\text{Mary}}{\text{PN}}}{[_{NP}\ \text{PN}]}(143c) \quad \dfrac{\dfrac{\text{saw}}{\text{V}} \quad \dfrac{\dfrac{\text{the}}{\text{Det}} \quad \dfrac{\text{film}}{\text{N}}}{[_{NP}\ \text{Det N}]}(143b)}{[_{VP}\ \text{V}\ [_{NP}\ \text{Det N}]]}(143d)}{[_{S}\ [_{NP}\ \text{PN}]\ [_{VP}\ \text{V}\ [_{NP}\ \text{Det N}]]]}(143a)$$

Thus the final derivation is in fact a bracketed list with a structure equivalent to (145)[9]. Crucially, while it is possible to carry the sub-proof as we construct the derivation (as in (146)), this structure is not *used* by the axioms of grammar (indeed, (144) does not build such a proof record). In early transformational grammar, however, we can write such structure sensitive rule, such as the passive[10]:

(147) $[_{S}\ [_{NP^1}\ \text{PN}]\ [_{VP}\ \text{V}\ [_{NP^2}\ \text{Det N}]]] \leftarrow [_{S}\ [_{NP^2}\ \text{Det N}]\ [_{VP}\ \text{V}\ [_{PP}\ \text{by}\ [_{NP^1}\ \text{PN}]]]]$

As we discussed above, current conceptions of the phrase-structure component within the principles and parameters approach are heading away from a rule-oriented characterisation. This is exemplified by the *Project Alpha* proposal of Speas, where the projection of structure from the lexicon is free, and only subsequently constrained by the syntax [Speas 90]. This move is essentially the final step in eliminating the rule component in favour of a pure 'licensing' grammar. In sum, the view is one where syntax simply constrains (or, licenses) virtually arbitrary structures, rather than generating them. While we do not attempt to provide a detailed axiomatisation of such a grammar here, the relevant consequences of the licensing grammar approach can be revealed by the following, rather simplistic 'principle-based' grammar[11]:

---

[9] See [Stabler 87] for a discussion relating the proof procedure assumed in (144) with an equivalent one which records the proof tree, as in (146).

[10] Note, this transformational rule is a simplified reformulation of (28), where the premise is the 'structural description' (SD), and the consequence is the 'structural change'.

[11] The definitions given for the named principles are only intended as simple approximations for the purposes on this exposition, and are not to be construed as axiomatisations of the actual grammatical principles. This toy fragment is only intended to demonstrate how multiple axioms can interact so as to determine the well-formedness of a particular construction.

(148) $\overline{X}$-theory:

(a)$[_{X^i}\ \mathrm{Y}^j\ \mathrm{Z}^k\ ] \rightarrow$
X = Z, i=j=2, k $\leq$ 1, Y is-spec-of X
*or,*
X = Y, j=0, k=2, 1 $\leq$ i $\leq$ 2.

(b) $[_{X^n}$ Word$] \rightarrow$ cat(Word,X), 2 $\geq$ n $\geq$ 0.

Case Filter:
(c) $[_Z\ \mathrm{X}\ \mathrm{N}^2\ ] \rightarrow$ is-a-case-assigner(X).

$\theta$-Criterion:
(d) $[_Z\ \mathrm{X}^0\ \mathrm{Y}^2\ ] \rightarrow \theta$-marks($\mathrm{X}^0$,$\mathrm{Y}^2$).

Lexicon/Parameters:
(e) cat('the',D).
(f) cat('film',N).
(g) cat('saw',V).
(h) is-a-case-assigner(V).
(i) $\theta$-marks($\mathrm{V}^0$,$\mathrm{N}^2$).
(j) D is-spec-of N.

In these rules, we simply assume that branch structures of the form $[_{Mother}$ LeftDaughter RightDaughter ] which are unifiable with the left-hand side of an axiom, must satisfy the conditions on the right. This axiomatisation differs from that for the CFG in (143) above in two respects: Firstly, the left hand side is a complex term consisting of the derived category and the premise categories of which it is composed. The result is that we can define principles which are sensitive to the local structure in which a constituent occurs. Secondly, the axioms are *necessary* conditions (i.e. '$\rightarrow$'), *contra* the sufficient conditions defined by the CFG rules. The result is that any derived structure must satisfy the conditions of all axioms which are relevant for that structure. Given this set of axioms, consider the following proof of VP $\rightarrow$ *saw the film*:

(149)

$$\text{Mary}\quad \cfrac{\cfrac{\text{saw}}{[_{V^0}\ \text{saw}]^{\{148b,g\}}} \quad \cfrac{\cfrac{\text{the}}{[_{D^2}\ \text{the}]^{\{148b,e\}}} \quad \cfrac{\text{film}}{[_{N^1}\ \text{film}]^{\{148b,f\}}}}{[_{N^2}\ \mathrm{D}^2\ \mathrm{N}^1]^{\{148a,j\}}}\,(150\mathrm{IIa})}{[_{V^2}\ \mathrm{V}^0\ \mathrm{N}^2]^{\{148a,c,d,h,i\}}}\,(150\mathrm{IIa})$$

The individual lexemes support the derivation of their projected category nodes by axiom (148b) (and, indeed the respective lexical axiom (148e,f, or g)). The derivation of the NP *the film* is consequently derived from the single axiom (148a). Note that only one axiom has a matching structure (i.e. left hand side) for each of these derivations, although further axioms are accessed within them (i.e. the 'cat' and 'is-spec-of' axioms). For the final step — the derivation of a VP from a verb and object NP — this is not the case. If, for example, we derive the $\mathrm{V}^2$ on the basis of the $\overline{X}$-rule (148a), we see that both the Case Filter (148c) and the $\theta$-Criterion (148d) have 'unifying' left hand sides, and hence must be satisfied (precisely because they are necessary conditions). It is also crucial to notice that their is no implied order in which (148a,c&d) are applied.

From this example, we can see that the axiomatisation and derivations for a principle-based — or, more relevantly, licensing — grammar are rather more complex than for a traditional 'construction-based' CFG. Most importantly, the derivation of a particular unit of structure is not supported by an individual (sufficient) axiom characterising precisely that construction, but must rather be consistent with the complete set of (necessary) axioms — each of which might only constrain some aspect of that unit of structure. Thus, while the derivation of the VP given in (149) does mirror the VP's constituent structure, the derivation is only an abbreviated proof — not a complete proof as was the the case for the CFG derivations. That is, the final step deriving $V^2$ only states that the structure $[_{V^2}$ $V^0$ $N^2]$ is well-formed with respect to the relevant principles of grammar (i.e. the axioms (148a,c&d)) — the derived structure does not directly correspond to the application of one particular rule.

Before proceeding, we should point out that while complex terms are used (e.g. $[_{V^2}$ $V^0$ $N^2]$ instead of just VP), the axioms above do not exhibit the power of transformational rules as exemplified by (147). The complex terms simply encode the constituent premises, so that we can define axioms with respect to local structural contexts. Clearly, a more complete axiomatisation incorporating movement, would entail the use of such *tree transducing* axioms, which permit the description and constraint of Move-$\alpha$[12]. This would would thus eliminate the strictly 'local' characterisation of principles as given in the toy fragment above. As we discuss in the following section, however, the reformulation of the transformational model proposed in chapter 3 should allow us to maintain the strictly local characterisation of principles with respect to their representation type.

To summarise, the difference between rule-based and licensing grammars can be conceptually characterised as follows: Assuming that the task of a grammar is to define the set of well-formed syntactic analyses of a language, the rule-based grammar provides a set of sufficient conditions for the construction of particular units of structure (e.g. to construct an NP it is sufficient to have a determiner and a noun, NP $\leftarrow$ Det N). In a licensing grammar however — and I assume that principle-based grammars are licensing grammars — the axioms do not 'generate' well-formed analyses (or, more precisely, well-formed formulae (wff)). Rather, the axioms are necessary conditions which when given some structure, determine whether or not it is a candidate member of the set of wffs (only once all the relevant axioms are satisfied can we be sure it is indeed a wff). Notice for the proof in (149), that it is not clear how the derivation is 'constructed' — we have simply shown that it is indeed well-formed. Indeed, a necessary addition to the axiomatisation of such a licensing grammar is the definition of a 'potential wff', in this case the set of (all) binary branching trees (with nodes uninstantiated), since this is the space of wffs restricted by the grammatical axioms[13]. This can be formalised straightforwardly as follows:

---

[12] See [Stabler 89b] for an elaborate exposition of this approach.

[13] But note that such a definition is not really part of the grammar. Rather it simply defines the kind of structures the grammar makes reference to. In the case of a rule-based grammar, the axioms actually generate these structures themselves.

(150) I. Two sets of nodes:
(a) The set $T$ of all terminals.
(b) The set $NT$ of all non-terminals.

II. The set $B$ of branches:
(a) if $X, Y, Z \epsilon NT$, then $[_X\ Y\ Z\ ]\ \epsilon\ B$
(b) if $X \epsilon NT$, $Y \epsilon T$, then $[_X\ Y\ ]\ \epsilon\ B$
(c) there is nothing else $\epsilon\ B$

III. The set $PB$ of proper branches:
(i) $\alpha\ \epsilon\ PB$ iff $\alpha\ \epsilon\ B$, and
(ii) and $\alpha$ meets all relevant necessary conditions in (148)

IV. The set $Tr$ of well-formed trees:
(a) if $\alpha = [_X\ Y\ ]\ \epsilon\ PB$, then $\alpha\ \epsilon\ Tr$
(b) if $A_t$, $B_t\ \epsilon\ Tr$ and $[_X\ A\ B\ ]\ \epsilon\ PB$, then $X_t = [_X\ A_t\ B_t\ ]\ \epsilon\ Tr$
(Note: we write $Z_t$ denotes a tree rooted at the non-terminal $Z$)

The definitions in (150) I & II simply define the space of nodes and branches under consideration, corresponding to the definition given in (81) in chapter 3. In (150) III, we then define the set of proper branches as those branches which are well-formed with respect to the necessary conditions (i.e. grammatical principles) defined in (148). Finally, a well-formed tree, is any tree which is composed entirely of proper branches.

This basic difference in the formalisation of rule-based versus licensing grammars has significant implications for (deductive) parsing. Given a rule-based axiomatisation, it is relatively simple to construct a parser or 'inference engine' which constructs a proof/analysis for a given theorem (i.e. the input string). Given a licensing grammar axiomatisation, we first need some mechanism which can generate arbitrary formulae (or, trees) so that we can then apply the axioms of grammar to determine which trees are grammatical (member of the set of wffs). When cast in these terms, the solution to developing an efficient deductive parser for a principle-based, licensing grammar seems rather elusive. One crucial property of this axiomatisation, however, is that by defining the principles of grammar as conditions on proper branches, we can directly define the set of well-formed syntactic trees. That is, we need not define the entire set of binary branching trees, and then subsequently restrict this set. Rather, the set of possible syntactic structures (trees, in this simplified grammar) is defined exclusively in terms of locally well-formed branches. As we shall see, this significantly contributes to the efficiency and incremental operation of model. It is this issue to which we now direct our attention.

### 5.2.3 Deduction in a Modular System

In the preceding section we observed that deductive principle-based parsers must make use of a set of necessary axioms, which act as constraints, combined with some procedure for generating potential formulae, such as trees. In his PAD parsers, Johnson achieves this by using 'tree constrainers' — axioms which (recursively) apply the various principles of grammar to branches of trees [Johnson 89]. Johnson's axiomatisation directly incorporates all the levels of representation of the standard transformational model, thus some principles constrain D-structure (such as $\overline{X}$-theory and $\theta$-theory), while other constrain S-structure (the Case Filter) and LF. In addition, Johnson's specification includes an axiomatisation of Move-$\alpha$, relating the various levels of representation. As he observes, the resulting Prolog interpretation functions essentially as a 'generate and test' parser — generating candidate D-structure from the lexicon, mapping them to S-structures, and then seeing if the PF yield (leaves of the S-structure tree) matches the input string. Unsurprisingly, this approach will be at best wildly inefficient, and at worst, won't terminate.

To improve the efficiency of the PAD parser, Johnson initially explored the use of both program transformations and varying control (i.e. using freeze) [Johnson 89], his more recent research has focussed on the former. In particular, Johnson has refined his earlier work on the use of the fold and unfold transformations (see [Tamaki & Sato 84]) for deriving efficiently parsable horn-clause realisations of a GB grammar axiomatisation [Johnson 91]. It is important to note, however, that while the transformed specification may be logically equivalent to the original, it does alter its organisational status. The unfold transformation, for example, is similar to a 'compiling-out' operation. Thus, while Johnson does not go to the extent of generating a set of phrase-structure rules from the original axioms, his transformations do move in this direction. The underlying modular, abstract organisation is partially evaluated (to some degree) so that it may be efficiently parsed by some existing inference engine (in this case, Prolog).

This use of transformations may well be fruitful in the 'engineering' of deductive, principle-based parsers which can be implemented efficiently using existing inference engines such as Prolog. Given the theory of sentence processing we have proposed, however, these techniques are inappropriate for constructing the computational model. In contrast with the single processor model employed by Johnson, the model we have developed consists of a number of processors over subsets of the grammar. Central to the model is a declarative specification of the principles of grammar, defined in terms of the representations listed in (179). A given principle of grammar is defined with respect to only one representation type. In addition, there are 'interface conditions' which control the mappings between the representation types. For example, the X-bar rule for complements determines local phrase structure (PS), while the mapping from PS to thematic structure further entails thematic assignment of the complement at TS.

In light of the above, it is essential that the computational model reflect both the *direct* use of grammatical principles and the modular organisation of the theory. Indeed, it is precisely our goal to determine the nature of the control strategies which are operative given these assumptions, and which further implement the strategies (AA, DSA, and ATS) motivated by the theory. Thus, to transform the theory so it conforms to some existing control strategy, like that of Prolog, would tell us very little. Rather, we begin with an axiomatisation of the model's organisation and the grammatical principles, and then implement the specialised inference engines which will use this axiomatisation in a manner which satisfies the performance requirements of the theory — i.e. concurrent operation of the modules, and obeyance of the proposed strategies. In this way, we demonstrate that the model has a natural and direct computational realisation and we can be certain principles are being used directly (not compiled together). As a result, we can inspect the precise control mechanisms which are involved in realising the model.

To maintain the modularity (of the process model, not just the syntactic theory), we propose that the computational model of the syntactic processor be realised as a set of specialised inference engines — one for each representation type. This is natural, given that we have already partitioned the various representation types; defining the representations themselves (PS,ChS,CiS, and TS) and outlining how the principles of grammar act of constraints on units of the local structure (or, schemas). In this way, the derivation of a syntactic analysis is not a single (abbreviated) proof as in Johnson's case (and in the toy example (148) discussed above). Rather, a complete analysis is the four-tuple of proofs for each representation type. In this way, we can treat each processing module as an independent theorem prover whose task is to show that its input is one of the wff's (whether they be trees, chains, etc.) of the structures determined by that module. So, for example, the chain module must prove that, for a given phrase structure input (i.e. the output of the PS module), we can derive a well-formed chain (or, in fact, set of chains).

This approach has a number of computational advantages: In the first place, there are relatively few axioms constraining any individual representation type. More importantly, however, these axioms are relative simply in nature. Recalling our discussion in the previous section, we remarked that, a complete formalisation of standard TG purely in terms of phrase structure would force us to significantly complicate the nature of the axioms given in (148). In particular we noted that it would be necessary to introduce some form of tree-transducing operations to axiomatise Move-$\alpha$[14]. However, in the context of our reformulation of the principles and parameters model in terms multiple representation types, this is not the case. As we stressed in chapter 3, the principles of grammar for each representation type are defined strictly with respect to local units of structure, and are not sensitive to global structures. That is, by factoring the traditional 'annotated surface structure' into its constituent informational structures, we observed that it was possible to state the axioms constraining these representations on local units of structure, and without the need to use axioms which transduce the structures (i.e. rewrite one structure with another) as illustrated by the passive example (147).

To summarise, we are no longer considering a single licensing grammar as discussed earlier, but rather we have defined our syntactic theory as a four-tuple of 'sub-grammars' each determining the well-formedness of a particular representation type. Having rejected the use of program transformations as inappropriate (recall, the discussion above), we will characterise each module as a meta-interpreter, whose task is as follows:

(151)
1. play the logical role of structure generators, proposing instances of uninstantiated structure for the particular representation.
2. sustain only those structures which are well-formed formulae with respect to the relevant 'necessary' axioms/constraints.
3. determine the control strategies and preferences where multiple structures are

[14] As indeed is this case in those systems developed by both [Johnson 89] and [Stabler 89b].

licensed, in accordance with the performance theory.

The first point above highlights the point that the axioms of our licensing grammar do not generate structures, but simply determine whether a proposed (uninstantiated or partially instantiated) structure is indeed licensed (i.e. the second point, above). The final point notes that the meta-interpreters are also the loci for implementing the various behavioural aspects of the modules outlined in the theory. This includes the incremental construction of representations, as well as realising the various strategies (AA, DSA, and ATS). In chapter 6 we proceed to a more detailed discussion of the meta-interpreters themselves, but first let us turn our attention to how the syntactic processor as a whole coordinates the operation of the individual processors.

## 5.3 Control in the Syntactic Processor

If we take this specification of the grammar to be the 'competence component', then the 'performance component' can be stated as a parse relation which maps the input string to a well-formed 'State'. The State is the four-tuple of representations used by the grammar: phrase structure (PS), thematic structure (TS), Chain structure (ChS), and coindexation (CiS), abbreviated as follows:

(152) State = { PS,TS,ChS,CiS }

The highest level of the parser specifies how each module may communicate with the others. Specifically, the PS processor acts as input to the other processors which construct their representations based on the PS representations and their own 'representation specific' knowledge. In a weaker model, it may be possible for processors to inspect the current State (i.e. the other representations) but crucially, no processor ever actually 'constructs' another processor's representation. The communication between modules is made explicit by the Prolog specification shown below:

[153]

```
synpro(Input,Rem,{PS,TS,CS}) :-
ps_module(PS,Input,Rem),
chain_module(PS,CS),
theta_module(PS,CS,TS).
```

The `synpro` predicate defines the parse relation according the organisation of the processors as shown in Figure 4.2. The algorithm specified in Prolog above, appears to suffer from the traditional depth-first, left-to-right computation strategy used by Prolog. That is, `synpro` seems to imply the sequential execution of each processor. As Stabler has illustrated, however, it is possible to alter the computation rule used [Stabler 89a], so as to permit incremental interpretation by each module: effectively coroutining the various modules.

In the present implementation we employ the 'freeze' directive. This allows processing of a predicate to be delayed temporarily until a particular argument variable is (partially) instantiated. In accord with the informational dependencies shown in Figure 4.2, each module is frozen (or 'waits') on representations as follows[15]:

(154)

| *Input dependencies* | |
|---|---|
| `ts_module` | PS |
| `chs_module` | PS |
| `cis_module` | PS |
| `ps_module` | LexInput |

Using this feature we may effectively 'coroutine' the four modules, by freezing the PS processor on Input, and freezing the remaining processors on PS. The result is that each representation is constructed incrementally, at each stage of processing. As shown in Figure 4.2, `sentence_processor` relates a lexical string (the InputString), to its corresponding thematic structure. The `increment` predicate is simply used to simulate incremental processing by instantiating 'LexInput' on a word-by-word basis, to allow inspection of 'State' at intermediate stages of processing.

[155]

```
sentence_processor(InputString,ThetaStructure) :-
State = {PS,ThetaStructure,ChS},
start(PS),
synpro(Input,['.'],State),
increment(Input,InputString,State),
well_formed(State).

increment(['.'],['.'],State) :-
pretty_print(State).
increment([Word|R],[Word|R1],State) :-
pretty_print(State),
increment(R,R1,State).
```

When Prolog first tries to execute `synpro` it will in turn call `ps_module`. Since `Input` is uninstantiated, however, `ps_module` will be frozen. As a result of this, the PS representation remains uninstantiated, and hence both `chain_module` and `theta_module` are similarly frozen. Once the first word of `Input` is instantiated, by the `increment` predicate, `ps_module` will be unblocked, and will begin parsing. As the `ps_module` begins instantiating the PS representation, the two subsequent modules will also be come unblocked, and will begin processing the partially instantiated PS tree (thus producing partially instantiated outputs of their own). After the first word is parsed, the `ps_module` will try to parse the next word,

[15] We list coreference structure (CiS) here for completeness, even though we do not include it in the implementation.

but since the rest of `Input` is uninstantiated, the `ps_module` will be frozen again. Similarly, once the other interpreters have processed the instantiated portion of `PS`, they too will be frozen. As the next word is instantiated by `increment`, the whole process begins again.

To illustrate this, consider once again the partial input string *What did John put ...*. The result of the call `synpro([what,did,john,put|_],['.'],State)` will yield the following instantiation of State (with the representations simplified for readability):

```
(156)  State = {  PS = cp/[np/what,
                           c1/[c/did,
                               ip/[np/John
                                   i1/[i/trace-1,
                                       vp/[v/put, _]]]]],

                  TS = [rel:put,grid:[agent:john | _]],

                  ChS = [ [what, _], [did,trace-1] ],

                  CiS = _   }
```

The PS representation has been constructed as much as possible, including the trace of the moved head of Infl. The ChS represents a partial chain for *what* and the entire chain for *did*, which moved from its deep structure position to the head of CP. TS contains a partial $\theta$-grid for the relation *put*, in which the Agent role as been saturated.

This is reminiscent of Johnson's approach, but differs crucially in a number of respects. First, we posit several processors which logically exploit the grammar, and it is these processors which are coroutined, not the principles of grammar themselves[16]. The result is that each interpreter is responsible for recovering only one, homogeneous representation, with respect to one input representation. This makes reasoning about the computational behaviour of individual processors much easier. Additionally, this 'meta-level' parsing as deduction approach permits more finely tuned control of the parser as a whole, and allows us to specify distinct parsing strategies for each module which are relevant to the particular representation.

