DIPLOMARBEIT

Titel der Diplomarbeit

"From Vagueness to Disaster:
On the Intricacies of Feature Checking"

Verfasser

Thomas Graf

angestrebter akademischer Grad

Magister der Philosophie (Mag.phil.)

Wien, im November 2007

Matrikelnummer	a0300809
Studienkennzahl lt. Studienblatt	A328
Studienrichtung lt. Studienblatt	Allgem./Angew. Sprachwissenschaft [Stzw.]
Betreuer	O.Prof. Mag. Dr. Dr. h.c. Wolfgang U. Dressler

L'envie vous prend parfois de crier aux ci-devant dieux: «Faites donc un petit effort, tâchez de réexister!»
J'ai beau maugréer contre tout ce qui est, j'y suis néanmoins attaché — si j'en juge d'après ces malaises qui s'apparentent aux premiers symptomes de l'être.

Émile Michel Cioran, «Le mauvais démiurge»

Acknowledgments

This thesis has profited from the feedback of several people, and I want to thank all of them — even if most of the time I was too stubborn to accept their criticism right away. In alphabetical order: Sebastian Beer, Wolfgang Dressler, Hans-Martin Gärtner, Bettina Gruber, Christian Huber, Dalina Kallulli, Martin Prinzhorn, Viola Schmitt, and foremost Edwin Williams, whose elaborate critique was the starting point for a lot of improvements to chapter 2. An honorable mention goes to Sarah Zobel for all the support she has given me and is still giving me on my ongoing journey into mathematics. As usual, none of these people should be assumed to agree with anything presented in the text, nor are they responsible for any mistakes I might have made.

Contents

Introduction

Since the very first moment of its inception, the Principles-and-Parameters approach (P&P) (Chomsky 1981; Chomsky and Lasnik 1993; Chomsky 1995c) has been characterized by a strong longing for explanatory adequacy. This has been reflected in a preference for underspecification and "rule-of-thumb" generalizations over precise implementations of technical aspects. Unnoticed by a great number of its advocates, this methodological decision nurtures a style of scientific exposition whose vague wording and reliance on intuitive notions gives rise to numerous problems related to descriptive adequacy as well as tractability.

This thesis is a case study of how various obscurities in the definitions of the very basics of a theory may endanger research based on it, and thus the whole enterprise. In particular, I will look at Reuland (2001) and his changes to the feature checking mechanisms developed in Chomsky (1995c). Depending on how one interprets the relevant details, there are two possible outcomes of Reuland's modifications, both undesirable: either the new theory is inherently contradictory and inapt to capture Reuland's main ideas, or it uses highly specific assumptions that render it incompatible with a great number of concurrent explanations of completely unrelated phenomena. Reuland's paper was picked because it is a condensation of typical P&P-traits; while the ideas put forward are very attractive and inspirational on an intuitive level, they fall short of formal soundness.

Because of the formally adept consideration of the relevant syntactic issues, the merits of my thesis go beyond simply advertising a technically more enlightened take on linguistics. It offers an ambitious exploration of feature checking in classical Minimalism, in particular how checking theory can be made more explicit and in which ways it must not be altered. Those results are of immediate importance to anyone who is still entertaining the idea that checking is feature deletion, not feature valuation. Furthermore, it is a first tentative step towards a comprehensive typology of minimalist feature checking mechanisms and how they compare on conceptual and formal grounds.

The thesis is laid out as follows: chapter 1 gives a short introduction to classic Minimalism. Readers still familiar with the details of this dated framework can easily skip that section and proceed to chapter 2, where I sharpen the notion of feature checking as put forward in Chomsky (1995b). The final result of those 30+ pages of conceptual discussion is a typology of feature checking systems which will lay the ground for the analysis of Reuland (2001) in chapter 3. There I show that Reuland's theory is incompatible with some classes of feature checking systems and that the remaining feature checking systems in conjunction with Reuland's modifications of Chomsky (1995b) are highly stipulative, or empirically inadequate, or inherently contradictory. I take this as evidence for how the lack of formal rigor may cause a theory which shines at an intuitive level to fall apart when subjected to a low-level inspection. I conclude that linguists have to take a more careful approach to theory construction that is neither dependant on intuitive notions nor prone to omitting crucial details. Building on this judgment, I assess the pervasiveness of vagueness in P&P-research and what its origins might be in chapter 4. I then investigate in how far logic or formal language theory might mitigate the impact of vagueness and infer that those tools, albeit highly useful, can show positive effects just in case linguists are already devoting a good deal of their attention to technical accuracy. That is to say, the solution to the problem won't be found in mathematical notation, but in a general awareness of the drawbacks of the current situation.

Chapter 1

Minimalism: A Minimal Introduction

1.1 A Word of Caution

In this thesis, discussions of Minimalist syntax, as interesting as they may be in and of themselves, ultimately serve the single purpose of establishing a firm base which the investigation of Reuland (2001) can be based on. Although Reuland's paper appeared when Minimalism had already matured (or at least changed) a lot compared to initial instantiations, it is set within the framework of that early version of Minimalism, i.e. Chomsky (1995b). Considering the dazzling mutability shown by the technical apparatus of Chomsky's youngest theory, refamiliarizing oneself with the original mechanics of the Minimalist Program is a wise move. That is exactly what this chapter is meant for. Consequently, Chomsky's later articles (Chomsky 1998, 2001, 2004, 2005a, 2007) will only play a minor role here.

But be aware that despite space restrictions I do not have the slightest intention to water down the formal aspects of the framework any more than absolutely necessary. Therefore this chapter is not meant for those who are completely unfamiliar with Minimalism. If a real introduction is needed, rather than a simple revision, I wholeheartedly recommend to start with Hornstein et al. (2005) and complement it with Boeckx (2006) and the first chapter of Uriagereka (1998) for a healthy dose of fanaticism (whether *pro* or *contra* Minimalism is up to the reader).

1.2 General Architecture

Let me start this section with a methodological note. Since its introduction in Chomsky (1995c), the most often stated and most seldomly understood fact about Minimalism has been that it is a framework, not a theory. Minimalism isn't a coherent set of proposals and analytical tools, it is a guideline for how one should do linguistic research, what constitutes an interesting question, which criteria one should apply for judging a theory's value, and so forth. At least that is what Chomsky asserts he intends it to be. Unfortunately, even most Minimalists apparently do not fully grasp this shift in perspective. They continue to follow their old habit of looking at Chomsky's main proposals, tinkering with some mechanism and molding the final result into a respectable paper which nonetheless won't enjoy notable resonance in the linguistic community because most Minimalists are mainly considering Chomsky's proposals as the point of departure for their own research. Due to this specific trait of the community, phase theory (Chomsky 2001, 2004) found rapid adoption despite various logical and empirical flaws (cf. Frampton et al. 2000; Epstein and Seely 2002; Abels 2003; Boeckx and Grohmann 2004), while other well-known and respectable flavors of Minimalism, like Level-free Syntax (Epstein et al. 1998; Epstein and Seely 2002, 2006) or Prolific Domains (Grohmann 2003) did not manage to increase their following. So, despite Chomsky's saying otherwise, Minimalism as the community views it is more like a theory than a framework and therefore I will present it here as if it were a coherent theory.[1]

An unintended consequence of this approach is that it mistakenly conveys that the shift to Minimalism was a revolution, rather than an evolution. For example, one could be under the impression that Chomsky (1995c) marked a sudden change in the technical framework, while in effect the first Minimalist lectures and manuscripts already appeared in the early nineties. Notions of economy in general were already present in Government and Binding Theory from the very beginning (in fact they can already be found in his earliest publications), for instance in the form of the Avoid-Pronoun principle. Features already played a certain role, too, as only a tensed I^0 with agreement features was assumed to assign Nominative Case. Minimalism thus did not introduce a completely new point of view, rather it emphasized up until then neglected aspects and tried to do as much work with them as possible. See Freidin and Vergnaud (2001) and Adger and Harbour (to appear) for further information on this issue.

In contrast to its predecessor, Government and Binding Theory (GB), Minimalism is derivational in nature, not representational.[2] GB opted for a single rule *Move* α, which generated all possible permutations of a given Deep Structure. Most of those representations were filtered out later on at one of three levels — Surface Structure (SS), Phonological Form (PF), Logical Form (LF) — if they violated a well-formedness condition of one of the numerous modules distributed over these levels.

Minimalism, on the other hand, employs no such modules and even strives for a reduction of the number of linguistic levels. As a counter measure, the number of structure-building and structure-manipulating operations is increased. The work formerly done by filters is now captured by the properties of those new operations and the restrictions on their application. Instead of generating myriads of representations that are filtered out later on, a single representation is built up incrementally by hierarchical concatenation

[1]In chapter 2 it will become obvious that, given the numerous uncertainties in Chomsky's work, this treatment is justifiable only from a pedagogical perspective.

[2]Nevertheless Chomsky (1995c) isn't devoid of a certain representational flavor, just think of chains, the uniformity condition, the definition of c-command and the checking domain. In addition, the entire assembled structure is accessible to syntactic computation, which leads Brody (2002) to classify Chomsky (1995c) as a weakly representational theory.

and manipulation of lexical items through various operations. The final shape of the representation is not due to some filter but to the properties of the operations needed to create it in the first place.[3]

The overall architecture Chomsky ascribes to the language faculty (also referred to as C_{HL}) is a simplification of the inverted T-model.

(1)

```
              Numeration
                  |
              Spell-Out
             /         \
           PF           LF
           |             |
  Sensory-motor   Conceptual-interpretative
     system              system
```

A so-called numeration stores all lexical items (LIs) of the intended utterance. From those LIs, narrow syntax constructs the set of all derivations which satisfy the conditions imposed by the interface levels PF and LF. Among the members of this set, the most economic derivation is chosen. A single Spell-Out separates the overt from the covert cycle, in stark contrast to the younger phase theory (Chomsky 2001, 2004, 2005a). Deep Structure and Surface structure are dispensed with in Chomsky (1993), leaving PF and LF as the only levels where certain well-formedness conditions apply. Presupposing that language is an extremely economic device, Chomsky concludes that the conditions enforced by these two levels, referred to as the *Bare Output Conditions* (BOC), are the driving force for all syntactic processes. Syntax is reduced to a device that supplies the means needed to create trees that are legitimate at both PF and LF. As PF and LF are uniform across languages, the BOC in conjunction with the assumption that the language faculty is economic and parsimonious suffice to derive all syntactic principles.[4] Parametric variation is captured by lexical differences between languages (cf. Borer 1984), encoded by the feature composition of LIs.

So far for the commonplaces on Minimalism, let us get a wee bit more formal now. Following Potts (2002), we define a derivation as a partially ordered set of trees.[5] Assuming

[3]This is actually a slight misrepresentation of Chomsky (1995b), just like the comment on Minimalism being not representational. Later on, in 1.4, we will see that classic Minimalism makes use of global economy conditions, which are in some sense mimicking the behavior of filters.

[4]This is to be understood as a research principle, not as an actual statement about the current status of the theory. Beyond some general intuitions, nobody knows what the BOC are, so it is obviously impossible to logically derive anything from them.

[5] This is a model-theoretic perspective on transformational frameworks (see section 4.2.3), and we will encounter it again several times. It does not reflect the stance of mainstream Minimalism, which the definition in Gärtner (2002:56) comes closest to.

(i) A *derivation* is a sequence of stages $\Sigma_0,\ldots,\Sigma_n$ such that for each i ($0 < i \leq n$), Σ_i is the outcome of exactly one syntactic operation applied to Σ_{i-1}.

(ii) A *derivational stage* Σ is a set of syntactic objects ($\Sigma = \{\mathrm{SO}_1, \ldots, \mathrm{SO}_n\}$)

that the final tree in this set serves as input for the interfaces, syntax as a whole is to be regarded as a function mapping sets of LIs to trees. This analogy begs the question how the function is to be provided with arguments, i.e. LIs, and which properties of the input it is sensitive to.

