Specifying Syntactic Structures

edited by
Patrick Blackburn
and Maarten de Rijke

Contents

1 **On Reducing Principles to Rules** 1
Marcus Kracht

1

On Reducing Principles to Rules

Marcus Kracht

ABSTRACT. According to the research principles of the tradition established by Noam Chomsky, one should always try to get rid of construction specific rules. In *Bare Phrase Structure* he has carried this out to the extreme for the basic generative component now known as X-bar-syntax. Based on this example, we will examine the dialectics between specific rules and conditions on rules or *principles*. We will show that there is a two-way reduction, one from rules to conditions on rules, and the other from conditions on rules to specific rules. As an effect of this reduction, a particular linguistic theory can be presented alternatively as a set of rules or as an axiomatic extension of the logical theory of all phrase structure trees — or in fact a suitable mixture of the two. Although this reduction is a purely formal one — and therefore less interesting for a linguist subscribing to the Principles and Parameters approach — it offers the possibility to draw on a large array of results both from logic and from ordinary formal language theory.

1 Introduction

In physics, and in science generally, we seek simple and elegant laws; laws that hold without exception and which each explain a great variety of facts. Chomsky, and many other linguists with him, have stressed that linguistics should not be different in this respect. Of course, if linguistics were only a so-called *soft* science we might not be able to expect laws to be 100% valid — we might have to allow for exceptions. However Chomsky has insisted that linguistics deals with a real object, the *language faculty*, and that we can expect hard results about this object. Chomsky 1980 successfully defends the place of linguistics in science and its particular mode of inquiry. Furthermore, in Chomsky 1986b he has reminded us of the fact that there is a difference between language the way it presents itself to us (for example, in the form of grammatical judgments) and the

Specifying Syntactic Structures
P. Blackburn and M. de Rijke, eds.

way it is realized and manipulated internally. The former has been called E-language, the latter I-language.

Probably few people dispute that this distinction is reasonable and that we can have nontrivial results about the I-language. However, quite often the insistence that it is I-language that we should be interested in has been abused to shake off critics. One example that concerns us here is the reduction of principles to rules as undertaken in Kracht 1995b. Remarking on this technique Chomsky writes (p. c.)

> [...] there is no nontrivial concept of weak generation, hence of weak generative capacity, for any theory of language that has been seriously proposed. Nor is there any explanatory gap that would be filled if such a notion were constructed. That is to say, there is no reason to believe that any such notion has the slightest relevance to natural language.

This passage at least implicitly uses the abovementioned distinction. The argument is that if the I-language does not make explicit use of a property then this property is of questionable interest in linguistics. I agree with the first two sentences, but not with the last. Any theory about I-language (or natural language for that matter) has to meet the facts, and these facts are — in standard practice — facts about E-language. Any attempt to systematize the latter will help in this process of evaluation. Even if we do not make any attempts to elucidate the structure of I-language, the description of E-language may be worth pursuing. What is more, we should welcome any result that mediates between various theoretical descriptions of language, be they of I-language or simply of E-language. The reason is the following. If in the course of inquiry we have to discuss problems concerning our theories, how do we settle the dispute? In the absence of any other knowledge we will probably try to evaluate the theories as far as possible, see where they differ and try to see which one is correct. To do this requires skill. For how do we know the consequences of a theory, if, as has been pointed out on numerous occasions, GB is so highly modular that small changes in one component can have bewildering global effects? It might be trivial to evaluate the effects given a specific construction, but who can foresee the *precise overall effect* of such changes? And who knows the exact limits of this enterprise in full? These are serious questions, and an answer is called for.

Perhaps this is not a problem for linguists. After all, linguists are interested mainly in empirical questions. Maybe such problems should be relegated to mathematics. However, Chomsky claims (p. c.)

> The problem of formalization is, in my opinion, not a very interesting one. It's an easy matter to formalize a theory, but also a pointless exercise, which is why it is not undertaken in the natural sciences (even in mathematics, until recently). When the need arises within a research program, the gap can easily be filled (as

it was in mathematics, when it became useful). The reason why it is pointless to formalize is that in doing so, we must make arbitrary decisions in areas where we do not know the answers, and that is simply a waste of time — though easy enough to do.

Although I half agree with these remarks, there is problem. In these remarks Chomsky essentially identifies *mathematics* with *formalization*. By doing so, Chomsky simply erects a strawman which he then successfully dismantles.

It's easy enough, though mistaken, to identify mathematics with formalization. Nearly all mathematics is formal in some sense — after all, typically lots of formulae get used. And if mathematics is not formal then what else is? However, formalization within mathematics, in the sense referred to by Chomsky in the above quotation, was a special development of the last century, aimed at placing mathematics on undisputable foundations. This did change mathematics since it tightened the requirements of what is to be counted a proof. Nonetheless, in everyday mathematics formalisation, in the strict sense, is absent: only *precision and clarity* are required. Much work in linguistics lacks both, and it is a noble task to look for remedies.

This distinction between formalisation and precision is well worth emphasizing. Strict formalization is often seen in linguistics. There is still the idea around in linguistics and other sciences that more symbols mean more precision. This is both false and dangerous, and Chomsky is right to criticize it.

However, mathematicians do not present their assumptions and theorems as formal systems. They would be unreadable like this. Mathematics is *not* the art of formalization. The latter is only a craft within it. Rather, mathematics is a special form of inquiry, which explores the absolutely necessary conclusions of certain assumptions. If presented with a theory a mathematician would ask questions like *What are the models of this theory?* or *To what theory might this one be equivalent under suitable interpretation?* or *What are the consequences of this theory?* Usually, as is demonstrated by group theory for example, a system of classifying the structures of interest is developed as a result of mathematical investigations — thus one learns more and more about the structure of these objects and the power of the theory. It is in this sense that mathematics is extremely useful for all sciences — including linguistics.

This paper is not intended to be an abstract debate on the role of mathematics in linguistics. Rather, I will illustrate the utility of mathematics by discussing in some detail the special relation between *principles* and *rules*. I will show how rules can be reduced to principles, and how principles can be reduced to rules. The prime example used throughout is *X-bar-syntax*.

Lately, in Chomsky 1995 we have witnessed the total elimination of specific rules in favour of principles. We will analyse this particular solution and point at some difficult problems concerning the overall strategy. We will show that the particular form of Principles and Parameters Theory deriving from GB has its limits in that it cannot reduce certain phenomena to principles. [1] In short, mathematics will provide us with a high level view of the architecture of transformational grammar, and will enable us indicate some of its problematic features.

This paper describes research that has been carried out in part in the Innovationskolleg INK II/A12 'Formale Modelle kognitiver Komplexität' funded by the DFG. I wish to thank Noam Chomsky, Carsten Grefe, Werner Simon, Juan Uriagereka and Christian Wartena for discussions on this matter.

2 X-bar-Syntax

We begin by recalling X-bar-syntax in the particular form it reached in the 1980s with the work of Stowell (see Stowell 1981). X-bar-syntax provides a rudimentary analysis of sentences. It consists of a list of primitive categories, n, v, infl, comp, and nowadays a variety of functional categories such as $\mathsf{agr{-}s}$, neg and so on. Their number is limited and can be seen as being derived from properties of the lexical items. Languages may or may not vary in their inventory of basic categories. Each category can project a phrase. By that we mean, roughly, that each category symbol not on a leaf of a syntactic tree can be traced to a leaf of the tree. [2] The totality of nodes thus corresponding to a given leaf form a subset of the tree which is linearly ordered by the dominance relation. This is called the *projection line* of the lexical item or leaf. Each node within that line is a *projection* of the leaf. In addition, projections have different *levels*, three for each category. Going up the tree, levels may not decrease. [3] Notation and terminology is as follows. With comp being a basic category, comp^0 is a zero-level projection, also called the *lexical* or *minimal* level projection, comp^1, comp' or $\overline{\mathsf{comp}}$ the *first-level* or *intermediate* projection and $\overline{\overline{\mathsf{comp}}}$, comp'' or compp the *second-level* or *phrasal* projection. With categories getting more complex, we will follow the GPSG-convention of writing them in the form of attribute-value pairs. The category $\overline{\overline{\mathsf{comp}}}$ would be written

[1] A note on terminology. There is a multitude of approaches that have been advanced by Chomsky. I use the cover term *transformational grammar* to refer to all of these theories. This includes in particular GB, the Principles and Parameters framework and the Minimalist Program.

[2] This is a nontrivial fact corresponding to the requirement of strict endocentricity of constructions. But this is only an exposition of X-bar syntax, not an explanation of why it is the way it is.