The individual processors act by consulting those axioms of the grammar which relate to the processor's representations. Conceptually speaking, each processor has its own, specialised 'view' of the grammar. Logically, each processor is self-contained, and strives to relate the input representation it 'looks at' to those it constructs, such that the latter are well-formed. At each stage of processing, the individual processors increment their representations if and only if, for the current input, the new unit of structure to be added is provable from the grammar. It is this deductive interface to the competence component which permits

[16] That is, Johnson's system defines a axiomatisation of the grammar, and then uses a single processor (the Prolog inference engine) to give the axioms a procedural interpretation. In the present system, we use a number of inference engines, specialised for each representation. It is important to note that these specialised meta-interpreters have no inherent grammatical knowledge, i.e. there is no $\overline{X}$-processor, for example. Rather, there is a phrase-structure processor which proves possible structures based on the axioms of $\overline{X}$ theory, move-$\alpha$, and other relevant principles.

preferences to be specified. That is, the individual theorem provers for each processor can be defined so as to prefer the deduction of particular constructions. In the next chapter, we discuss the implementation of the individual modules in greater detail.

The most important point we have demonstrated here is how the logical organisation of modules can be explicitly stated in the specification of the parser in [153] above. The concurrent operation of the modules is then achieved by the quite independent specification of an appropriate control mechanism based on coroutining.

## 5.4 Summary and Discussion

In this chapter we have argued for the adoption of the *parsing as deduction* paradigm in constructing a computational model of the performance theory. At the highest level, this approach ensures a clear distinction between the competence and performance theories, such that the relationship between the two can be transparently identified. This also removes to a large extent the danger of 'procedurally' implementing the declarative principles of grammar. In this way, once the grammar is specified we have the freedom to experiment with variations in the control strategies and interpreters.

Perhaps one of the most interesting observations we have made is the fundamental difference between rule-based and principle-based grammars with regards to deductive parsing. Rule-based grammars may be more accurately termed 'construction-based' grammars, in that an individual rule is *sufficient* to license a particular unit of structure. Principle-based grammars, however, fall into a category we have termed 'licensing-grammars' (following [Speas 90]). In the case of licensing grammars, the principle are typically *necessary* conditions on particular structural configurations, and indeed numerous conditions may apply to a particular unit of structure. Since the individual principles are not *sufficient* to license some structure, such an axiomatisation also entails the existence of axioms which (over) generate the space of possible structure, such as binary-branching trees, for example.

Given this characterisation of principle-based grammars we argued for the use of meta-interpreters in each processing module. The interpreters play the role of the structure generating axioms, and also provide the loci for implementing particular control strategies. Furthermore, this approach is more appropriate than one which employs program transformations, since we can directly realise the modular organisation of the performance theory.

# Chapter 6

# The Specification of Modules

In this chapter we examine more closely the implementation of the meta-interpreters which determine the operation of the respective modules. At the end of the last chapter, we demonstrated both the organisation and operation of the syntactic processor, with only a minimal discussion of the constituent module's processors — stressing above all that their individual incremental behaviour is imperative to the concurrent and incremental operation of the sentence processor as whole. In §5.2.3, however, we did make some more theoretical remarks concerning the advantages of decomposing syntactic analysis into simpler representational types. In addition, we defined the role of the respective meta-interpreters (repeated from (151) above), stating that they should,

(157)
1. play the logical role of structure generators, proposing instances of uninstantiated structure for the particular representation.
2. sustain only those structures which are well-formed formulae with respect to the relevant 'necessary' axioms/constraints.
3. determine the control strategies and preferences where multiple structures are licensed, in accordance with the performance theory.

Perhaps the most striking point made in §5.2.3 concerned the fundamental difference in the axiomatisation of licensing grammars as compared with traditional construction-based grammars: While the latter use a set of axioms which are able to directly generate wff's, the former simply provide axioms which, given some formula, can determine whether or not it is well-formed. In addition, there may exist any number of necessary conditions which must be satisfied by any candidate formula (or structure). This approach further assumes the existence of some fundamental axioms which define the space of 'potential wff's'. From a procedural perspective, we can imagine these axioms generating arbitrary structures which are then accepted or rejected by the set of necessary conditions, or constraints. This task of generating structure is precisely what the meta-interpreters do: They define the space of possible representations for their respective processors (the first point in (151) above).

Standing back, let us consider the various components of a module. From an external perspective, they simply take some input representation and derive a corresponding output representation. We can, however, articulate the internal structure of modules as follows:

(158) **Input visibility:** Determines which aspects of the input are relevant to the module.

**Interpreter:** Builds representations with respect to visible input, and verifies well-formedness according to its 'view' of the grammar (see next point).

**View relation:** Specifies which axioms of the grammar are relevant to the particular representation type.

**Grammar:** The axiomatisation of the grammar as necessary conditions defined on local units of structure.

The notion of 'visible input' varies from module to module. As we shall see, in the case of the PS module, all lexical input is visible (i.e. there are no words which are ignored). For the ChS module, on the other hand, we are only concerned with traces and moved constituents — i.e. those element in PS which can and must participate in ChS. Each interpreter, as we have discussed above, is responsible for generating representations — building them up from recursive instances of schematic, structural units. As each unit of structure is postulated, the interpreter consults the view relation, to determine whether or not the structure is locally well-formed with respect to the relevant principles of grammar. Note, there may be multiple ways in which the view relation can license and instantiate a particular structure, and it is the view relation's responsibility to decide which is preferred. This allows us to specify as 'selection rules' which grammatical axioms are to be considered first, contributing to the implementation of the range of strategies for dealing with structural ambiguities.

In the following sections we present the meta-interpreters[1] for each of the modules, and also discuss the axiomatisations of the grammatical principles the interpreters make use of. Throughout, we will make our discussion of the interpreters more concrete through reference to concrete examples of output from the system. Since it is not particularly fruitful to show just the representation produced by an individual processor, we will show the complete state. While the display of the various representational structures does try to closely match those developed in §3.3, it is perhaps worthwhile to consider a simple example.

In Figure 6.1, we show the final state of representations for *Who will the girl kiss?* The phrase structure representation is reasonably standard, with the node in the tree being abbreviated using traditional $\overline{X}$ conventions. In addition, traces are denoted by the symbol `t-IDnumber`, where the IDnumber is the category of the trace followed by a unique integer (e.g. `t-n147`). Chain structure consists of the set of chains, where each chain is represented as a list of C-Nodes separated by the link operator (here, we use `#`, rather than $\infty$). Each C-Node is represented as <`IDnumber;Level/Pos;Type`>, where IDnumber is the unique identifying symbol for the constituent (as above). Level denotes if the constituent is a `head` or `phrase`, and Pos indicates the position occupied — either `abar` for an $\overline{A}$-position, `apos` for an A-position, or `home` for a base generated (D-structure) position (note, for heads

[1] We use the terms 'interpreter' and 'meta-interpreter' almost interchangeably. From a purely logical perspective we are simply building interpreters, but since we are implementing them 'on top of' Prolog's existing interpreter, they might be more appropriately considered as meta-interpreters.

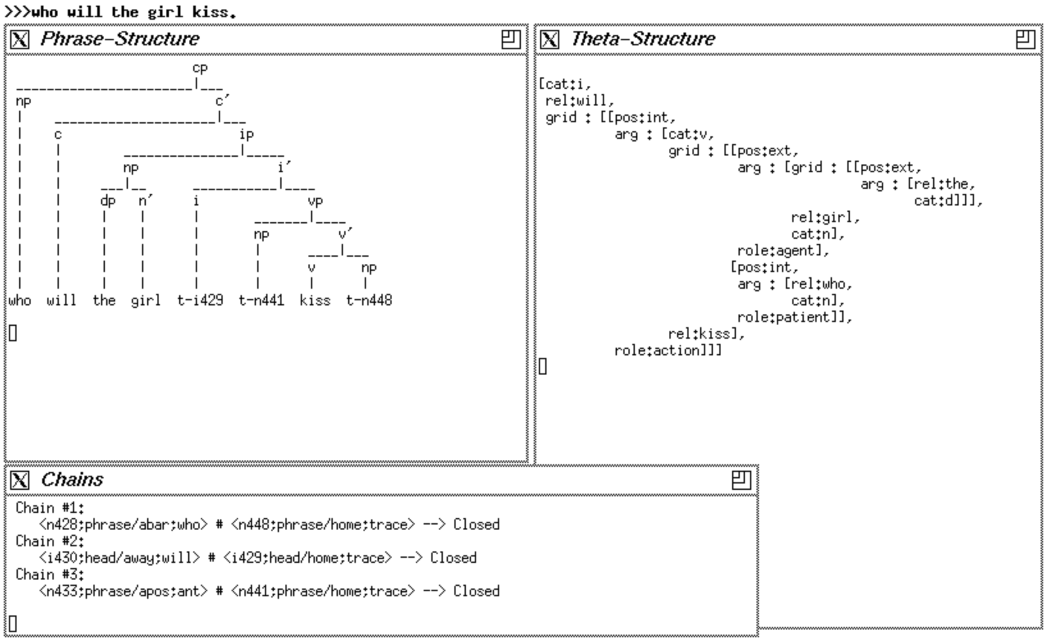



Figure 6.1: An Example Parse of *Who will the girl kiss?*

which are not `home`, we simply indicate their position as `away`). Finally, the last field, Type, simply expresses whether the C-Node is a lexical antecedent (i.e. expressed as `ant` or the word itself, in the case of a simple constituent) or a `trace`.

The representation employed for thematic structure is a feature structure. The structure corresponds directly with that given in (87), although the reader should notice that features occur in no particular order. In Figure 6.1, for example, the relation `rel:kiss` appears below its `grid`. This is not significant, but simply an artefact of when information is encountered, and the way in which unification of feature structures takes place.

## 6.1 The Phrase Structure Module

The phrase structure (PS) module is responsible for constructing a constituent structure representation from the input string of lexical items. As we remarked above, the entire lexical input string is visible, so no special visibility relation is required. The structure of the PS module can therefore be depicted as in Figure 6.2.

### 6.1.1 Representations and Grammatical Knowledge

The role of the interpreter, is to build a phrase structure tree for the input string, and ensure that this tree is licensed by the well-formedness conditions of the grammar. Before proceeding to a discussion of the interpreter, however, let us briefly consider the view relation which determines access to relevant grammatical axioms. Below, we list some examples of

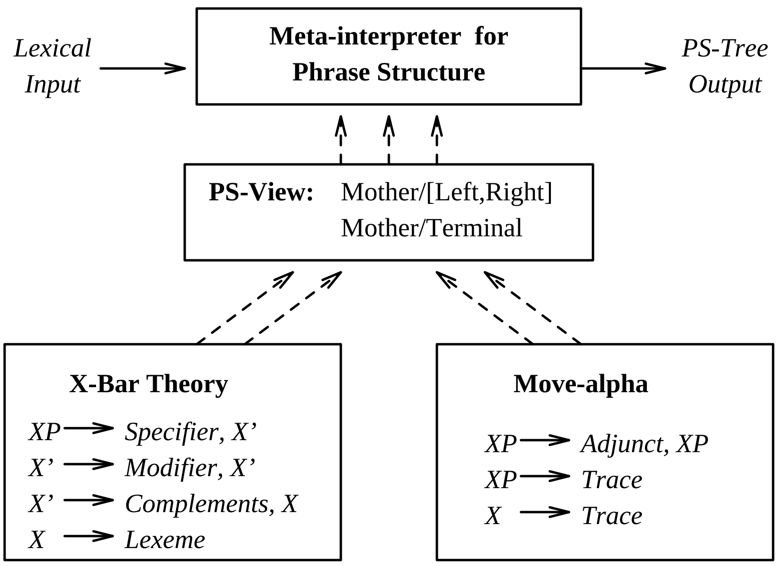



Figure 6.2: The Phrase Structure Module

the phrase structure axioms (see appendix E):

[159]

```
ps_view(Node/t:{C,lex:W,ID,Fs}) :–
xbar( Node/t:{C,lex:W,ID,Fs} ).
ps_view(Node/[Left,Right]) :–
xbar( Node/[Left,Right] ).
ps_view(Node/[Left,Right]) :–
move_a( Node/[Left,Right] ).
```

These axioms state the conditions (the right-hand side) which must hold for any instances of the structure given on the left-hand side. Note that the two `xbar` clauses are quite similar to the those in the toy grammar we gave in the last chapter (148). The first clause, for example, states that a leftward satellite may occupy the specifier position of a phrase, only if that phrase (i) is specifier initial (i.e. the specifier occurs as a left-branch), and (ii) the specifier is of the appropriate category and features (e.g. the specifier of CP can be a NP or PP with wh-features). The `move_a` clause specifies that the specifier of non-lexical phrase (C and I) may be landing sites for those phrasal constituents which are permitted to be moved (typically NP and PP for English).

The `ps_view` predicate defines the phrase structure 'view' of the grammar, and returns those possible instantiations of the PS schema (of (81) above) which are determined to be well-formed by the relevant principles of grammar as described above. The following is an exemplary subset of the clauses which define `ps_view`:

[160]

```
nt:{Cat,[m(+),s(−),c(−)],ID,FM}/
[nt:{S,[m(+),s(−),c(−)],IDS,FS},nt:{Cat,[m(−),s(+),c(−)],ID,FM}] ) :−
spec_pos(Cat,initial), spec(S,Cat,FS).

nt:{Cat,Lev,ID,FM}/t:{Cat,lex:Phon,ID,FM} ) :−
lexicon(Cat,Phon,FD).

nt:{C1,[m(+),s(−),c(−)],ID,FM}/
[nt:{C,[m(+),s(−),c(−)],IDS,FS},nt:{C1,[m(−),s(+),c(−)],ID,FM}] ) :−
non_lexical(C1), spec(C,C1,FS), moveable(C,phrase).
```

These clauses consider each of the basic local structural configurations: binary branches and terminal branches. The ordering of the clauses allows us to specify preferences in applying a particular grammatical axiom. The final two clauses, for example, are both concerned with licensing binary branches, but `ps_view` will first try to instantiate the structure using the `xbar` axioms, and only if this fails will it try `move_a`. In this way we encode the preference for postulating base-generated structures (a requirement of the D-structure Attachment principle). Similarly, we could express a preference for those `xbar` axioms which license A-positions (as required by A-Attachment)[2].

In general, `ps_view` will return any possible branch structure (in the appropriate order of preference), but is usually constrained by partial instantiation of the query. We might for example, call `ps_view` with the Mother node instantiated, in an attempt to derive a branch 'top-down'. If, for example, we are trying to parse an IP, `ps_view` would instantiate the branch IP/[??,??] as IP/[NP,Ibar].

### 6.1.2 The Phrase Structure Interpreter

Before we proceed to a discussion of the actual PS interpreter, let us first consider the following trivial specification of an interpreter for deriving phrase structures:

[161]

```
ps_interpreter([Word|L0]−L0,NTnode/{C,Word,ID,Ftrs}) :−
ps_view(NTnode/{C,Word,ID,Ftrs}).
ps_interpreter(L−L0,NTnode/[Left/LD,Right/RD]) :−
ps_view(NTnode/[Left,Right]),
ps_interpreter(L−L1,Left/LD),
ps_interpreter(L1−L0,Right/RD).
```

The `ps_interpreter` predicate recursively derives a phrase structure representation for the lexical input (represented as a difference list), such that it is well-formed with respect to

[2] This is not sufficient to completely realise these strategies, since they also depend on the interpreter trying to find all potential AA/DSA attachment sites. We take this up in more detail at the end of this section.

both the input (i.e. the terminal yield of the PS tree is identical to the input string) and the principles of grammar relevant to the PS representation. The two clauses correspond to the two types of possible branching structures (recall the phrase structure schema given in (81), chapter 3): The first derives a unary-branching schema where the leaf is a terminal node. The result above is a top-down, left-to-right parser which tries to postulate binary branching nodes recursively, unless a unary leaf node can be derived which absorbs the current lexical item.

The parsing strategy specified above does meet our requirement of incrementality: That is, it begins building the tree top-down, structuring lexical items into the tree as they are encountered in the string. The result is that, if frozen on the input string, this parser will construct a partial representation of the tree for the instantiated portion of the input (we will show how this may be specified below). This simple interpreter is, however, deficient is several respects: In the first place, it is well known that such a strategy cannot deal with left-recursive grammars (i.e. it may recurse infinitely). While we have argued for some top-down processing, notably for functional categories, we have largely assumed that parsing should be data-driven (or bottom-up) for the remaining lexical constituents. Finally, the above interpreter will only posit leaves which absorb a lexical item, and is therefore not able to parse traces, or other empty categories. To address these points, consider the following, slightly more articulated interpreter[3]:

[162]

```
ps(Tree,L,L0) :-
freeze(L, ps_int(Tree,L,L0)).

ps_int(Node/[Left/LD,Right/RD]) -->
{ non_lexical(Node),
  ps_view(Node/[Left,Right]) },
ps(Left/LD),
ps(Right/RD).

ps_int(Tree) -->
ps_ec_eval(Tree).

ps_int(Tree) -->
ps_lex_eval(Tree).
```

The first clause above simply specifies that `ps_int` will not be executed unless the input string L, is at least partially instantiated. Thus given a partially instantiated input, such as the list `[who, did, john,|_]`, `ps_int` will only attempt to construct a partial tree covering those three words. Once the uninstantiated portion of the list is reached, `ps_int` will be 'blocked' (i.e. won't do anything) until that part of the list is subsequently instantiated. The second clause, is not unlike the second clause in [161] above, except we have added a number

[3] In presenting example code from the present implementation, we will occasionally simplify less relevant aspects of it. The complete, unadulterated source code is, however, available in the relevant appendix.

of 'preconditions' for the application of this clause[4] (enclosed in the { }'s). In addition to the call to `ps_view`, to determine if the proposed branch in well-formed (and possibly instantiate it further), we require that the mother node be `non_lexical` (i.e. functional). In this way we restrict the application of this top-down rule to functional structures, as proposed in our theory.

The third clause above, invokes the routine for parsing empty categories (`ps_ec_eval`), while the final clause calls the interpreter for parsing lexical items into the (sub-)Tree bottom-up (`ps_lex_eval`). We won't go into details here about the bottom-up parsing component, since it is essentially a standard left-corner parser (see [Pereira & Shieber 87] for discussion), which, on the basis of the current lexical item, will 'project' a phrase structure tree, until it unifies with the Tree argument of the `ps_lex_eval` call. It is important to note that the ordering of the clauses specifies the priorities for building the various kinds of structures: Ranked highest is the prediction of functional structure (CP and IP), then the postulation of traces (implementing part of the ATS), and finally the attachment of lexical items.

The interpreter as just described will parse traces in the traditional manner, positing them only once their corresponding position in the input string is reached (although, of course, they are not present in the input string). As we argued in §4.6.3, however, the ATS demands that traces be postulated at the earliest possible moment — i.e. as soon as their structural attachment site is available. While a complete interpretation of this strategy has not been implemented[5], we can postulate the following interpreter predicate, which handles the head-final traces which arise in German:

[163]

```
ps_int(Node/[Left/LD,Right/RD],X,X0) :-
   (non_lexical(Node);cat(Node,v)),
   branch_right(Node/[Left,Right]),
   ps_view(Node/[Left,Right]),
ps_ec_eval(Right/RD,X0,X0) ->
ps(Left/LD,X,X0).
```

This clause only considers right-branching (i.e. where the daughter projection is right of the satellite) structures for the non-lexical categories C and I, and also V (since these are the only categories which may have empty/moved heads). If these conditions are met, then the

[4] For this clause we have employed a DCG notation, which simply hides the difference lists. Thus the normal Prolog clause `ps(Tree/[Left,Right],L,L0) :- ps(Left,L,L1), ps(Right,L1,L0).` can be abbreviated as `ps(Tree/[Left,Right]) --> ps(Left), ps(Right).` To simplify the PS interpreter rules, we will use the DCG notation wherever possible, using the standard Prolog only when explicit access to the difference lists is required. Note also, that goals enclosed within { }'s are not considered DCG terms, but are direct Prolog calls. See [Clocksin & Mellish 81] for detailed discussion of this.

[5] In fact, experimentation with the parser, revealed that fully implementing the ATS itself is not particularly difficult. It does, however, lead to a great deal of local overgeneration which is currently not ruled out by the subset of principles we have incorporated, but would be, given a more complete specification of the grammatical constraints. As an example, the Case Adjacency Principle would significantly reduce the space of possible complement configurations.

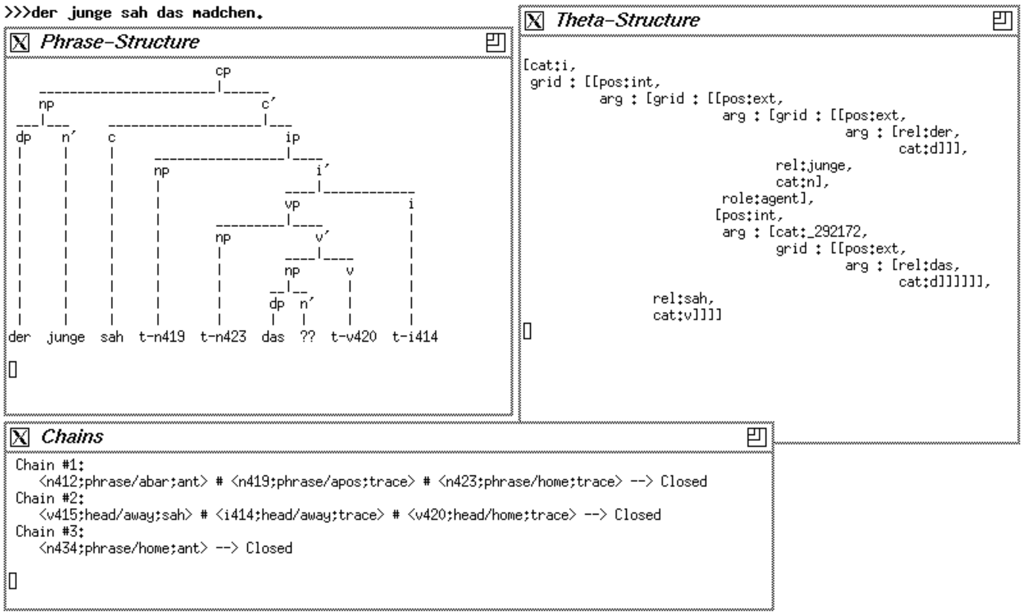



Figure 6.3: Example of *Der Junge sah das ...*

clause attempts to first parse the right daughter as a trace, thus increasing and potentially closing a chain, before it proceeds with parsing the left daughter. To illustrate the operation of the parser, consider the partial parser of *Der Junge sah das Maedchen*. We can see in Figure 6.1.2, that by the time *das* is encountered (in fact, immediately after *sah* is parsed), we have already completed the chain for the raised verb *sah*, as indicated by `Chain #2`. This in turn allows us to build the thematic grid projected by the raised verb as soon as it it encountered, making maximal use of the available information. This contrasts with the traditional view, which would force a delay in the use of the verb's $\theta$-grid until the end of the sentence (i.e. the position of the traces) was reached permitting the chain to be recovered.