LIs are stored in the lexicon, wherefrom they can be added to the numeration. The numeration is a multiset containing all the LIs from which the utterance is to be built. More precisely, it contains ordered pairs $\langle LI_i, o \rangle$, where $o \in \mathbb{N}^+$ is the occurrence index of LI_i. Said index encodes the number of LI_i's occurrences in the tree and is decreased by 1 when LI_i is selected and enters the derivation.[6] Crucially, a derivation can't converge before all indices in the Numeration are 0. If one contends that a derivation won't yield a legitimate representation unless the entire numeration is exhausted, a precise mechanism for the correct construction of the latter becomes indispensable. Otherwise, the numeration could contain material not needed for the derivation, hence preventing it from converging. This task is handled by a nameless economy principle which ensures that only the smallest numeration needed for the intended utterance is constructed.[7]

(2) α enters the numeration only if it has an effect on output. (Chomsky 1995b:294)

When an LI enters the derivation, its feature composition is already fully specified and has an immediate impact on the steps syntax has to take in order to cause the derivation to converge. The set of features that constitute an LI comprises phonological, semantic and formal features, with only the latter playing a distinctive role for syntactic computation. The set of formal features is not known in its entirety, but it is usually said to contain categorial features, Case features, ϕ-features and the EPP-feature.

In contrast to phonological and semantic features, formal features can be subclassified with recourse to two properties, feature interpretability and feature strength. Uninterpretable features must not reach LF if the derivation shall converge, and strong features have to get eliminated as soon as they enter the derivation.

(3) a. [±interpretable, +strong]: Eliminated immediately
b. [-interpretable, -strong]: Eliminated before LF
c. [+interpretable, -strong]: Does not need to get eliminated

The necessity to purge specific features is the driving force behind movement, as the reader will soon see for himself.

In my opinion, the model-theoretic definition often is the more illuminating one and easier to work with, so I will switch between both points of views as it fits my needs.

[6]Indices express the occurrence of one specific LI with a single referent, i.e. two numerations $N_1 := \{\langle John, 1\rangle, \langle John, 1\rangle, \ldots\}$ and $N_2 := \{\langle John, 2\rangle, \ldots\}$ differ with respect to the referents of *John*. In the utterance generated from N_1, the speaker is referring to two distinct individuals that coincidentally share the same name, whereas N_2 encodes that a specific individual called John enjoys the honour of being mentioned twice in a single utterance.

[7]It does not take a lot to figure out that this principle is highly problematic when it comes to computability. I shall postpone the relevant discussion until 1.4, where economy conditions and reference set computation will get all due attention.

1.3 Operations and Conditions

In order to decrease the number of illegitimate derivations, LIs have to be chosen from the numeration in a principled manner, i.e. via a syntactic operation, in this case Select.

(4) *Select* (Chomsky 1995b:226)
Suppose that the derivation has reached the stage Σ, which we may take to be a set $\{\mathrm{SO}_1, \ldots, \mathrm{SO}_n\}$ of syntactic objects. One of the operations of C_{HL} is a procedure that selects a lexical item LI from the numeration, reducing its index by 1, and introduces it into the derivation as SO_{n+1}.

Select comes in two different variants, c-selection and s-selection (cf. Adger 2003). The former refers to the usual mechanism of subcategorization, while the latter denotes semantic selection, i.e. selection of LIs due to their semantic impact, e.g. in the case of adjunction or when it can't be decided by any other means which DP should become the object of the verb. Although it remains a difficult issue how s-selection could be modeled in a Minimalist theory, the operation is assumed as a given without further questioning (hence without further research on its specifics).

Select is the only option to supply syntax with the material from which phrase markers are built. Syntax cannot make use of anything that was not stored in the numeration prior to the first rule application. This constraint is also known as the Inclusiveness Condition.

(5) *Inclusiveness Condition* (Chomsky 1995b:228)
[A]ny structure formed by the computation [...] is constituted of elements already present in the lexical items selected for N [the numeration; TG]; no new objects are added in the course of computation apart from rearrangements of lexical properties (in particular, no indices, bar levels in the sense of X-bar theory, etc.; [...]).

Selection of LIs evidently does not suffice for the generation of an utterance, at least one additional operation that concatenates the selected LIs into a string is indispensable. If hierarchical relations are to be expressed as well, even more is needed. The operation Merge fulfills both duties by taking two arguments and concatenating and labelling them. The structure of the label may vary: arguments are set-merged, while adjuncts are pair-merged.[8]

(6) *Set-Merge* (cf. Chomsky 1995b:243)
$\mathrm{Merge}(\alpha, \beta) = \big\{\underline{\gamma}, \{\alpha, \beta\}\big\}$, where α, β are terms and $\gamma \in \{\alpha, \beta\}$ is the label.[9]

[8] Pair-Merge in its current form does not fit well into Minimalism and there is remarkably little literature on the topic, with the notable exception of Chametzky (2003), who provides a devastating critique on conceptual grounds.

[9] The status of labels is not made explicit in Chomsky (1995b), and Chomsky (1995a:397) simply states that the label of $\{\alpha, \beta\}$ is "one or the other of α, β". Citko (2006) notes that they have to be copies of the head of a phrase, according to the Inclusiveness Condition (5). This conclusion does not carry over to the system of Chomsky (1995b), though, for Chomsky states that copies are not accessible to further syntactic computation. This leaves us with two options, either the projecting head is not accessible anymore, ruling out Head-Movement, or the label is not accessible, defeating the purpose of the entire labeling mechanism.

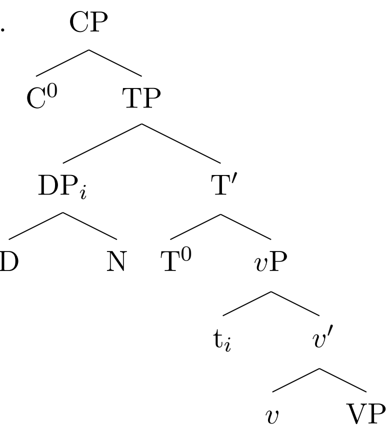
(7) *Pair-Merge* (cf. Chomsky 1995a:402)
Merge(α, β) = $\left\{\underline{\langle\gamma, \gamma\rangle}, \{\alpha, \beta\}\right\}$, where α, β are terms and $\langle\gamma, \gamma\rangle$, $\gamma \in \{\alpha, \beta\}$, is the label.

Both definitions contain a notion we did not encounter yet, namely *term*. Simplifying somewhat, terms are maximal and minimal nodes, so the respective definitions of Merge allow both merger at the root of the tree and at one of its leaves. Things will become clearer when we consider how terms are defined, but before we can do this we have to familiarize ourselves with the relation between trees and the sets generated by Merge.

The attributes of phrase structure follow directly from the definition of Merge. Merge is a binary operation, hence phrase structure has to be binary branching. Neither unary nor n-ary ($n \geq 3$) branching nodes exist in syntax, because there is no operation to generate them. For the same reason, there are no bar-levels. Instead, the status of the projection is determined relationally. The highest instance of a label γ is a maximal projection, the lowest instance a minimal projection, all other projections are intermediate. The simple phrase structure thus described is termed Bare Phrase Structure (BPS). In contrast to classic $\overline{\text{X}}$-theory, BPS relies solely on naive set-theory, which is an integral part of human cognition according to Hauser et al. (2002). This is an instance of rather abstract Minimalist deduction: phrase structure is not a linguistic construct in and of itself, but a byproduct of the interaction of a general cognitive mechanism, namely set formation, and the faculty of language, allowing us to use a language which relies on hierarchical information. That way, phrase structure comes for free and does not need to be stipulated as a linguistic primitive (cf. Chomsky 2005b).[10]

The following tree can easily be expressed in set-theoretic form.

(8) a.



b. $\{\underline{\text{C}},\{\text{C},\{\underline{\text{T}},\{\{\underline{\text{D}},\{\text{D},\text{N}\}\},$
$\{\underline{\text{T}},\{\text{T},\{\underline{v},\{\{\underline{\text{D}},\{\text{D},\text{N}\}\},\{\underline{v},\{v,\text{VP}\}\}\}\}\}\}\}\}\}$

Let us now return to terms, for which I did not supply any definition yet.

(9) *Terms* (Chomsky 1995a:399)

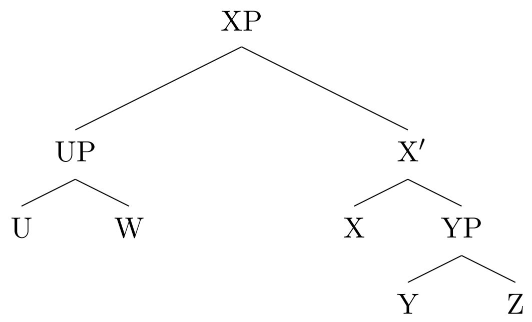
a. K is a term of K;

[10]Actually, this only applies for the concatenating part of Merge, one still has to stipulate the existence of labels and different bar-levels.

b. if L is a term of K, then the members of the members of L are terms of K.

Take a look at the structure in (10), represented both as a tree and as a set.

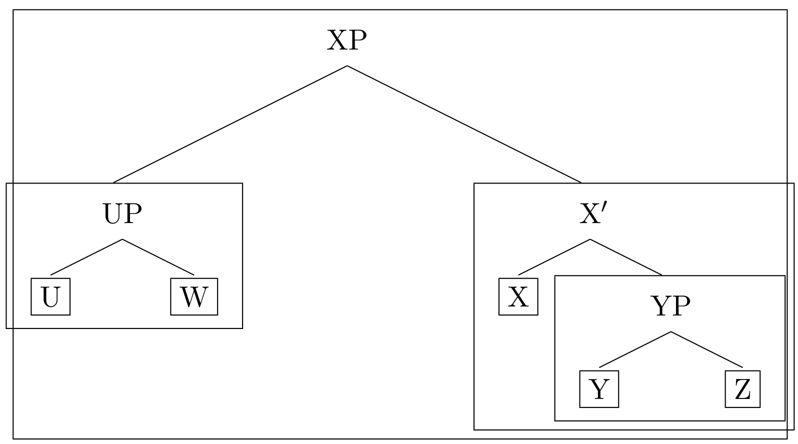
(10) a.



b. $\{\underline{X},\{\{\underline{U},\{U,W\}\},\{\underline{X},\{X,\{\underline{Y},\{Y,Z\}\}\}\}\}\}$

Applying the definition in (9), we get the following terms (indicated by boxes). As can be seen, only terminal nodes and syntactic objects constructed from them qualify as terms, labels do not.

(11) a.



b. $\boxed{\{\underline{X},\{\boxed{\{\underline{U},\{\boxed{U},\boxed{W}\}\}},\boxed{\{\underline{X},\{\boxed{X},\boxed{\{\underline{Y},\{\boxed{Y},\boxed{Z}\}\}}\}\}}\}\}}$

The reader can verify for himself that Merge always applied to terms during the construction of this tree.

Besides Merge, there is another structure-building operation, Move. Move functions like Merge, except that one of its arguments is provided by the operation Copy, instead of Select. Copy targets the element α which is to be displaced, creates a duplicate of it and hands this as input to Move.[11] If movement takes place in the overt cycle, both α and the landing site are terms. If it is delayed post Spell-Out, only the formal features of an LI have to move covertly and adjoin to a head.

[11] Chomsky (2005a) criticizes that this point of view is actually mistaken, for already in Chomsky (1993) he intended the copy theory to be what is currently known as Remerge. This will be mooted in 4.1, but let me mention in advance that I find it highly amusing that the canonical theory is actually due to a misreading of Chomsky's dubious writings. On the other hand, I can't help but wonder why it took Chomsky 12 years to clarify this misunderstanding.

(12) *Move*[12]
Let α and β be terms, or β a head and α an LI's set of formal features. Then
a. Move(Copy(α), β) := $\{\underline{\beta}, \{\alpha, \beta\}\}$, if Move is triggered as an instance of substitution.
b. Move(Copy(α), β) := $\Big\{\underline{\langle\beta, \beta\rangle}, \{\alpha, \beta\}\Big\}$, if Move is triggered as an instance of adjunction.
c. Move(Copy(α), β) is immediately followed by the construction of a chain CH := (α, t_α). Every chain obeys the conditions in (13).

(13) *Conditions on chains* (Chomsky 1995b:253)
a. C-Command Condition
α must c-command its trace.
b. Uniformity Condition
A chain is uniform with regard to phrase structure status.

While (13a) prohibits any kind of lowering or sidewards movement, (13b) blocks movement of α into a position β with a different bar-level. Move itself is subject to various constraints, too.