[3] Again, this is a nontrivial assumption.

in this notation as follows.

$$\begin{bmatrix} \text{CAT} & : & \mathsf{comp} \\ \text{LEVEL} & : & \mathsf{2} \end{bmatrix}$$

We assume the following X-bar rules, in traditional notation:

$$\begin{array}{llllllllll}
(u) & X'' & \to & X' & & X' & \to & X^0 & \\
(pp) & X'' & \to & Y'' & X'' & X'' & \to & X'' & Y'' \\
(pi) & X'' & \to & Y'' & X' & X'' & \to & X' & Y'' \\
(ii) & X' & \to & Y'' & X' & X' & \to & X' & Y'' \\
(iz) & X' & \to & Y'' & X^0 & X' & \to & X^0 & Y'' \\
(zz) & X^0 & \to & Y^0 & X^0 & X^0 & \to & X^0 & Y^0
\end{array}$$

This schema is just *one* of the possibilities. Specifically, we have chosen a form of X-bar-syntax that holds throughout the derivation, not just at D-structure. We remain uncommitted with respect to this choice. The variables X and Y each stand for any particular choice of basic category. [4]

On the basis of these rules we can define what I want to call the *centricity role* of a node. There are four types of centricity roles, namely *head*, *complement*, *specifier* and *adjunct*. In each rule, one of the daughters is called the *head*. The head is of the same category as the mother. The other daughter (if there is one) is called a *complement* if one of the rules (iz) has been applied, *specifier* if one of the rules (pi) has been applied, and *adjunct* in all other cases.

Immediately, a problem appears. Nothing prevents us from choosing $X = Y$, in which case we cannot tell which is the head. [5] However, we can also move to a fully articulated X-bar syntax by introducing explicit *centricity* roles. In attribute-value notation we have a feature CENT with values head, comp, spec, adct. To keep notation short, we will use booleans. First one needs to introduce the centricity grammar. It is based on the *head grammar*, which itself only distributes the headness role.

$$\begin{array}{llllllllll}
(u) & \top & \to & \mathsf{head} & & & & & \\
(b) & \top & \to & \neg\mathsf{head} & \mathsf{head} & \top & \to & \mathsf{head} & \neg\mathsf{head}
\end{array}$$

(See Kracht 1995b for more details.) Here, $\top$ is a constant for the unit, that

[4] A notational point. Chomsky 1986a uses in his exposition the variable X to denote both X and Y, Sternefeld 1991 lets Y'' appear to both sides in a rule, but hastens to add that this means it can appear on either side but not on both. Both are very unhelpful ways of stating the facts. Incidentally, many have followed Chomsky in using X just as a piece of ink to which the level symbols are tagged rather than as genuine variables.

[5] Fanselow 1991 is the only place where I have found this problem being discussed. In his view there can be only one head, and to identify it means that one has to resort to a more complex definition. If we assume that there are no base generated adjuncts of phrases and minimal projections then the problematic case appears only at S-structure and LF. It has then been produced by movement. It is clear that the adjoined constituent cannot be called a head. As long as the derivational history can be recovered it is then in principle possible to say which daughter is the head and which one is not.

is, the element which is always true. As a consequence of this notation, the value of the mother is totally unimportant for determining the centricity role of a node. The roles comp, adct and spec are added by stating that all four centricity roles are exclusive. This completes the definition of the centricity grammar. In the rules (b), we are allowed to specialize ¬head to any of the three roles adct, comp and spec. Thus, effectively the centricity grammar makes no real distinction between adjuncts, complements and specifiers.

On top of the centricity grammar we can now define the X-bar syntax. Namely, we say that (i) the category value is passed from the mother to the head daughter, (ii) the levels are defined as follows: the level goes one down if there is no adjunct, otherwise it is the same at the mother and the head daughter. The non-head has level 0 if it is adjunct to a zero-level head, and is phrasal otherwise. We can see that with the centricity grammar introduced, the levels are redundant. Notice that GPSG and HPSG make use of the centricity roles, although it seems that GPSG would rather regard the notion of a head as epiphenomenal. The complete phrase structure rules (without levels), when put into (order irrelevant) attribute value notation, are like this:

$$\begin{bmatrix} \text{CENT} & : & 1 \\ \text{CAT} & : & \alpha \end{bmatrix} \to \begin{bmatrix} \text{CENT} & : & \mathsf{head} \\ \text{CAT} & : & \alpha \end{bmatrix} \quad \begin{bmatrix} \text{CENT} & : & \neg\mathsf{head} \\ \text{CAT} & : & \beta \end{bmatrix}$$

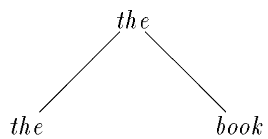
In Chomsky 1995 a bold attempt is made at eliminating X-bar syntax by apparently reducing it to its bare minimum. The attack is chiefly directed towards the level grammar. We have already seen that it is in principle dispensible, but it is worthwile to see Chomsky's solution here. If we put adjunction aside for the moment, we see that X-bar syntax mainly provides a reason for there being a single complement and a single specifier. We can achieve the same by saying that in a projection line the lowest node is lexical and the highest one is phrasal. Assuming that we can always identify heads (for example by banning the rules $X^0 \to X^0\ X^0$ and $X'' \to X''\ X''$) we can identify the level of each node. However, since the levels are not intrinsic characteristics of single nodes, but are in fact relational, the level of a node can only be recovered given a suitably large context. If we want to abandon the restrictions posed by the levels on how to rewrite these level-projections, we encounter the problem that we have to stipulate that without adjunction any projection line has length three. (Kayne 1994, by the way, achieves this by pairing linear order with asymmetric c-command.)

Chomsky is now led to assume that this restriction must be lifted. Given that eventually there will be no explicit distinction between specifier, (adjunct and) complement, this is most sensible, because it allows for the standard recursion within phrases (for example, it allows, contra Kayne, any number of base generated X'-adjuncts, to use the old terminology). Furthermore, every rule must now be branching. Chomsky now codes trees as sets in the following way. We define a function t on binary branching trees with explicit marking of a head, and a function h which will define the head of the entire tree. If the tree τ consists only of a single node, carrying the lexical entry α, then $t(\tau) := \alpha$ and $h(\tau) := \alpha$. This is the ground clause; for the inductive clause suppose that we have a tree τ consisting of two immediate subtrees σ_1 and σ_2. We now need to know which of σ_1 and σ_2 is the head. This is given to us by the assumption that the heads are explicitly marked. Suppose the node y_1, the root of σ_1, is the head. Then

$$\begin{array}{rcl} t(\tau) & := & \{h(\tau), \{t(\sigma_1), t(\sigma_2)\}\} \\ h(\tau) & := & h(\sigma_1) \end{array}$$

There is the possibility that both terms are equal, in which case the set reduces to $\{\alpha, \{t(y_1)\}\}$. On the assumption that trees are strictly binary branching, we can nevertheless recover the original tree. However, this does not seem to be an intended effect. If σ_2 is the head, then $h(\tau)$ has to be defined as $h(\sigma_2)$. Furthermore, since in the bare structure we need no levels, it is possible to use the head as a label. Thus for the structure *the book* we have the following code

$$\{the, \{the, book\}\}$$



Notice that nothing is assumed about the relative order of the elements. Thus, with the set notation, everything is derived from the lexicon.

Now, in order to incorporate adjunction, Chomsky uses a trick. Rather than using the term $\{h(\sigma_1), \{t(\sigma_1), t(\sigma_2)\}\}$ he now defines (for σ_1 the head of τ)

$$t(\tau) := \{\langle h(\sigma_1), h(\sigma_1)\rangle, \{t(\sigma_1), t(\sigma_2)\}\}$$

Thus, rather than using the lexical entry of the head of the construction as a marker of the head of construction, he just uses the pair $\langle h(\sigma_1), h(\sigma_1)\rangle$. Given the typical definition of a pair $\langle x, y\rangle$ as the set $\{x, \{x, y\}\}$ we get

$$t(\tau) = \{\{h(\sigma_1), \{h(\sigma_1)\}\}, \{t(\sigma_1), t(\sigma_2)\}\}$$

This encoding can also be used to give labels to each node in a tree; simply identify the label of x with $t(\downarrow x)$, where $\downarrow x$ is the constituent rooted at x. Seen this way, each node label is actually also a code of the structure of the constituent underneath that node. This explains why movement can never go down in a tree, only upwards. (This was pointed out to me by Juan Uriagereka, who refers to this as the *Grandmother Problem.*) Moving down would mean inserting some nodes into the tree, whereby all labels further up would fail to correctly encode the structure of the tree. Thus we have an explanation of why already built structure resists manipulation. This explanation, however, is conditional on an analysis of movement as a process that takes place in time, as does real movement. As much as I agree with this — since it means that the term *movement* is well chosen as a metaphor — it is precisely that interpretation that Chomsky rejects. We will not pursue such issues further here. [6]

3 The Human Computational System

In order to understand the motivation of *Bare Phrase Structure* one has explain the framework in which it has been put, namely the *Minimalist Program* of Chomsky 1993. There Chomsky explores the consequences of the assumption that a linguistic theory should employ only those concepts that follow — as Chomsky puts it — with *virtual necessity* from the fact that the language faculty mediates between sound and meaning. The difference between the Minimalist Program and the aims of Chomsky 1995 is that in the latter he is more concerned with minimizing the additional components of the representation itself rather than the global architecture of the theory. The latter contains now two so-called *interfaces*, namely (from syntax to) the *articulatory-perceptual* system (A-P) and (from syntax to) the *intentional-conceptual* system (C-I). Ideally, what we observe as nontrivial properties of the grammar should be consequences of conditions imposed by the two systems. Chomsky assumes that there is one and only one lexicon, that there is a single computational system (C_{HL}) and that both together derive representations that are fed to the two systems A-P and C-I. D-structure and S-structure have been abandoned, mainly because by definition a level of representation must serve the purpose of checking nontrivial properties and if there is nothing to check at these levels, they cannot be called levels of representation. It is formally possible (and sometimes also useful for comparison) to think that D- and S-structure are there nevertheless, but the difference is that nothing is checked there. From an explanatory point of view, however, we would loose something. Namely, it is a specific conceptual advance that in the Minimalist Program conditions on representations must be motivated by external demands. The computational system is free to derive any structure it can, but those which violate output conditions, that is, those which give rise to problems at the interface, are filtered out because the systems to which they are sent are unable to handle them.