### 6.1.3 Discussion

The phrase structure processor is essentially a data-driven parser, using traditional left-corner (LR) parser, but is augmented with a top-down component to deal with the prediction of functional structure and also to predict traces at the earliest possible point (as just illustrated above). The result is a realisation of the incrementality requirement, since the interpreter will yield a 'connected'[6] partial parse tree for any partial input string. We also incorporate a preference for postulating traces over lexical items, including the top-down prediction of empty structure which occurs down rightward branches (as just illustrated above). This implements a significant portion of the ATS, although some further work is required to completely generalise this strategy. It is important to note, that while the PS

[6] This contrasts with other parsing algorithms, such as shift-reduce and chart parsers, which may construct a number of subtrees for the partial input, but leave them as unconnected elements of a chart or stack. Again, the reader is referred to [Pereira & Shieber 87] for discussion of various parsing techniques.

interpreter will try to postulate traces wherever possible, most will be ruled out immediately by the Chain module (see below), if they cannot participate in a well-formed chain.

The current implementation of the model has not completely incorporated the AA and DSA strategies. Indeed, as we outlined at the beginning of chapter 5, it was not our intention to do so. We did, however, remark that part of the realisation of these strategies lies simply in the expression of preferences in the `ps_view` relation. In addition, however, the interpreter would have to consider all possible attachment sites in the tree. That is, the `ps_view` preferences will only prefer a particular instantiation of a particular branch, so, given an NP followed by a PP in Dutch, the `ps_view` relation will try to attach the PP as an argument of the NP but since this is not possible (usually), it will then try attaching it as modifier, and probably succeed (unless the NP is, say, a pronoun). A complete implementation of AA and DSA, however, would require looking at all potential attachment sites rather than simply accepting whatever structure `ps_view` proposes at the first site encountered. Thus in this example it might be possible to attach the PP as an argument of upcoming VP, which is precisely the behaviour we argued for in §4.4.3. To find this site, the interpreter would have to search the partial tree for all potential attachment sites looking for an A-position, and if unsuccessful do the same looking for a D-structure position, etcetera.

## 6.2 The Chain Module

While the PS module recovers a phrase structure representation for an input string, the chain structure (ChS) module recovers the chains associated with an input phrase structure. The structure of the Chain Module is shown in Figure 6.4, and is virtually identical to that of the PS Module (in Figure 6.2). However rather than recovering branches of phrase structure, it recovers links of chain, determining their well-formedness with respect to the relevant grammatical axioms.

Unlike the PS module, however, the ChS module is not concerned with its entire input representation. Rather, the ChS module only considers those constituents of phrase structure which are relevant to the formation of chains. In particular, we stated in §3.3.3 that all 'moved' constituents must head exactly one well-formed chain, and further, all traces must be a member of exactly one well-formed chain.

### 6.2.1 Representations and Grammatical Knowledge

As we did for our discussion of the phrase structure processor, above, let us first consider the specification of the grammatical conditions on chain well-formedness, and the view relation. The `chain_view` predicate is responsible for determining the well-formedness of links in a chain, just as `ps_view` was concerned with branches in a tree. In particular, there are three fundamental configurations in a chain; the head, the tail, and the linking of two C-Nodes in the body of the chain. Each of these possibilities is dealt with by the three clauses for `chain_view` given below:

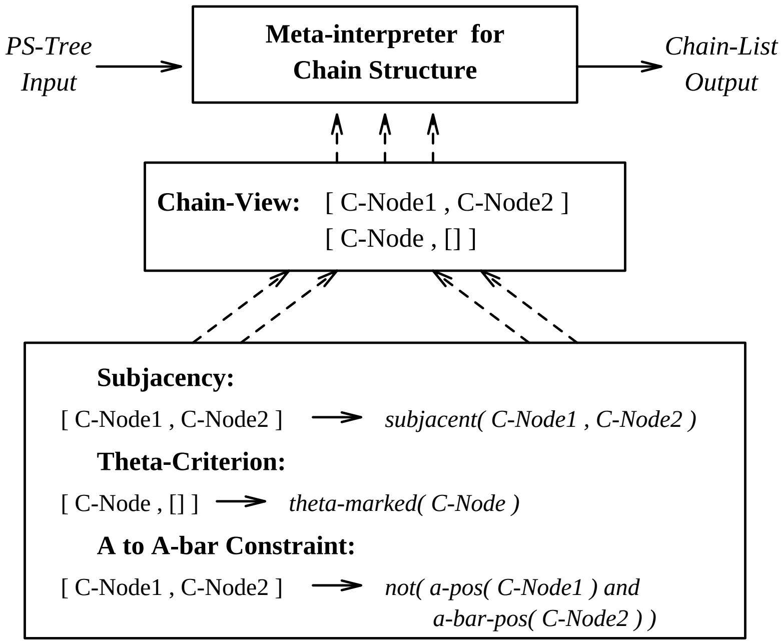



Figure 6.4: The Chain Module

[164]

```
chain_view(Cnode # []) :- !,
theta_criterion(Cnode # []).

chain_view(Cnode # head_of_chain) :-
case_filter(Cnode # head_of_chain).

chain_view(Cnode1 # Cnode2) :-
cat(Cnode1,Cat), cat(Cnode2,Cat),
case_filter(Cnode1 # Cnode2).
```

The first clause considers the tail of a chain (the `C-Node # []` link), and invokes (a part of) the $\theta$-criterion (see below). The second and third clauses apply to the head of the chain, and intermediate links, respectively. Note, that for the current implementation the only conditions on chains are the Case Filter, and the relevant part of the $\theta$-criterion. There is, however, no reason we couldn't add further constraints to the grammar, and call them from the appropriate `chain_view` predicates. The specification of some current link constraints are as follows:

[165]

```
theta_criterion(c:{Cat,_,_/home,_,_} # []).

case_filter(c:{n,_,phrase/POS,_,Ftr} # head_of_chain) :-
case_markable(POS),!,
wait_exist(Ftr,case:yes).

case_filter(c:{n,_,phrase/abar,_,F1} # c:{n,_,phrase/POS,_,F2}) :-
case_markable(POS),!,
wait_exist(F2,case:yes).

case_filter(c:{n,_,phrase/apos,_,F1} # c:{n,_,phrase/POS,_,F2}) :-
case_markable(POS),!,
wait_exist(F2,case:no).
```

What we have called the `theta_criterion` simply ensures that any `home` position in a chain is necessarily the tail of that chain. The home position of a constituent is its D-structure, or base-generated, position. In the case of arguments, this will be their $\theta$-position, and thus this constraint ensures that an argument chain includes exactly one $\theta$-position, namely the tail. Similarly, this constraint applies to moved heads, and ensures that a head does not occupy multiple $\theta$-assigning positions. The three `case_filter` predicates realise the formulation of the Case Filter we proposed in (85 (in chapter 3). The first of these states that if the head of the chain occupies a potentially Case-markable position (i.e. an A-position) then it must be Case marked (85ii). The second states that the highest A-position in a chain (the A-position which immediately follows an $\overline{A}$-position) must be Case marked (85i). Finally, the last clause requires that no *lower* A-position receive Case, since a Chain may only receive Case once — at the highest A-position.

The implementation of other Chain conditions, most notably Subjacency, can be defined purely as a constraint on C-Node links. This does, however, entail that the PS tree representation be augmented with an indexing procedure. This procedure, developed in [Latecki 91], assigns sets of indices to each node in the tree which permits various command relations (of which subjacency can be defined as a special case) to be determined as a simple operation over the sets for any pair of nodes in the tree. This mechanism has been implemented in the GB-parser developed by [Millies 90].

### 6.2.2 The Chain Interpreter

In the case of building the PS interpreter, we were able to consider a variety of parsing algorithms. Our principal constraint was to ensure that the PS interpreter constructed the PS tree incrementally as input was received. Since trees are two-dimensional structure (i.e. they encode both dominance and precedence information), there are a number of algorithms which met this demand. We began with a simple top-down, left-to-right approach, and refined this into a parser which uses a mixture of top-down and left-corner (bottom-up) strategies. In the case of chains, however, we are dealing only with one dimensional structures: a sequence of nodes related to each other by links. Thus, we are only presented with one option when building the chain structure: We must append successive C-Nodes to existing chains (or, indeed, initiate a new chain) as those C-Nodes are encountered.

As we remarked above, the chain module is only concerned with particular elements in the PS input representation. At the highest level, the chain interpreter traverses the input tree, identifying and processing precisely those elements[7]:

[166]

```
chain_int(NTnode/[Sat/SD,Right/RD],CS) :-
branch_right(NTnode/[_,Right]),
visible(NTnode/[Sat,Right],LP),!,
create_cnode(Sat/SD,LP,Cnode),
chain_member(Cnode,CS),
chain(Right/RD,CS).
chain_int(NTnode/[Left/LD,Right/RD],CS) :- !,
chain(Left/LD,CS),
chain(Right/RD,CS).
chain_int(NTnode/Tnode,CS) :-
visible(NTnode/Tnode,LP), !,
create_cnode(NTnode/Tnode,LP,Cnode),
chain_member(Cnode,CS).
chain_int(NTnode/Tnode,CS).
```

The first clause identifies if the current satellite (Sat) phrase is `visible`. Visibility is defined as any constituent which immediately dominates a trace, since traces must be members of chains, and any lexical constituent which occupies a non-base-generated position (such as [Spec,IP], and [Spec,CP]). Furthermore all NP's are visible, even if they only form unit chains (i.e. occur in their D-structure position), since the Case Filter is verified with respect to chains (following [Chomsky 86b]). If a constituent is visible, the C-Node is created for it, and we require the C-Node to be a `chain_member`. The third clause performs exactly the same function for constituents which do not occur as satellites — namely, moved heads and their traces. Finally, the second and fourth clauses handle those instances where the current branch is not visible.

The `chain_member` predicate is responsible for ensuring that a C-Node is a member of a chain. As we remarked above, there are two cases: If the C-Node denotes a lexical antecedent, then a new chain must be created for it. If the C-Node represents a trace, then that trace must be a member of an existing chain:

[7] The predicates shown here have been simplified to some extent. In particular, a number of `freeze` directives have been removed to improve readability. The use of freeze is primarily to ensure that relevant variables are indeed instantiated before any operations are carried out with respect to them. Thus the removal of `freeze` does not affect the declarative reading of the clauses.

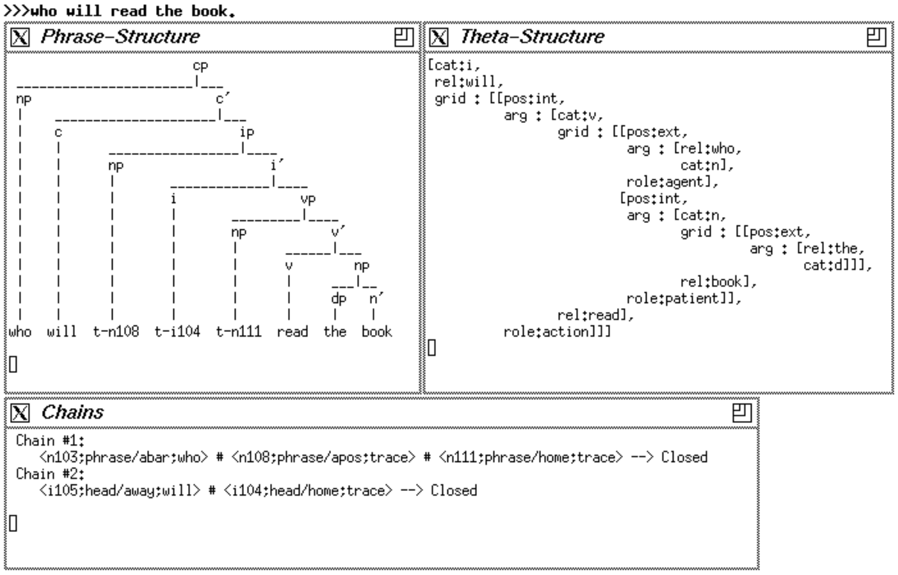



Figure 6.5: Example of *Who will read the book?*

[167]

```
chain_member(Cnode,CS) :-
antecedent(Cnode),!,
initiate_chain(Cnode,CS),
chain_view(Cnode # head_of_chain).
chain_member(Cnode,CS) :-
trace(Cnode),!,
find_chain(Cnode,CS).
```

The first clause simply identifies an antecedent, initiates a chain for it, and ensures that the `chain_view` relation is satisfied, enforcing those grammatical constraints which apply to the heads of chains. The second identifies a trace, and then proceeds to search the set of existing chains, until the C-Node can be successfully appended to one of them. This appending operation again consults the `chain_view` relation to ensure that the link created by the C-Node and the current tail of the chain being appended to is well-formed.

To illustrate these various operations of the ChS module and the various chain conditions, consider `Chain # 1` for the subject in Figure 6.5. When *who* is encountered, it occurs in a non-canonical position, so we initiate a chain for it. While it is a `head_of_chain`, it is not in a `case_markable` position, so (first clause of) `case_filter` fails, and no other constraints apply. We then posit a trace in the [Spec,IP] position which is appended to the chain. Since the resulting link connects an `abar` C-Node with an `apos` C-Node (which is a `case_markable` position), the second clause of `case_filter` applies and ensures that this trace has indeed received Case, which it has since [Spec,IP] of a tensed clause is Case marked. Subsequently, we posit a trace in the D-structure subject position, [Spec,VP] thus creating a C-Node in a `home` position. When this is appended to chain, linking with the previous trace, we can see that the third clause of `case_filter` will be triggered. This occurs, since the previous trace is an `apos`, and the current `home` trace (where `home` positions are a those A-positions which must be occupied at D-structure) is a potentially `case_markable` position. The purpose of this clause is to ensure that the the current position is, indeed, not assigned Case, since this would mean the chain had received Case multiply. The constraint succeeds, since we assume that V Case marks to the right, and therefore does not mark its subject. Finally, we can attempt to close the chain, by appending the null `[]` C-Node, which is licensed by our so-called `theta_criterion` constraint.

### 6.2.3 Discussion

We remarked above that there is little room for variation in constructing a chain interpreter which recovers chains incrementally. For leftward moved constituents (the only movement really being considered here), we must simply create a chain for the antecedent, and then append successive traces, recovering the history of movement on the base-generated position is reached and the chain is closed.

Interestingly, if we assume that the incremental recovery of a chain begins with the antecedent, followed by the the appending of each link, we are presented with even more support for our active trace strategy. If we recall the example structure shown in Figure 6.1.2, we can see that the linear order of the links in the `Chain # 2` do not correspond with their linear position in the string. That is, in the string, the base-generated trace for the verb (`t-v420`) *precedes* the higher trace in $I^0$ (`t-i414`). In the chain, however, we can see that the traces appear in their correct order. This result is only possible because the PS interpreter parses the trace in $I^0$ before attempting to parse the VP. The $I^0$ trace, is therefore incorporated into the chain headed by *sah* before the final trace in $V^0$ is posited. Given the traditional interpretation that traces only be postulated once the appropriate string position is reached, we would first encounter the trace in $V^0$. We would then be unable to append this C-Node to a chain until after the $I^0$ trace was parsed and incorporated into the chain. Thus, we would both violate the incremental construction of chains, and, as argued in the first place, be unable to use the moved heads thematic information to assist in the parsing of pre-verbal complements.

As a final point here, we should mention that there is some scope for ambiguity in the construction of chain structure: Given a trace, there may be more than one chain to which the trace may be grammatically appended. This issue has been considered by Fodor [Fodor 78], who argues for a 'nesting preference', where the trace is preferentially structured into the most recently postulated chain. Consider the following example:

(168) a. ? "Which pot$_i$ is this rice$_j$ easy to cook $\varepsilon_j$ in $\varepsilon_i$"
b. ?? "Which rice$_i$ is this pot$_j$ easy to cook $\varepsilon_i$ in $\varepsilon_j$"

While both utterances are somewhat awkward, Fodor argues that there is a strong preference for (168a) (given the intended reading). To account for this we could very simply employ a strategy which, when $\varepsilon_j$ is postulated (in (168a)), would incorporate the trace into the most recent chain, i.e. the chain for *rice*$_j$.[8]

## 6.3 The Thematic Module

The thematic module plays a somewhat distinguished role in the syntactic processor. Not only does it construct the output to subsequent semantic and pragmatic systems, but it is also responsible for coordinating the three other syntactic representations; phrase structure, chains, and coreference (of which we are only concerned with the first two). In addition, thematic structure (TS) is not subject to a battery of well-formedness axioms in the same way that the other representations are. Rather, in the construction of TS, we need simply ensure that the $\theta$-criterion (as stated in (88) in chapter 3) is obeyed. To this end, the thematic module does not contain a view relation, *per se*, but rather the thematic assignment relation (discussed) below, embodies the requirements of the $\theta$-criterion. The mechanism for assigning $\theta$-roles also makes use of the Canonical Structural Realisation (CSR) properties of the particular language (recall discussion in §3.2.1).

While the thematic module takes the PS representation as its 'primary' input, it also makes crucial use of chain information. In particular, the thematic module must associate dislocated constituents with their thematic, D-structure positions. Interestingly, however, the thematic module is only concerned with the head and tail of a chain, and not any of the intermediate positions. So in addition, to traversing the PS tree as it is constructed, the TS module also incrementally examines chains as they are instantiated, and constructs its own 'simplified' chain structure:

[169]

```
theta_module(Tree,CS,TS) :-
simplify_chains(CS,SCS),
theta(Tree,SCS,TS).

theta(M/D,SCS,TS) :-
freeze(D, theta_int(M/D,SCS,TS) ).
```

So, just as the `theta_int` is frozen on phrase structure, the `simplify_chains` predicate is frozen on chain structure, constructing the simplified ChS (i.e. SCS) as chains are instantiated. The simplified chain structure consists of terms of the following kind:

(170) `Head # Tail @ TS`: for moved maximal projections (phrases).

`Head # Tail`: for moved minimal projections (heads).

[8] Note, there might in fact be grammatical reason to rule out (168b). Our only point here is to note that it would be straightforward to incorporate Fodor's proposals should we wish to. Whether or not the 'nesting preference' can be sustained cross-linguistically also remains an issue for empirical inquiry (see [Engdahl 82], [Christensen 82]).

Simply, Head is the initial C-Node of a chain, and Tail is the final, home C-Node — the thematically relevant position in the chain. In addition, for phrases we must transmit the theta structure (TS) for the Head so that it may be instantiated into the correct location in the overall thematic structure, as determined by Tail. This is not necessary for moved heads, since all the necessary information is present in the Head C-Node (i.e. the relation itself, and its $\theta$-grid), so we can essentially just copy the features of the Head to the Tail. We will see below, the details of how the simplified chain structure is used by the thematic interpreter.

### 6.3.1 The Thematic Interpreter

The highest level of the thematic interpreter (theta_int) simply traverses the PS tree and, for each node in the tree, associates a thematic structure with it (via the @ operator). The thematic structure associated with a node in the tree denotes the entire thematic structure for the sub-tree dominated by the node. The result is that the thematic structure for the mother in a particular branch can be determined 'compositionally' from the thematic structures of its daughters:

[171]

```
theta_int(Mom/[Dtr/D,Sat/S],CS,TM) :–
branch_left(Mom/[Dtr,Sat]),
freeze(S, theta_int1(Mom@TM/[Dtr@TD,Sat@TS],CS) ),
theta(Dtr/D,CS,TD),
theta(Sat/S,CS,TS).
theta_int(nt:M/t:D,CS,TS) :–
theta_int1(nt:M@TS/t:D,CS).
```

At this point, theta_int calls a secondary level of the interpreter, called theta_int1, which is responsible for dealing with the simplified chain structures discussed above. Shown below are the clauses for dealing with phrasal chains:

[172]

```
theta_int1(Mother@TM/[Dtr@TD,Sat@TS],CS) :–
exists(H#T@Theta,CS),
nonvar(H), equiv(H,Sat),!,
Theta <=> TS,
TM <=> TD.

theta_int1(Mother@TM/[Dtr@TD,Sat@TS],CS) :–
exists(H#T@Theta,CS),
nonvar(T), equiv(T,Sat),!,
TS <=> Theta,
theta_visible(Mother@TM/[Dtr@TD,Sat@TS]).
```

The first clause checks to see if the current satellite is the antecedent (or, head) of an existing (simplified) chain. If so, the theta structure for the simplified chain (`Theta`) is unified with that of the satellite (`TS`), and since the satellites thematic structure is not associated with the current branch, the thematic structure of the mother node (`TM`) is simply unified with that of the daughter (`TD`). The second clause deals with the case where the current satellite corresponds with the tail position of an existing chain. If so, the satellites thematic structure (`TS`), which will be uninstantiated since it is a trace, will be unified with the thematic structure of the chains head (`Theta`) which will have been instantiated previously by the clause just described above. Since the current branch is the D-structure position of the satellite, it must be visible to the thematic structure. To determine how the satellite thematic structure is to be incorporated into the global structure, we call the `theta_visible` relation.

[173]

```
theta_visible( nt:{Cat,Lev,ID,FM}@H/t:{Cat,lex:Phon,ID,F} ) :-
(lexical(Cat) ; Cat = i),
get_roles(F,Roles),!,
H:rel <=> Phon,
H:cat <=> Cat,
H:grid <=> Grid,
theta_assign(Grid,Roles).

theta_visible(nt:{C,[m(+),s(-),c(-)],ID,FM}@TM /
    [nt:{C,[m(-),s(+),c(-)],ID,F}@TD, nt:{SC,[m(+),s(-),c(-)],IDC,FC}@TS] ) :-
lexical(C),!,
TM <=> TD,
Arg:pos <=> ext,
Arg:arg <=> TS,
TD:grid <== Arg.
```

The role of the `theta_visible` predicate is to determine how the thematic structures of two sister sub-trees are to be combined to yield the thematic structure of their mother. Before discussing the first clause, let us consider the second. This clause identifies those branches where the satellite is a specifier (sister to a node with the `s(+)` feature) of a phrase whose category is lexical (such as V, or N). The body of this clause first unifies the theta structure of the mother with that of the daughter. It then constructs an $\Theta$-Relation (recall our definition of thematic structure in (87)) denoted by the variable `Arg`. Since the constituent is a specifier, we set the `pos` feature as external, and then instantiate the `arg` feature with the thematic structure of the satellite (`TS`). Finally, we insert `Arg` into the $\Theta$-Grid for the daughter (which has already been unified with the mother).