(14) *Conditions on Move*
a. Minimal Link Condition (Chomsky 1995b:296)
α can raise to target K only if there is no legitimate operation[13] Move β targeting K, where β is closer[14] to K.
b. Procrastinate (cf. Chomsky 1995b:254)
Covert movement is cheaper than overt movement.
c. Last Resort (Chomsky 1995b:280)
Move F raises F to target K only if F enters into a checking relation with a sublabel of K.

While (14a) and (14b) are easy to understand, (14c) is painfully incomprehensible given our current vocabulary, so let us simplify it: movement is allowed only in those cases where it leads to the checking of a feature. A feature is checked if it is strong or uninterpretable and in a specifier-head configuration with an identical feature.

Consider example (15). When T is merged with vP, a strong categorial D-feature (better known as the EPP-feature) is added to the tree, and it has to be checked immediately. The DP bears the corresponding checker and hence has to move to [Spec,TP]. The

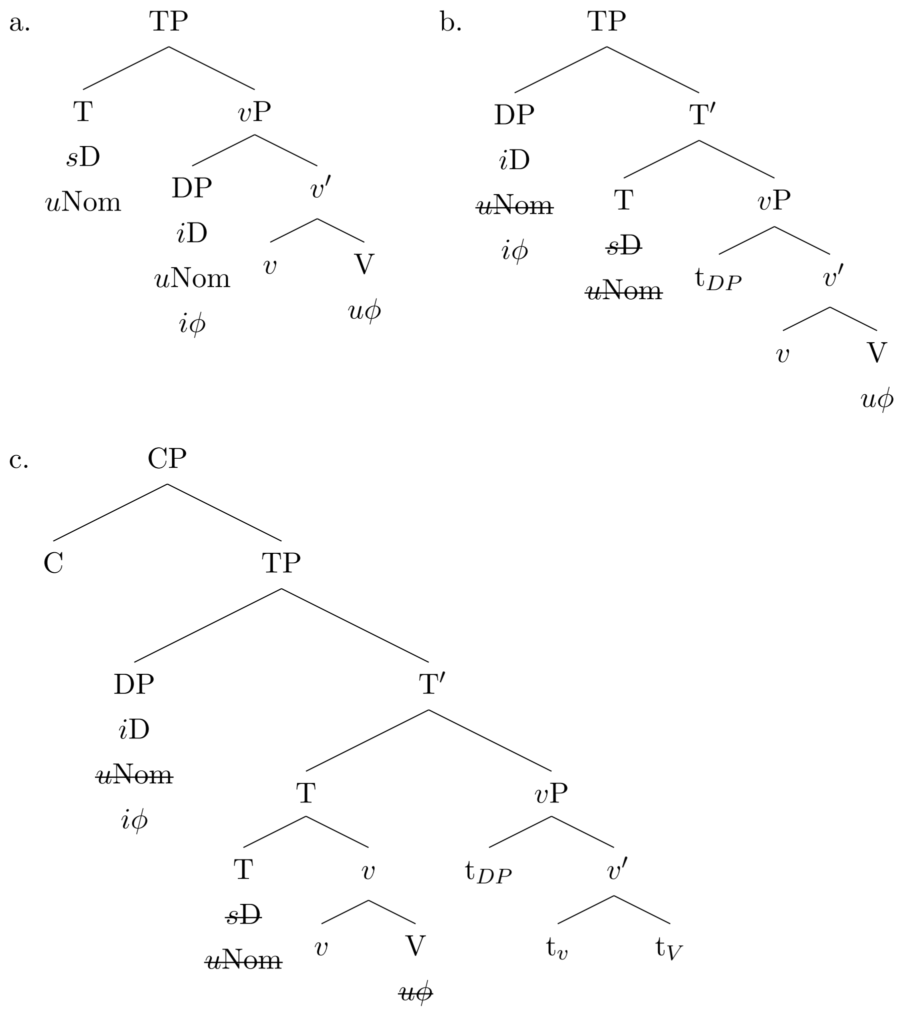
[12]This is a somewhat simplified definition. Actually, there are two movement operations, Move F and Move α. Move F copies a formal feature (or the entire set of formal feature), and adjoins it to a head. Move α, in turn, is a complex operation such that some feature of LI_i is adjoined by Move F to a head, say X^0, followed by copying LI_i and merging the copy with XP. Then, by some undefined repair mechanism, the formal features adjoined to X^0 move back into the copy of LI_i.

[13]A "legitimate operation" satisfies Last Resort as it is stated in (14c).

[14] Closeness is to be defined in terms of c-command and equidistance. The notion of equidistance relies on Minimal Domains, defined in (48) in chapter 2. "γ and β are equidistant from α if γ and β are in the same minimal domain." (Chomsky 1995b:356)

strong feature is checked and removed from the structure, while the interpretable categorial feature of the DP remains accessible for further computation. As a byproduct of the immediate dislocation, both the DP and T can check their uninterpretable Nominative features prior to Spell-Out.[15]

If T did not have a strong D-feature, Case would have to be checked covertly, due to Procrastinate. After C has been merged with TP, all occurrence indices in the numeration are zero and Spell-Out takes place. V covertly raises and adjoins to v, which in turn adjoins to T, such that V ends up as a possible feature checkee of the DP, and consequently its uninterpretable ϕ-features are checked by the DP's interpretable ϕ-features.

(15)


1.4 Economy and Reference Set Computation

Up to now I neglected the role of economy in the Minimalist Program. This shortcoming has to be rectified, firstly because it is the center piece of Minimalist reasoning, and

[15]Spell-Out takes place as soon as the numeration has been exhausted, unless there are still strong features present that have to be checked. This is the case only if the strong feature is introduced into the derivation by the last item selected from the numeration. After the strong feature has been checked (if there was one), Spell-Out takes the current structure Σ_i as its input and generates an identical Σ_j which is sent to PF, while a different representation Σ_L which is devoid of any material solely relevant to PF is sent to LF (cf. Chomsky 1995b:229).

secondly because the technical implementation of economy in Chomsky (1995b) differs significantly from his later works. The general role of economy in Minimalism has already been discussed many a dozen times in various articles and interviews the interested reader can get immersed in (see Chomsky 2002 and Uriagereka 2000, among others). I do not see any point in replicating those fine expositions. The technical aspects of economy, however, are a challenging topic well worth closer scrutiny.

The conjecture that syntax were a device optimally tailored to doing its job while obeying the constraints imposed by other cognitive modules entails that the language faculty is as economic as possible, for computational ressources are limited and thus to be saved whenever possible. Unfortunately, determining what constitutes a maximally economic device is anything but trivial. The first thing to decide is whether it is global or local economy conditions it adheres to. Global economy conditions compare all possible outputs and pick the most economic one as sole viable input for further computation. Local economy conditions, on the other hand, define certain economically motivated restrictions on the computation itself such that only economical output can be generated. Consider the following minimal pair taken from Chomsky (1995b:344).

(16) a. * there seems someone t to be in the room.
b. there seems t to be someone in the room.

Given the technical apparatus of Chomsky (1995b), both sentences should be fine. Neither *seems* nor *someone* ends up with some crucial feature unchecked, because it does not matter whether features are checked overtly or covertly. Just like *someone* in (16a) can raise overtly to Spec,TP of the embedded clause and thus get its features checked, so can its formal features do covertly in (16b). If checking features is the only thing that matters, the results are basically the same. However, with recourse to global economy we are able to capture the contrast and rule out (16a) as ungrammatical. We do so by comparing both constructions and concluding that the first one violates Procrastinate, while the second one does not. But we could also explain the difference by a local economy condition which prefers Merge over Move wherever possible.

Most readers are probably familiar with local economy conditions from recent Minimalist research, whereas global economy conditions perhaps feel awkward to them. This isn't particularly surprising given that global economy is one of the most overlooked, yet decidedly distinctive traits of classic Minimalism. Nevertheless it is one of the most disfavored, too, and that for good reasons indeed.

Chomsky (1995b) has both local and global economy conditions in his repertoire. The Minimal Link Condition is an important example for the former, while the latter is represented by reference set computation. Instead of constructing a single well-formed and economically optimal representation which is shipped to the interfaces later on, syntax produces a set comprising all representations that it can generate from the numeration, and from this set it chooses the most economic one — a system of overgeneration and filtering reminiscent of GB. The crucial difference between a grammar using global economy and one using local economy then is that in the case of the former, it does not suffice for a representation to be well-formed in order to be grammatical, it also has to be the most economic candidate.

Obviously, the computational load of global economy conditions peaks way higher than that of local economy conditions. In an attempt to reduce the computational overhead of his theory, Chomsky introduces a distinction between what one could call local and global reference sets. The terminology is particularly misleading here, as both kinds of reference sets belong to the class of global economy conditions. A local reference set is a subset of a global reference set, as it contains only those converging derivations which are compatible with the current derivational stage. That is to say, only possible continuations of the derivation are considered, in contrast to a global reference set, which also contains all derivations already obsoleted. Johnson and Lappin (1997) point out that this does not reduce computational load at all, because a local reference set is only a subset of a global reference set, not a proper subset. At the very first stage of the derivation, when no LI has yet been selected from the numeration, the sets are identical. Therefore, local reference sets may eventually require less memory, but the computational load of constructing one equals that of global reference sets, if it isn't even higher. After all, derivations in a local reference-set have to be detected as incompatible with the current stage and marked accordingly, which might be more resource-consuming than expected.

Collins (1996), Sternefeld (1996), Johnson and Lappin (1997) and Potts (2001, 2002) provide even more arguments against global economy conditions, but I will stop my general critique here, as the main point should be sufficiently bolstered by now: global economy is a computational nightmare.

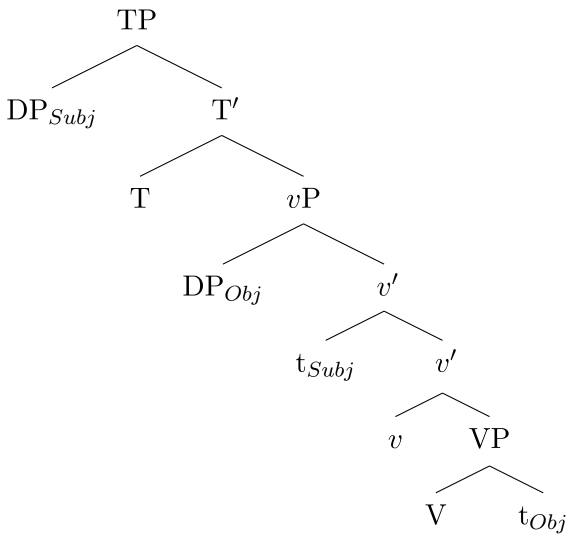
Despite the questionable status of global economy conditions, I will exemplify Chomsky's implementation a little bit more in detail, since it is not without relevance for our discussion of Reuland (2001). Chomsky employs three principles as global economy metric, two of which we are already familiar with.

(17) *Economy metrics*

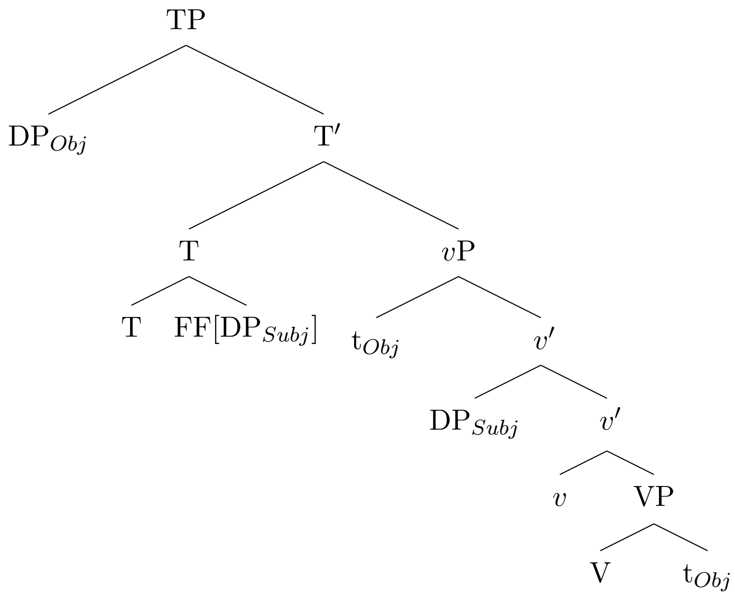
a. *Procrastinate*
   Covert movement is cheaper than overt movement.
b. *Smallest Derivation Principle* (SDP) (Johnson and Lappin 1997:21)
   Let OD be the set of distinct operations in a derivation D. [...] For any two convergent derivations D and D' from a numeration N, D is more optimal than D' if $|OD| < |OD'|$.
c. *Has an Effect on Output Condition* (HEOC)
   α enters the numeration only if it has an effect on output.