[6] Another relevant question is why the representation should be in terms of sets, as presented here. Such labellings should be excluded on economical grounds since they code what is present in the structure anyway.

The computational system operates as follows. It draws from the lexicon a multiset (Chomsky uses the term *numeration*, but this is practically the same thing) and composes the items into trees via an operation called *Merge*, performing on its way some other nontrivial operations (forming chains, moving items, and so on). There is no point at which the former D-structure can be said to be present. In that sense it has really been abandoned. Thus, C_{HL} may draw the multiset $\{reads, the, book\}_m$ [7] and then form a tree-term

$$\{the, \{the, book\}\}$$

identifying *the* as the head of the phrase (due to properties derived from the lexical entry itself). It may then continue to incorporate the verb into the tree to yield

$$\{reads, \{reads, \{the, \{the, book\}\}\}\}.$$

The notation leaves a lot of things unseen. The entry for *book* contains extra features, as do the verb and the determiner. In particular, all items are inserted with full morphological specification, for example, for case, gender etc. (though that is not a necessary assumption in the Minimalist Program). Features come in two classes, *interpretable features* and *uninterpretable features*. Uninterpretable features (e.g. Case) must be seen as genuinely syntactic, that is, they belong to C_{HL} and must therefore disappear before the (derived) structure is presented at the interface. Interpretable features are allowed to survive the derivation, because the C-I component does not reject them (as it would with the uninterpretable ones). At the cost of misrepresenting the original proposal we can picture the elimination of features as follows. A feature FEAT can have several values, $\mathsf{val}_1, \ldots, \mathsf{val}_n$. C_{HL} operates by moving features; if that feature is connected with a constituent then that constituent is taken along with that feature.

A note on terminology is actually necessary here. Features in the Minimalist Program are by intention pairs [FEAT : val]. However, there is also talk of the Case-feature of a DP. Here, we use the following convention. Only CASE is the feature, while the pair [CASE : nom], say, is an *elementary feature-structure*. An item can get rid of its elementary feature-structure [FEAT : val_i] either by moving into the specifier of or by adjoining to the head of a functional projection carrying [FEAT : val_i]. The word order variation is derived by assuming that features are either *strong* or *weak* and

[7] The subscript m is used to identify this as a multiset. Recall that in a multiset the number of occurrences of an element matters, not however the order in which they are listed. In the present example the multiset is not different from the mere set, but in the following example, it is:

$$\{the, pope, wrote, the, book\}_m.$$

that at the point at which the structure is prepared for the articulatory interface, only elementary feature structures corresponding to weak features are stripped off (this point is called *Spell-Out*). If strong feature-value pairs remain, A-P cannot process it and the derivation is said to *crash*; that is, it terminates with no output.

The Spell-Out point can be equated with S-structure. However, in the new system Spell-Out can take place at any point in the derivation, though of course many of the attempts at Spell-Out will yield a crash. Thus, with the numeration fixed we can have many Spell-Out points. Notice, however, that in GB there need not be a single S-structure for a given D-structure. The difference with the Minimalist Program is only that in the latter the derivational component does not identify itself the S-structure; rather, the S-structure is identified indirectly as those points of Spell-Out which do not yield a crashing derivation. The identification is thus made into an interface condition.

Of course, there must be the possibility of having several S-structures for a given D-structure; otherwise it is not clear, for example, why Scrambling in Germanic languages or Rightward Movement can be optional. Notice, however, that in the original system the apparent options for Spell-Out quickly reduce to one and only one point. Thus, strictly interpreted, free word order in Germanic languages receives no explanation. The Minimalist Program, although accounting for cross language variation, has no explanation for such language internal variation.

The distinctive feature of the Minimalist Program is the use of global and abstract principles that have no equivalent in GB and the Principles and Parameters framework. First, there is the principle *Procrastinate*, which simply forbids any movement. The reason why there is movement at all is that this principle has lowest priority and can be overridden. Second, there is the principle *Greed* which says that movement must result in a benefit. Originally, *Greed* required the benefit to be for the item that is being moved. Thus, an item may only move if it itself receives a benefit from doing so. In particular, this benefit lies in getting rid of a feature. If in the course of a derivation we are before (a non crashing) Spell-Out point then we can only move if we thereby strip off a strong feature, because there is another chance after Spell-Out and the latter is preferred by *Procrastinate*. By moving an element it may serve to check a strong feature of another element, as has been noted in Ćavar and Wilder 1994, who call this phenomenon *Early altruism*. This principle is compatible with Greed. For movement need not be beneficial exclusively for the element that is being moved; side effects for other elements are allowed to occur.

To save the whole process from overgenerating Chomsky also employs a set of economy principles, namely the requirement that (i) only shortest moves may be made, and (ii) the fewest number of steps must be chosen. The first is a condition for convergence (that is, the derivation will otherwise crash). The second is covered by another principle called *Economy*. In the Minimalist Program it assumed a specific form, namely as the transderivational constraint that one must choose the shortest possible derivation.

The actual wording of these constraints or principles has given rise to a lot of confusion. There have been debates about the compatibility of these economy principles, but we will refrain from commenting on them. Suffice it to say that it is still unclear how economy can be properly formulated to meet the facts of language.

4 Principles as Axioms as Rules

I have argued at length in Kracht 1995a that it is possible to reduce principles of grammar to axioms within a special kind of modal logic which has been dubbed the *constituent logic*, **CL**. That much may in this context be granted, since axioms and principles are of similar nature. Rogers 1994 has conducted similar investigations, using a second order logic over structures with successor functions. These two approaches are similar in spirit, and there seem to be deeper connections, which are however not fully understood.

In this section I will briefly outline the syntax and semantics of **CL**. **CL** is in fact a variety of special languages, each a fragment of dynamic logic. Recall from Harel 1984 or Goldblatt 1987 that dynamic logic has two types of well-formed expressions, propositions and variables. There are denumerably many propositional variables, $var = \{p_1, p_2, \ldots\}$, denumerably many propositional constants, $con = \{c_1, c_2, \ldots\}$, the usual boolean connectives $\wedge$, $\vee$, $\neg$, $\rightarrow$, any fixed number of elementary *programs*, $prg = \{\pi_1, \pi_2, \ldots, \pi_m\}$, the program connectives $\cup, \circ, ^*$, and finally the test ? and the brackets $[-]$, $\langle - \rangle$. If α and β are programs, so is α^*, $\alpha \cup \beta$ and $\alpha \circ \beta$. If ϕ is a proposition, $\phi?$ is a program. Finally, if α is a program and ϕ a proposition, then $[\alpha]\phi$ and $\langle\alpha\rangle\phi$ are both propositions. A *frame model* is a triple $\langle f, \beta, \eta \rangle$ where $\beta : var \cup con \rightarrow \wp(f)$ and $\eta : prg \rightarrow \wp(f \times f)$. Both β and η can be extended over all propositions and programs as follows.

$$\begin{array}{lcllcl}
\beta(\neg\phi) & := & f - \beta(\phi) & \beta(\phi \wedge \psi) & := & \beta(\phi) \cap \beta(\psi) \\
\beta(\phi \vee \psi) & := & \beta(\phi) \cup \beta(\psi) & & & \\
\eta(\phi?) & := & \{\langle x, x\rangle | x \in \beta(\phi)\} & \eta(\alpha \cup \delta) & := & \eta(\alpha) \cup \eta(\delta) \\
\eta(\alpha \circ \delta) & := & \eta(\alpha) \cdot \eta(\delta) & \eta(\alpha^*) & := & \eta(\alpha)^* \\
\beta([\alpha]\phi) & := & \{x | (\forall y)(\langle x, y\rangle \in \eta(\alpha) \rightarrow y \in \beta(\phi))\} & & & \\
\beta(\langle\alpha\rangle\phi) & := & \{x | (\exists y)(\langle x, y\rangle \in \eta(\alpha) \wedge y \in \beta(\phi))\} & & &
\end{array}$$