The first clause above deals with thematically relevant heads, namely everything except head of C. This clause first determines the set of `Roles` licensed/required by the head, and then creates a $\Theta$-Node for the head, instantiating the relation (`rel`) and category (`cat`) appropriately. Finally, we access the `grid` for the $\Theta$-Node (which is instantiated by the clause just described in the previous paragraph), and assign the `Roles` of the head to each argument, or Θ-Relation in the `Grid`. As we will see shortly, the `Grid` may or may not be partially instantiated by the time the head is encountered. Thus the assignment of thematic roles is frozen on the instantiation of the Θ-Grid.

We remarked earlier that the θ-Criterion (88) , repeated below, is embodied in the definition of thematic assignment.

(174) **θ-Criterion:** For every Θ-Node,
(i) all θ-roles of the relation must be associated with exactly one Θ-Relation in the Θ-Grid.
(ii) all Θ-Relations in the Θ-Grid must receive exactly one θ-role.

The task of the `theta_assign` relation, then, is to ensure that each argument in a particular relation's Θ-Grid, receives exactly one role from the grid associated with the relation's lexical entry. Furthermore, we must make certain that all obligatory roles are assigned:

[175]

```
theta_assign([],[]) :- !.
theta_assign([],Grid) :- member(Role,Grid), \+ Role = {_}, !, fail.
theta_assign([Arg|Args],Grid) :-
Arg:arg:cat <=> C,!,
remove(R/T,Grid,Grid1),
csr(R/T,Arg),
theta_assign(Args,Grid1) )).
```

The first clause defines the trivial case, where the list of arguments (i.e. Θ-Grid) is empty, as is the set of roles required by the head. The second clause states that, if the argument list is empty, there must be no obligatory roles still unassigned (note, we identify optional roles by enclosing them in {,}'s). The third clause, considers the first argument in the Θ-Grid, and selects/removes a role from the set associated with the relation (in `Grid`). The role/position (`R/T`, where T indicates if the roles is assigned internally or externally) is then matched with the argument Θ-Relation in accordance with the CSR relation. In particular, `csr` ensures that the constituent receiving a role is of the appropriate category and relation (i.e. some roles can only be assigned to PP's with a locative head, etc.). We give several example `csr` clauses below:

[176]

```
csr(agent/ext, Arg) :-
Arg:arg:cat <=> n,
Arg:pos <=> ext,
Arg:role <=> agent.

csr(agent/int, Arg) :-
Arg:arg:cat <=> p,
Arg:arg:rel <=> by,
Arg:pos <=> int,
Arg:role <=> agent.

csr(proposition/T, Arg) :-
Arg:arg:cat <=> i,
Arg:pos <=> T,
Arg:role <=> proposition.

csr(inst/T, Arg) :-
Arg:arg:cat <=> p,
Arg:arg:rel <=> mit,
Arg:pos <=> T,
Arg:role <=> inst.
```

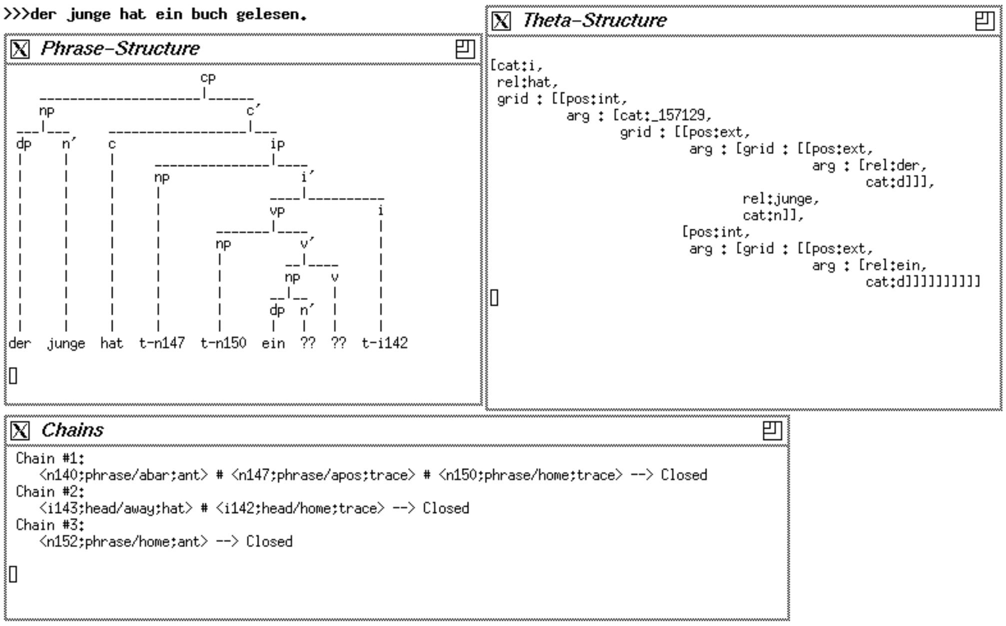
The first three are taken from the English lexicon (since the CSR may vary slightly from language to language). In particular, the first two show that the AGENT role is realised externally as an NP, but internally (as in the passive example) as *by*-PP. The final clause indicates how German INSTRUMENTAL roles might be realised as *mit*-PP.

### 6.3.2 Discussion

While there might be a variety of ways in which the thematic interpreter could recover a thematic structure, this is not a particularly interesting issue. The PIC demands that, given the partial states of phrase structure and chain structure (and, ultimately, coreference structure), we must make maximal use of the available information to construct a maximal partial thematic structure as input is received. To this end, we simply perform a top-down, left-to-right traversal of the partial PS tree, as it is instantiated. Simultaneously, we construct the simplified chain structure representation, which is also accessed during the recovery of thematic structure.

It is interesting to note that some degree of thematic interpretation can be accomplished even before the head/relation is encountered. This contrasts with so-called head driven parsers of the sort advocated by Pritchett (recall discussion in §2.3.3). This is of considerable interest in the context of head-final language such as Japanese and, to some degree, German. Consider the partial state for the sentence *Der Junge hat ein Buch gelesen.* in Figure 6.3.2. Before the verb is reached, we can create a thematic structure for the VP, and add both the subject NP as an external argument and the partial object NP (*ein ...*) as an internal

**State for *Der Junge hat ein* ...**




**State for *Der Junge hat ein Buch gelesen.***

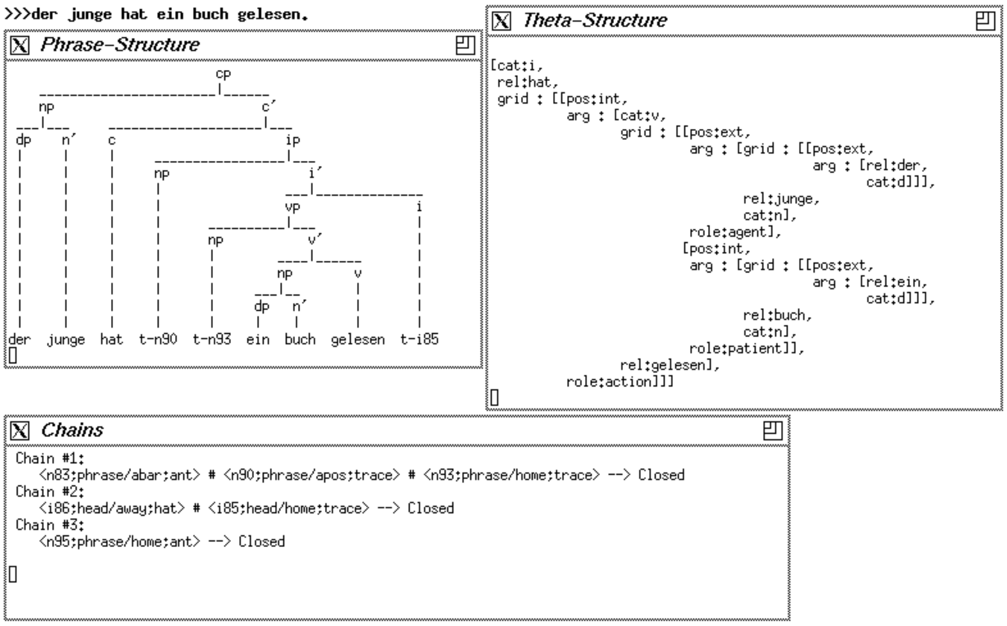



Figure 6.6: Example of *Der Junge hat ein Buch gelesen.*

argument. While we cannot yet assign roles to these $\Theta$-Relations, we can assume that they will be co-participants in the same $\Theta$-Node. When the verb is reached, the roles will be assigned. Crucially, if the $\theta$-grid for the verb does *not* license the two arguments, backtracking will be required. In this way we predict the increase in processing complexity analogous to that which Frazier obtains for Dutch in (107)[9].

## 6.4 Summary

In this chapter we have outlined how the various interpreters may be implemented. For the most part, the PIC heavily constrains the way in which we can construct representations, especially for simpler types of structures such as chains (and presumably coreference relations). For the PS module there is some scope for variation, and there are doubtlessly a number of other parsing strategies which would suffice. We are required by the theory, however, to have some degree of top-down prediction for functional constituents and also for the prediction of traces in accordance with the ATS. The combination of LL (top-down) and LR (bottom-up) strategies used here do lead to the incremental construction of a connected partial parse tree and make maximal use of available information, both grammatical and lexical, at each stage of processing the input.

As we have previously remarked, it is not our intention here to provide a thorough implementation of the competence and/or performance models. Rather, our aim was to show how interpreters could be built which constructed their representations incrementally, and crucially, relied on the other modules operating concurrently and incrementally. In addition, we have shown and discussed how the various strategies can be implemented within the interpreters (as in the case of the ATS) and the view relations (as for part of AA and DSA).

[9] Note, that while this example contains an NP object, rather than a PP object as in Frazier's examples, the present system would behave similarly for PP objects.

# Chapter 7

# Summary and Discussion

This thesis has undertaken the development of a model of human linguistic performance which integrates three fundamental assumptions:

(177) **Principle-based:** The HSPM uses the principles of grammar directly, and processing strategies are defined with respect to the grammar.

**Incrementality:** There is a global demand for maximal, incremental comprehension which must be met by the sentence processor.

**Modularity:** Partitioning of informational/representational domains into modules facilitates speed of processing via distributed, concurrent operation. This is a paradigm for process organisation to be invoked whenever possible.

We have noted that there is some tension among these assumptions: processing principles must be grammar-based while also maximising incremental comprehension; we must reconcile the notion of modularity in an incremental system; and finally we must consider how the 'modularity paradigm' impacts upon the syntactic processes, if at all.

As we discussed in chapter 1, there are a number of deeper issues which arise in the context of this research programme. In particular the basis of the model upon the UG, and indeed the innateness hypothesis in general, raise questions concerning the universality of the sentence processing machinery: Is it developed from scratch, deriving from general cognitive mechanisms? Is it constructed from a general set of language processing 'routines' which are selected, perhaps, on the basis of the parameter settings for the particular languages being acquired? Or is the HSPM invariant, capable of successfully parsing, presumably with similar efficiency, all acquirable languages? In sum, what is the relationship between competence and performance in the HSPM, and how did it originate?

In addition, some rather more subtle issues have presented themselves in the course of this thesis, as a result of a integrating modern theories of competence, mental organisation, and computation in a comprehensive theory of performance. We have, however, presented the material in rather isolated chunks. Chapter 3 sketched a theory of competence which emphasises the *types* of information which must be recovered during syntactic analysis. The motivation for this was to move away from the mechanisms used in traditional transformational grammar, which have led to some confusion and conflation in previous attempts to construct strongly competent models of performance. The result of this revised model is an equivalent system which removes emphasis from the particular mechanisms used to characterise principle-based grammars (most notably, the use of Move-$\alpha$ and levels of representation). Rather, the model we proposed highlights the underlying informational content embodied in these mechanisms. We therefore propose that any theory of performance which can outline how such information is recovered, is directly consistent with the principles and parameters competence model in the only relevant sense.

In chapter 4 we developed such a theory of performance. The model makes crucial use of the informational domains present within syntactic theory to explain a wide range of empirical processing phenomena. The model is carved up into modules, precisely on the lines dictated by our revised syntactic model, containing distinct processors for the recovery of constituent structure, chains, coreference, and thematic structure. The process model further incorporates several grammar-based processing strategies which have been demonstrated to hold for English and Dutch, and which also seem consistent with preliminary evidence from German and Japanese. In addition, we argue that these syntactic strategies may be derived from our over-arching *Principle of Incremental Comprehension*.

Finally, we have constructed an explicit computational model which demonstrates that the fundamental organisation of syntactic processor and operation of the modules have a natural computational realisation. The model further emphasises the direct use of grammatical principles. These points are made more perspicuous through the use of a deductive approach to implementation which distinguishes a declarative specification of the model's organisation and the axioms of the grammar, from the control mechanisms which cause processing to occur in precisely the manner demanded by the proposed theory of performance.

Having presented the competence, performance, and computational aspects individually, we will now take the opportunity for a more general examination and discussion of the proposals made here. In addition to the issue of innateness and universality, we will address the more subtle point of 'psychological reality': How are the principles of grammar and their representations actually realised in the HSPM? The aim of this discussion is not so much to argue for a particular proposal, but rather to highlight the spectrum of possibilities concerning psychological reality, and their potential impact on modern theories of competence and performance.

## 7.1 A Summary of the Theory

As a basis for discussion in this chapter, let us briefly summarise the theory of performance we have constructed, and consolidate the proposals we have made. With the support of a wide range of empirical evidence, we made the fundamental claim that the operation of the Language Comprehension System (LCS) must satisfy the Principle of Incremental Comprehension (95), repeated below:

(178) **Principle of Incremental Comprehension** (PIC): The sentence processor operates in such a way as to maximise comprehension of the sentence at each stage of processing.

We assumed that the PIC characterises an external demand placed upon the LCS by General Cognitive System (GCS), and further noted that any subsystems of the LCS (e.g. the syntactic processor) must also obey the PIC. Thus, the PIC is *not* a syntactic processing principle, but is rather an external condition which must be met during the course of the sentence processors operation. This demand entails that the syntactic processor make maximal use of the information available to it — both syntactic and lexical — at each stage of processing (i.e. as each lexical item is encountered in the input). That is, structure is projected bottom-up, as lexical input is received, and where structure can be predicted top-down, by the syntax, it is.

### 7.1.1 The Modular Syntactic Processor

In addition to the PIC we have also assumed that modularity, in roughly the Fodorian sense [Fodor 83], is a paradigm for the organisation of mental processes which is to be exploited maximally. That is, to the extent that we can identify representationally and informationally encapsulated domains, their modularisation permits distributed operation: Modules can operate concurrently, constructing their outputs as input is received[1]. Thus while Fodor argues only for a modular 'language faculty' similar to what we have called the LCS (where, admittedly, the theoretical and informational boundaries are somewhat unclear), we assume distinct phonological/lexical and syntactic modules within the LCS (we also assume a 'semantic/pragmatic' (henceforth, S/P module), with some pragmatic, real-world knowledge, but little hinges on this). This sort of organisation has been widely assumed in the psycholinguistic literature — e.g. by Frazier and her colleagues [Frazier 87b] — but it is important to note that it assumes a more pervasive notion of modularity than Fodor proposes.

The task before us in chapter 4 was therefore to construct a model of the syntactic processor which would account for interaction with non-syntactic systems and also obey the PIC. Additionally, however, we assumed that, in the absence of evidence to the contrary, the syntactic processor must use the principles of grammar directly. On the basis of the revised model of grammar developed in §3.3, combined with the paradigm of modularity argued for above, we suggested that the syntactic processor could be naturally decomposed further into a set of modules precisely determined by the four representation/information types identified in the model of grammar:

[1] Note, this contrasts with the sort of serial modularity refuted by Crain and Steedman, where they assumed that modules operated in a sequential manner (not concurrently) and that information could only flow in units of structure such as clauses or phrases [Crain & Steedman 85]. We would similarly argue against such a version of modularity, since the serial assumption results in no performance benefit, and the units of structure they stipulate simply slow things down, inhibit incrementality and in no way contribute to the modular status.

(179)

| Modules | |
|---|---|
| (a) | Phrase structure (PS) |
| (b) | Chains (ChS) |
| (c) | Thematic structure (TS) |
| (d) | Coindexation (CiS) |

As we remarked at the beginning of the chapter, a tension begins to emerge here: can we sustain the principled basis and modular status of the model and still meet the overarching requirement of maximal incremental comprehension. That is to say, the modular organisation predicts that only certain types of information are available during the initial construction of a syntactic analysis, thereby limiting the types of information which may be used 'top-down' to predict structure. It therefore falls upon us to (i) demonstrate that the modules can indeed operate so as to construct the maximal possible interpretation, (ii) outline the strategies used by the modules in the face of ambiguity, and (iii) to relate these strategies to both the grammar and the PIC.

In our discussion of the phrase structure module — the locus for the bulk of ambiguity — we argued for the use of two fundamental attachment strategies, repeated below:

(180) **A-Attachment** (AA): Attach incoming material, in accordance with $\overline{X}$ theory, so as to occupy (potential) A-positions.

(181) **DS-Attachment** (DSA): When an A-position is unavailable for attachment, prefer attachment of incoming material into its canonical, Deep Structure position.

In addition to providing a descriptive account of a broad range of data in English and Dutch (and to some degree German and Japanese, though on the basis of intuitive evidence), the strategies successfully met the three criteria required to sustain our set of assumptions:

(182)
1. AA and DSA are both defined with respect to notions made available by UG, and are thus principle-based and language universal.
2. The strategies are located within the PS module, and only make use of information available to that module, and are thereby consistent with modularity.
3. The strategies lead to the optimal reconstruction of thematic structure (the output to the semantic/pragmatic system) during strictly incremental processing, and thus satisfy the PIC.

In our discussion of the chain module and the data concerning the recovery of antecedent-trace relations, we noted that — while descriptively successful — the Active Filler Strategy proposed by Frazier was incompatible with the modular organisation we have proposed. In particular, it requires that the PS module access Chain information for purposes of postulating gaps. We observed, however, that a naive strategy of active trace postulation satisfies the restrictions of modularity:

(183) **Active Trace Strategy** (ATS): When constructing the phrase structure analysis, initially try to posit a trace in any potentially vacated position, i.e. as daughter of $X^{max}$ or $X^{min}$ where the category of X is visible for phrase or head movement, respectively.

First consideration of the ATS might lead us to reject it on the grounds that it appears to overgenerate wildly. The concurrent operation of modules, however, ensures that inappropriately posited traces will be immediately identified and removed, by the Chain module. This aspect of the model's operation was explicitly demonstrated by the implementation constructed in chapter 5. We argue that the cost of this operation is essentially equivalent to the AFS, where at each step the Chain information must be consulted to see if a gap should be postulated. Finally, the ATS successfully meets the three criteria of (182) above: the notion of trace is defined by the grammar, the ATS operates according to the encapsulation of the PS module, and the ATS will lead to the earliest resolution of antecedent-trace relations[2], thereby maximising incremental comprehension.

Recalling our discussion above, we stated that the PIC entailed the prediction of structure where possible. In particular, we assume that functional projections predict their complements top-down; C predicts IP, and I predicts VP. This property of the PS module's operation interacts crucially with AA and DSA to account for Frazier's evidence from Dutch and the evidence we presented for German in §4.4.3 for the sentences in (107)). In addition, this predictive aspect of the parser is instrumental in explaining the processing of traces with respect to the data of Pickering and Barry: I.e. when combined with the ATS, this leads to postulation of a trace as soon as the attachment site becomes available, rather than delaying until the PF position of the trace (if this is even a relevant notion) is encountered.

### 7.1.2 A Concentric Theory of Complexity

One of the most fundamental divergences the present theory makes from traditional models of sentence processing, is that we do not take computational or representational complexity to be fundamental determinates of the processor's operation. That is, the strategies we propose are not motivated by the desire to minimise time or space complexity, but are rather based on the desire to satisfy grammatical relations in such a way as to achieve maximal and incremental comprehension. Furthermore, the modularity hypothesis, as we have invoked it here, assumes that a primary advantage of modularisation is that individual processors are very fast. This predicts that the process of backtracking need not itself be a particularly costly operation. Broadly speaking, given the PIC, we assume that reanalysis is costly when the interpretation we have constructed requires revision in some significant sense (such as defined by Pritchett's Θ-Reanalysis Constraint (TRC)). Furthermore, the precise cost of such revision may vary according to the extent to which we have *committed* to the current interpretation:

[2] Note the ATS operates in conjunction with AA and DSA, to prefer postulation of traces in argument or base-generated positions.

(184) **Commitment Principle** (CP): Successful integration of lexical items into the current context results in commitment to that analysis. Implausible interpretations do not.