We already looked at (16) as an example for Procrastinate, so let us immediately proceed to the SDP. Chomsky (1995b:357) gives the following example for its usefulness: in Icelandic, v may host an optional D-feature in order to allow for overt object shift. T evidently has a D-feature too. In such a configuration, two different derivations can converge, but only one actually is grammatical.

(18) a.



b.



In both derivations, all Case and ϕ-features are checked, just like T's strong D-feature. There are no instances of feature mismatch either. Such a mismatch would induce cancellation of the derivation, but as the Case feature of the object is checked prior to its movement to Spec,TP, no conflict arises with T's Case feature. Nor does a violation of the Minimal Link Condition ensue, because the subject and the object occupy specifiers of the same phrase and are thus equally close to TP (cf. footnote 14). If both derivations converge, but only the first one is grammatical, it follows from our definition of a grammatical structure as the most economic converging derivation that the second derivation is less economic. Resorting to the Smallest Derivation Principle reveals that this is the case indeed. While they do not differ in the number of Select and Merge operations, the second derivation needs three applications of Move, the first one only two. Hence the latter is more economic.

The third metric, called HEOC (due to Johnson and Lappin 1997), is very special insofar as it introduces reference sets for different numerations, with far reaching consequences. In order to determine whether adding an LI has an effect on output, one has to construct a set containing at least two numerations, such that all numerations in the set differ in their cardinality at most by 1. For each of those numerations a full reference set of converging derivations has to be computed, followed by selection of the most economic derivation for each set. Then those derivations are sent to LF, where their denotations are computed, which are in turn compared to each other. If the denotations of the respective derivations turn out to be equivalent, the numeration containing the fewest LIs is chosen as starting point for the derivation. If the denotations differ, the numeration with the most LIs is chosen. That way, if we have three numerations $N_1 := \{LI_1\}$, $N_2 := \{LI_1, LI_2\}$, $N_3 := \{LI_1, LI_2, LI_3\}$ and $[\![N_1]\!] \neq [\![N_2]\!] = [\![N_3]\!]$[16], N_2 is chosen as the optimal numeration.

As is to be expected, Chomsky does not give any hints how equivalence classes at LF should be defined, and the issue inevitably gets even more complex as soon as PF and pragmatics are taken into account. Furthermore, the principle is deeply flawed insofar as it apparently does not allow N_3 to be chosen in cases where there is an additional $N_4 := \{LI_1, LI_3\}$ such that $[\![N_1]\!] = [\![N_2]\!] = [\![N_4]\!] \neq [\![N_3]\!]$, because neither LI_2 nor LI_3 make a difference to output on their own, wherefore neither is allowed to enter the numeration.

One may doubt the viability of the whole discussion and maintain that the HEOC should not be taken as a metric for economy but as a methodological guideline or as an allusion to the performance mechanisms responsible for choosing LIs according to the speaker's conceptions. Both conjectures lack rhyme and reason. The latter deprives the principle of any relevance for any theory of grammar considered to be a theory of competence, while the former is almost cynical in claiming that Chomsky intended to create a doppelganger of Occam's razor, a somewhat amusing, yet misguided idea.

When it comes to the empirical foundations of the principle, Chomsky himself does not provide any supporting data, but see Fox (1995) for a possible application in ellipsis constructions and Johnson and Lappin (1997) for a rebuttal.

1.5 Recapitulating the Crucial Differences

After having read this chapter, those readers primarily acquainted with the latest incarnation of Minimalism will have discovered several traits that distinguish Chomsky (1995b) from its successors.

First, LIs enter the derivation with all their features fully specified, in contrast to Chomsky (2001) and subsequent works, where some features may lack a value which has to be provided later on via Agree. Consequently, movement is driven by the need to check any uninterpretable features, whereas only specific features may trigger movement in an approach based on Agree. In addition to uninterpretability, features are also specified for strength such that a strong feature has to be checked immediately after it has entered the derivation.

Second, there is a clear division between the overt and the covert cycle, marked by a single Spell-Out. Neither is covert movement allowed to take place at any time

[16] My notation here is rather sloppy, for a numeration obviously cannot have a denotation. I am confident that everybody realizes that it is meant to be read as "the denotation of the most economic converging derivation to be constructed from this numeration".

— although this could easily be implementend given that it is couched in terms of pure feature movement — nor is there any segregation of the derivation into smaller units akin to phases or structured numerations.

Third, Chomsky (1995b) features three global economy conditions, Procrastinate, the SDP and the HEOC. Most of the time, these conditions act invisible and do not have any impact on empirical analysis, wherefore one is prone to overlook them. Yet they represent classic Minimalism's most distinctive trait and most significant weakness.

Chapter 2

Exploring Feature Checking

This chapter will deal with the details of features and feature checking in Chomsky (1995b). I will examine the status of features, feature bundles and explore the mechanics of the feature deletion processes Delete(α) and Erasure. These steps have to be taken in order to analyze the feasibility of Reuland's proposal, which will be the topic of chapter 3.

I proceed as follows: first I discuss the status of features in Chomsky (1995b) and in which way it differs from the assumptions in later version of Minimalism, beginning with Chomsky (1998). Then I show how lexical items can be defined set-theoretically without jeopardizing the strict division between lexical items and syntactic structure. This is followed by a simpler reformulation of feature strength by making use of the sharpened notion of feature bundles. The new solution provokes some intricate questions concerning the encoding of feature interpretability, which I will address in section 2.4.

In section 2.5, I finally move beyond features and their architecture and focus on feature checking instead. Interestingly, this section also marks a step into realms of much higher lucidity. As we will see more than once, Chomsky's exposition is surprisingly conclusive as long as it does not need to drop below the syntactic level. As soon as the structure of LIs and the manipulation of features themselves becomes a part of the picture, things get messy.

In order to clarify the gloomy parts, I first introduce the relevant notions entertained in Chomsky (1995b) and then give a more explicit formulation of feature matching and feature identity. I look at the properties of Erasure and Delete(α) and demonstrate that the latter faces some conceptual problems which can easily be accounted for by measures already familiar from my implementation of feature strength. Building on Nunes (2000), I then show the redundancy of having both Erasure and Delete(α), and that theories employing only one feature checking process still cover a comparable range of empirical phenomena. I conclude with a classification of feature checking systems compatible with the main assumptions in Chomsky (1995b).

With regard to my methodology, a clarifying note is in order: ultimately, I am interested in a valid analysis of the consequences of Reuland's modifications to feature checking. Collecting the bits and pieces scattered over Chomsky (1993, 1995a,c) is a necessary evil, not my main objective. Consequently, I will prefer precise stipulations over vague derivations whenever it suits my needs, as long as the result does not look totally "unminimalistic". However, I do realize that many syntacticians will consider this chapter the most relevant one of the thesis, and I do not want to put off this part of the audience. Therefore I decided that the inclusion of a notable amount of what is sometimes referred to as *minimalist reasoning* would not do any harm, as long as it did not interfere with the rhythm and pace of the presentation. As for the fundamental conceptual issues, like interpretability, I also consider more recent proposals in the literature, although their results evidently are not represented in the early formulations of Minimalism. The result of this painstaking enterprise is a number of explicit variants of the Minimalist Program which are sufficiently close to Chomsky (1995b) for being considered a viable basis of Reuland (2001) while at the same time retaining the overall minimalist spirit of the system. Hopefully, this will satisfy the expectations of the entire audience, diverse as it might be.

2.1 Features

In Minimalist Syntax, a representation is constructed derivationally by application of several operations to a specified input, with the input consisting of LIs and nothing else. An LI, in turn, is a feature bundle. This is a very vague way of putting it, but unfortunately it is as good as it gets in Chomsky (1995c), where features and related topics definitely are not treated with due precision. Whoever wants to correct his shortcomings has to start by providing clear answers to two questions: first, what is a feature, and second, what is a feature bundle? I will now turn to the first question, leaving the latter to section 2.2.

The status of features[1] within minimalist syntax has changed over the course of time. The underlying concept in classical Minimalism (Chomsky 1993, 1995a,b) is neatly captured by the notion *building block*, inspired by Kobele (2005). Metaphorically speaking, features are the atoms from which bigger molecules, that is LIs, are assembled. Just like atoms, they exist on their own and can be manipulated separately. Therefore it is perfectly possible to move only formal features after Spell-Out and nothing else, leaving the LI in situ. Furthermore, syntax can directly affect the features themselves, for example by marking them as invisible at LF. The introduction of Agree in Chomsky (1998), on the other hand, marks the change to a characterization of features as properties. It is no longer possible to move features without moving the whole LI, nor is syntax capable of manipulating any attribute of a feature except its value. Just like the mass of a particle, features cannot be separated from their host or be affected independently, and they also can't be present more than once in an LI, which is at least in principle compatible with the metaphor of features as building blocks.

Under their most common reading, both views of features imply a one-to-one relation between features and linguistic phenomena.In layman's terms this means that every feature should represent at most one aspect of natural language. The feature PAST, for instance, represents a certain aspect of language we want to capture explicitly, just like case or

[1] In best minimalist tradition, I will neglect PF- and LF-features, devoting my attention entirely to formal features.

categories. I am just highlighting the habit of Minimalists to represent integral parts of natural language as features, by no means am I adding an unprecedented assumption to the properties of features.

The one-to-one relation implicitly entertained by Minimalist researchers is a direct consequence of the assumptions on the nature of features and those traits of language, which I will now simply refer to as functions. Features are taken to be relatively independent of each other, so we do not want to map any function to anything bigger than a feature. If we did, we would accept the possible existence of a feature f_i that does not have any contribution to make unless the structure contains a feature f_j, too. This would amount to a direct attack against compositionality. As far as functions are concerned, they are by convention atomic too and hence cannot be spliced into smaller units, so we should not map functions to any non-atomic units. Considerations of parsimony finally dictate that no two different features may represent the same function. Taken together, this yields (19).

(19) *One-to-one relation between features and functions* (F→F)
Every feature represents exactly one function and every function is represented by at most one feature.

Could this principle be strengthened, such that we arrive at (20)?

(20) *One-to-one correspondence between features and functions* (F↔F)
Every feature represents exactly one function and every function is represented by exactly one feature.

The short answer is "no". The main difference between both principles lies in the relevance of structural configurations. The weaker formulation, F→F, allows certain functions to be expressed in a purely structural way, e.g. Θ-roles.

But didn't I just say above that functions cannot be decomposed, and that we therefore should not represent them by any decomposable unit?[2] Yes, indeed, but the crucial point to observe is that the set of atomic elements does in fact comprise more than just features, e.g. chains, which consist of complex structure yet can't be decomposed. Doing so gives a set of identical LIs, but none of them is in any way relevant for the properties of the chain. Nor can a chain be reduced to the rule of chain-formation, which would be as pointless as reducing all features to the process which added them to UG many tenthousand years ago. So F→F is still an adequate summary of the preceding discussion.

Adopting F↔F, on the other hand, is equivalent to banning non-feature-like constructs from narrow syntax. This might be a wise move in the case of Θ-roles, but a lot of interesting research on the interaction of syntax and semantics would then be ruled out as well, e.g. Svenonius (2002). For this reason, F→F is a more accurate representation of the underlying intuitions of the community, and I will stick to it for the rest of my thesis.

[2]There might of course be functions that seemingly are not atomic, e.g. if it turned out that LIs are classified as adjectives by virtue of spanning less structure than verbs. But then it is mainly a matter of definition what the function actually is, the concrete, discrete category label, or the interaction of structure that results in different behavior for different LIs.

Now let us see what concretely qualifies as a feature according to F→F, and what does not. Imagine a feature-like object which marks an LI as both definite and singular. Such an object collides with the postulated homomorphism, as both functions can already be expressed by separate features whose distribution isn't necessarily the same. That nothing qualifies as an atomic building block if there are already two of them whose conjunction serves the same purpose is plausible from an intuitive point of view, too. If, however, we consider a language $\mathcal{L}_i$ where all definite LIs are singular and all singular LIs are definite, it makes sense to treat a feature f_i encoding both singularity and definiteness as a building block. As the features can never surface separately, one should also dispense with the separate features f_g and f_k denoting singularity and definiteness, respectively.