Here, we have made use of the relation composition $\cdot$, and Kleene Star, *, which are defined as follows.

$$\begin{array}{lcl} R \cdot S & := & \{\langle x,z\rangle | (\exists y)(\langle x,y\rangle \in R \textit{ and } \langle y,z\rangle \in S)\} \\ R^* & := & id \cup R \cup R\cdot R \cup R\cdot R\cdot R \cup \ldots \end{array}$$

It would take too much to explain more about dynamic logic in general, instead I will now turn to the logic(s) **CL**. The specialty is that we assume two different types of programs, features and orientation programs. Of the latter there are only three or four, depending on necessity. They serve to look around inside the syntactic tree. We assume that our models are ordered trees (in the sense that for each node the set of daughters is linearly ordered) with each node being a feature structure consisting of possibly stacked attribute-value pairs. The tree relations are *one step down*, *one step left* and *one step right* and the fourth one is *one step up*. They are abbreviated by *down*, *left*, *right* and *up*. Instead of $\langle down\rangle$ we write $\Diamond_{\downarrow}$, and instead of $[down]$ we write $\Box_{\downarrow}$. The symbols $\Diamond_{\leftarrow}$, $\Box_{\leftarrow}$, $\Diamond_{\rightarrow}$ and $\Box_{\rightarrow}$, $\Diamond_{\uparrow}$, $\Box_{\uparrow}$ should be self-explanatory. We assume any fixed number of features $\text{FEAT}_1, \ldots, \text{FEAT}_m$ and a finite set of constants. **CL** has axioms which force the interpretation of *down*, *left* and *right* to form an ordered tree. Moreover, any formula stacking the orientation programs inside feature-programs will be false; thus we have axioms of the form $[\text{FEAT}_i \circ down]\bot$. Finally, each feature is deterministic, that is, the relations corresponding to these features are partial functions. This is easy to axiomatize.

Almost any particular grammar using some finite set of features can be rewritten as an axiomatic extension of **CL**. This has been demonstrated in Kracht 1995a. Moreover, assume that the trees of the grammars have an upper bound n on the number of daughters for a single node. Furthermore, let $\mathbf{CL}_n$ be the extension of **CL** by an axiom stating that the bound is n. Then the following is a theorem (Kracht 1995b).

Theorem 4.1 (Coding Theorem) *Let Λ be an extension of $\boldsymbol{CL}_n$ by finitely many axioms. The finite trees characterized by Λ can be generated by a context-free grammar iff Λ is axiomatizable by finitely many constant formulae over $\boldsymbol{CL}_n$.*

This theorem is called the Coding Theorem because the method by which it is proved is by enriching the language with constants, which stand for certain properties of nodes. A grammar that distributes a constant exactly at those nodes that satisfy a property P is called a *syntactic code* of P. To enforce a condition P on a given grammar G one first writes a code H for P; then H is fused with G. This fusion is a straightforwardly definable procedure. This theorem establishes a strong correspondence between two ways of characterizing the structures of a language, either directly via axioms (alias universal principles) or indirectly by a generating system (alias grammar).

It is important not to underestimate the fact that such a grammar can generate these structures directly — for a given logic, after all, it may not be clear how to build correct models. On the other hand, axiomatic extensions of **CL** are expressible also in a fragment of second-order logic with only universal prefixes for predicate variables. For the latter it is known that the validity of sentences can be checked on a finite model in polynomial time. So, at least from this perspective, **CL** is not too difficult a language.

An informal account of the ideas underlying the proof of the coding theorem may be of interest to some readers, for the proof itself is technically demanding. Following Kracht 1995b we start with a grammar that has only n-ary branching rules and is completely arbitrary, that is, all rules are admitted. [8] Basically, it is clear that any kind of local dependency (mother-daughter and sister-sister) can be encoded into the rule system by just throwing away all those rules not conforming to these dependencies. For example, if there is a law stating that a feature FEAT should be instantiated to a value at a given node iff it is instantiated to that same value at the mother, then simply throw away all rules in which mother and daughter do not agree in the value of FEAT. In order to handle the non-local dependencies we must introduce additional features. For example, if an axiom requires a node to agree with its *grandmother* with respect to FEAT then we add a new feature N-FEAT together with the axiom that the N-FEAT-value of a node should agree with the FEAT-value of the mother, and another to require that the FEAT-value of a node should be identical to the N-FEAT-value of the mother. This clearly encodes *grandmother-granddaughter-agreement*. Similarly, *spec-head-agreement* would require the addition of an extra feature. However, it is not always necessary to introduce new features. In the second case, we also have *mother-head-daughter-agreement* for the original feature, and so the dependency from the specifier to the zero-level head of the phrase proceeds through a series of local dependencies. The latter can thus be coded into the grammar directly, without additional features. Thus, what the result above comes down to is the observation that at the cost of having to add new features we can extend the range of dependencies considerably to include — for example — what is known in LFG as functional uncertainty, and many other relations.

As it turns out, there are types of principles which do not correspond to constant axioms because they force the grammar to be non-context free. A familiar construction are the cross-serial dependencies. [9] Despite of the fact that they cannot be reduced to context-free grammars, there are some

[8] Here we need the finite branching assumption. Without it the set of rules would be infinite.

[9] In Kracht 1995a it is shown how simple grammars for such cross-serial-dependencies can be reduced to axioms. In the present circumstances it is however not necessary to know how this particular solution works.

more powerful grammars to which they too can be reduced. Particularly suited in this respect are *definite clause grammars*, DCGs for short. A rule in a DCG may for our purposes just be of the form $\mathsf{F} \rightarrow \mathsf{G}_1 \ldots \mathsf{G}_m$, where F and G_i, $1 \leq i \leq m$, are feature-structures. A DCG is context-free iff it can be written using a finite number of such feature-structures. (This is not to say that it must use finitely many symbols, only that up to syntactic distinguishability it uses finitely many.) Characteristically, DCGs allow for more than that, yet they too can encode only strictly local dependencies. The same trick may therefore be used with DCGs to show that non-local dependencies can be reduced to local dependencies a the cost of introducing new features.

Theorem 4.2 *Let n be a natural number and Λ be an extension of $\boldsymbol{CL}_n$ by finitely many axioms. The finite trees characterized by Λ can be generated by a definite clause grammar.*

This result show that it is one and the same thing to speak of an axiomatic system and to speak of definite clause grammars modulo the fact that a DCG can be translated into an extension of $\mathbf{CL}_n$ for some n. However, the correspondence is not entirely straightforward. First, only finitely axiomatizable logics may be considered (which form a sublattice of the entire lattice of extensions). Secondly, there may be logics which are not determined by their finite models. So the translation from logic to grammar and back may yield a *stronger logic*. In the case of context-free grammars, however, there is no danger of that happening.

5 Transformations and their Stratification

That logics correspond to DCGs shows rather directly that GPSG- and HPSG-style grammars can be directly analysed as axiomatic extensions of the constituent logic. The same holds for LFG, even with functional uncertainty; this was demonstrated in Kracht 1995a. Thus, with the exception of transformational grammar and categorial grammar, we are able to translate back and forth between grammars and logics describing their parse trees and thus to analyse these theories and compare them on — so to speak — neutral territory.

However, it is possible to use the same methods for transformational grammar as well. All that is needed is a way of reducing movement. For, since transformations operate on structures to produce new structures, we must find a way to let a single structure represent all there is to the derivational history of a structure once generated by the base component (if there is any). So, we need to flatten out the derivations in such a way that we can talk about a single structure rather than a sequence of them. Let us call the process of reducing transformational grammars *stratification*.

There are versions of transformational grammar that are stratified, for example Koster 1986 and Brody 1995. The latter also raises many points against derivational analyses, some of which overlap with those given below. It should be stressed that the point we are making is a formal one, not one pertaining to the empirical coverage of any theory, and less also to the internal structure of the theory. We could, as does Brody, simply argue in favour or a stratified account by pointing at economy principles, which should favour a movement-free account. We will not do that; the reader is referred to the quoted book instead. Moreover, although there is talk of evidence against a stratified account, we will show that from a purely descriptive point of view there is no way to decide which of the two is more adequate, because they are reducible to each other.

We assume that transformations are simply *movement* operations. There is no deletion in the sense of removing structure, only in the sense of marking an element as phonetically empty. (Thus after deletion, the structure is still there but will not be pronounced.) Now, using a trick akin to abstract incorporation we can encode LF into S-structure. [10] All this is fairly standard. If we want to pursue the reduction to **CL** we must, however, also get rid of indices. To do that we assume that the structure of a trace is (almost) isomorphic to that of the antecedent. This was in fact proposed in Chomsky 1993 but is rejected in Chomsky 1995. Commenting on a structure where a noun is incorporated into a verb which itself has raised he notes

> In all such cases it is possible to formulate the desired result in terms of outputs. For example, in the movement-case, one can appeal to the (plausible) assumption that the trace is a copy, so the intermediate V-trace includes within it a record of the local $V \rightarrow N$ raising. But surely this is the wrong move. The relevant chains at LF are $(N, \underline{t}_N)$ and $(V, \underline{t}_V)$, and in these the locality relation by successive raising is not represented.