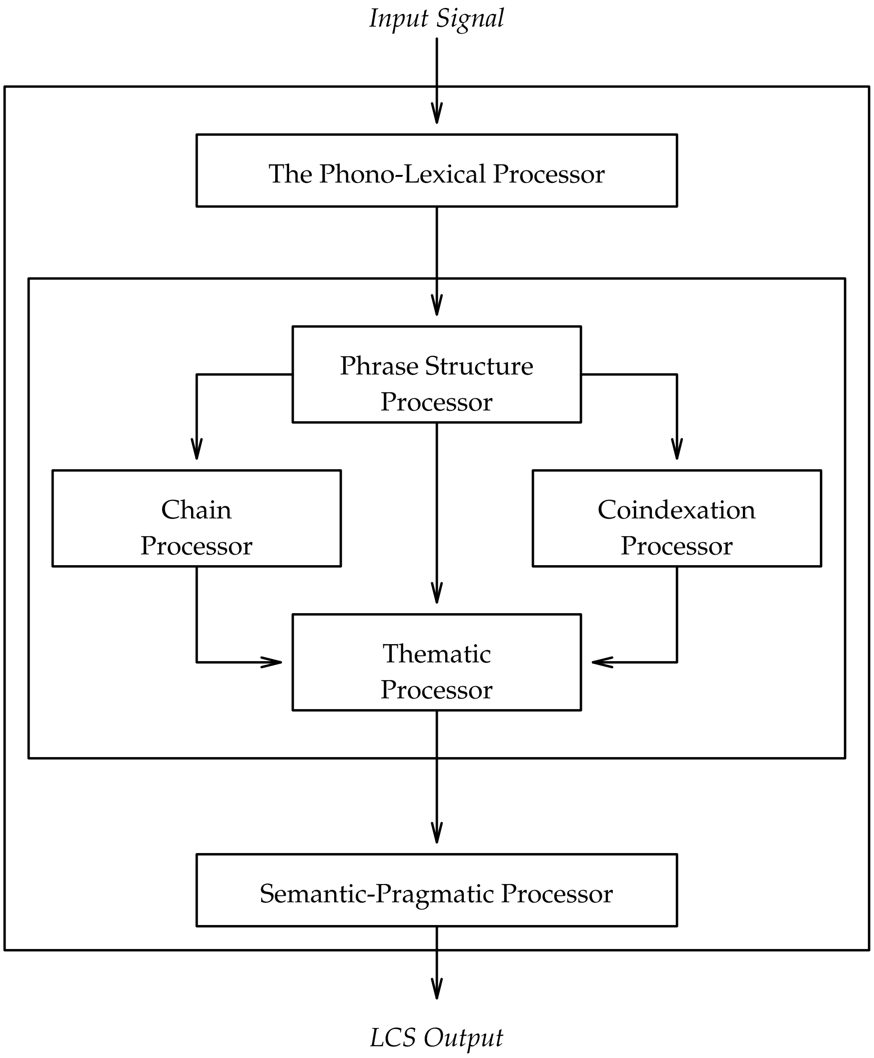



Figure 7.1: The Organisation of Modules

At the end of §4.2, we suggested that this principle might simply be a descriptive characterisation of some more fundamental property of the modular system. Now that we have developed a more detailed account of the LCS's organisation, let us reconsider this proposal. Figure 7.1 shows the the modular organisation we propose, including the basic paths of communication (and crucially, their direction), as well as the nesting of modules: There are four modules within the syntactic processor, three (approximately) within the language comprehension system[3], and the LCS is presumably just one of several distinct modules

[3] Assuming the four syntactic modules are seen as just one at this level.

within the general cognitive system.

We assume, as stated above, that backtracking into an individual module is relatively cost-free, in and of itself. If however, this backtracking leads to a 'significant' change in that module's revised output, then the reanalysis of the subsequent module, whose *input* has now changed, will be costly. Considering, for example, the phrase structure and thematic modules with respect to the following three sentences:

(185) a. "After the child sneezed the doctor prescribed a course of injections."
b. "I broke the window with my sister."
c. "While Mary was knitting the sock fell off her lap."

With respect to (185a) we suggested that the PS module initially tries to attach *the doctor* as a complement of *sneeze*. This construction is, however, immediately rejected by the thematic module, forcing the PS modules to backtrack and construct the correct analysis. Thus, no reanalysis is required by the thematic module, and the cost of backtracking into the PS processor predicts only a minimal increase in complexity, as is indeed observed [Mitchell 89b]. For sentences such as (185b), the PS processor will initially attach the *with*-PP as an instrument of the verb, and this is consistent with the thematic processor[4]. The semantic-pragmatic system (henceforth S/P module), however, will reject this analysis as implausible, forcing the PS module to attach the PP as a modifier. Backtracking into the PS is, as before, unproblematic. This time, however, the thematic module must reanalyse the new input thereby increasing complexity somewhat more. No reanalysis by the S/P module is required, since it rejects the initial analysis. This suggests greater complexity than the previous example (due to the reanalysis required by the thematic processor), but there is still no conscious garden path effect, since reanalysis has been 'contained' within the syntactic module.

Finally, consider (185c). As for (185a), the NP *the sock* is attached by the PS module as a complement of *knitting*. This time it is accepted by the thematic processor, and added into the grid for the relation *knit* (as THEME). This is in turn consistent with the S/P module, and presumably with the general cognitive system[5]. When *fell* is encountered, the PS module is forced to reanalyse (not for semantic reasons, but simply because there is no grammatical continuation), and attach *the sock* as the subject of *fell*. There is added cost again, when the thematic module must reanalyse its input. Furthermore, the change in thematic structure violates the TRC and causes the S/P module, and hence the GCS, to substantially revise the interpretations to which they had been committed. This propagation of reanalysis, I claim, is reflected as a conscious garden path effect. As we observed in §4.2, the severity of the revisions required by S/P module and the GCS may be reduced if the choice point where Θ-Reanalysis occurs was not committed to. This was stated as the Commitment Principle, repeated above in (184).

[4] Since we assume a purely syntactic notion of thematic structure, where pragmatic and semantics interpretation is a post-syntactic process.

[5] Some real world or contextual knowledge might exist to lead the GCS to reject the output of the LCS. However it is unclear what types of information are available in the S/P module, and what can only be evaluated by the GCS. For the purposes of discussion here, nothing hinges on this.

To summarise, the modular organisation provides a mechanism for distinguishing the relative complexity of various instances of reanalysis: if backtracking simply occurs into a simple (or *atomic*) module (i.e. one which does not contain submodules), then the cost is neglible as in (185a). If the revised result forces reanalysis by a subsequent module, as in (185b), then the cost is increased but since reanalysis has still been contained within the syntactic processor as a whole, no real garden path results. If, however, backtracking leads to substantial reanalysis by post syntactic processes, or even the GCS, then a garden path will result. We will refer to this account as the *Concentric Theory of Complexity*, where local reanalysis is relatively cost-free, increasing as it propagates to subsequent and 'outer' modules.

## 7.2 Computational Properties of the Model

The logical view of grammar takes a system of rules or principles as its axioms from which sentences of the language can be derived as theorems. More specifically, the axioms determine the set of well-formed formulae (such as the set of grammatical phrase structure trees), and it is the task of the deductive parser to find such a formula whose terminal yield matches the lexical input. In the case of construction-based grammars, such as CFG's, a particular axiom is sufficient to license a unit of structure, resulting in a direct rule-to-structure correspondence. Thus the axioms 'generate' the set of well-formed formulae directly. In the case of principle-based, licensing-grammars, however, the principles are stated as a set of necessary conditions or particular structures. In this case we also need the axioms which determine the space of formulae over which these necessary constraints are defined. For example, if the principles determine the well-formedness of local binary branches, as in our toy axiomatisation (148), then we also require those axioms which generate the space of binary branching trees. Our approach here has been to treat these structure generating axioms as the focal-point for determining the way in which analyses are constructed.

### 7.2.1 The Role of Meta-interpretation

In implementing the core aspects of the performance model, we chose to characterise each module as an independent deductive parser for its own particular representation type. For each module, we define a meta-interpreter in Prolog which realises the structure generating axioms for the representation the module is responsible for. That is, the PS interpreter strives to construct arbitrary phrase structures, but as units of structure are proposed it ensures that the necessary conditions of grammar are satisfied. We implement particular processing strategies by defining the interpreters such that they postulate structures in a particular way, and also by stating preferences for the particular instantiations of structures returned by the view relations (i.e. the interface to the grammar).

The most important behavioural characteristic of the modules is that they are able to recover their representations incrementally, as their input is received. Within the syntactic processor as a whole, the interpreters are coroutined to simulate the concurrent operation of modules assumed by the process model. To achieve incremental construction of the representations, it is necessary that each interpreter be able to add units of structure into the partial representation, as new input information is encountered. To make this possible, we have argued that the principles of grammar must be defined as strictly local conditions on units of structure (i.e. the representational schemas) as defined in §3.3. In this way we will never add a unit of structure which is ill-formed with respect to the grammar and the current partial input. It is, however, possible that there will be no grammatical continuation for subsequent input (thus entailing the interpreters to backtrack). Since we assume that the interpreters make direct use of the grammatical principles, this locality requirement suggests that there may be some performance oriented constraints on the syntax, i.e. grammatical principles must be defined with respect to local structures. Were this not the case, we would possibly be able to transform or compile the grammar into a set of local structural constraints, but this would significantly weaken the relationship between competence and performance.

The theory of processing we have proposed underdetermines a 'complete computational model'. That is, there are potentially numerous possible algorithms for the interpreters which would remain compatible with the theory. The theory does, however, heavily constrain what the processors might look like: They must be incremental, yielding a connected, partial output representation for a partial input. This determines quite specifically how the one-dimensional representations such as chains and coindexation relations may be constructed, and also places significant constraints on the phrase structure, and thematic processors. In particular we rule out the use of stack- and chart-based algorithms which may leave constituents unstructured *during* processing. Furthermore, we motivated the a degree of top-down prediction, necessary for attachment in head final languages and also for early trace postulation, thereby constraining the space of possible interpreters even further.

### 7.2.2 Decomposition and Parallelism

One important aspect of the computational model is that it stresses the potential performance benefits which can be accrued by invoking the modularity paradigm. Rather than using a single processor to recover a 'conglomerate' annotated phrase structure, we have decomposed the task of syntactic analysis into a number of constituent analyses, each representing one particular aspect of syntactic informational structure. Crucially, to reconcile the modular organisation with incremental operation we further assumed that the modules operated concurrently, building their individual representations simultaneously. At the next highest level, we similarly assume that the syntactic processor operates incrementally, and concurrently with the phono-lexical and semantic-pragmatic systems.

While the concurrent operation of modules was motivated by the PIC, it is important to note that this is really the only sensible operation in a modular system. That is, if modularity is to be justified as a paradigm for mental organisation it must have some benefits. Where modules operate strictly in sequence, as assumed by Crain and Steedman, the result would be a deterioration in performance compared with the single processor architecture [Crain & Steedman 85]. Assuming, however, modules operate concurrently, the distribution of processing tasks will result in improved performance over the single processor model (assuming the underlying computational architecture can support multiple processes). Given that the individual modules in a distributed system will each be simpler, and hence faster, than a single process which does the combined work, the distributed system should, in the worst case, be able to operate as fast as the slowest module — which will still be faster than the single processor implementation.

In the context of the present model, it seems uncontroversial that the PS module will operate faster than a parser which must build a conglomerate, annotated phrase structure: The PS representation is simpler, and fewer principles of grammar apply. As the PS representation is constructed the other modules are simultaneously building their own structures. Assuming, for example, the PS module is the slowest, i.e. the other modules are 'eagerly' waiting for PS input, the complete syntactic analysis can be accomplished in the time it takes the PS module to finish. Contrary to the claim that incrementality and modularity are at odds, as is assumed by Crain and Steedman among others, we argue that the two are inextricably linked by the common requirement of concurrency among modules.

### 7.2.3 Must Representations be Explicitly Constructed?

As a final point in our discussion of the computational model, let us turn our attention to a rather more philosophical issue. Throughout our exposition of the performance model, we have characterised the processing task as one of 'constructing representations'. While this may indeed be true, or at least a valid abstraction for purposes of theory formulation, it is not necessarily the case that a process explicitly constructs the representations used in characterising what that process does.

If we consider it from an external perspective, the syntactic processor relates some lexical input to an output thematic structure. The internal representations — PS, ChS, and CiS — are not 'visible' outwith the syntactic module. We might therefore ask whether or not it is essential to construct them at all. We know from related work, for example, that it is possible to parse 'with respect to' a syntactic analysis without actually constructing it. For example, DCG rules in Prolog permit the specification of CFG's. Given a sentence, Prolog will then determine if the sentence is grammatical, but won't recover a representation of the parse tree. Crucially, however, the proof that Prolog traverses to recognise the sentence corresponds directly to the parse tree (recall the example in (144)). If desired, we might augment the CFG with arguments to record the parse tree as the derivation is built (as in (146)). Alternatively, however, if the phrase structure tree is going to be mapped into some logical form (LF) representation, on a rule-to-rule basis[6], we might simply construct the LF representation directly. This is precisely the approach adopted in [Pulman 86] and [Crain & Steedman 85]. Thus the syntactic component of these systems parse *with respect to* a set of syntactic rules, but only construct the semantic (LF) representation.

While this technique is possible for rule-based grammars, assuming the rule-to-rule hypothesis, it is not immediately apparent whether or not it could be extended to principle-based parsers. Johnson has shown, however, that for his PAD parsers it is indeed possible to construct a parser based on GB theory which maps PF directly into an LF representation

[6] By rule-to-rule, we essentially mean that each rule of syntax structure corresponds directly with a rule defining the semantic (LF) interpretation of that structure.

without ever explicitly constructing the intermediate DS or SS representations (while still making use of principles which hold at those levels of representation) [Johnson 89]. It is important to note though, that Johnson accomplishes this through a series of transformations which essentially fold in the D-structure and S-structure levels, such that the relevant principles apply without the need to explicitly recover these levels of representation[7]. Crucially, this transformation process does alter the original axiomatisation of the theory, resulting in a 'logically equivalent' but organisationally different grammar.

In the present system it is not immediately clear whether or not the construction of the syntax internal representations could be eschewed. In the first place, we have prohibited the use of any transformational techniques, as used by Johnson. Secondly, a given unit of structure must satisfy (potentially) a number of principles before it is well-formed and can be interpreted by a subsequent system. Thus if it is not represented, it is unclear how individual principles could be applied to the same piece of structure. Interestingly, however, once a unit of structure has been constructed and is locally well-formed, and all subsequent processes have interpreted it, it is no longer required, and could 'fade'. Because the information is immediately transmitted to the partial output (i.e. thematic structure), and is necessarily (locally) well-formed (i.e. no further grammatical constraint will be applied), there is no need to retain the structure after the interpreters have 'moved on'. Thus it seems possible that the various interpreters need only ever represent the current unit of structure they are considering, and, after all subsequent modules have interpreted it, that unit of structure can be discarded. To do away with representations entirely, however, is difficult in a principle-based system, since the rule-to-rule property of traditional construction based syntax and semantics no longer holds.

These examples do highlight the fact that there is potentially some danger in making crucial reference to whether or not certain representations are *actually* constructed, without being able to define what this means in terms of the mental architecture. If there is some distinct process whose performance is formally characterisable in terms of a particular representation (regardless of whether or not it is actually constructed), and such that subsequent processes are formally defined as using that representation as input, then perhaps we may legitimately say this representation is real in some sense. This position seems the most fruitful in view of the current state of knowledge concerning the nature of mental representations.

It is interesting that Frazier's proposals do crucially rely on the notion that complete syntactic structures are represented, and crucially determines complexity on this basis [Frazier 87b]. Minimal Attachment (MA), for example, is motivated by the desire to minimise representational complexity by choosing the structure with the fewest numbers of nodes. While it might be possible to translate this strategy into a non-representation-based context[8], their fundamental basis on minimising syntactic representational complexity (to the point of actually weighting nodes in [Frazier 85]) would be undermined.

---

[7] The details of how Johnson achieves this are rather subtle, and detailed discussion would take us too far afield.

[8] For example, the parser could choose the shortest path in the searching through the phrase structure rules. Interestingly, however, searching all possible paths for the 'minimal' derivation, would involve more effort than simply choosing the first one.

## 7.3 The Innate Sentence Processor

At the beginning of this thesis, we considered the shift in emphasis which has directed modern linguistic inquiry. The broader goal of this research programme concerns the study of the I-language — the actual state of mind which constitutes our knowledge of language. This entails a deeper, more explanatory consideration of language than that sought by earlier, descriptive approaches which typically led to relatively superficial accounts of linguistic behaviour, i.e. the E-language. Chomsky further distinguished three basic areas of inquiry within the study of language, shown in (1), and repeated below:

(186) (i) What constitutes knowledge of language?
(ii) How is knowledge of language acquired?
(iii) How is knowledge of language put to use?

Despite the distinction of these domains, Chomsky tacitly observes that a particular theory of grammar (186i) must satisfy relevant 'interface' conditions. Most notably, Chomsky suggests that the language acquisition problem — when considered in the context of a poverty of stimuli — can only be accounted for by a theory of UG, wherein people are genetically endowed with some *a priori* knowledge of language; a set of language independent principles and parameters of variation which are settable on the basis of early linguistic experience.

This reasoning suggests that important results may also be gained from the examination of the interface between grammar and processor, as well as between processor and LAD. Concerning the latter, we suggested that initial 'bootstrapping' of the LAD can only be explained by assuming the availability of some initial processing mechanism to deal with early linguistic input. The primary focus of this thesis, however, concerns the former: what is the relationship between competence and performance. In particular we have developed a theory of processing, which makes direct use of the principles of UG and a set of language invariant processing principles, to account for a broad range of attachment, gap-filling, and garden-path phenomena in several languages. To the extent that the model holds across languages, it suggests that the entire HSPM may indeed be a constituent of the human genetic endowment — a Universal Parser (UP). Note, that this does not necessarily predict that the child has complete access to the same processing machinery as adults, since there is the possibility that the UP matures, similar to recent proposals concerning the principles of grammar [Borer & Wexler 87], [Borer & Wexler 88].

Clearly, for us to seriously defend this position, the empirical support for the theory must be extended to both a broader syntactic coverage and a greater spectrum of languages. While we have sketched an account of the processing of scrambled constituents in German, for example, there remains a lack of experimental support. Similarly, the Japanese data we discussed remains speculative at best, and only the most sensitive experimental studies will be useful in confirming the predictions made by the theory. Furthermore, the vast majority of experimental evidence which exists is for English, and where there does exist evidence for other languages, it is typically conducted for similar phenomena (e.g. Frazier's evidence from Dutch is concerned with the Dutch equivalent of PP attachment ambiguity in English). To sustain a universal model of parsing, we must also study the various interesting phenomena which arise exclusively in other languages, and demonstrate that these phenomena follow from the universal principles of the theory[9]. For example, it would be interesting to explore the interaction of scrambling (preposing) and extraposition in the processing of languages like German and Japanese. While these phenomena rarely occur simultaneously in English, a universal performance model must be able to account naturally for them in languages where they may.

For the moment, however, the hypothesis of an invariant sentence processor — innate and unparameterised — is the strongest, and remains consistent with the available data. In the remainder of this section we will discuss this hypothesis in greater detail, and compare it with contrary proposals made in [Frazier & Rayner 88] and [Mazuka & Lust 90], both of which assume that at least some parameterisation of the HSPM is needed.

### 7.3.1 Acquisition in the Deductive Sentence Processor

We have discussed how current syntactic theory has been influenced by the innateness hypothesis, leading to the postulation of UG; a system of innate principles which are simultaneously (i) rich enough to account for learning under a poverty of stimuli, and (ii) abstract enough to permit the acquisition of any (attainable) language.

In this thesis we have assumed the theory of UG in our efforts to develop a theory of human syntactic performance. We also suggested that, for acquisition to begin, there must be some sort of innate sentence processor capable of hypothesising syntactic analyses for early linguistic experience, on the basis of UG principles. That is, without some means to construct analyses, it is difficult to see how the LAD could make enough sense of the input to begin setting parameters. Furthermore, it is unclear how children would acquire the ability to construct such analyses.

In developing a computational model of the HSPM, we have recruited the PAD hypothesis of logic programming. This approach allows us to distinguish the declarative specification of the grammatical principles from the inference engine used to construct analyses (or, 'proofs'). Cast in these terms, which aspects of the computational model might we take to represent our innate endowment? Most obviously, the axiomatisation of the grammar must exist, since this is a prior claim of UG[10]. The question that face us, then , concern the nature of this initial processing mechanism, or 'theorem prover':

(187) 1. Are we simply equipped with some general inference engine which later develops into a more refined/specialised sentence processor?

[9] We might discover that additional strategic principles are required to explain phenomena in radically different languages. If this is the case, we must show that the strategies are both grammar-based and compatible with the PIC, and also that they hold universally. Clearly, however, it is undesirable to have numerous strategies, some of which are relevant to only particular languages.

[10] It should perhaps be made clear that we are using the PAD perspective purely metaphorically. As Johnson also points out [Johnson 89], we do not mean to say that people actually possess some logical representation of the principles of grammar. We are simply assuming that the knowledge is somehow represented in the innate system, either as a declarative 'facts' or possibly embedded within processing machinery.

2. Does 'maturation' of the processing mechanism occur on scheduled, predetermined lines, or is it language dependent?
3. Are we equipped with a rich sentence processing mechanism, which is language invariant?

The hypothesis we have suggested here is that the modular organisation, and the individual interpreters constitute part of the human genetic endowment.

### 7.3.2 Against Parameterisation of the HSPM

The search for a universal theory of sentence processing is a relatively recent initiative. Frazier has sought in recent years to generalise her proposals across a number of languages, most notably to Dutch and Japanese [Frazier & Rayner 88]. For the most part, Frazier and Rayner consider the sentence processor to be universal. They do however observe that postulation of the top-most S (the so-called 'partial top down constraint') must be relaxed in left-branching languages, since sentences such as the following are unproblematic (recall our discussion in §4.4.3) even thought the initial S node does not remain the top-most one:

(188) "[$_S$ [$_{\overline{S}}$ Bill ni Tom ga nanika o hanasita to ]$_i$ John wa $\varepsilon_i$ omotte-iru]"
*[ [ Bill-DAT Tom-NOM something-ACC spoke that] John-TOP thinking ]*
*John thinks that Tom said something to Bill*

In another recent article, Mazuka and Lust propose that parameterisation of the sentence processor is rather more significant [Mazuka & Lust 90]. In particular they argue that top-down parsing strategies are most appropriate for right-branching languages, such as English, while bottom-up approaches are better suited to left-branching languages, such as Japanese (recall §4.4.3 for a summary of their arguments).

Both of the above proposals struggle with the problem of parsing both left-branching and right-branching languages, and both account for this by parameterising the parsing in accordance with the relevant parameters in the grammar. In the present system the PS interpreter makes use of both top-down and bottom-up strategies, depending on the nature of the constituents being parsed. In general, the parser will always try to build structure top-down, using grammatical knowledge, when such information contibutes to the construction of maximal structure. Similary, the parser also makes maximal use of lexical information, projecting structure bottom-up, as the words of an utterance are encountered. The result is a parser which makes maximal use of lexical and syntactic information, at the earliest possible moment. Clearly, this parser will behave quite differently for different parameter settings in the grammar: In German for example (where I and V are head-final) we may not discover that a particular mittelfeld attachment was inappropriate until the verb is reached (potentially quite far 'downstream'). In English, on the other hand, the head initial structure means such errors are identified much more quickly. It is not, however, necessary to characterise such differences by explicitly setting parameters in the PS interpreter. In contrast, however, by using the same parsing algorithm across languages, we can ensure that the fundamental requirement of incrementality will be met universally. Thus the parser is flexible enough to make use of both top-down and bottom-up strategies where appropriate (and when entailed by the PIC) but remains uniform across languages. Thus, in the absence of any convincing arguments for the parameterisation of the HSPM, we assume the parsing mechanism is universal, and that any variations in performance result from the way in which a particular parameterisation of UG interacts with the UP.