There are two alleged counterexamples to F→F. The first one is constituted by work like Pesetsky and Torrego (2001, 2007) and Kratzer (2004), where it is assumed that one and the same feature can serve very different functions on different LIs, depending on the categorial feature of the LI. Pesetsky and Torrego assume that nominative case and tense are related, following a suggestion by Williams (1994), whereas Kratzer relates accusative case to telicity. This apparent conflation of functions, though, does not entail that the relevant feature represents two or even more functions. Rather the function encoded by the feature is very abstract and may be influenced indirectly by the presence of other features; in the specific instances above, the value of the categorial feature has an effect on the interpretation of the "ambiguous" feature. Crucially, that is not a property of the feature itself but of its interpretation at the interfaces.[3] Paraphrasing what I just said, Kratzer or Pesetsky and Torrego do not claim that one feature accounts for two phenomena, they maintain that there is actually only one phenomenon, although surface properties may lead us to expect otherwise.

The second problem concerns dependencies between features: what if an LI may contain f_1 without f_2, but not the other way round? Are both features, none, or only one of them, and if the latter, which one? And what are the ramifications for F→F? First of all, let us exclude all the cases where f_2 is always present when f_1 is, but the effects of f_2 unfortunately can't be detected in all instances. Even if UG did in fact list both features, F→F would then force us to replace them by a single feature f_{1+2} which exhibits a special behavior in some situations. For example, some parts of the denotation of a feature may already be entailed by the remaining structure. If in some of those cases the feature is spelled out differently, a situation arises where a single feature seems to consist of two.

But this is not the logical possibility the question forces us to address, so let us return to the main case, f_2 always showing up together with f_1, but not the other way round. Such dependencies between features cannot threaten F→F either. Firstly, the interpretation of f_2 might rely on some element that has to be supplied by f_1. As can easily be verified, this

[3]The assumption that the content of a feature and its interpretation at the interfaces may diverge is needed independently. Just think of a third person feature. At LF, we want it to be an identity function introducing some presupposition, whereas at PF it is needed for the correct inflection. Furthermore, we want the feature to give rise to different morphology depending on whether its LI is a pronoun or a verb. Those in principle unexpected quirks should not be attributed to the feature but to the interpreters at the interfaces.

is just a generalization of my account for feature ambiguity in the sense of Kratzer and Pesetsky&Torrego, but now f_1 is obligatory in some instances and optional in others for independent reasons. Secondly, that f_2 depends on the presence of f_1 isn't necessarily due to the functions of the features, a feature geometry reflecting well-formedness conditions on LIs (Harley and Ritter 2002a,b) could cause a similar behavior. Whether one assumes such a construct is a purely empirical issue and completely independent of F→F.

From this discussion of possible counterexamples, I conclude that F→F is a good representation of Minimalists' take on features and that it does not face any insurmountable challenges, neither conceptual nor empirical ones.[4]

Speaking of empirical challenges, some readers are perhaps missing considerations of a distinctively more grounded nature, namely in how far morphological reflexes should be relevant to the postulation of (formal) features, given that a very strong connection between morphology and syntax is already assumed in Chomsky (1993). Chomsky argues that LIs enter the derivation fully inflected, that is to say with all their formal features specified. Those features have to be checked against various inflectional heads and whence drive movement.

First of all, Chomsky's idea should not be confused with accounts which try to establish a connection between the richness of the morphological paradigms of a specific language and which movement happens overtly in that language (Vikner 1997; Rohrbacher 1999). The interdependence they maintain between syntactic movement and morphological richness is rather puzzling from a conceptual point of view and faces several empirical problems (cf. Bobaljik 2003). Therefore one should not ascribe too much importance to the overt realization of a formal feature within a specific language. Nevertheless the relevance of morphology on a universal level must not be underestimated. If we adopt the null hypothesis that the mapping from features to sounds is arbitrary, it follows that for every feature there must be some overt reflex in some language. A formal feature that never surfaces overtly in any language of the world is at odds, although it is not unthinkable considering that the set of possible languages most likely is a superset of the set of languages known to us.

Returning to the assumption that LIs enter the derivation fully inflected, let me briefly discuss the criticism it was subjected to. Halle and Marantz (1993) note that the architecture proposed in Chomsky (1993) draws a dubious distinction between inflectional heads which are not pronounced in general, and other terminal nodes which are. However, this argument loses its force with the abandonment of Agr-nodes in Chomsky (1995b), because C, T and v do show PF reflexes even in English, at least sometimes.[5]

---

[4]From an epistemological point of view, this was to be expected anyway, for F→F is an axiom. It is indispensable for the partitioning of the empirical realm, so there is no empirical method to refute it.

[5]Whereas instantiations for C and v are rather trivial, the evidence for T being pronounced at rare occasions is slightly more intricate. Consider (i):

(i) He did not arrive yesterday.

Assuming that unaccusative verbs lack a vP-shell, supportive *do* in (i) has to reside in T.

But Halle and Marantz (1993) are not the only ones who doubt the validity of Chomsky's lexicalist assumptions, semanticists do too. Look at the example in (21), which is discussed by Kratzer (1998) and which she attributes to Irene Heim.

(21) Only I got a question that I understood.

This sentence has two readings, a strict one where I understood no question save the one that I got, and a sloppy one where nobody except me got a question that he understood. If the bound pronoun *I* in the embedded clause had entered the derivation with all its features specified, only the strict reading should be possible at LF, because a variable specified for first person arguably cannot be bound by any non-first person entity without giving rise to a presupposition failure.[6]Semanticists therefore commonly assume that pronouns enter the derivation without their morphological features specified (Heim 2005; Kratzer 2006). Yet there are accounts which opt for deletion of the features of a bound variable at LF, foremost Stechow (2003a,b). Evidently, such approaches work flawlessly within the framework of Chomsky (1993, 1995b).

In summing up we conclude that the stance taken in classical Minimalism isn't necessarily inferior to the feature valuation system of Chomsky (1998) on conceptual or empirical grounds. But still we should not put too much emphasis on the role of morphology when it comes to determining the set of features in natural language. Furthermore, I assert that there should be no instance where morphology could be more than just one of many indicators.

2.2 Feature Bundles

Now that we have familiarized ourselves with the very nature of features, we will turn to feature bundles, another concept whose specifics are rarely made precise. That, of course, is in line with the general indeterminacy of lexical structure within the Minimalist Program. Obviously the specifics of lexical structure form a huge area of (potential) research that can't be treated in all due detail here, but that isn't what I am heading for anyway. Rather, I want to determine the smallest set of conditions we have to stipulate for bundles and LIs in order to arrive at a working theory of syntax. Being aware of those conditions will in turn provide us with solutions to syntactic problems we did not even have the chance to look at yet, as we will see in 2.3.

The idea behind feature bundles is a very simple one: feature bundles are the pots into which features are put to keep them tightly together. Bundles are crucial for the operation Select (both s-selection and c-selection). If there were none, only categorial and semantic features would be selected from the numeration, and as a result, no derivation could converge, for the numeration would never be completely exhausted. If features are grouped in bundles, the selection of a feature is equivalent to the selection of the whole bundle, as required.[7] In addition, Merge depends on feature bundles, too. Merge can be

[6] I will give a more thorough discussion of the denotation of features in section 2.4.

[7] This is a stipulation, of course. It is based on the idea that a feature cannot leave the bundle

viewed as the complex function *label*(*concatenate*(LI_1, LI_2)) (cf. Chametzky 2000), so if LIs were not feature bundles, dozens of features would have to be taken as arguments, violating the diadicity of the operation. Bundles also play a role in pied-piping of features and hence movement, for they encode which features have to move simultaneously with the attracted feature such that the derivation does not crash. It follows logically from those three arguments that feature bundles are visible to narrow syntax and differ significantly from features in their syntactic status.

When it comes to turning the intuitive notion of feature bundles into a formal one, there are two competing views. Chomsky (1995a:392) explicitly treats them as sets, whereas Uriagereka (1998) defines them as matrices. To decide which one is a better fit, it helps to remind oneself that bundles do not have a lot of work to do except containing features. This is exactly what sets do, whereas matrices also encode linearity. Unless linearity is obligatorily needed in syntax as well as at the interfaces, we should stick with sets for reasons of parsimony.

Odd as it might sound, matrices nevertheless have an advantage over sets by virtue of not being sets. Consider (22), a set-theoretic definition of the structure of an LI.

(22) *Lexical Item* (LI) (to be revised)
$\mathcal{F} := \{f_i : f_i \text{ is a feature }\}$
$LI \subseteq \mathcal{F}$

In the set-based approach, an LI is a set of features, which gives rise to a rather perplexing conflict. Recall that BPS builds directly on set theory, that is, a tree is a graphical representation of a set consisting of either an atomic element and a set (Set-Merge) or two sets (Pair-Merge).[8] So how does one keep syntactic and lexical structure separate, although they are based on the same concept, namely sets? If we take our technical implementation seriously, that's anything but a trivial question, and the whole applicability of the set approach relies on a convincing answer to that question, as the consequences of a failure to do so would be devastating: feature bundles subjected to a structural analysis along the lines of BPS are not necessarily binary, whence misinterpreting them as a part of phrase structure is equivalent to looking at phrase markers where at least one non-terminal node is not binary branching. This is a violation of the Linear Correspondence Axiom (LCA) of Kayne (1994), and as every representation shipped to PF contains feature bundles,

by which it is immediately contained. However, this intuition does not apply to Select, given that selection of an element from the numeration isn't related in any way to movement, but to External Merge of a copy of the selected element plus reduction of the numerical index of this element by one (see 1.2 and 1.3). To account for this technical problem, we have to assume that features do not bear such a numerical index and that selection obligatorily has to reduce the index of the selected item — if there is no such index, the operation is illicit and the whole derivation crashes. Hypothesizing further that a parser should create as few crashing derivations as possible, no object will be selected if there is a bigger object containing it. It follows that nothing but LIs can be selected. This way we also prohibit the selection of FF[LI].

[8]For Pair-Merge, this isn't obvious at first sight, as the label is by definition a pair $\langle\gamma,\gamma\rangle$. But such pairs can easily be defined in purely set theoretic terms, such that $\langle\gamma,\gamma\rangle = \{\{\gamma\},\{\gamma,\gamma\}\} = \{\{\gamma\},\{\gamma\}\} = \{\{\gamma\}\}$. Therefore, a term generated by Pair-Merge is a set containing two sets.

virtually every derivation should crash at PF, the locus of the LCA. If this problem is unsolvable, adopting matrices instead of sets is a wiser choice.

One could stipulate a representational pattern matching algorithm which tries to find the typical structures associated with BPS, $\{\underline{a}, \{a, b\}\}$ and $\left\{\underline{\langle a, a\rangle}, \{a, b\}\right\}$, where a and b are LIs and $\underline{a}$, $\langle a, a\rangle$ are the respective labels. Whatever does not fit that bill, for instance the set $\{f_1, f_2, f_3, f_4, \ldots, f_{32}, f_{33}\}$, could then be considered an LI. This is anything but computationally efficient, and in addition, it crucially relies on labels, which renders it incompatible with current attempts to simplify phrase structure (Collins 2002). Even worse, it is prone to failure when encountering $LI_i := \{\{f_1\}, \{\{f_1\}, \{f_2\}\}\}$ or comparable LIs.[9]

We could considerably improve on that if the structure of LIs was more principled. Fortunately, that is exactly what Chomsky (1995a:392-394) assumes.[10]

> I assume that an item in the lexicon is nothing other than a set of lexical features, or perhaps a further set-theoretic construction from them (e.g. a set of sets of features). [...] We assume then that each lexical entry is of the form P, S, F, where components of P serve only to yield π (phonological features), components of S serve only to yield λ (semantic features) and components of F (formal features, e.g. the categorial features [±N, ±V]) may enter into computations but must be eliminated (at least by PF) for convergence.

If we adopt this proposal, we get (23).