I fail to see why this is the wrong move. Nobody knows what the structure of traces is at LF. It can only be deduced with the help of assumptions on the global architecture of the grammatical system and the structure of representation(s). Further, notice that even if there is every reason to believe that the current program is on the right track, one should still acknowledge the importance of this reduction of indices. If the purpose is to compare different theories of grammar with respect to their empirical predictions, then in absence of any hard evidence for LF and its structure, appeal to its 'evident' properties is illegitimate. Transformationalists may be granted their different levels, whether they are in fact part of the human symbolic system or not — but they must be prepared to accept comparison

[10] Recall that S-structure is nonexistent in the minimalist program. Nevertheless, we may regard it as identical to the Spell-Out point. What is important is that we need to encode the surface alignment of the lexical items.

to theories without them. To give a crude example, all symbolic processes can be described by using natural numbers and recursive functions. This means that anything can be reduced to a game played with numbers. This however is no ontological claim; all that is claimed is the existence of a faithful encoding of a symbolic process into a process involving numbers. Mathematically this is advantageous because a recursion theorist can then go on studying recursive functions on natural numbers only. What numbers *are* is not the problem of a recursion theorist.

There are alternative ways to express the same facts which also do not make use of movement. One very attractive proposal is that we should think in terms of links or reentrant structures, as for example in HPSG. Instead of copying material we just make or add a link from the antecedent to the trace or conversely, analogous to manipulating a pointer structure in a computer. Notice that reentrant structures are more economical representations than the ones standardly used in transformational grammar.

Now assume that traces are copies, isomorphic to their antecedents with the exception that one is marked as a trace the other as antecedent. Since there are no indices there must be independent ways to identify the trace-antecedent pairs. In the GB this was possible due to two facts. One was that there were limits on the distance between trace and antecedent and the other was that within these limits many landing sites for the antecedent were ruled out as improper. The latter restrictions can be coded by additional features (such as whether the item originated in an A-position, a case-position, and so on). The former are formulated in terms of mutual command. In Kracht 1993 I have shown what relations are relevant for the usual cases in GB. I put aside a proof that all these relations can be encoded as **CL**-programs (easy) even with the by now standard distinction between nodes and segments of a category (less easy, but still straightforward). Moreover, I will also omit the proof that the fact that trace and antecedent are related to each other via a relation R remains valid throughout all subsequent levels, so that we can check whether trace and antecedent are R-related at any level, thus at S-structure. (The latter is a nontrivial property of the command relations in syntax and the way adjunction works, and a proof of that fact is not entirely straightforward.)

If all these assumptions are granted, all versions of GB can be stratified. But what about the Minimalist Program? Here we have no straightforward definitions of domains, but only certain principles on when and where to move. I can see no direct way to state an axiom to account for the requirement that a derivation must be optimal among all competing ones, nor an axiom to the effect that the fewest number of steps is being made. However, the cases for which these principles have been invented, can (in my opinion) be accounted for. Certainly the other principles, *Greed*, *Procrastinate* and *Shortest Steps* can be given an axiomatic encoding. To see this, consider first Shortest Steps. It says that an element must move to the closest possible landing site. Since landing sites are identified by their category (and some features), this requirement boils down to a command relation between trace and antecedent in the sense of Kracht 1993. Namely, for every trace there must be an antecedent within a certain command domain of the trace. In order to specify what the antecedent must look like given the trace, we say that a feature structure α is a *minor* of τ if (i) all elementary feature-value pairs of α are in τ, (ii) if τ contains strong features then some pair $[\textsc{feat} : \mathsf{val}]$ such that FEAT is strong is contained in τ but not α, (iii) some pair $[\textsc{feat} : \mathsf{val}]$ is contained in τ but not in α. (Here, α *contains* $[\textsc{feat} : \mathsf{val}]$ if $\alpha \models \langle \mathsf{feat} \rangle \mathsf{val}$. (i) encodes the fact that the antecedent is obtained by nibbling off structure from τ, (ii) is Procrastinate, (iii) is Greed.) So, finally, Shortest Steps says that given an occurrence of a structure τ in a tree that contains a feature, there exists within some command domain of τ a unique α which is a minor of τ.

One should take this formalization with a grain of salt, though. First, the domain extension has to be taken into account. That is to say, domains change in the course of derivation. Second, there are versions of Procrastinate that do not allow an encoding as proposed above. We might for example minimize over the length of derivations leading to a Spell-Out point. These issues tend to get very intricate when one looks at the possible ramifications. It is nevertheless a plausible hypothesis that while local economy principles can be replaced by axioms on the structure (or output filters), transderivational economy principles generally cannot.

6 An Analysis of Bare Phrase Structure

Let us now return to the reduction of X-bar syntax to bare phrase structure. Consider the case where we start out with an articulated X-bar grammar using categories, levels and centricity roles. Trees are binary branching, so we work over $\mathbf{CL}_2$. Let us introduce new features JOI and DIR, with values $+$ and $-$. Technically, they can be defined as follows.

$$\begin{array}{lcl}
[\textsc{joi} : +] & \leftrightarrow & \Diamond_{\downarrow}\mathsf{adct} \\
[\textsc{joi} : -] & \leftrightarrow & \neg\Diamond_{\downarrow}\mathsf{adct} \\
[\textsc{dir} : +] & \leftrightarrow & \Diamond_{\downarrow}(\mathsf{head} \wedge \neg\Diamond_{\leftarrow}\top) \\
[\textsc{dir} : -] & \leftrightarrow & \Diamond_{\downarrow}(\mathsf{head} \wedge \neg\Diamond_{\rightarrow}\top)
\end{array}$$

In words, JOI encodes whether or not a node is an upper segment, that is, resulted from adjoining to its daughter. DIR encodes the position of the head. $[\textsc{dir} : +]$ states that the head is to the left, while $[\textsc{dir} : -]$ states that it is to the right. In terms of these features it is possible to recover the category of the mother from that of its daughters, by simply passing up the category of the head daughter. [11] The directionality feature tells us which of the two is the head. Furthermore, the mother node can tell whether or not it is the result of adjunction. Thus we can recover also the phrase level in the following way. We know that a completed phrase is a non-head (if it is not the root), and from the highest non-adjoined node within it we have level 2; however, we want to express this in terms of the primitives JOI and DIR rather than the centricity roles. Note that head is definable as follows:

$$\mathsf{head}. \leftrightarrow .\neg\Diamond^{\leftarrow}\top \wedge \Diamond^{\uparrow}[\textsc{dir} : +] \vee \neg\Diamond^{\rightarrow}\top \wedge \Diamond^{\uparrow}[\textsc{dir} : -] \vee \neg\Diamond^{\uparrow}\top^{12}$$

In other words, we are a head if we are on the left and the mother has its head on the left hand side, or we are on the right hand side and the mother has its head there, too. We introduce the constants $\mathsf{u{-}seg}$ and $\mathsf{l{-}seg}$, standing for *being an upper segment* and *being a lower segment.* Then we have

$$\begin{array}{lcl} \mathsf{u{-}seg} & \leftrightarrow & [\textsc{joi} : -] \\ \mathsf{l{-}seg} & \leftrightarrow & \neg\Diamond^{\uparrow}\top \vee \Diamond^{\uparrow}(\mathsf{head} \wedge \mathsf{u{-}seg}) \end{array}$$

Now we can recover the levels by saying that *level: 2* or *phrasal* means either being the root or a non-adjoined non-head or else, on going up until the next non-head, every node must be adjoined. This can be defined in **CL**:

$$\mathsf{phr}. \leftrightarrow .\neg\Diamond^{\uparrow}\top \vee \langle((\mathsf{head}?); (\neg\mathsf{u{-}seg}?); up)^*\rangle(\mathsf{u{-}seg} \wedge \neg\mathsf{head})$$

Being lexical — or zero-level — and *being first-level* can also be defined. A zero-level category is either non-adjoined and preterminal (or terminal in Chomsky's new version, but this is really unimportant) or adjoined and travelling down adjoined nodes we will eventually meet a non-adjoined preterminal.

$$\mathsf{lex}. \leftrightarrow .\langle(((\mathsf{head}?); \neg\mathsf{l{-}seg}?); down)^*\rangle\Box^{\downarrow}\bot$$

Finally, a node is *level: 1* iff it is $\neg\mathsf{lex} \wedge \neg\mathsf{phr}$. A node may be both lexical and phrasal.

---

[11] Talk of feature passing is meant to be metaphorical. It is totally irrelevant for the result whether we talk of passing a feature up, passing it down, or whether we phrase this as principles of feature agreement.

[12] Technically, the root is herewith also classified as a head. This is done for purely technical reasons, and has no further implications.