Interestingly, Mazuka and Lust reject the notion of a more flexible parser (namely one based on Marcus' D-theory [Marcus *et al* 83]) on the grounds that it will fail to rule out a number of garden path phenomena, such as reduced relatives[11]. The present model, however, which accounts for such garden paths at the level of thematic structure would not be subject to such a criticism, should we choose to incorporate such a parser in the PS module.

[11] Their rejection on these grounds is somewhat premature, since a D-theory parser could be constrained in a number of natural ways which would still account for such phenomena.

# Chapter 8

# Conclusions

This thesis has investigated the way in which principled theories of syntax and modular theories of mind may participate in an incremental model of human linguistic performance. In particular, we suggest that the architecture and operation of the human sentence processor are determined by the desire to achieve maximal modularity, for reasons of efficiency, while simultaneously demanding that the behaviour of these modules maximise incremental comprehension.

The existing psycholinguistic literature has interpreted the core empirical data in two different ways: On the one hand, there is clear evidence, both experimental and intuitive, that we incrementally interpret and comprehend utterances as they are received. We take it as a given fact that, if we comprehend sentences incrementally, we must similarly perform all prerequisite tasks — such as phonological, syntactic, and semantic analysis — in an incremental fashion as well. On the other hand, there is a variety of evidence which suggests that not all types of information are brought to bear simultaneously. In particular, we highlighted evidence suggesting that the initial analysis assigned to the input is directed by purely syntactic considerations — without regard to semantic information. This led us to postulate a modular language processor containing encapsulated phonological, syntactic, and semantic systems. Crucially, these constituent processors must operate concurrently so as to maintain incremental comprehension. As we remarked above, however, this is in fact the most attractive sort of modularity on independent grounds. That is, the whole motivation for the modularity paradigm is presumably that it permits distributed, concurrent processing of those tasks which can be decomposed into encapsulated subsystems.

Focusing our attention on the syntactic processor, we observed that current transformational grammar encodes syntactic information and structure in a number of ways: In addition to making use of 'annotated' surface structures — encoding constituency, coindexation, and thematic relations — the theory also makes use of the derivational (tree-transducing) mechanism, Move-$\alpha$, to characterise movement. To make these aspects of syntactic analysis more explicit, we gave a purely representational account of the theory, which distinguishes four information/representation types: (1) phrase structure, (2) chain structure, (3) coreference, and (4) thematic structure. Each representation is then constrained by the relevant principles of grammar. An important advantage of this reconstruction is that the grammar is agnostic with respect to particular mechanisms such as movement. This eschews any temptation to ascribe some procedural interpretation to the declarative principles of grammar. Our claim is simply that any theory of performance which can outline how syntactic information is recovered, such that structures are licensed by the principles of grammar, is directly consistent with the principles and parameters competence model in the only relevant sense.

The natural partition of the syntax into four representational domains, combined with the modularity maxim, led us to propose a modular syntactic processor wherein each representation type defines a sub-processor. In addition, we illustrated how this organisation could successfully account for a variety of empirical phenomena, such as the apparent delay in the use of detailed lexical information (i.e. subcategorization/$\theta$-grid information), as accounted for by our distinction of phrase structure and thematic processors. To account for preferences in the face of ambiguity, we argued for a number of specific processing principles: A-Attachment (AA) prefers the attachment of constituents into argument positions (e.g. subject and complement positions), while the default strategy of DS-Attachment (DSA) prefers the attachment of constituents into their canonical position (i.e. unmoved), in the event that an A-position is unavailable. The encapsulation of phrase structure from chain information further led us to hypothesize the Active Trace Strategy (ATS) to account for the early resolution of antecedent-trace relations. While this strategy may seem intractable at first consideration, we demonstrated in the computational model that it is in fact not significantly less efficient than a standard 'active-filler' strategy, since unlicensed traces are immediately detected by the chain processor, and withdrawn. Perhaps the most crucial point here, however, is that in addition to being defined in terms of grammatically relevant notions, each of these strategies can be considered as an instance of our more general principle of incremental comprehension (PIC). That is, AA, DSA, and the ATS are simply the natural realisations of the PIC in the context of syntactic processing.

The modularity paradigm for which we have argued — i.e. one in which the mind seeks to encapsulate sub-systems which can be processed concurrently — assumes the mind has a wealth of computational resources. That is, concurrent, distributed processing is easy, and should be maximised. Thus in presenting a preliminary explanation of processing complexity, we have assumed that reanalysis within a given processor is not particularly difficult, precisely because these modules are fast, dumb, and good at what they do. Rather, we propose that observable increases in complexity arise precisely when reanalysis *cannot* be contained or isolated within a particular module, but propagates through the surrounding modules — what we have labelled the Concentric Theory of Complexity (CTC). While a precise characterisation of this complexity metric remains an issue for future research, it seems the most natural way to explain processing complexity in a distributed, concurrent, modular architecture. Put simply, in a system which can, for the most part, efficiently achieve maximal incremental comprehension on a distributed processing platform, it follows that breakdown in the latter (i.e. among modules) will correspondingly disrupt comprehension in roughly proportionate terms.

In our reconstruction of the syntactic framework, we emphasised that principles could be defined as strictly *local* conditions on well-formed units of structure. In our formalisation of the computational model, we further noted that this contributed to both the efficient and incremental operation of the various modules. In particular, we noted that as partial representations are built, we can determine that each unit of structure added is well-formed. Only if the subsequent input is inconsistent with that representation will backtracking need to take place. Whether this strictly local application of principles can be maintained for more complete theories of syntax is far from clear, but if so it lends credence to the hypothesis that the grammar obeys these locality constraints (for the relevant representation type) precisely because they are required by the processing mechanism. Whether or not other aspects of syntactic theory are reducible to performance considerations remains an import question in our pursuit of an explanatory theory of the competence-performance relationship.

Finally, in the absence of any convincing arguments for the parameterisation of the HSPM, we assume the parsing mechanism is universal, and that any variations in performance result from the way in which a particular parameterisation of universal grammar interacts with the universal parser. This view has a number of interesting implications: It suggests that the processing mechanism and language acquisition device are completely innate, and that only the grammar is 'parameterised' (i.e. subject to variation). Clearly the existing cross-linguistic evidence in support of this position is rather thin on the ground, and remains as perhaps the most important direction for future inquiry. If this position may be successfully defended, however, it may be taken as evidence that constraints concerning the acquisition and performance mechanisms are the relevant *a priori* determinants of possible languages. That is, UG may simply define the class of interpretable languages (i.e. by human) which are both parsable and acquirable.

# Bibliography


[Abney 87] Stephen Abney. *The English Noun Phrase and its Sentential Aspect.* Unpublished PhD thesis, MIT, Cambridge, Massachusetts, 1987.

[Abney 89] Stephen Abney. A Computational Model of Human Parsing. *Journal of Psycholinguistic Research*, 18(1), 1989.

[Abramson *et al* 88] H. Abramson, M. Crocker, B. Ross, and D. Westcott. A Fifth Generation Translator Writing System: Towards an Expert System for Compiler Development. In *International Workshop on Programming Language Implementation and Logic Programming*, Orleans, France, 1988.

[Ades & Steedman 82] A. Ades and M. Steedman. On the Order of Words. *Linguistics and Philosophy*, 4:517–558, 1982.

[Altmann & Steedman 88] Gerry Altmann and Mark Steedman. Interaction with Context during Human Sentence Processing. *Cognition*, 30, 1988.

[Berwick & Weinberg 84] Robert C. Berwick and Amy S. Weinberg. *The Grammatical Basis of Linguistic Performance.* Current Studies in Linguistics. The MIT Press, Cambridge, Massachusetts, 1984.

[Berwick & Weinberg 85] Robert C. Berwick and Amy S. Weinberg. Deterministic Parsing and Linguistic Explanation. *Language and Cognitive Processes*, 1(2):109–134, 1985.

[Bever & McElree 88] Thomas G. Bever and Brian McElree. Empty categories access their antecedents during comprehension. *Linguistic Inquiry*, 19, 1988.

[Borer & Wexler 87] Hagit Borer and Kenneth Wexler. The Maturation of Syntax. In T. Roeper and E. Williams, editors, *Parameter Setting*, pages 273–317. Reidel, Dordrecht, The Netherlands, 1987.

[Borer & Wexler 88] Hagit Borer and Kenneth Wexler. The Maturation of Grammatical Principles. unpublished manuscript, University of California, Irvine, August 1988.

[Borer 84] Hagit Borer. *Parametric Syntax.* Foris, Dordrecht, 1984.

[Breidt 89] Elisabeth Breidt. Thematic Roles in Language Processing. unpublished ms., Centre for Cognitive Science, University of Edinburgh, 1989.

[Bresnan & Kaplan 82] J. Bresnan and R. Kaplan. Lexical-Functional Grammar: A formal system for grammatical representation. In J. Bresnan, editor, *The mental representation of grammatical relations.* MIT Press, Cambridge, Mass., 1982.

[Bresnan 82] J. Bresnan. *The mental representation of grammatical relations.* MIT Press, Cambridge, Mass., 1982.

[Cann 88] Ronald Cann. Heads Without Bars: A Theory of Phrase Structure. unpublished manuscript, University of Edinburgh, 1988.

[Caplan & Hildebrandt 88] David Caplan and Nancy Hildebrandt. *Disorders of Syntactic Comprehension.* Issues in the Biology of Language and Cognition. The MIT Press, Cambridge, Massachusetts, 1988.

[Carlson & Tanenhaus 88] Greg N. Carlson and Michael K. Tanenhaus. Thematic Roles and Language Comprehension. *Syntax and Semantics*, 21:263–288, 1988.

[Chomsky 57] Noam Chomsky. *Syntactic Structures.* Mouton, The Hague, 1957.

[Chomsky 65] Noam Chomsky. *Aspects of the Theory of Syntax.* MIT Press, Cambridge, Massachusetts, 1965.

[Chomsky 73] Noam Chomsky. Conditions on Transformations. In S. R. Anderson and P. Kiparsky, editors, *A Festschrift for Morris Halle.* Holt, Rinehart and Winston, New York, 1973.

[Chomsky 80] Noam Chomsky. *Rules and Representations.* Basil Blackwell, Oxford, 1980.

[Chomsky 81a] Noam Chomsky. *Lectures on Government and Binding.* Foris Publications, Dordrecht, 1981.

[Chomsky 81b] Noam Chomsky. Principles and Parameters in Syntactic Theory. In Norbert Hornstein and David Lightfoot, editors, *Explanation in Linguistics*, chapter 2, pages 32–75. Longman, London, 1981.

[Chomsky 82] Noam Chomsky. *Some Concepts and Consequences of the Theory of Government and Binding.* Linguistic Inquiry Monograph Six. The MIT Press, Cambridge, Massachusetts, 1982.

[Chomsky 86a] Noam Chomsky. *Barriers*. Linguistic Inquiry Monograph Thirteen. The MIT Press, Cambridge, Massachusetts, 1986.

[Chomsky 86b] Noam Chomsky. *Knowledge of Language: Its Nature, Origin and Use*. Convergence Series. Praeger, New York, 1986.

[Chomsky 88] Noam Chomsky. Some Notes on Economy of Derivation and Representation. unpublished ms., MIT, 1988.

[Christensen 82] Kirsti Christensen. On Multiple Filler-Gap Constructions in Norwegian. In Elisabet Engdahl and Eva Ejerhed, editors, *Readings on Unbounded Dependencies in Scandinavian Languages*. Almqvist and Wiksell International, Stockholm, Sweden, 1982.

[Clifton & Frazier 89] Charles Clifton and Lyn Frazier. Comprehending Sentences with Long-Distance Dependencies. In Greg Carlson and Michael Tanenhaus, editors, *Linguistic Structure in Language Processing*, pages 273–317. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1989.

[Clifton 90] Charles Clifton. The Use of Lexical Information in Sentence Comprehension. Five College Cognitive Science Paper 90-1, University of Massachusetts, Massachusetts, April 1990.

[Clifton *et al* 89] Charles Clifton, Shari Speer, and Steven Abney. Parsing Arguments: Phrase Structure and Argument Structure as Determinants of Initial Parsing Decisions. unpublished manuscript, University of Massachusetts, Amherst, 1989.

[Clocksin & Mellish 81] W.F. Clocksin and C.S. Mellish. *Programming in Prolog*. Springer Verlag, 2nd edition, 1981.

[Crain & Fodor 85] Stephen Crain and Janet Dean Fodor. How Can Grammars Help Parsers? In David R. Dowty, Lauri Karttunen, and Arnold M. Zwicky, editors, *Natural Language Parsing*, chapter 3, pages 94–128. Cambridge University Press, Cambridge, England, 1985.

[Crain & Steedman 85] Stephen Crain and Mark Steedman. On not being led up the garden path: the use of context by the psychological syntax processor. In David R. Dowty, Lauri Karttunen, and Arnold M. Zwicky, editors, *Natural Language Parsing*, chapter 10, pages 320–358. Cambridge University Press, Cambridge, England, 1985.

[Crocker 88] Matthew W. Crocker. A Principle-Based System for Natural Language Analysis and Translation. Unpublished M.Sc. thesis, University of British Columbia, Vancouver, Canada, August 1988.

[Crocker 91a] Matthew W. Crocker. A Principle-Based System for Syntactic Analysis. *The Canadian Journal of Linguistics*, 3, 1991.

[Crocker 91b] Matthew W. Crocker. Multiple Interpreters in a Principle-Based Model of Sentence Processing. In *Proceedings of the 5th Conference of the European ACL*, Berlin, Germany, 1991.

[Crocker 91c] Matthew W. Crocker. Multiple Meta-Interpreters in a Logical Model of Sentence Processing. In C. Brown and G. Koch, editors, *Natural Language Understanding and Logic Programming, III.* Elsevier Science Publishers (North-Holland), 1991.

[Cuetos & Mitchell 88] F. Cuetos and D. C. Mitchell. Cross-linguistic differences in parsing: Restrictions on the use of the Late Closure strategy in Spanish. *Cognition*, 30:73–105, 1988.

[Dorr 87] Bonnie Dorr. UNITRAN: A Principle-Based Approach to Machine Translation. Unpublished M.Sc. thesis, MIT, Cambridge, Massachusetts, 1987.

[Emonds 76] Joseph Emonds. *A Transformational Approach to English Syntax.* Academic Press, New York, 1976.

[Engdahl 82] Elisabet Engdahl. Interpreting Sentences With Multiple Filler-Gap Dependencies. Working Papers 24, Lund University, 1982.

[Ferreira & Clifton 86] Fernanda Ferreira and Charles Clifton. The Independence of Syntactic Processing. *Journal of Memory and Language*, 25:348–368, 1986.

[Fodor & Pylyshyn 88] J. A. Fodor and Z. W. Pylyshyn. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28:3–71, 1988.

[Fodor 78] Janet D. Fodor. Parsing strategies and constraints on transformations. *Linguistic Inquiry*, 9:427–473, 1978.

[Fodor 83] Jerry A. Fodor. *Modularity of Mind.* MIT Press, Cambridge, Massachusetts, 1983.

[Fong 91] Sandiway Fong. *Computational Properties of Principle-Based Grammatical Theories.* Unpublished PhD thesis, MIT, Cambridge, Massachusetts, 1991.

[Ford *et al* 82] M. Ford, J. Bresnan, and R. Kaplan. A competence-based theory of syntactic closure. In J. Bresnan, editor, *The mental representation of grammatical relations.* MIT Press, Cambridge, Mass., 1982.

[Frazier & Clifton 89] Lyn Frazier and Charles Clifton. Successive Cyclicity in the Grammar and Parser. *Language and Cognitive Processes*, 4(2):93–126, 1989.

[Frazier & d'Arcais 89] Lyn Frazier and Giovanni B. Flores d'Arcais. Filler Driven Parsing: A Study of Gap Filling in Dutch. *Journal of Memory and Language*, 28:331–344, 1989.

[Frazier & Rayner 82] Lyn Frazier and Keith Rayner. Making and Correcting Errors During Sentence Comprehension: Eye Movements in the Analysis of Structurally Ambiguous Sentences. *Cognitive Psychology*, 14, 1982.

[Frazier & Rayner 88] Lyn Frazier and Kieth Rayner. Parameterizing the Language Processing System: Left- vs. Right-branching Within and Across Languages. In J. Hawkins, editor, *Explaining Linguistic Universals*, pages 247–279. Basil Blackwell, Oxford, 1988.

[Frazier 79] Lyn Frazier. *On Comprehending Sentences: Syntactic Parsing Strategies.* Unpublished PhD thesis, University of Connecticut, Connecticut, 1979.

[Frazier 84] Lyn Frazier. Modularity and the Representational Hypothesis. In *Proceedings of NELS 15*, pages 131–144, Brown University, Providence, Rhode Island, 1984.

[Frazier 85] Lyn Frazier. Syntactic Complexity. In David R. Dowty, Lauri Karttunen, and Arnold M. Zwicky, editors, *Natural Language Parsing*, chapter 4, pages 129–189. Cambridge University Press, Cambridge, England, 1985.

[Frazier 87a] Lyn Frazier. Syntactic Processing: Evidence from Dutch. *Natural Language and Linguistic Theory*, 5:519–559, 1987.

[Frazier 87b] Lyn Frazier. Theories of Sentence Processing. In Jay L. Garfield, editor, *Modularity in Knowledge Representation and Natural Language Understanding*, pages 291–307. MIT Press, Cambridge, Massachusetts, 1987.

[Frazier 90] Lyn Frazier. Exploring the Architecture of the Language System. In Gerry Altmann, editor, *Cognitive Models of Speech Processing: Psycholinguistic and Computational Perspectives.* MIT Press, Cambridge, Massachusetts, 1990.

[Gibson 91] Edward Gibson. *A Computational Theory of Human Linguistic Processing: Memory Limitations and Processing Breakdown.* Unpublished PhD thesis, Carnegie Mellon University, Pittsburgh, 1991.

[Gorrell 87] Paul Gorrell. *Studies of Human Syntactic Processing: Ranked-Parallel Versus Serial Models.* Unpublished PhD thesis, University of Connecticut, Connecticut, 1987.

[Gorrell 89] Paul Gorrell. Establishing the Loci of Serial and Parallel Effects in Syntactic Parsing. *Journal of Psycholinguistic Research*, 18(1), 1989.

[Hasegawa 90] Nobuko Hasegawa. Comments on Mazuka and Lust's Paper. In Lyn Frazier and Jill de Villiers, editors, *Language Processing and Language Acquisition*, pages 207–223. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1990.

[Hornstein & Lightfoot 81] Norbert Hornstein and David Lightfoot. Introduction. In Norbert Hornstein and David Lightfoot, editors, *Explanation in Linguistics*, chapter 1, pages 9–31. Longman, London, 1981.

[Huang 82] C.-T. James Huang. *Logical Relations on Chinese and the Theory of Grammar.* Unpublished PhD thesis, MIT, Cambridge, Massachusetts, 1982.

[Johnson 89] Mark Johnson. Use of Knowledge of Language. *Journal of Psycholinguistic Research*, 18(1), 1989.

[Johnson 91] Mark Johnson. Program Transformation Techniques for Deductive Parsing. In C. Brown and G. Koch, editors, *Natural Language Understanding and Logic Programming, III.* Elsevier Science Publishers (North-Holland), 1991.

[Kayne 84] Richard Kayne. *Connectedness and Binary Branching.* Foris Publications, Dordrecht, 1984.

[Kimball 73] John Kimball. Seven Principles of Surface Structure Parsing in Natural Language. *Cognition*, 2(1):15–47, 1973.

[Kolb & Thiersch 90] Hans Peter Kolb and Craig Thiersch. Levels and Empty Categories in a Principles and Parameters Approach to Parsing. Research Report 19, Institute for Language Technology and Artificial Intelligence, Tilburg University, The Netherlands, July 1990.

[Koster 87] Jan Koster. *Domains and Dynasties: the radical approach to syntax.* Foris, Dordrecht, 1987.

[Lasnik & Saito 89] Howard Lasnik and Mamoru Saito. Move-$\alpha$: Conditions on its application and output. Unpublished ms. University of Connecticut, 1989.

[Lasnik & Uriagereka 88] Howard Lasnik and Juan Uriagereka. *A Course in GB Syntax: Lectures on Binding and Empty Categories.* Current Studies in Linguistics. MIT Press, Cambridge, Massachusetts, 1988.

[Latecki 91] Longin Latecki. An Indexing Technique for Implementing Command Relations. In *Proceedings of the 5th Conference of the European ACL*, Berlin, Germany, 1991.

[Lightfoot 82] David Lightfoot. *The Language Lottery: Towards a Biology of Grammars.* MIT Press, Cambridge, Massachusetts, 1982.

[Linebarger 89] Marcia Linebarger. Neuropsychological Evidence for Linguistic Modularity. In Greg Carlson and Michael Tanenhaus, editors, *Linguistic Structure in Language Processing*, pages 197–238. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1989.

[Manzini 88] M. Rita Manzini. Constituent Structure and Locality. In A. Cardinaletti, G. Cinque, and G. Giusti, editors, *Constituent Structure: Papers from the 1987 Glow Conference.* Foris, Dordrecht, 1988.

[Marcus 80] Mitchell P. Marcus. *A Theory of Syntactic Recognition for Natural Language.* The MIT Press Series in Artificial Intelligence. The MIT Press, Cambridge, Massachusetts, 1980.

[Marcus *et al* 83] M. Marcus, D. Hindle, and M. Fleck. D-theory: Talking about talking about trees. In *Proceedings of 21st Conference of the ACL*, pages 129–136, 1983.

[Marslen-Wilson & Tyler 87] W. D. Marslen-Wilson and L. K. Tyler. Against Modularity. In Jay L. Garfield, editor, *Modularity in Knowledge Representation and Natural Language Understanding.* MIT Press, Cambridge, Massachusetts, 1987.

[Marslen-Wilson 75] William Marslen-Wilson. Sentence Perception as an Interactive, Parallel Process. *Science*, 189:226–228, 1975.

[Mazuka & Lust 90] Reiko Mazuka and Barbara Lust. On Parameter Setting and Parsing: Predictions for Cros s-Linguistic Differences in Adult and Child Processing. In Lyn Frazier and Jill de Villiers, editors, *Language Processing and Language Acquisition*, pages 163–206. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1990.

[McClelland *et al* 89] J. McClelland, M. St. John, and R. Taraban. Sentence Processing: A Parallel Distributed Processing Approach. *Language and Cognitive Processes: Special issue on Pars ing and Interpretation*, pages 287–335, 1989.