(23) *Lexical Item* (LI) (to be revised)
$LI := \{PF, LF, FF\}$
$\mathcal{F}_{PF} := \{f_i : f_i \text{ is a PF-feature}\}$
$\mathcal{F}_{LF} := \{f_i : f_i \text{ is a LF-feature}\}$
$\mathcal{F}_{FF} := \{f_i : f_i \text{ is a formal feature}\}$
$PF \subseteq \mathcal{F}_{PF}$
$LF \subseteq \mathcal{F}_{LF}$
$FF \subseteq \mathcal{F}_{FF}$

Distinguishing the structure of an LI from BPS is now a trivial task. Every set $\{\underline{a}, \{a, b\}\}$ is a non-terminal node, and every set of the form $\{PF, LF, FF\}$[11] is a terminal node. If one uses a variant without labels, it suffices to redefine non-terminal nodes as sets of the

[9] The simpler $LI_j := \{f_1, \{f_1, f_2\}\}$ isn't problematic, because terminals in BPS are now sets. Hence the confusion should only arise with LI_i, not LI_j.

[10] Uriagereka (1998:250) assumes a similar structure for matrices.

[11] Two minor points are in need of clarification. First, LF and FF are disjoint sets, i.e. interpretable features belong to FF, but not LF. Second, an LI can be defined as a set instead of a triple without any side-effects for the following reason: syntax obviously recognizes syntactic features, so it has no problem with detecting which set FF is. Similarly, it is aware of the interpretability of features at LF (cf. section 2.4), so it is perfectly able to recognize LF. It follows that if the content of one of the sets cannot be read by syntax, this set is PF. Spell-Out is a syntactic operation and hence recognizes the sets correctly, too. At LF, an LI contains only interpretable material, so the partitioning of the features does not matter. The same reasoning applies to PF, if we maintain, *contra* Chomsky (1995b), that Spell-Out deletes LF-features in the PF-copy of the

form $\{a, b\}$. As even an LI with only two features won't ever have that form, no confusions arise.

When studying the definition in (23), most Minimalists will probably wonder why PF, LF and FF must not contain sets besides features. Sure, there may be no compelling reason for adding additional structure, yet nothing prohibits it either, and following minimalist traditions, we should shun every stipulation which is not forced onto us by conceptual or empirical necessity. Moroever, although nothing explicit is said about the microstructure of LIs in Chomsky (1995b), Chomsky (1998:40) insinuates that the set of ϕ-features is a proper subset of FF: "We take deletion to be a 'one fell swoop' operation, dealing with the ϕ-set as a unit. Its features cannot selectively delete: either all or none." This is additional support for generalizing the current definition of LIs such that an LI is a set of the three sets PF, LF, FF, which in turn may contain features and sets of arbitrary complexity.

(24) *Lexical Item* (LI) (final version)
$LI := \{PF, LF, FF\}$
$\mathcal{F}_{PF} := \{f_i : f_i \text{ is a PF-feature }\}$
$\mathcal{F}_{LF} := \{f_i : f_i \text{ is a LF-feature }\}$
$\mathcal{F}_{FF} := \{f_i : f_i \text{ is a formal feature }\}$
FF is either the empty set or a finite subset of the smallest set M such that

(i) $\forall x[x \in \mathcal{F}_{FF} \cup \{\emptyset\} \rightarrow x \in M]$, and

(ii) $\forall x[x \subseteq M \rightarrow x \in M]$

Analogously for PF and LF.

Allowing LIs to be highly structured introduces some notational hurdles. In principle, we can no longer write $f_i \in LI_i$ to designate some feature of LI_i, because f_i could belong to a set $M \in LI_i$. I will therefore adopt my notation such that ϵ denotes the transitive closure of $\in$. That is to say, if f_i is a member of set A, I still write $f_i \in A$, and $f_i \notin A$ if it is not. If f_i is a feature contained by A, no matter how deeply it is embedded, I write $f_i \epsilon A$. This covers both $A := \{f_1, \ldots, f_i, \ldots, f_n\}$ and $A := \{f_1, \ldots, f_n, \{\ldots, f_i, \ldots\}\}$ and other variants.

(24) licenses more complex structures than (23), e.g. LIs with all their ϕ-features grouped together as a subset.

(25) {{PF-features}, {LF-features}, {FF, {ϕ-features}}}

But that is just the tip of the iceberg, (26) is licensed by (24), too.

(26) $\{\{\text{stressed}, \{\text{vowel}, \{\text{ATR}, \{\text{back}, \text{high}\}\}\}\},$
$\{\text{alive}, \{\text{mammal}, \emptyset, \{\text{cat}, \{\text{tomcat}\}\}\}, \{\text{black}, \text{white}\}, \{\{\{\text{has stripes}\}\}\}\},$
$\{\text{N}, \{\text{masculine}, \text{third person}, \text{plural}\}\}\}$

---

phrase marker. Consequently, there is no risk of confusing sets at any point, even if their order with respect to each other is not fixed.

Such complex structures allow us to emulate both feature geometries and feature hierarchies, which are then to be interpreted as additional conditions on the well-formedness of LIs imposed by other components of grammar.[12] Owing to the new complexity of LIs, a serious question arises concerning parsing. If they can indeed have such a rich structure, don't we predict that trees are parsed top-down at the interfaces? Only by proceeding in a top-down manner does the parser recognize an LI immediately when it has reached it. Bottom-up parsing is more demanding, as an LI may contain a set which is itself tripartitioned, wherefore the parser can never be sure whether he has already left the LI and entered syntactic structure. If resource usage matters at the interfaces, a top-down parser should have to be used. This reasonsing, however, holds just in case syntax could not classify features. If formal features can be told apart from LF- and PF-features (cf. fn. 11), the problem does not arise because the highest level of an LI can be unambiguously determined in a strictly local fashion.

Our pattern-matching algorithm can effectively work with both definitions, (23) and (24), even if an LI lacks features of a specific type, e.g. PF-features, because the LI still contains three subsets, one of them the empty set. Unfortunately we encounter a problem as soon as the two empty sets are present. Particles merged at PF, for example, may lack both semantic and formal features. Some semantic operators arguably contain nothing but LF-features, and some inflectional heads might only have formal features (at least in classical Minimalism). But by definition a set never contains the same element twice, hence those LIs are of the form $\{\{f_1, \ldots, f_i\}, \emptyset\}$, rather than $\{\{f_1, \ldots, f_i\}, \emptyset, \emptyset\}$. This structure isn't recognized as an LI by the parser and again disaster is bound to strike.

We can easily overcome the problem if we assert that LIs are multisets (just like the numeration). In a multiset, an element can occur multiple times, wherefore $A := \{\{f_1, \ldots, f_i\}, \emptyset, \emptyset\} \neq B := \{\{f_1, \ldots, f_i\}, \emptyset\}$. Adopting multisets also offers us an elegant solution to our original problem, namely how to distinguish LIs from BPS. If LIs are multisets, we just need to add the provision that syntax and the interfaces can distinguish between sets and multisets.

In addition, multisets are able to account for yet another problem, namely covert movement. Covert movement is pure feature raising, i.e. a copy of the set of formal features is merged with syntactic structure in the covert cycle. If we look at LIs as sets, this puts us back to where we started. The set of formal features with its flat structure is more or less a variant of an LI as it was defined in (22). Logic dictates that this requires the resurrection of the first algorithm to prevent disaster. Well, logic did not scrutinize the facts with due care. In Minimalism, movement creates chains which are visible and parsed at the interfaces, and chains can thus encode the origin of some element in a straightforward way. The only thing left to do for the parser is to determine the status of the element in its original position, the rest is taken care of by the Uniformity Condition (see 13b). However, chains face some serious problems on their own, for they do not fit

[12]This allows for a conception where the lexicon functions more or less like an unrestricted generator whose final inventory is directed by linguistic input during language acquisition and the restrictions of the computational system.

well into a derivational framework (Brody 2002) and are technically flawed (Epstein and Seely 2006). Besides, having some very fundamental and mostly theory-neutral issue like the detectability of terminal nodes depend on a specific high-level device like chains does not look like a farsighted decision to me. We can spare ourselves the complications and just assume that FF is a multiset, too. That way, we also adhere to the metaphor of features as building blocks which insinuates that features may be present more than once.

But multisets also open the door to new complications. Consider an $LI_\emptyset := \{\emptyset, \emptyset, \emptyset\}$. We definitely don't want it to enter the numeration, as it can neither be s-selected nor c-selected. As a result, a numeration containing $LI_\emptyset$ will never be exhausted and the derivation won't converge. If LIs are sets, $LI_\emptyset := \{\emptyset\}$ is no licit LI anyhow and the whole issue does not arise. If LIs are multisets, though, we would better come up with a neat solution. Either we add an additional well-formedness condition on LIs, according to which every LI must contain at least one feature, or we stipulate that such an LI would never enter the numeration in the first place. I will do the latter, referring to Chomsky's HEOC discussed in section 1.4.[13]

Independent of multisets, a rather perplexing asymmetry arises as the immediate result of allowing LIs to have subsets. Neither features nor subsets can leave the set they are contained by, except in the case where the subset in question happens to be FF. If an LI is a set just like any other, why can its subset FF move separately? And why is the same impossible for PF and LF? Apparently this is due to the conception of covert movement in Chomsky (1995b) and nothing else. There is little use in arguing about the soundness of that assumption, so I can't do much but capture this quirk explicitly.

(27) *Impermeability of bundles*
For all features f and all sets M, N, such that $f \in M \epsilon N \in LI$:
a. f cannot move out of M, and
b. M cannot move out of N, and
c. N cannot move out of LI, unless $N = FF$

To sum up, we saw that bundles must be visible to syntax. Treating LIs as sets of features initially creates some tough challenges when it comes to keeping the distinction between LIs and phrase structure easily tractable, but this can be rectified by defining them as multisets. As an additional benefit we don't need any specific algorithms, nor are we locked into a specific kind of parsing method. Further, we do not have to depend on chains and we do not encounter any problems with LIs whose features belong to only one class. We can faithfully maintain that we have found a powerful yet simple alternative to matrices that does not obligatorily comprise linearity. In the next section, we will use our

[13]The details work out as follows. Suppose that the numeration contains pairs of LIs and numerical indices, such that two numerations $N_1 := \{\langle LI_i, m\rangle\}$ and $N_2 := \{\langle LI_i, m\rangle, \langle LI_\emptyset, n\rangle\}$, $m, n \in \mathbb{N}^+$, can be constructed. By the HEOC, all derivations based on those numerations have to be constructed in order to determine the effect of $LI_\emptyset$ on the output. If we refine Chomsky's ideas and maintain that a derivation crashes if all possible operations have been carried out but the numeration still isn't completely exhausted, we conclude that N_1 will always be preferred over N_2.

new knowledge about the properties of feature bundles and the structural conditions on LIs for a simpler implementation of feature strength.

2.3 Strong Features and their Consequences

Chomsky (1995b) does not give a complete list of formal features, stressing that there just isn't enough research done on that topic. But he contends that FF comprises at least the following features (Chomsky 1995b:277):

(28) a. categorial features
b. ϕ-features
c. Case features
d. strong F, where F is categorial [my own notation: $F := f \in \mathcal{F}_{FF}$; TG]

I do not have a lot to say about (28a) to (28c). Categorial features are mainly needed for subcategorization and may thus proof superfluous provided that somebody comes up with a different version of the operation Select which does not need c-selection in addition to s-selection. They also fall short on grounds of explanatory adequacy, as they are just a restatement of the facts. Regarding Case features, they were one of the main mechanisms for triggering movement, but with the introduction of Agree in (Chomsky 1998) this duty was delegated to the EPP-/OCC-feature (Chomsky 2004), so Case features lost importance. Svenonius (2002), among many others, started their foreseeable demise and Sigurðsson (2003, 2006a,b) tries to get rid of them completely. Concerning ϕ-features, their microstructure recently entered the focus of linguistic research (cf. Adger and Béjar to appear), so new results can be expected soon. Diverse as the current state of (28a) to (28c) might be, we see that each of them is at least partially a matter of empirical research. As such, they are beyond the scope of my thesis.

Strong features, on the other hand, are purely theory internal constructs and merit further discussion. Chomsky (1995b) entertains a strict division between an overt and a covert cycle, which are separated by a single Spell Out. Strong features are used as a device to enforce overt movement, i.e. movement before Spell Out. If a strong feature isn't checked immediately after entering the derivation, the derivation is canceled. Whether this behavior is due to PF- or LF-requirements is issue to debate — Lasnik (1999) gives reasons to prefer an analysis whereby properties of PF are the main culprit.