Notice that here we have used the upgoing modality. Elsewhere in Kracht 1995a I have shown that practically speaking this modality is unnecessary. This requires explanation. Notice that the coding results really show that there is a close link between features and expressive power of the language that we use to express the principles. In other words, all we are doing is to boost up the non-logical symbolism to compensate for the fact that certain strong constructors are not present, such as Kleene Star and others. The less expressive power, the more additional basic constants or feature-constructs we must add to ensure a proper formulation of the principles. At present our aim is to use as few constants or constructors as possible, and this means that we may be forced to use a stronger language to define the otherwise explicit constants.

There still remain the centricity roles. Being adct means being a daughter of a non-lower segment. spec is the highest non-adjunct that is not a comp and comp is a non-adjunct sister of a lexical head.

$$\begin{array}{lll} \mathsf{adct} & .\leftrightarrow. & \neg\mathsf{head} \wedge \Diamond^{\uparrow}(\neg\mathsf{l{-}seg}) \\ \mathsf{comp} & .\leftrightarrow. & \Diamond^{\leftarrow}(\mathsf{head} \wedge \mathsf{u{-}seg} \wedge \mathsf{lex}) \vee \Diamond^{\rightarrow}(\mathsf{head} \wedge \mathsf{u{-}seg} \wedge \mathsf{lex}) \\ \mathsf{spec} & .\leftrightarrow. & \neg\mathsf{comp} \wedge \neg\mathsf{head} \wedge \Diamond^{\uparrow}(\mathsf{l{-}seg} \wedge \mathsf{phr}) \end{array}$$

Suppose now that the following axiom holds

$$(\dagger) \qquad \mathsf{head} \vee \mathsf{spec} \vee \mathsf{comp} \vee \mathsf{adct}$$

Then immediately we would get the same X-bar syntax back again. However, Chomsky's aim in this case is to get rid of such axioms since they put global constraints on the representation within the syntactic component so he dismisses it, with the consequence that we have a fifth type of centricity role, which might suitably be translated as *base generated intermediate adjunct*. This is because there can be any number of intermediate level nodes with feature [JOI : $-$]. It is *not* an axiom that there are exactly five of them, this follows already from the way things are set up, that is, it is a theorem. Chomsky on the other hand redefines the notion of a specifier to include this fifth type of a centricity role (and so he gets back ($\dagger$) as a theorem).

Let us summarize what we have got so far. We can code a traditional phrase structure tree as follows. (i) The category is labelled only at the leaves, (ii) all other nodes are of four types only, indicating whether they are adjoined or not and whether the head daughter is found on the left or on the right. Lifting the ban on base generated intermediate adjunction we can observe that there are absolutely no restrictions left on the representation, so any tree satisfying (i) and (ii) is legitimate.

Notice, by the way, that the way in which these trees are represented according to Chomsky 1995 is highly inefficient, because it introduces great deal of redundancy. To copy the head into the mother given that a lexicon contains hundreds of thousands of words, is a waste of memory. Instead, the two distinctive features are enough. It's true that they do not derive from the lexicon, but then neither do the sets in the set notation. I should also point out that in using sets Chomsky also gets rid of representations of order in the structure. Presumably, ordering should be derivable from other properties. I have no stance on that here, but I'd like to point out that rather than *sets*, it seems that *lists* are the primitives out of which everything is formed (at least in a computer). I do not know of any compelling argument that information is stored in the form of a set rather than a list. In the case at hand we have two options. (i) List all rules twice, (ii) disconnect linear order in the string from the physical alignment and let the latter be explicitly encoded by another feature REAL, where

[REAL : +] means that the daughters are aligned in the same way as coded, and [REAL : −] that they are aligned in opposite order.

But now let us return to the bare phrase structure. What is still worrying is the fact that we need the additional features for adjunction and head positioning. Ultimately, we may want to get rid of them, thus turning to a representation which is just a set encoding the constituent structure. The idea presented by Chomsky is to derive it from independent principles of grammar. This, however, requires a rather deep analysis of the total system. Let us illustrate how it can be performed. Suppose that we start with the multiset $\{book, the, read\}_m$. It is a fact that determiners select nouns. [13] So we project a structure $\{the, \{the, book\}\}$, assuming the difference between sets and multisets to be irrelevant at the structural level. We know where the head is, and that there is a complement. Next we project a structure

$$\{read, \{read, \{the, \{the, book\}\}\}\}$$

The head is *read*, from subcategorization. Suppose now that the complement moves for checking. Then we the following structure

$$\{\langle read, read\rangle, \{\{the, \{the, book\}\}, \{read, \{read, \underline{t}\}\}\}\}$$

Next we must assume the verb to have been raised, and so on. If we follow this process close enough and ensure that trace-antecedent relations are recoverable after stratification, then a suitably complex definition will capture the notion of head position and adjoinedness.

We shall not pursue this further but simply note that in addition to there being a genuine concern in spotting the redundancies of the system there is much of shadow boxing as well. We can make the representation compact in the way suggested, but the price we pay is that the control structure will be complex. Thus, this is the opposite of the Coding Method, where we seek to reduce control structure and explode the feature system instead. There is an antagonism between explicit symbolism and the power of the language in which principles are being formulated. Powerful languages require very complicated control structures so that they can be executed correctly, whence a reduction of the explicit symbolism increases the complexity of the control structure. This would require spelling out the control strategy rather carefully; this has not been undertaken in the Minimalist Program. On the other hand, control can be identified with structure in a rather precise way. Recall from Kracht 1995b the idea of

[13] It is unclear where to place this requirement. It is probably not part of the lexicon because it is a regular fact of language, at least if we assume it correct that NPs should be seen as DPs. The selection facts, be they universal (that is, part of UG) or language particular, as many propose (for example, Laka 1991) do not belong to the computational system qua transformational component but qua structure building process (although the two are now non-distinct).

a *memory*. We may forget as suggested the category labels in the representation, but while parsing a sentence these labels must be recorded somewhere. A rule based grammar of X-bar syntax has them present in each node (whence the redundancy), while for bare phrase structure we need to compute them. However, rather than recomputing them all the time we will simply store them in a *memory*. The amount of extra storage needed to correctly parse sentences is called the *size* of memory. This just measures in the finite case the distance from this grammar to a context-free grammar performing the same analysis. A final note. Chomsky's particular reduction of the level grammar to properties of C_{HL}, in particular the visibility properties of projections, is not without major problems, as has been pointed out by Gaertner 1995.

7 Why Rules are Sometimes Better

Most of the data discussed in the transformational tradition consists of alignment facts. Very little is being said about agreement, although at present — in the Minimalist Program — agreement seems to be given a major role in the theory. Another aspect of language receiving marginal attention in transformational grammar is coordination. Many rival theories owe their success to substantial progress in analyzing coordination (for example GPSG, HPSG, combinatorial categorial grammar). Here we will be concerned with agreement in the context of coordination and show that none of the theories has a good answer to the phenomena at hand. Worse still, it seems that this particular phenomenon cannot be incorporated into any current framework other than by making a list of possible combinations. Certainly, at a low level these facts can be accommodated in any framework — but they do not conform with any higher level principle.

Before we begin outlining the facts let us make clear that there is a fundamental difference between individual noun phrases and the coordinated structure as regards agreement. In a construction $[np_1 \ and \ np_2]_3$ the conjuncts wear their own marking directly on their sleeves; the coordinated noun phrase itself is nowhere marked directly for its agreement features, since the coordinator does not show any overt agreement. Hence, we can test the intrinsic character of the noun phrase only by the fact that it agrees with verbs or other elements in the construction. Lack of a one-to-one-correspondence between the agreement forms of the subject and the agreement forms of the verb may therefore obscure the facts somewhat.

Let us present the data, starting from the most simple facts. In English, a noun phrase such as *Mary and John* is plural, while both *Mary* and *John* itself are singular. On the other hand, *Mary or John* is (or may at least be) singular. In German, in similar constructions, I would accept both singular and plural, though the acceptance varies with the contextualization. But with *entweder ... oder* (exclusive or) judgments are clear. The coordinated noun phrase must be singular. In languages where there is in addition to singular and plural also a dual, matters are naturally a bit more complex. There may be straightforward semantic explanations for this behaviour so that one may be inclined to simply not regard this as a syntactic phenomenon. In the Minimalist Program one may see this as a problem of LF.

But let's proceed further. Baker 1992 discusses a peculiar agreement phenomenon of Mohawk, exemplified in the next two sentences, which appear as (20a,b) in the reference.

(1)	John wa'-t-hy-atskahu-'	ne	Uwari.
	John ate.masc.dual	with	Mary.
(2)	John tanu Uwari	wa'-t-hy-atskahu-'.	
	John and Mary	ate.masc.dual.	

Observe that the verb shows *dual* agreement in both cases. In their German translations (1) differs from (2) in that the verb is plural in (2) and singular in (1).

(1′)	Karl und Maria aßen.
	Karl and Mary ate.plur.
(2′)	Karl aß mit Maria.
	Karl ate.sing with Mary.