[McElree & Bever 89] Brian McElree and Thomas G. Bever. The Psychological Reality of Linguistically Defined Gaps. *Journal of Psycholinguistic Research*, 18(1), 1989.

[Millies 90] Sebastian Millies. Ein Modularer Ansatz fur Prinzipienbasiertes Parsing. unpublished ms., Universitaet Saarbruecken, 1990.

[Mitchell & Holmes 85] D. C. Mitchell and V. M. Holmes. The role of specific information about the verb in parsing sentences with local ambiguity. *Journal of Memory and Language*, 24:542–559, 1985.

[Mitchell 89a] Don Mitchell. Lexical Guidance in Human Parsing: Locus and Processing Characteristics. *Attention and Performance XII*, pages 123–154, 1989.

[Mitchell 89b] Don Mitchell. Verb-Guidance and Other Lexical Effects in Parsing. *Language and Cognitive Processes: Special issue on Pars ing and Interpretation*, pages 123–154, 1989.

[Muysken 83] Pieter Muysken. Parameterizing the Notion 'Head'. *Journal of Linguistic Research*, 2:57–76, 1983.

[Pereira & Shieber 87] Fernando C.N. Pereira and Stuart M. Shieber. *Prolog and Natural-Language Analysis*. CSLI Lecture Notes. Center for the Study of Language and Information, Stanford, California, 1987.

[Pereira & Warren 83] Fernando Pereira and David Warren. Parsing as Deduction. In *Proceedings of Twenty-First Conference of the ACL*, Cambridge, Massachusetts, 1983.

[Pereira 85] Fernando C. N. Pereira. A New Characterization of Attachment Preferences. In David R. Dowty, Lauri Karttunen, and Arnold M. Zwicky, editors, *Natural Language Parsing*, chapter 9, pages 307–319. Cambridge University Press, Cambridge, England, 1985.

[Pickering & Barry 91] Martin Pickering and Guy Barry. Sentence Processing without Empty Categories. *Language and Cognitive Processes*, 6(3):229–259, 1991.

[Postal 64] Paul Postal. Limitations of Phrase Structure Grammars. In Jerry Fodor and Jerrold Katz, editors, *The Structure of Langauge: Readings in the Philosophy of Language*, pages 137–151. Prentice-Hall, Englewood Cliffs, 1964.

[Pritchett 88] Brad Pritchett. Garden Path Phenomena and the Grammatical Basis of Language Processing. *Language*, 64:539–576, September 1988.

[Pritchett ar] Bradley L. Pritchett. Locality in language processing: head-driven parsing, head-position, & the CED. In H. Goodluck and M. Rochemont, editors, *Psycholinguistics of Island Constraints*, (to appear).

[Pullum & Gazdar 82] Geoffrey Pullum and Gerald Gazdar. Natural Languages and Context-free Languages. *Linguistics and Philosophy*, 4:471–504, 1982.

[Pulman 86] Steven G. Pulman. Grammars, Parsers and Memory Limitations. *Language and Cognitive Processes*, 1(3):197–225, 1986.

[Pylyshyn 84] Zenon Pylyshyn. *Computation and Cognition: Toward a Foundation for Cognitvie Science.* The MIT Press, Cambridge, Massachusetts, 1984.

[Rayner *et al* 83] Keith Rayner, Marcia Carlson, and Lyn Frazier. The Interaction of Syntax and Semantics During Sentence Processing: Eye Movements in the Analysis of Semantically Biased Sentences. *Journal of Verbal Learning and Verbal Behavior*, 22:358–374, 1983.

[Rizzi 82] Luigi Rizzi. *Issues in Italian Syntax.* Foris, Dordrecht, 1982.

[Rizzi 86] Luigi Rizzi. On Chain Formation. *Syntax and Semantics*, 19:65–95, 1986.

[Rochemont & Culicover 90] Michael Rochemont and Peter Culicover. *English Focus Constructions and the Theory of Grammar.* Cambridge University Press, Cambridge, England, 1990.

[Ross 67] John R. Ross. *Constraints on Variables in Syntax.* Unpublished PhD thesis, MIT, Cambridge, Massachusetts, 1967.

[Selkirk 84] Elizabeth Selkirk. *Phonology and Syntax: The relation between sound an structure.* Linguistic Inquiry Monograph Thirteen. The MIT Press, Cambridge, Massachusetts, 1984.

[Sells 85] Peter Sells. *Lectures on Contemporary Syntactic Theories.* CSLI Lecture Notes. Center for the Study of Language and Information, Stanford, California, 1985.

[Sharp 85] Randall M. Sharp. A Model of Grammar Based on Principles of Government and Binding. Unpublished M.Sc. thesis, University of British Columbia, Vancouver, Canada, October 1985.

[Sharp 86] Randall M. Sharp. A Parametric NL Translator. In *11th International Conference on Computational Linguistics*, pages 124–126, Bonn, West Germany, August 1986.

[Shieber 85] Stuart Shieber. Evidence Against the Context-freeness of Natural Languages. *Linguistics and Philosophy*, 8:333–343, 1985.

[Speas 90] Margaret Speas. *Phrase Structure in Natural Language*. Studies in Natural Language and Linguistic Theory. Kluwer, Dordrecht, 1990.

[Stabler 87] Edward P. Stabler. Logic Formulations of Government-Binding Principles for Automatic Theorem Provers. Cognitive Science Memo 30, University of Western Ontario, London, Ontario, June 1987.

[Stabler 89a] Edward Stabler. Avoid the pedestrian's paradox. unpublished ms., Dept. of Linguistics, UCLA, 1989.

[Stabler 89b] Edward P. Stabler. *The Logical Approach to Syntax*. The MIT Press (forthcoming), Cambridge, Massachusetts, 1989.

[Steedman 90] Mark Steedman. Structure and Intonation in Spoken Language Understanding. In *Proceedings of 28th Conference of the ACL*, pages 9–16, 1990.

[Stowe 86] Laurie Stowe. Parsing Wh-constructions: Evidence for on-line gap location. *Language and Cognitive Processes*, 1:227–246, 1986.

[Stowe 89] Laurie A. Stowe. Thematic Structures and Sentence Comprehension. In Greg Carlson and Michael Tanenhaus, editors, *Linguistic Structure in Language Processing*, pages 319–357. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1989.

[Swinney 79] D. A. Swinney. Lexical access during sentence comprehension: (Re)consideration of contextual effects. *Journal of Verbal Learning and Verbal Behavior*, 18:645–659, 1979.

[Swinney *et al* 88] D. Swinney, M. Ford, U. Frauenfelder, and J. Bresnan. On the temporal course of gap-filling and antecedent assignment during sentence comprehension. In B. Grosz, R. Kaplan, M. Macken, and I. Sag, editors, *Language Structure and Processing*. CSLI, Stanford, CA, 1988.

[Tamaki & Sato 84] H. Tamaki and T. Sato. Unfold/Fold Transformation of Logic Programs. In *Second International Conference on Logic Programming*, pages 127–137, Uppsala, Sweden, 1984.

[vanRiemsdijk & Williams 86] Henk van Riemsdijk and Edwin Williams. *Introduction to the Theory of Grammar*. Current Studies in Linguistics. The MIT Press, Cambridge, Massachusetts, 1986.

[Wehrli 83] Eric Wehrli. A Modular Parser for French. In *Proceedings of the Eighth International Conference on Artificial Intelligence*, pages 686–689, Karlsruhe, West Germany, 1983.

[Wehrli 88] Eric Wehrli. Parsing with a GB-Grammar. In U. Reyle and C. Rohrer, editors, *Natural Language Parsing and Linguistic Theories*, pages 177–201. Reidel, Dordrecht, 1988.

[Wexler & Culicover 80] Ken Wexler and Peter Culicover. *Formal Principles of Language Acquisition.* MIT Press, Cambridge, Massachusetts, 1980.

# Appendix A

# The Syntactic Processor

```
%%_______________________________________________________
%%
%% Module: SynPro
%% Paradigm: sentence_processor(InputString,ThetaStructure)
%% Description: Specification of the syntactic processor.
%%_______________________________________________________
%% Written: February 26 1990 Revised: Jul 1991
%% (c) 1991 M. W. Crocker, University of Edinburgh
%%_______________________________________________________

%_______________________________________________________
% sentence_processor(InputString,ThetaStructure): generates
a ThetaStructure output for an Input String. The
output is constructure incrementally, via co–routining.

sentence_processor(InputString,ThetaStructure) :–
abolish(state/1),
start_symbol(NTnode), State = {NTnode/_,ThetaStructure,_},
synpro(Input,['.'],State),
increment(Input,InputString,State),
well_formed(State),
assert(state(State)),!.

start_symbol(nt:{c,[m(+),s(−),c(−)],ID,F}) :– language(_).

%_______________________________________________________
% SynPro: The syntactic processor relates a given input to a
well–formed State = {PS, TS, ChS} (Coreference structure
is absent from this version.)
e.g. synpro([the,boy,will,see,the,girl,'.'],['.'],State).
```

```
synpro(Input,Rem,{PS,TS,CS}) :-
ps_module(PS,Input,Rem),
chain_module(PS,CS),
theta_module(PS,CS,TS).

%___________________________________________________________
% increment(Input,InputString,Tree): Incrementally instantiates
Input based on InputString, and display State as each
word is processed. Illustrates incremental processing.

increment(['.'],['.'],State) :-
pretty_print(State),!.
increment([Word|Rest],[Word|Rest1],State) :-
wait_step(State),
increment(Rest,Rest1,State).

%___________________________________________________________
% pretty_print(State): Display the current instantiated State.

pretty_print({PS,TS,CS}) :-
        pretty(PS,ps),
        chain_write(CS,chain),
        theta_write(TS,theta),!.

wait_step(State) :- step_mode(off),!.
wait_step(State) :- step_mode(on), !,
pretty_print(State), write('Press any key to proceed.'), get0(_),!.

%___________________________________________________________
% well_formed(State): Ensure that the final state is well-formed.
I.e. there must be no remaining uninstantiated structure.

well_formed({PS,TS,CS}) :-


% Make sure all chains are complete.


cs_ok([Chain|Rest]) :-
member(X,Chain), var(X),!,fail.
cs_ok([Chain|Rest]) :-
cs_ok(Rest).
```

# Appendix B

# PS Module

```
%%________________________________________________
%%
%% Module: PS Interpreter
%% Paradigm: ps_module(Tree,String,[]).
%% Description: Construct a phrase struct. Tree for the input String.
%%________________________________________________
%% Written: Oct 15 1990 Revised: Jan 24 1991 (DCG conversion)
%% (c) 1991 M. W. Crocker, University of Edinburgh
%%________________________________________________

%________________________________________________
% ps(Tree,Head,Tail): constructs a PS Tree for the input string
(represented by the difference list Head–Tail of a DCG).
Incrementality is achieved by freezing execution on input
(i.e. Head). Difference lists are hidden by using DCG's
wherever possible.

ps_module(Tree) -->
{sister_hood(Tree)}, ps(Tree).

ps(Tree,L,L0) :-
freeze(L, ps_int(Tree,L,L0)).

% Try to parse an 'empty'
% Right branch.
branch_right(Node/[Left,Right]),
ps_view(Node/[Left,Right]),
Daughters = [Left/LD,Right/RD],
ps_ec_eval(Right/RD,X0,X0) ->
ps(Left/LD,X,X0).
```

```
% Build 'functional' structure
% top down.
ps_view(Node/[Left,Right]),
Daughters = [Left/LD,Right/RD],true,! },
ps(Left/LD),
ps(Right/RD).
% Parse v-specifiers top down.
% (I.e. D-structure Subjects.)
          ps_view(Node/[Left,Right]),
  branch_right(Node/[Left,Right]),
  what_pos(Node/[Right,Left],spec),
          Daughters = [Left/LD,Right/RD],true,! },
        ps(Left/LD),
        ps(Right/RD).
% Try to parse and EC.
        ps_ec_eval(Tree,X,X1), X0 = X1.
% Try to parse a CP or IP.
ps_non_lex_eval(Tree).
% Project a lexical structure.
ps_lex_eval(Tree).

%_____________________________________________________
% ps_ec_eval(Tree): postulate an empty category as the daughter or
left corner of Tree.

ps_ec_eval(Tree) -->
{ compatible(Node1,Tree),
  ps_view(Node1/t:{C,ec:[p(-),a(A)],ID,Fs}),
  moveable(C,head),
  attachable(Node1/t:{C,ec:[p(-),a(A)],ID,Fs},Tree) },
project(Node1/t:{C,ec:[p(-),a(A)],ID,Fs},Tree).
ps_ec_eval(Tree) -->
{ compatible(Node1,Tree),
  ps_view(Node1/t:{C,ec:[p(-),a(A)],ID,Fs}),
  moveable(C,phrase),
  Tree = Node1/t:{C,ec:[p(-),a(A)],ID,Fs} },
[].

%_____________________________________________________
% ps_lex_eval(Tree): project structure from the current lexical item
and attach as Tree (roughly, an LR bottom-up parse).

[W], { ps_view(Node/t:{C,lex:W,ID,Fs})},
project(Node/t:{C,lex:W,ID,Fs},Tree).

%_____________________________________________________
```

```
% ps_non_lex_eval(Tree): parse Tree as a non-lexical consitiuen (C/I),
by simply calling 'ps' (which will parse top down).

ps_non_lex_eval(Node/Daughters) -->
{ Node = nt:{C,[m(+),s(-),c(-)],_,_},
  var(C), non_lexical(C) },
ps(Node/Daughters), {!}.

%________________________________________________________________
% project: basically a left corner parser, to achieve incremental and
data-driven operation. Distinguishes left/right branching cases
to enforce relevant constraints (to avoid infinite recursion).

project(T:{C,L,ID,FD}/D,T:{C,L,ID,FM}/D) --> [],{FM <=> FD}.
project(Left/LD,HigherNode) -->
{ compatible(Node,HigherNode),
  branch_left(Node/[Left,Right]),
  ps_view(Node/[Left,Right]),
  attachable(Node/[Left/LD,Right/RD],HigherNode) },
ps(Right/RD),
project(Node/[Left/LD,Right/RD],HigherNode).
project(Node,HigherNode/HD) -->
{ branch_right(HigherNode/[Left,Right]),
  ps_view(HigherNode/[Left,Right]),
  HD = [Left/LD,Right/RD] },
project(Node,Left/LD),
ps(Right/RD).

attachable(Node,Tree) :-
\+ (Node=Tree,true), !, fail.
attachable(Node,Tree).

:- wait sister_hood/1.
sister_hood(X/D) :- !,
id_mark(X),
sister_hood(D,X/D).
sister_hood(X) :- !,
id_mark(X).

:- wait sister_hood/2.
sister_hood(D,X/D) :-
D = [Left/LD,Right/RD],true,!,
case_mark(X/[Left,Right]),!,
sister_hood(Left/LD),
sister_hood(Right/RD),!.
sister_hood(D,X/D) :- !,
```

```
sister_hood(D).

id_mark(X:{Cat,_,ID,_}) :- freeze(Cat, gensym(Cat,ID)).

case_mark(X/[Sat,Right]) :-
% left-ward direction,
cat(Sat,Cat),
freeze(Cat, case_assign(X/[Right,Sat],left) ).
case_mark(X/[Left,Sat]) :-
% Same for right-ward case.
        cat(Sat,Cat),
        freeze(Cat, case_assign(X/[Left,Sat],right) ).

case_assign(M/[D,S],Dir) :-
cat(S,n),
cat(D,Cat),
what_pos(M/[D,S],Pos),
assigns_case(Cat,Pos,Dir),!,
D = _:{_,_,_,Fs},
freeze(Fs, (node_ftrs(trans,D), add_ftr(case:yes,S))),true,!.
case_assign(M/[D,S],Dir) :-
        cat(S,n),
add_ftr(case:no,S),true,!.
case_assign(M/[D,S],Dir) :- !.

node_ftrs(Ftr,_:{_,_,_,Fs}) :-
exists(Ftr,Fs),!.
add_ftr(Ftr,_:{_,_,_,Fs}) :-
member(Ftr,Fs),!.

%________________________________________________________________
% ps_view(Branch): determines possible instantiations of Branch
which are licensed by the principle governing PS.

% Try to attach the
% current lex. item.

% Posit a branch node.
xbar( Node/[Left,Right] ).

% Posit a branch node.
move_a( Node/[Left,Right] ).

% Try to attach the
% current lex. item.
```

*% Try to attach the*
*% current lex. item.*

# Appendix C

# Chain Module

```
%%________________________________________________
%%
%% Module: Chain Interpreter
%% Paradigm: chain_module(Tree,ChainStructure)
%% Description: Construct the set of chains (i.e. ChainStructure)
for the input phrase structure Tree.
%%________________________________________________
%% Written: Oct 15 199 Revised: Aug 1991
%% (c) 1991 M. W. Crocker, University of Edinburgh
%%________________________________________________

chain_module(Tree,CS) :-
close_chains(CS),
chain(Tree,CS).
chain(M/D,CS) :-
freeze(D, chain_int(M/D,CS) ).

%________________________________________________
% chain_int(Branch,CS): finds 'visible' branches in the PS input,
and structure them into chains:
1. Antecedents must head exactly one chain.
2. Traces must be appended to exactly one chain.

:- wait chain_int/3.
% Finds a visible satellite
% in a branch.
cat(Sat,Cat),
freeze(Cat, visible(NTnode/[Sat,Right],LP) ),!,
freeze(SD, create_cnode(Sat/SD,LP,Cnode) ),
freeze(Cnode, chain_member(Cnode,CS) ),
```

```
chain(Right/RD,CS).
chain_int(NTnode/[Left/LD,Right/RD],CS) :- !,
chain(Left/LD,CS),
chain(Right/RD,CS).
% Finds visible terminals.
visible(NTnode/Tnode,LP), !,
        freeze(Tnode, create_cnode(NTnode/Tnode,LP,Cnode) ),
freeze(Cnode, chain_member(Cnode,CS) ).
chain_int(NTnode/Tnode,CS).

%________________________________________________________
% visible(NTM,NTD,Satellite,Cnode): determines if the Satellite
is a visible element for the chain processor, i.e.,
must it be a member of a well-formed chain.

visible(NTnode/[Sat,Daught],phrase/Pos) :-
        move_a( NTnode/[Sat,Daught] ), !,
cat(Sat,Cat),
moveable(Cat,phrase),
position(NTnode/[Daught,Sat],Pos).
visible(NTnode/[Sat,Daught],phrase/home) :-
        cat(Sat,n), !.
visible(NTnode/Tnode,Lev/Pos) :-
move_a( NTnode/Tnode ), !,
cat(Tnode,Cat),
cat(NTnode,C),
moveable(Cat,Lev),
determine_pos(C,Cat,Lev,Pos).

create_cnode(nt:_/t:{C,W,ID,F},LP,Cnode) :- !,
Cnode = c:{C,W,LP,ID,F}.
create_cnode(nt:{C,L,ID,F}/_,LP,c:{C,ant,LP,ID,F}).

determine_pos(C,C,head,home).
determine_pos(C,Cat,head,away) :- \+ C = Cat.
determine_pos(C,C,phrase,home).

chain_member(Cnode,CS) :-
antecedent(Cnode),!,
initiate_chain(Cnode,CS),
chain_view(Cnode # head_of_chain).
chain_member(Cnode,CS) :-
trace(Cnode),!,
find_chain(Cnode,CS).

antecedent(X) :- \+ trace(X).
```

```
trace(c:{_,ec:[p(-),a(A)],_,_,_}).

% adds a new chain, with
% head = Cnode to CS.
initiate_chain(Cnode,Rest).

% find a chain to append to.
nonvar(Chain),
ch_append(Cnode,Chain),!.
find_chain(Cnode,[Chain|Rest]) :-
nonvar(Chain),
find_chain(Cnode,Rest),!.

% If Cnode = [], then close
nonvar(Cnode1), var(Tail),
% the chain if Cnode # [].
Tail = [].
% Try to append Cnode to the
nonvar(Cnode1), var(Tail),
% chain, if chain_view permits.
Tail = [Cnode|_].
ch_append(Cnode,[Cnode1|Rest]) :-
nonvar(Cnode1),!,
ch_append(Cnode,Rest).

:- wait close_chains/1.
close_chains([]).
close_chains([Chain|Rest]) :-
close_chain(Chain),
close_chains(Rest).

:- wait close_chain/1.
close_chain([Link|Tail]) :-
var(Tail),
freeze(Link,chain_view(Link # []) ), true, !,
Tail = [].
close_chain([Link|Rest]) :-
freeze(Link,
close_chain(Rest) ).

%______________________________________________________
% chain_view(Cnode1 # Cnode2): determine if the chain link
Cnode1 # Cnode2 is licensed by the grammar. At present
this only enforces the A-to-Abar constraint, the Case
Filter, and the relevant part of the theta criterion.
```

```
chain_view(Cnode # []) :- !,
theta_criterion(Cnode # []).

% Condition on Chain heads.
case_filter(Cnode # head_of_chain).

% Condition on links.
cat(Cnode1,Cat), cat(Cnode2,Cat),
case_filter(Cnode1 # Cnode2).
```