Feature strength has been subjected to a lot of criticism from the very beginning. Various arguments have been made against its empirical implications (cf. Sauerland 1995; Broekhuis 2000), and the soundness of the whole concept has been questioned, too. Both Lotfi (2002) and Þráinsson (2003) object that the methodology behind feature strength is fundamentally flawed.

(29) *Tautology of strong features* (Lotfi 2002:6)
If a moves then a moves and [if it is not the case that a moves then it is not the case that a moves and F moves iff two features enter a checking relation.

Paraphrasing (29), strong features are merely a description of the basic facts without any added explanatory value. If it is assumed that LI_1 moves overtly, one assigns its attractor LI_2 a strong feature, and if it is the case that LI_2 has a strong feature, then LI_1 moves overtly. This is one of those coding tricks Chomsky explicitly condemns (Chomsky 1995b:224).

Leaving aside its tautological status, we see that feature strength is rather odd in another respect. Consider (30).

(30) If F is strong, then F is a feature of a nonsubstantive category and F is checked by a categorial feature. (Chomsky 1995c:232)

If we assume that a feature can be checked only if it is identical to its checker (as Chomsky does, see 2.5), it follows from (30) that strong features are categorial. This is explicitly confirmed by Chomsky (1995b:232): "If so, nouns and main verbs do not have strong features, and a strong feature always calls for a certain *category* in its checking domain (not, say, Case or ϕ-features)." Recall that F→F, Minimalists' underlying intuition of features, establishes a tight relation between features and functions. Apparently it is not respected by strong features, for they are not only strong, but also categorial.

Confronted with this rather unexpected clash, we may seek help from other linguists who tackled the issue in their work. Unfortunately, this does not prove helpful at all, quite to the contrary. One encounters claims about heads having a strong Case feature (Bobaljik and Jonas 1996; Collins and Þráinsson 1996) or a strong wh-feature (Müller 1999), which obviously isn't in agreement with Chomsky's quote above. The only piece of advice we can gain from this detour is that we should not restrict our attention to the details of feature strength in Chomsky (1995b), because the concept has been generalized in such a way that any feature can be specified for strength. This puts F→F even more at odds than the original proposal.

Yet we do not want to give up on our axiom, as it is more or less the only guiding rule we have for what constitutes a good feature system in Minimalism. However, if it is impossible to reconcile feature strength with F→F, one of them has to go, and at first sight it is more likely that feature strength won't be the one to be dropped, as it forms an integral part of Chomsky's technical machinery. Fortunately, the first option, redefining feature strength such that it does not conflict with F→F, is indeed valid. Incidentally, this move is needed anyhow if we want to prevent feature strength from undermining the internal machinery of the Minimalist Program. Let me explain this in more detail. The field of linguistics is home to a multitude of feature based systems of differing complexity that can be hierarchically grouped as follows (cf. Adger 2006):[14]

(31) *Types of features (from less to more complex)*

[14] Adger's hierarchy is challenged by Asudeh and Toivonen (2006), who observe that it is not *a priori* evident that two grammars that differ in no aspect but their feature system do not show the same level of power. Rather than a real argument, that is an example of how simplicity and economy can be understood in different ways — although Asudeh and Toivonen's notion of simplicity is at least well-defined.

a. privative features

b. binary features

c. non-recursive attribute-valued features

d. recursive attribute-valued features

Privative features are the simplest option, they have no values but are either present or not. Such a system is entertained by Government Phonology (Kaye et al. 1985, 1990), for example. Binary features are familiar from Chomsky and Halle (1968), they can be valued [+] or [−]. Non-recursive attribute-value systems are frequently used in morphology, for instance when specifying a specific verb form as [Category: V; Tense: Present; Number: Singular; Person: 3^{rd}]. This system is also being used in Minimalist syntax since Chomsky (1998). The most powerful framework is provided by recursive attribute-value feature systems, which are a distinctive trait of Head-driven Phrase Structure Grammar (Pollard and Sag 1994). In such a system, it is possible for a feature to take another feature or feature bundle as its value, resulting in very powerful feature structures that can do a lot of work which is handled by separate syntactic mechanisms in the Minimalist Program.

Chomsky (1993, 1995b) probably uses a mixture of privative, binary and valued features. It must not use a recursive attribute-valued feature system, as most of its operations would then be devoid of virtual conceptual necessity. That is to say, if most of the technical apparatus can be emulated by complex feature structures, methodological concerns force us to simplify one of them. Introducing second-order features, that is diacritic features like [±strong] that assign additional properties to features, is a kind of recursive feature system. It might be restricted to special features and does not allow for more than one instance of recursion,[15] but as soon as this is a valid option in our theory, we have to explain why it is restricted to special features, why deeper embedding is impossible and so forth. If the reader still has some doubts, he or she may take a look at the following two notations for a strong feature, and think off a reason why they should not be considered equivalent:

$$[f_i^S : y] \equiv [\text{strong} : [f_i : y]]$$

The dilemma is rectified in Chomsky (1998) by abandoning feature strength altogether. Instead, movement is said to be induced by an EPP feature taking a feature type as its value. That the value of the EPP feature is restricted to general types such as Case or Person, but ignores the actual value of the feature, prevents the issues depicted above.[16] Yet this workaround forces us to commit ourselves to an attribute value feature system, for which there is no compelling reason in the older framework of Chomsky (1995c). But due to our refined notion of feature bundles, we can accommodate a feature neutral alternative.

(32) *Feature strength*

[15] Actually, Chomsky's treatment allows for two instances of recursion, because checking of a strong feature entails marking it for removal at Spell-Out.

[16] Note though that we are then forced to treat the wh-feature as a discrete feature, whereas its interpretable counterpart, Q, should be the value of the clause type feature of C^0.

> If F_i and F^S are members of a feature bundle B, F^S privative[17] and interpretable, then the closest feature F_j which is identical to F_i has to move immediately to check F_i (pied-piping whatever is needed for reasons of convergence). Else the derivation is canceled.

The logic behind (32) is simple. Strong features merely force the immediate deletion of every feature that is a member of the set they are a member of. From this point of view, a strong Case feature is actually a bundle $B := \{\text{Case}, F^S\}$. F^S itself won't be checked, as we take it to be interpretable (it denotes the identity function). That is in line with the claim in Lasnik (1999) that strong features have to be checked due to requirements at PF, not LF. What more to say, not allowing F^S to be checked has the distinctive advantage that we need not concern ourselves with the question which features may under which circumstances check F^S, which is a vacuous question anyhow considering the doubtful status of feature strength. From a practical point of view, (32) does the same work as (30).

It might be objected that (32) still introduces bloat into the feature system. This criticism is only partially warranted. The new implementation requires nothing but that FF may contain subsets and that syntax is sensitive to the presence of bundles. Both assumptions were already established independently. We also do not need to worry about the new implementation of feature strength obsoleting parts of the Minimalist apparatus, for the bundles themselves do not interfere with syntax as long as no feature makes us of them. All the power resides in F^S, so keeping the number of such privative features small guarantees an unchanged syntax. I have to admit, though, that the system can in principle model recursive attribute-value systems if the definition of LIs and the number of such diacritic features is adopted accordingly. Nevertheless I chose to implement feature strength in this way, because the idea can be generalized to other diacritic features, as we will see later on. This is not the case for Chomsky's EPP-based account.[18]

Cautious readers may further point out that another, rather special assumption is indispensable, namely that the numeration is not only able to add feature bundles to an LI, but also to generate a completely new bundle and fill it with features. This is due to the timing of the insertion of optional features. Intrinsic features might already come bundled with a strong feature from the lexicon, but optional features are not added

[17] If F^S was a binary feature [±strong], it would have a [-strong] feature itself, which would in turn have such a feature too, and so on.

[18] A third implementation is proposed in Gärtner (2002), where features aren't treated as atomic units but as 4-tuples $\langle A, V, i, j \rangle$, where A is the attribute, e.g. Case, V is the value, e.g. nominative, and $i, j \in \{0, 1\}$ such that $i = 1$ iff the feature is strong and $i = 0$ otherwise, and $j = 1$ iff the feature has been checked and $j = 0$ otherwise. This solution is straightforward and does not introduce the problem of recursiveness, but it isn't in line with $F \to F$, nor with general Minimalist intuitions. In section 2.6, we will see in particular that it isn't faithful of Chomsky's own ideas regarding checking. Finally, it is always a wise move to push existing tools to their limits before introducing additional machinery. In the worst case, we simply learn that we can't reach our goal without enhancing the technical apparatus. To the end of sticking with what we already have at our disposal, I opt for a feature and bundle based approach.

earlier to an LI than in the numeration. If an optional feature is strong, it has to be bundled with F^S in the numeration, it seems. This argument, though, is only conclusive if a certain optional feature can be present in both strong and weak instances within the same language. Otherwise the optional features may already be marked as strong or weak in the lexicon, that is before entering the numeration. One could even claim that the lexicon could contain two instances of an optional feature, the single feature and a bundles containing the feature and F^S. In the numeration, one of the two is added to an LI depending on its categorial feature. Finally, we can simply maintain that every FF contains a set $\{F^S\}$ to which optional features can be added in the numeration. That way we can stick to the assumption that the bundling of features can only be done in the lexicon, where it is indispensable for the assembly of LIs.

2.4 (Un)Interpretability

Besides feature strength, Chomsky (1995b) makes heavy use of another diacritic feature, namely interpretability. Naturally, the question arises whether this property of features should be handled the same way as feature strength. I conjecture that it does not.

There is one fundamental property that distinguishes interpretability from feature strength. The latter is language specific, hence coded in the lexicon (Borer 1984) and thus a feature by minimalist assumptions.The interpretability of a given feature F, on the other hand, is universal. Granted, F may be interpretable on a noun and uninterpretable on a verb, but even this variation will never be subject to parametric variation. True as this assumption may be, it is only an indication that we need not use diacritic features, it does not tell us whether there are viable alternatives to those second-order features. If there aren't any, interpretability should indeed be treated analogously to feature strength and I have to withdraw my previous claim.[19]

The first proposal is due to Chomsky himself, who implicitly treats interpretability as a part of UG. The interpretability of a feature in relation to a category is stored in UG, telling narrow syntax that e.g. ϕ-features are licensed on a noun, but have to be deleted when they belong to a verb. Feature strength is a parameter, so the values cannot be fixed in UG for every feature. Consequently, a lexical mechanism is needed to set the property.

Chomsky's treatment of interpretability is reasonably sound and definitely an improvement over encoding it by features. Unfortunately, it is not without its caveats. The Inclusiveness Condition demands that syntax has no access to any non-syntactic information that isn't encoded by features. Yet interpretability is usually considered an LF property, so it is rather questionable why syntax should have access to this information if it isn't represented by features. If we allow syntax to know about interpretability, why

[19]Asudeh and Potts (2004) propose to capture the distinction between interpretable and uninterpretable features by typing features. It remains to be seen how such an approach fares for the behavior of those features whose interpretability changes in accordance with the categorial feature of the LI. If there is an elegant way to implement this, it could be extended to strong features to circumvent the problems mentioned in the previous section.

does not it also know about QR or type shifting? Apparently there is something about interpretability which makes it less of a pure LF property than the latter two, so letus do a short investigation of this concept.

Interpretability allows us to partition the set of features into three sets. First those which are always interpretable, then those which are always uninterpretable, and finally those which can be both, depending on where they occur. Regarding the first group, it is interesting to note that being interpretable is not equivalent to being interpreted. Recall the discussion of Heim (2005) in section 2.1. There we saw that in some instances, ϕ-features may not be interpreted even if they are interpretable. The relevant example is repeated here for the reader's convenience.

(33) Only I got a question that I understood.

From this we conclude that an interpretable feature is a feature with a specific semantic denotation, but whether this denotation is actually interpreted depends on other conditions. That is to say, being interpreted implies being interpretable, but the entailment does not hold in the other direction. The denotation of at least some interpretable features is the identity function plus some presupposition (cf. Heim and Kratzer 1998; Sauerland 2003). Relevant examples (due to Heim 2005) are given in (34).