The facts of German are readily explained. The subject in (1′) is plural while singular in (2′). However, the Mohawk agreement system disconnects the notion of subject and agreement target of the verb. The consequences of this may be far reaching, but we have not considered what they will be.

There is another simple construction, in Latin, in which we can see a morphological agreement in gender between subject and verb, namely in predicative constructions.

(3)	Carolus est magnus.
	Karl is big.
(4)	Maria non est magna.
	Mary is not big.

If we insert a coordinated noun phrase we can detect from the attributive adjective which gender the noun phrase must have. It turns out that (i) if both noun phrases agree in gender, the coordinated noun phrase will agree with both, (ii) if they disagree, then masculine wins over feminine in case of humans, while in case of things the one nearer to the verb wins. [14] For

[14] See Bayer and Lindauer 1974. Facts are a bit obscured by the fact that they speak of

example we have

(5) Murus et porta fulmine icta est.
Wall-m.sg and door-f.sg by lightning struck-f.sg been-sg.
Wall and door have been struck by a lightning.

(6) In oratorem omnium ora atque oculi conversi erant.
Towards speaker of-all faces-n.pl and eyes-m.pl directed-m.pl were.
Everybody's faces and eyes were directed towards the speaker.

Indeed, in the case of things, agreement is with the nearest conjunct, not with the conjoined noun phrase. Notice in passing that these facts might be solved by the analysis provided by Munn 1993. Munn assumes that in coordinated structures the coordinator (for example *and*) projects a so-called *boolean phrase* which in turn is adjoined to the first conjunct. This solution has a number of advantages. It can explain why in Arabic the verb agrees with the first conjunct in a VSO structure. How we reconcile this with the standard facts — that coordinated structures are nonsingular even when the individual coordinated elements are singular — is left unexplained.

All constructions show a dependency between three items. For example, in the coordinated construction np_1 *and* np_2 the two noun phrases np_1 and np_2 and the conjunction are interrelated. We find that the value of the feature PLU can be calculated for English with the following list.

	−	+
−	+	+
+	+	+

It may be tempting to dismiss this problem by saying that in coordinated structures of this kind we can safely assume that the result is plural, so no real dependency is observed. However, in languages with a dual, the table is as follows if we assume that it faithfully reflects the cardinality (i. e. 1, 2, and more than two). [15]

persons and things rather than syntactic gender. To be fair, intuitions in coordination, especially with respect to number agreement, seem to be driven by semantic consideration. It may very well be the case that due to the lack of the specifically problematic constructions there are no clear syntactic judgments, and that they get replaced by semantic considerations. This applies for example to the cases of constructions such as *Johan oder Maria* (english *John or Mary*). Since there is the possibility that we mean both rather than one, we expect to find either of (i) and (ii).

(i) Johan oder Maria hat dies getan.
John or Mary has done this.

(ii) Johan oder Maria haben dies getan.
John or Mary have done this.

I find (ii) less acceptable, but with the right context it might also be ok.

[15] Note that zero has no proper home in any of the number systems. The analysis in terms of cardinality is therefore not accurate. It seems that in many languages two

	s	d	p
s	d	p	p
d	p	p	p
p	p	p	p

Here the marking of the conjunct depends on the marking of the individual noun phrases. Moreover, this relation is a ternary relation (since it is a function on two arguments) but it cannot be reduced to a binary relation. The same holds with respect to gender marking.

Nothing lets us conclude that transformational grammar can handle ternary relations — unless they arise from phrase structure rules. Let us see why this is so. We have in the pre-Minimalist tradition the basic phrase-structure component called X-bar syntax. There is no indication as to how coordination can fit into it, since coordination is basically a polymorphous construction. (See Keenan and Faltz 1985 for extensive arguments.) Moreover, apart from distributing levels and categories, X-bar syntax did not provide special mechanisms to account for agreement. Surely, most people will agree that in principle a rule based grammar could be written to encode these facts about coordination in languages, although none of the accounts I am familiar with (including that of Gazdar et al. 1985) are fully correct. The problem is that the grammars take it for granted that agreement can be regulated by identity criteria. For example, in GPSG it is part of the head-feature percolation mechanism which distributes the PLU-value from mother to head. Since in coordinated NPs both noun phrases are heads it can never happen that they are singular while the mother is plural. This gap is due to the focus on subject-verb agreement, plainly visible also in the Minimalist Program. Be it with the help of transformations which move items into checking position or without, the Minimalist Program, just as any of its predecessors, relies on an identity check with respect to agreement. This gets subject-verb agreement (almost) right but leaves the difficult cases of coordinated NPs out of discussion. To add to the complication, the cases of disagreement that Baker gives in the quoted paper show that identity checks are insufficient for those languages which treat a comitative phrase as if it was a coordinated conjunct of the subject.

Some of the problems can be solved by allowing the feature GENDER to take a *set* of gender values. The gender conflict can then be handled by realization rules on these sets. In Latin, we will let the set $\{\mathsf{masc}, \mathsf{fem}\}$, $\{\mathsf{masc}, \mathsf{neut}\}$ come out as masculine agreement, while $\{\mathsf{fem}, \mathsf{neut}\}$ will be feminine. (We disregard here for simplicity the distinction between people and things.) However, such additional symbolism would be no less less

contrasts are melted into one, namely one between one individual and several, and the other between a specified amount and an unspecified amount. Consider, for example, the quantifier *all*, which shows plural agreement.

stipulative than simple phrase structure rules if not backed by additional evidence. In fact, such evidence may come from the semantics of such coordinated structures, and in particular from a theory of plurals. [16] For on the one hand syntax requires (at least in some languages) that noun phrases have a gender, but what gender a group of different people, or a combination of people and things should have, is a highly conventional affair. Groups, however, may as a first approximation be analysed as sets, so sets of gender values qualify as syntactic reflexes from that perspective. It is at least intuitively clear that under such an interpretation a unification account of gender is incorrect. A proof of that needs careful investigation, however. [17]

There are other cases of apparent disagreement. For example, plural neuter subjects in Greek cause singular agreement on the verb (see Kaegi 1972). There are many more such examples, but on the whole they do not pose problems of a similar sort for GB. If ϕ-features are transmitted to the verb, we will simply posit a special realization rule for the verb to assure it comes out singular. Or we may deny that neuter are plural, at least as concerns the agreement feature transmitted to the verb. In the case at hand, the latter approach is at least historically correct. Namely, the ending in the nominative neuter plural was originally a feminine in a collective singular ending in -$\overline{\alpha}$.

8 Gains of the Coding Method

Proving intertranslatability between logical systems, grammars and syntactic theories does not amount to saying that syntactic theories have to be translated into rules. Nor does it mean that there is an ontological reduction of one to the other. But it does tell us how to recast theories if we need to. There is a lot to say in favour of establishing such techniques — even though Chomsky and many others see no significance in them. For a start, even if two theories are notational variants, depending on the problem at hand it may be easier to work with one rather than the other. Also, transformational grammar has persistently been a model of sentence production; the design of parsing or recognition procedures is typically thought of as a secondary problem. [18] However, in formal language theory there is a lot of serious work on parsing; connecting with

[16] The latter subject has attracted much more research and various proposals have been made (see Baker 1992 and references therein). There have been debates particularly about the role of indices in connection with agreement. The additional twist in the problems raised here comes from an interaction with basic syntactic constructions.

[17] I have recently become aware of literature in which this problem is being discussed. I have not had the time to evaluate it properly, so I have to leave a discussion of it to another occasion.

[18] Notice also the word *generative* in transformational generative grammar. To be fair, many syntactic theories begin with the production side, and it is not our aim to criticise

that work directly may be a much better strategy than starting afresh. In principle-based parsing, however, the strategy has been that even in the context-free case, principles allow much better parsing algorithms to be derived (see for example Berwick 1991). That can only be so if languages form a proper subset of the context-free languages. But from the angle of complexity all that matters is the inherent combinatorial complexity of the problem itself, abstracting away as much as possible from its particular presentation. Thus additional research into the complexity of the human language presented within GB-theory is made redundant by the possibility of giving a formal translation into rule systems. Although this translation results in an explosion of the underlying feature system, this does not affect the complexity class. Moreover, the exact factor by which the feature system gets expanded can also be determined given any theory $\mathbf{CL}_n \oplus \Phi$, Φ constant. Thus, if we end up with a context-free grammar (which is the case surprisingly often) then we can make use of the standard techniques for CFGs.