# Appendix D

# Thematic Module

```
%%_______________________________________________________
%%
%% Module: Thematic Interpreter
%% Paradigm: theta_module(Tree,ChainStructure,ThematicStructure)
%% Description: Construct a ThematicStructure on the basis of theta
positions in the Tree, and resolve filler-gap relations
from ChainStructure.
%%_______________________________________________________
%% Written: Jul/Aug 1991 Revised:
%% (c) 1991 M. W. Crocker, University of Edinburgh
%%_______________________________________________________

:- op(650,xfx,@).

theta_module(Tree,CS,TS) :-
% reduces chains.
theta(Tree,SCS,TS).

theta(M/D,SCS,TS) :-
freeze(D, theta_int(M/D,SCS,TS) ).

%_______________________________________________________
% theta_int: traverses the PS tree, puts left/right branches in a
canonical form and calls theta view.

theta_int(Mom/[Dtr/D,Sat/S],CS,TM) :-
branch_left(Mom/[Dtr,Sat]),
freeze(S, theta_int1(Mom@TM/[Dtr@TD,Sat@TS],CS) ),
        theta(Dtr/D,CS,TD),
        theta(Sat/S,CS,TS).
```

```
theta_int(Mom/[Sat/S,Dtr/D],CS,TM) :-
        branch_right(Mom/[Sat,Dtr]),
        freeze(S, theta_int1(Mom@TM/[Dtr@TD,Sat@TS],CS) ),
        theta(Dtr/D,CS,TD),
        theta(Sat/S,CS,TS).
theta_int(nt:M/t:D,CS,TS) :-
theta_int1(nt:M@TS/t:D,CS).

%___________________________________________________
% theta_int1: resolves any long-distance dependencies. Calls
theta_visible when appropriate.

% Is Sat an antecedent.
% Find a chain,
% that matches Sat,
% Set Theta
TM <=> TD.

theta_int1(Mother@TM/[Dtr@TD,Sat@TS],CS) :-    % Is Sat an Deep-S trace.
% Find a chain,
% that matches Sat,
% Instant. TS
theta_visible(Mother@TM/[Dtr@TD,Sat@TS]).

theta_int1(nt:M@TS/t:D,CS) :-                  % For head traces.
        exists(H#T,CS),                        % Find a chain,
        nonvar(H),                             % whose head H matches
        equiv(H,t:D),!.                        % the current terminal

% For head traces.
% Find a chain,
% whose tail T matches
% the current terminal
cnode_to_ps(H,PS),
theta_visible(nt:M@TS/PS).

% other traces are intermed.
:- !.

% If there's nothing to do, call theta_visible.
theta_visible(Branch).

equiv(c:{C,Phon,phrase/_,ID,_},nt:{C,[m(+),s(-),c(-)],ID,_}).
equiv(c:{C,Phon,head/_,ID,F}, t:{Cat,Phon,ID,F}).
cnode_to_ps(c:{C,Phon,P/L,ID,Fs},t:{C,Phon,_,Fs}).
```

```
%________________________________________________
% theta_visible: determines if the current branch is for a
moved constituent, an intermediate position (ignored),
or is thematicallt relevant.
%     1st clause for heads with args,
%     2nd for "atomic" heads.

theta_visible( nt:{Cat,Lev,ID,FM}@H/t:{Cat,lex:Phon,ID,F} ) :-
% Cat is N,P, or V.
% Get the grid for the head.
H:rel <=> Phon,
H:cat <=> Cat,
% The list of args in the grid.
% Assign roles -> args.

theta_visible( nt:{Cat,Lev,ID,FM}@H/t:{Cat,lex:Phon,ID,F} ) :-
        lexical(Cat),!,                    % Cat is N,P, or V.
        H:rel <=> Phon,
        H:cat <=> Cat.

theta_visible( nt:{Cat,Lev,ID,FM}@H/t:{Cat,lex:Phon,ID,F} ) :- Cat = c,!.

get_roles(Fs,Roles) :- Fs:grid <=> Roles,!,nonvar(Roles).

%_________________________
% The rule for complement satellites.

theta_visible(nt:{C,L,ID,FM}@TM /
   [nt:{C,[m(-),S,c(+)],ID,F}@TD, nt:{CC,[m(+),s(-),c(-)],IDC,FC}@TS] ) :-
C = c,!,
TS:cat <=> i,
% CP doesn't theta-mark its complement.

theta_visible(nt:{C,L,ID,FM}@TM /
   [nt:{C,[m(-),S,c(+)],ID,F}@TD, nt:{CC,[m(+),s(-),c(-)],IDC,FC}@TS] ) :- !,
% mothers structure = daughters.
% construct an internal argument.
% the argument is the satellite.
% add it to the mothers grid.

%_________________________
% The rule for specifier satellites.

theta_visible(nt:{C,[m(+),s(-),c(-)],ID,FM}@TM /
   [nt:{C,[m(-),s(+),c(-)],ID,F}@TD, nt:{SC,[m(+),s(-),c(-)],IDC,FC}@TS] ) :-
% only for spec of lexical phrase.
```

```
        TM <=> TD,              % mothers structure = daughters.
        Arg:pos <=> ext,        % construct an external argument.
        Arg:arg <=> TS,         % the argument is the satellite.
        TD:grid <== Arg.        % add it to the mothers grid.

%______________________________
% Do nothing, if sat is intermed trace.

theta_visible(M@TM/[D@TD,S@TS_]) :-
TM <=> TD.


% theta_assign(Args,Grid): where Grid is a list of roles,
and Args is a list of constituents to receive roles,
assign each constituent a role, and ensure each role
is assigned.
% Roles: { agent, pat, theme, prop, loc, instr, source, dest }
% Pos:   { ext, int }
% Entry: Role/Type or {Role/Type} for optional roles.
% ex:     "saw" grid:[ agent/ext, pat/int, {inst/int} ]
% Each Arg has the structure [role: R, pos: P, arg: A], where A
is the feature structure for the actual constituent.

theta_assign(Args,[]) :- !.
theta_assign(Args,Grid) :- theta_assign1(Args,Grid).

:- wait theta_assign1/2.
theta_assign1([],[]) :- !.
theta_assign1([],Grid) :- member(Role,Grid), \+ Role = {_}, !, fail.
theta_assign1([Arg|Args],Grid) :-
Arg:arg:cat <=> C,!,
remove(R/T,Grid,Grid1),
csr(R/T,Arg),
theta_assign(Args,Grid1) )).

%__________________________________________________________
% simplify_chains: Takes chains of the form [Head, ... , Tail]
and creates a < Head # Tail @ TS > term, where TS is the
theta structure of Head.

:- wait simplify_chains/2.
simplify_chains([],[]) :- !.
simplify_chains([C|R],[Rchain | NewR]) :-
        freeze(C, reduce_chain(C,Rchain) ),
        simplify_chains(R,NewR).
```

```
reduce_chain(Chain,Head # Tail) :-
% The Head of the Chain.
% The chain is for a head.
% Get the Chain's Tail.
% Retrieve the Heads features.

reduce_chain(Chain,[]) :-
% The Head of the Chain.
% The chain is for a phrase.
% It's a unit chain -- ignore.

% Has a slot for TS.
% The Head of the Chain.
% The chain is for a phrase.
% Get the Chain's Tail.

copy_ftrs(c:{_,_,_,_,Fs},c:{_,_,_,_,Fs}).
chain_type(c:{_,_,T/_,_,_},T).

:- wait get_tail/2.
get_tail([Cnode|Rest],Tail) :-
        freeze(Rest, is_tail(Cnode,Rest,Tail) ).
is_tail(Cnode,[],Cnode) :- !.
is_tail(Cnode,List,Tail) :-
        get_tail(List,Tail).
```

# Appendix E

# Universal Grammar

```
%%________________________________________________
%%
%% Module: Universal Grammar
%% Paradigm: knowledge base
%% Description: Specification of all the UG principles
%%________________________________________________
%% Written: Oct 18 1990 Revised: Jun 1991
%% (c) 1991 M. W. Crocker, University of Edinburgh
%%________________________________________________

%________________________________________________
% PS-Representations:
nt:{Cat,Level,ID,Ftrs}   (non-terminals)
t:{Cat,Phon,ID,Ftrs}     (terminals)
%
Branch: nt:{..}/[nt:{..},nt:{..}]
Branch: nt:{..}/t:{..}

?- op(500,xfy,':').
?- op(550,xfy,'/').

%_____________________
% X-bar Theory

nt:{Cat,[m(+),s(-),c(-)],ID,FM}/
[nt:{S,[m(+),s(-),c(-)],IDS,FS},nt:{Cat,[m(-),s(+),c(-)],ID,FD}] ) :-
spec_pos(Cat,initial),spec(S,Cat,FS), FM <=> FD.
nt:{Cat,[M,S,Comp],ID,FM}/
[nt:{Cat,[m(-),S,c(+)],ID,FD},nt:{C,[m(+),s(-),c(-)],IDC,FC}] ) :-
head_pos(Cat,initial),comp(C,Cat), FM <=> FD,
```

```
comp_features(Cat,Comp).
xbar(    nt:{Cat,[M,S,Comp],ID,FM}/
         [nt:{C,[m(+),s(-),c(-)],IDC,FC},nt:{Cat,[m(-),S,c(+)],ID,FD}] ) :-
                 head_pos(Cat,final),comp(C,Cat),
comp_features(Cat,Comp), FM <=> FD.
nt:{Cat,Lev,ID,FM}/t:{Cat,lex:Phon,ID,FD} ) :-
lexicon(Cat,Phon,FD), FM <=> FD.

%_________________________
% Move-alpha

% Head movement.
nt:{Cat,Lev,ID,F}/t:{C,lex:Word,ID1,F1} ) :-
non_lexical(Cat),lexicon(C,Word,F1),\+ C = Cat,moveable(C,head).
move_a( nt:{C1,[m(+),s(-),c(-)],ID,FM}/
[nt:{C,[m(+),s(-),c(-)],IDS,FS},nt:{C1,[m(-),s(+),c(-)],ID,FD}] ) :-
non_lexical(C1), spec(C,C1,FS), moveable(C,phrase), FM <=> FD.

%_________________________
% Trace Theory

nt:{Cat,[m(-),s(+),c(+)],ID,FM}/t:{Cat,ec:[p(-),a(A)],ID,FD} ) :-
non_lexical(Cat), FM <=> FD.
nt:{Cat,Lev,ID,FM}/t:{Cat,ec:[p(-),a(A)],ID,FD} ) :-
moveable(Cat,head), lexical(Cat), FM <=> FD.
move_a( nt:{Cat,[m(-),s(+),c(+)],ID,FM}/t:{C,ec:[p(-),a(A)],ID,FD} ) :-
         non_lexical(Cat), moveable(C,head), FM <=> FD.
nt:{Cat,[m(+),s(-),c(-)],ID,FM}/t:{Cat,ec:[p(-),a(A)],ID,FD}) :-
moveable(Cat,phrase), FM <=> FD.

%_____________________________________________
% Chain-Representations:
c:{Cat,Phon,Level/Pos,ID,Ftrs}
%
c:{..} # c:{..}

?- op(550,xfy,'#').

%theta_criterion(c:{Cat,ec:_,head/away,_,_} # []) :- fail.
theta_criterion(c:{Cat,_,_/home,_,_} # []).

case_filter(c:{n,_,phrase/abar,_,F1} # c:{n,_,phrase/POS,_,F2}) :-
         case_markable(POS),!,
wait_exist(F2,case:yes).

case_filter(c:{n,_,phrase/apos,_,F1} # c:{n,_,phrase/POS,_,F2}) :-
```

```
          case_markable(POS),!,
          wait_exist(F2,case:no).

case_filter(c:{n,_,phrase/POS,_,Ftr} # head_of_chain) :-
case_markable(POS),!,
          wait_exist(Ftr,case:yes).

:- wait wait_exist/2.

wait_exist([],Ftr) :- !,fail.
wait_exist([X|Rest],A:V) :-
nonvar(X),X=A:_,!,X=A:V.
wait_exist([X|Rest],Ftr) :-
nonvar(X), wait_exist(Rest,Ftr).

case_markable(apos).
case_markable(home).

%____________________________________________________
% Some fundamental definitions:

position(Mother/[Sat,Daught],Status) :-
branch_right(Mother/[Sat,Daught]),
pos_status(Mother/[Daught,Sat],Status).
position(Mother/[Daught,Sat],Status) :-
          branch_left(Mother/[Daught,Sat]),
          pos_status(Mother/[Daught,Sat],Status).

%_________________
% Def of A-position.
% complement of a lexical
% phrase, or,
% the specifier of a verbal
% phrase (basicallly v,i).
          level(D,[_,s(+),_]),
          cat(D,C),
verbal(C).

%_________________
% Def of A-bar-position.
% simply [Spec,CP].
          cat(D,c).

%_________________
% Def of Adjunct-position.
% simply [Spec,CP].
```

```
what_pos(M/[D,S],spec) :-
          level(M,[m(+),s(-),_]),
          level(D,[m(-),s(+),_]).
what_pos(M/[D,S],comp) :-
level(D,[_,_,c(+)]).

branch_right(Node/[_,Right]) :-
          compatible(Node,Right/_).
branch_left(Node/[Left,_]) :-
          compatible(Node,Left/_).

%_________________________
% compatible: ensure two nodes are
of the same projection.

compatible(T:{Cat,_,ID,_},T:{Cat,_,ID,_}/_).


%_________________________
% Some general definitions.

verbal(C) :- member(C,[i,v,p]).

lexical(C) :- member(C,[n,v,a,p,d]).
lexical(C) :- cat(C,Cat),nonvar(Cat),lexical(Cat).

non_lexical(C) :- member(C,[c,i]).
non_lexical(C) :- cat(C,Cat),nonvar(Cat),non_lexical(Cat).

cat(Type:{Cat,_,_,_},Cat) :- member(Type,[t,nt]).
cat(Type:{Cat,_,_,_,_},Cat) :- member(Type,[c]).

level(nt:{_,Lev,_,_},Lev).

word(t:{_,Word,_,_},Word).
```

# Appendix F

# English Parameters and Lexicon

```
%%______________________________________________
%%
%% Module: English Lexicon and Parameters
%% Paradigm: knowledge base
%% Description: Contains all the language specific parameter
settings, and the lexicon for English.
%%______________________________________________
%% Written: 29 May 1991 Revised:
%% (c) 1991 M. W. Crocker, University of Edinburgh
%%______________________________________________

lexicon(Cat,Word,Ftrs) :-
lex(Cat,Word,Fs),
append(Fs,_,Ftrs).

%_____________________________________
% Parameters for English

%__________________
% X-bar Parameters
%
% can remove 'comp' settings once
% theta module is in place.

spec(d,n,_). spec(n,v,_). spec(n,i,_).
spec(n,c,FS) :- wait_exist(FS,wh:yes).
spec(p,c,_).

comp(n,p). comp(i,c). comp(v,i).
comp(X,v) :- verify_comp(X,v),!.
```

```
:- wait verify_comp/2.
verify_comp(n,v). verify_comp(p,v).
% verify_comp(i,v).
verify_comp(c,v).

comp_features(C,Comp) :- member(C,[p,c,i]),!,Comp=c(-).
comp_features(Cat,c(_)).

% spec_pos(Category,SpecPosition).
spec_pos(_,initial).

% head_pos(Category,HeadPosition).
head_pos(_,initial).

%______________________
% Movement Parameters

moveable(v,head). moveable(i,head).
moveable(n,phrase). moveable(p,phrase).

%______________________
% Case Parameters

assigns_case(v,comp,right).
assigns_case(p,comp,right).
assigns_case(i,spec,left).


%____________________________________________
% Lexicon

lex(d,the,[num:N]).
lex(d,a,[num:sing]).

lex(n,john,[wh:no,num:sing,proper:yes]).
lex(n,mary,[wh:no,num:sing,proper:yes]).
lex(n,boy,[wh:no,num:sing]).
lex(n,boys,[wh:no,num:pl]).
lex(n,girl,[wh:no,num:sing]).
lex(n,girls,[wh:no,num:pl]).
lex(n,who,[num:N,wh:yes]).
lex(n,what,[num:N,wh:yes]).
lex(n,book,[wh:no,num:sing]).
lex(n,table,[wh:no,num:sing]).

lex(p,on,[trans,grid:[patient/int]]).
```

```
lex(p,in,[trans,grid:[patient/int]]).
%lex(p,to,[grid:[patient/int]]).
lex(p,under,[trans,grid:[patient/int]]).
lex(p,where,[wh:yes]).

lex(v,kiss,[trans,grid:[agent/ext,patient/int]]).
lex(v,kissed,[trans,grid:[agent/ext,patient/int]]).
lex(v,see,[trans,grid:[agent/ext,patient/int]]).
lex(v,saw,[trans,grid:[agent/ext,patient/int]]).
lex(v,put,[trans,grid:[agent/ext,patient/int,location/int]]).
lex(v,read,[trans,grid:[agent/ext,patient/int,{concern/int}]]).
lex(v,know,[trans,grid:[agent/ext,proposition/int]]).

lex(i,will,[trans,grid:[action/int]]).
lex(i,does,[trans,grid:[action/int]]).
lex(i,did,[trans,grid:[action/int]]).
lex(i,do,[trans,grid:[action/int]]).
lex(i,to,[grid:[action/int]]).

lex(c,that,[]).

%______________________________________________________
% CSR(Role/Type, Arg): defines the "canonical structural realisation" for
%         a particular Role/Type.
%
% Language: English

csr(agent/ext, Arg) :-
        Arg:arg:cat <=> n,
        Arg:pos <=> ext,
        Arg:role <=> agent.

csr(agent/int, Arg) :-
        Arg:arg:cat <=> p,
        Arg:arg:rel <=> by,
        Arg:pos <=> int,
        Arg:role <=> agent.

csr(patient/T, Arg) :-
        Arg:arg:cat <=> n,
        Arg:pos <=> T,
        Arg:role <=> patient.

csr(location/T, Arg) :-
        Arg:arg:cat <=> p,
        Arg:arg:rel <=> W, member(W,[on,under,in]),
```

```
        Arg:pos <=> T,
        Arg:role <=> location.

csr(inst/T, Arg) :-
        Arg:arg:cat <=> p,
        Arg:arg:rel <=> with,
        Arg:pos <=> T,
        Arg:role <=> inst.

csr(action/T, Arg) :-
        Arg:arg:cat <=> v,
        Arg:pos <=> T,
        Arg:role <=> action.

csr(proposition/T, Arg) :-
        Arg:arg:cat <=> i,
        Arg:pos <=> T,
        Arg:role <=> proposition.
```

# Appendix G

# German Parameters and Lexicon

```
%%____________________________________________
%%
%% Module: German Lexicon and Parameters
%% Paradigm: knowledge based
%% Description: The language specific parameters setting and
lexicon for German
%%____________________________________________
%% Written: 29 May 1991 Revised:
%% (c) 1991 M. W. Crocker, University of Edinburgh
%%____________________________________________

lexicon(Cat,Word,Ftrs) :-
lex(Cat,Word,Fs),
append(Fs,_,Ftrs).

%____________________________________
% Parameters for German

%__________________
% X-bar Parameters
%
% can remove 'comp' settings once
% theta module is in place.

spec(d,n,_). spec(n,v,_). spec(n,i,_). spec(n,c,_). spec(p,c,_).

comp(n,p).  comp(i,c). comp(v,i).
comp(X,v) :- verify_comp(X,v),!.
:- wait verify_comp/2.
verify_comp(n,v). verify_comp(p,v).
```

```
verify_comp(c,v).

comp_features(C,Comp) :- member(C,[p,c,i]),!,Comp=c(-).
comp_features(Cat,c(_)).

% spec_pos(Category,SpecPosition).
spec_pos(_,initial).

% head_pos(Category,HeadPosition).

head_pos(c,initial).
head_pos(p,initial).
head_pos(n,initial).

head_pos(v,final).
head_pos(i,final).

%________________________
% Movement Parameters

moveable(v,head). moveable(i,head).
moveable(n,phrase). moveable(p,phrase).

%________________________
% Case Parameters

assigns_case(v,comp,left).
assigns_case(p,comp,right).
assigns_case(i,spec,left).

%_______________________________________________
% Lexicon

lex(d,der,[num:N]).
lex(d,die,[num:N]).
lex(d,das,[num:N]).
lex(d,dem,[num:N]).
lex(d,ein,[num:sing]).
lex(d,eine,[num:sing]).
lex(d,einer,[num:sing]).
lex(d,einen,[num:sing]).
lex(d,einem,[num:sing]).

lex(n,peter,[num:sing,proper:yes]).
lex(n,maria,[num:sing,proper:yes]).
lex(n,junge,[num:sing]).
```

```
lex(n,jungen,[num:pl]).
lex(n,madchen,[num:sing]).
lex(n,madchenen,[num:pl]).
lex(n,wer,[num:N,wh:yes]).
lex(n,was,[num:N,wh:yes]).
lex(n,buch,[num:sing]).
lex(n,tisch,[num:sing]).

lex(p,auf,[trans,grid:[patient/int]]).
lex(p,mit,[trans,grid:[patient/int]]).
lex(p,in,[trans,grid:[patient/int]]).
lex(p,unter,[trans,grid:[patient/int]]).
lex(p,wo,[wh:yes]).

lex(v,gesehen,[trans,grid:[agent/ext,patient/int]]).
lex(v,sehen,[trans,grid:[agent/ext,patient/int]]).
lex(v,sah,[trans,grid:[agent/ext,patient/int]]).
lex(v,gab,[trans,grid:[agent/ext,theme/int,patient/int]]).
lex(v,gegeben,[trans,grid:[agent/ext,theme/int,patient/int]]).
lex(v,gelegt,[trans,grid:[agent/ext,patient/int,location/int]]).
lex(v,legt,[trans,grid:[agent/ext,patient/int,location/int]]).
lex(v,lesen,[trans,grid:[agent/ext,{patient/int},{concern/int}]]).
lex(v,gelesen,[trans,grid:[agent/ext,patient/int]]).
lex(v,weiss,[trans,grid:[agent/ext,proposition/int]]).

lex(i,wird,[trans,grid:[action/int]]).
lex(i,werde,[trans,grid:[action/int]]).
lex(i,hat,[trans,grid:[action/int]]).
lex(i,haben,[trans,grid:[action/int]]).
lex(i,zu,[grid:[action/int]]).

lex(c,dass,[]).

%____________________________________________________________
% CSR(Role/Type, Arg): defines the "canonical structural realisation" for
%          a particular Role/Type.
%
% Language: German

csr(agent/ext, Arg) :-
        Arg:arg:cat <=> n,
        Arg:pos <=> ext,
        Arg:role <=> agent.

csr(agent/int, Arg) :-
        Arg:arg:cat <=> p,
```

```
        Arg:arg:rel <=> bei,
        Arg:pos <=> int,
        Arg:role <=> agent.

csr(patient/T, Arg) :-
        Arg:arg:cat <=> n,
        Arg:pos <=> T,
        Arg:role <=> patient.

csr(theme/T, Arg) :-
        Arg:arg:cat <=> n,
        Arg:pos <=> T,
        Arg:role <=> theme.

csr(location/T, Arg) :-
        Arg:arg:cat <=> p,
        Arg:arg:rel <=> W, member(W,[auf,unter,in]),
        Arg:pos <=> T,
        Arg:role <=> location.

csr(inst/T, Arg) :-
        Arg:arg:cat <=> p,
        Arg:arg:rel <=> mit,
        Arg:pos <=> T,
        Arg:role <=> inst.

csr(action/T, Arg) :-
        Arg:arg:cat <=> v,
        Arg:pos <=> T,
        Arg:role <=> action.

csr(proposition/T, Arg) :-
        Arg:arg:cat <=> i,
        Arg:pos <=> T,
        Arg:role <=> proposition.
```