Matters tend to be less favorable if the reduction does not end up in a context-free grammar. However, not all is lost. In any case we can use a result of Fagin 1974 (which can also be found in Ebbinghaus and Flum 1995), which states that universal second-order sentences of the form $(\forall \overline{P})\phi(\overline{P})$ where $\phi(\overline{P})$ is first-order can be checked on finite structures in polynomial time. If in addition there is a constant c such that for a given input string there is a parse tree with at most $c{\cdot}n$ terminal nodes, then there is a parsing strategy which requires at most exponential time. It proceeds as follows; first guess a tree or a directed acyclic graph with at most $c \cdot n$ leaves such that the yield with null elements erased is the original string. (This is the exponential part.) Then check all principles on that structure. This result is less spectacular, but one might hope to have substantially better results if one makes use of the specific characteristics of syntactic theories (as opposed to just arbitrary axiomatic extensions). Notice that the Minimalist Program allows to deduce that there is a constant c of the required kind. This is so because each overt element has a bounded number of features, and so there is a bound on the number of movements for each element. [19]

Another problem area where the method is useful is in deciding the adequacy or the equivalence of syntactic theories. Here we say that a theory is *adequate* in a given sense if it is *equivalent to the real grammar* in that sense, where the real grammar — so to speak — is the one in our heads. Several degrees of adequacy for a grammar have been distinguished, weak

this qua research strategy. Rather, we want to emphasize that a full theory of language must also be able to explain our abilities as listeners in addition to those as speakers.

[19] For that we must work with the reentrant structures à la HPSG and not with copy movement. Otherwise the bound on the number of nodes is far too large.

adequacy if it admits the correct strings, strong adequacy if it admits the same structures, and finally explanatory adequacy. The latter is difficult to define, let alone make rigorous in a mathematical sense. Although it has been emphasized over and over that explanatory adequacy is the most important thing we should look for, the other criteria should not at all be regarded useless. It is a legitimate to ask whether two given theories differ weakly or strongly or in any other sense. If, for example, natural languages are not context-free then any syntactic theory predicting that they are is simply wrong.

In this way questions of generative complexity, perhaps in terms of a given language hierarchy, are of some importance for the actual inquiry into language. These questions, even though not of the same fundamental nature, are nevertheless surprisingly tricky to answer and will hopefully add substance to the claim that theories of language are not so easy to analyse. First of all, it is known that weak equivalence of context-free grammars is undecidable, while strong equivalence evidently is decidable. (You just have to define a bijection between the nonterminals that preserves the rules in both directions.) The first question is certainly the most interesting one because acceptability of surface strings is the most readily accessible empirical data. Strong equivalence, on the other hand, is uninteresting because it takes into account structure about which we have only indirect evidence (empty elements, unary rules). In between the two, however, we can recognize a third kind of equivalence, namely the question of whether two grammars generate the same bracketed strings. Although to analyse strings into bracketed structures requires syntactic skill, it may be considered a fairly theory-neutral affair. It has the great advantage of being decidable, as has been shown in McNaughton 1967. To generalize this in absence of context-freeness is a real challenge.

However, be it transformational grammar, GPSG or other formalisms, one of their main characteristics is the internal architecture of the system. GPSG is not just about context-free grammars, it also tries to detect global mechanisms and properties such as head- versus foot-features, metarules, defaults etc. Our system for translating between axioms and rules immediately shows a way to encode the facts once we have been able to spell them out in sufficient detail. This gives a lot of insight into the way in which the feature system of language (or UG) should be designed without going through many painful details as in Pollard and Sag 1987, to give just one example.

Take again the example of agreement in coordination. It is a simple matter to write down the axiom concerning the distribution of number features (or ϕ-features in general) once we know the facts themselves. It is equally simple to refine any grammar without ϕ-feature assignment into a grammar that distributes them correctly. But it is rather difficult to incorporate these insights into a syntactic formalism that has an articulate inner structure. The Coding Method is a method that mechanically incorporates any facts about language into a rule system. The problematic aspect of it is that it disrespects totally the architecture of the theories; in other words, it is in itself rather simple minded. On the other hand, if that is so, it is not clear how the given architecture is justifiable. For example, it is a complete mystery as to why certain features are classified as head-features and others as foot-features, and why the particular percolation mechanisms are the way they are. We have demonstrated above that PLU cannot be a head-feature although it is classified as such in the standard reference Gazdar et al. 1985. So, we need independent criteria for what is to count as a head-feature, or else give up the head-feature convention in its present form. Thus, the Coding Method may be used to determine the particular mode of percolation for a given feature and thus ultimately connect GPSG up with the empirical data in a systematic and less ad hoc fashion.

References

Baker, Mark. 1992. Unmatched Chains and the Representation of Plural pronouns. *Journal of Semantics* 1:33–74.

Bayer, Karl, and Joseph Lindauer. 1974. *Lateinische Grammatik*. München: Oldenbourg.

Berwick, Robert C. 1991. Principle–Based Parsing. In *Foundational Issues in Natural Language Processing*, ed. Peter Sells, M. Shieber, Stuart, and Thomas Wasow. 31 – 81. MIT–Press.

Brody, Michael. 1995. *Lexico–Logical Form – a Radically Minimalist Theory*. Linguistic Inquiry Monographs, No. 27. MIT Press.

Ćavar, Damir, and Chris Wilder. 1994. Word order variation, verb movement, and economy principles. *Studia Linguistica* 48:46 – 86.

Chomsky, Noam. 1980. *Rules and Representations*. Columbia University Press.

Chomsky, Noam. 1986a. *Barriers*. Cambrigde (Mass.): MIT Press.

Chomsky, Noam. 1986b. *Knowledge of Language. Its Nature, origin and Use*. New York: Praeger.

Chomsky, Noam. 1993. A Minimalist Program for Linguistic Theory. In *The View from Building 20: Essays in Honour of Sylvain Bromberger*, ed. K. Hale and Keyser S. J. 1 – 52. MIT Press.

Chomsky, Noam. 1995. Bare Phrase Structure. In *Government and Binding Theory and the Minimalist Program*, ed. Gert Webelhuth. 385 – 439. Blackwell.

Ebbinghaus, Hans-Dieter, and Jörg Flum. 1995. *Finite Model Theory*. Perspectives in Mathematical Logic. Springer.

Fagin, Ronald. 1974. Generalized first–order spectra and polynomial–time recognizable sets. In *Complexity of Computation, SIAM–AMS Proceedings*, ed. R. M. Karp, 43 – 73.

Fanselow, Gisbert. 1991. *Minimale Syntax*. Groninger Arbeiten zur germanistischen Linguistik, No. 32. Rijksuniversiteit Groningen.

Gaertner, Hans-Martin. 1995. Has Bare Phrase Structure Theory superseded X–Bar Theory? In *FAS Papers in Linguistics*, ed. Artemis Alexiadou et. al. 22 – 35.

Gazdar, Gerald, Ewan Klein, Geoffrey Pullum, and Ivan Sag. 1985. *Generalized Phrase Structure Grammar*. Oxford: Blackwell.

Goldblatt, Robert I. 1987. *Logics of Time and Computation*. CSLI Lecture Notes, No. 7. CSLI.

Harel, D. 1984. Dynamic Logic. In *Handbook of Philosophical Logic*, ed. Dov M. Gabbay and Franz Guenthner. Reidel.

Kaegi, Adolf. 1972. *Griechische Schulgrammatik*. Weidmann Verlag.

Kayne, Richard S. 1994. *The Antisymmetry of Syntax*. Linguistic Inquiry Monographs, No. 25. MIT Press.

Keenan, Edward L., and Leonard M. Faltz. 1985. *Boolean Semantics for Natural Language*. Synthese Language Library, No. 23. Dordrecht: Reidel.

Koster, Jan. 1986. *Domains and Dynasties: the Radical Autonomy of Syntax*. Dordrecht: Foris.

Kracht, Marcus. 1993. Mathematical Aspects of Command Relations. In *Proceedings of the EACL 93*.

Kracht, Marcus. 1995a. Is there a genuine modal perspective on feature structures? *Linguistics and Philosophy*.

Kracht, Marcus. 1995b. Syntactic Codes and Grammar Refinement. *Journal of Logic, Language and Information* 4:41 – 60.

Laka, Itziar. 1991. Negation in syntax: on the nature of functional categories and projections. *International Journal of Basque Linguistics and Philology* 25:65 – 138.

Lakatos, Imre. 1976. *Proofs and Refutations – The Logics of Mathematical Discovery*. London: Cambridge University Press.

McNaughton, Robert. 1967. Parenthesis Grammars. *Journal of the Association for Computing Machinery* 14:490 – 500.

Munn, Alan B. 1993. *Topics in the Syntax and Semantics of Coordinate Structures*. Doctoral dissertation, University of Maryland.

Pollard, Carl, and Ivan A. Sag. 1987. *Information-Based Syntax and Semantics, Part I.* CSLI Lecture Notes, No. 13. CSLI.

Rogers, James. 1994. *Studies in the Logic of Trees with Applications to Grammar Formalisms*. Doctoral dissertation, Department of Computer and Information Sciences, University of Delaware.

Sternefeld, Wolfgang. 1991. *Syntaktische Grenzen. Chomskys Barrierentheorie und ihre Weiterentwicklungen*. Opladen: Westdeutscher Verlag.

Stowell, Tim. 1981. *Elements of Phrase Structure*. Doctoral dissertation, MIT.

Tappenden, Jamie. 1995. Geometry and Generality in Frege's Philosophy of Arithmetic. *Synthese* 102:319 – 361.