(34) a. $[\![\mathbf{sg}]\!] = \lambda x_e : x$ is an atom. x
  b. $[\![\mathbf{pl}]\!] = \lambda x_e : x$ is a plurality. x
  c. $[\![\mathbf{1st}]\!]^c = \lambda x_e : x$ includes s_c. x
  d. $[\![\mathbf{2nd}]\!]^c = \lambda x_e : x$ includes h_c and excludes s_c. x
  e. $[\![\mathbf{3rd}]\!]^c = \lambda x_e : x$ excludes s_c and h_c. x

If this treatment can be generalized to all interpretable features, we can claim to have a certain understanding of what constitutes an interpretable feature, although we have to keep in mind that this still does not correlate directly with LF-behavior. We are also still lacking an account for what the precise properties of an uninterpretable feature are, a more pressing question for syntacticians.

Without a doubt the most striking property of uninterpretable features is that they might not exist. This is an unexpected conjecture, but it definitely has a certain appeal. Consider the set of purely formal features, that is the set of those features which are always uninterpretable. The cardinality of this set is very low, as its only members are Case features and the EPP-feature. I already mentioned attempts to get rid of the former, or at least tie them closer to semantics, and comparable research is currently done on the EPP-feature (Epstein and Seely 2006). So it might well turn out in a few years that the set of purely formal features is the empty set.

As a consequence, the "androgynous" features remain as sole instance of uninterpretable material. ϕ-features are the most prominent example of this group. They are interpretable on nouns and uninterpretable on verbs. Yet it should be doubted that those features really have uninterpretable instantiations. Zeijlstra (2006) emphasizes that there is no way to determine that features of this group sometimes aren't interpreted. Consider the case of plural, which is uninterpretable on the finite verb of the sentence, but interpretable on the subject. Both categories are in a direct relation to each other, such that the verb cannot surface inflected for plural without a corresponding DP in its neighborhood. Given that logically $A \wedge A \Leftrightarrow A$, no one can prove that the plural feature on the verb isn't actually interpreted, because interpreting this feature does not add any new information.

If the allegedly special behavior of "androgynous" features is nothing but an artifact, i.e. if they are always interpreted, but in some cases they have no tractable effect on interpretation, we can treat them as purely interpretable features. And even if those features aren't always interpreted, this does not necessitate that they are uninterpretable, as we already saw above. If, in addition, the EPP and Case features are also shown to be interpretable, or even non-existent, interpretability is no meaningful category any longer, for all features (except PF-features), are then interpretable.

No matter whether Minimalist research will eventually reach this ambitious goal or not, it shows that the notion of interpretability does not neatly line up with LF behavior and is rather immature and bound to change. Therefore we should not needlessly concern ourselves with the contradictions and obscurities it might give rise to. Instead, I recommend to accept them as symptoms of our failure to fully capture the phenomenon. Although it is quite strange that syntax knows about interpretability even though it isn't encoded via features, we should not hesitate to assume it nevertheless.

2.5 The Primitives of Checking Theory

In the preceding sections, I was concerned with ensuring a stable ground on which I could base the next steps of the discussion. I focused especially on the status of feature bundles, using the technical implications of feature strength as a way to establish the usefulness of bundles. In the following sections, I will turn to the center piece of both Minimalism and this paper, namely feature checking.

I will start by providing the exact definitions (most of them are indeed exact) and analyzing the interaction between checking theory and movement. Later on, I will give a detailed treatment of Delete(α) and Erasure, resulting in various formulations of feature checking, some of them more plausible than others.

Let us begin with the introduction of the vocabulary we need in order to define the checking domain. (35)–(44) are taken from Nunes and Thompson (1998), who, to my knowledge, are the only ones to provide a set-theoretic definition of those basic notions. Concerning the definitions of domination and containment, a very important point needs to be mentioned beforehand. They are cited unchanged, although the wording is dangerously sloppy insofar as it is in no way obvious that (36b) and (37b) can be applied recursively. Yet if those conditions were not recursive, domination and containment could not reach further down the tree than three levels.

(35) *Syntactic object*
σ is a syntactic object if it is
a. a lexical item or the set of formal features of a lexical item, or

b. the set $K = \{\underline{\gamma}, \{\alpha, \beta\}\}$ or $K = \Big\{\underline{\langle\gamma,\gamma\rangle}, \{\alpha, \beta\}\Big\}$ such that α and β are syntactic objects and $\underline{\gamma}$ or $\underline{\langle\gamma,\gamma\rangle}$ is the label of K.

Given a syntactic object K such that $K = \{\underline{\gamma}, \{\delta, \mu\}\}$ or $K = \Big\{\underline{\langle\gamma,\gamma\rangle}, \{\delta, \mu\}\Big\}$:

(36) *Domination*
K dominates a syntactic object α if and only if

a. for every set L such that $L \in K$, $\alpha \in L$, or

b. for some set M, K dominates M and M contains α.

(37) *Containment*
K contains a syntactic object α if and only if

a. for some set L such that $L \in K$, $\alpha \in L$, or

b. for some set M, K contains M and M contains α.

(38) *Minimal Projection*
A syntactic object α is a minimal projection if and only if there is no syntactic object β such that α dominates β.[20]

(39) *Maximal Projection*
A syntactic object α is a maximal projection if and only if there is no syntactic object β such that β dominates α and β has the same label as α.

(40) X^{0max}
The syntactic object K is an X^{0max} projection if and only if

a. K is a minimal projection, and

b. there is no minimal projection L such that L contains K.

(41) *Max(α)*
Max(α) is the maximal projection P such that P dominates α and for every maximal projection $Q \neq P$, if Q dominates α, then Q dominates P.

(42) *Sisterhood*
The syntactic objects α and β such that $\alpha \neq \beta$ are sisters if and only if for every syntactic object such that γ contains α, γ also contains β, and conversely.

(43) *Specifier*
A syntactic object α is a specifier of the head H if and only if α is a sister of an intermediate projection P such that P has the same label as H.

[20] This definition is rather tricky, so let me elaborate on it. Consider the case where V has undergone head-movement and adjoined to T. As we are dealing with an instance of adjunction, the projected label is a set, so we get the term $A := \{\{\{T\}\}, \{V, T\}\}$. According to our definition of domination, A does not dominate V or T. Consequently, A (represented as T^0 in a tree structure) is a minimal projection.

(44) *Complement*
A syntactic object α is a complement of the minimal nonmaximal projection H if and only if α is a sister of H.

I continue with my own paraphrases of the relevant domain definitions in Chomsky (1993).[21] Note that domination as defined above isn't a reflexive relation, but of course this does not imply that it is illicit to make reference to its reflexive closure.

(45) *Domain of* α ($\mathrm{D}(\alpha)$)
Let $\mathrm{D}(\alpha) := \{x : x \neq \alpha \wedge x$ does not contain $\alpha \wedge \mathrm{Max}(\alpha)$ contains $x\}$ be the domain of α.

(46) *Complement Domain of* α ($\mathrm{D}_C(\alpha)$)
Let $\mathrm{D}_C(\alpha) := \{x \in \mathrm{D}(\alpha) : x$ is (reflexively) dominated by the complement of the X^{0max} projection (reflexively) dominating $\alpha\}$ be the complement domain of α.

(47) *Residue of* α ($\mathrm{D}_R(\alpha)$)
Let $\mathrm{D}_R(\alpha) := \mathrm{D}(\alpha) - \mathrm{D}_C$ be the residue of α.

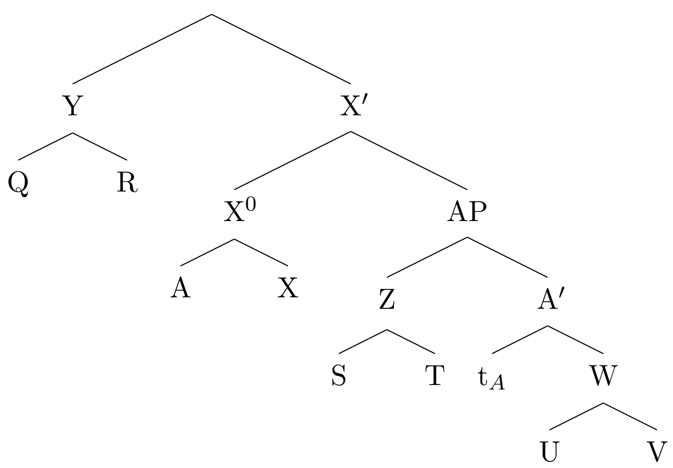
(48) *Minimal Domain of* α ($\mathrm{D}_M(\alpha)$)
Let $\mathrm{D}_M(\alpha)$ be the smallest subset M of $\mathrm{D}(\alpha)$ such that for every $x \in \mathrm{D}(\alpha)$ there is a $y \in M$ that (reflexively) dominates x. We call $\mathrm{D}_M(\alpha)$ the minimal domain of α.

(49) *Internal Domain of* α ($\mathrm{D}_I(\alpha)$)
Let $\mathrm{D}_I(\alpha) := \mathrm{D}_C(\alpha) \cap \mathrm{D}_M(\alpha)$ be the internal domain of α.

(50) *Checking Domain of* α ($\mathbb{C}(\alpha)$)
Let $\mathbb{C}(\alpha) := \mathrm{D}_R(\alpha) \cap \mathrm{D}_M(\alpha)$ be the checking domain of α.

Let us see how we can derive from (45)–(50) that $\mathbb{C}(\alpha)$ comprises α's specifier as well as anything adjoined to α, but nothing adjoined to $\mathrm{Max}(\alpha)$.

(51)



[21] The definitions given in Nunes and Thompson (1998) produce slightly different results with regard to first-merged constituents.

I will first determine $\mathbb{C}(\alpha)$ for α = X, as this represents the most common case. It is obvious that Max(X) is XP and that D(X) is {Y, Q, R, A, AP, Z, S, T, A', t_A, W, U, V}. Note that if some category YP was adjoined to XP, the lower XP would still be Max(X) because adjuncts have a different label generated by Pair-Merge instead of Set-Merge.

D_C(X) is a proper subset of D(X), namely {AP, Z, S, T, A', t_A, W, U, V}. It follows that D_R(X) contains Y, Q, R, and A. The members of D_M(X) are a little bit more difficult to retrieve. Trivially every node in D(X) is reflexively dominated. We are looking for the smallest subset, though. This subset is formed by the nodes in D(X) which are only reflexively dominated by one member of D(X). The set D_M(X) therefore contains Y, A and AP. We can now enumerate the nodes within $\mathbb{C}$(X), as those are the nodes which are both in D_M(X) and D_R(X), in our case Y and A. And as the reader can see for himself, Y is the specifier of X and A a head adjoined to it.

If we want to know $\mathbb{C}$(A), we have to keep in mind that the checking domain of the head of a nontrivial chain is the checking domain of the chain, in our case $\mathbb{C}$((A,t_A)). The domains of (A,t_A) are as follows:

(52) a. D((A,t_A))= {Y, Q, R, X, Z, S, T, W, U, V}
b. D_C((A,t_A)) = {W, U, V, Z, S, T}
c. D_R((A,t_A)) = {Y, Q, R, X}
d. D_M((A,t_A)) = {Y, X, Z, W}
e. $\mathbb{C}$((A,t_A)) = {Y, X}

Y and A are in a spec-head-relation and X is the head to which A is adjoined. Hence the predictions for nontrivial chains are the same as for trivial ones.

I conclude that Chomsky's proposals on how to specify $\mathbb{C}(\alpha)$ do indeed work as expected and no further revisions are necessary. There are many pieces of the checking theory left to scrutinize, though. First of all, nothing has been said yet about how a functional node may get the opportunity to check its features. In other words, I didn't talk about the connection between checking and movement yet. Thus a cautious look at the concept of attraction and related terms is in order.

(53) *Term* (Chomsky 1995b:247)
For any syntactic object K,
a. K is a term of K;
b. if L is a term of K, then the members of the members of L are terms of K.

(54) *Sublabel* (Nunes and Thompson 1998:513)
σ is a sublabel of K if and only if
a. σ is a formal feature of a term of K, and
b. K is an X^{0max} projection.[22]

[22]Observe that owing to the second condition, K being an X^{0max} projection, the definition of D_C introduces some redundancy into the system. I defined D_C such that it comprises only the complement of X^{0max}, even for nodes adjoined to it. This however, isn't actually necessary. If

I will abbreviate "a sublabel of K" as $\int$(K).

(55) *Attract* (cf. Chomsky 1995c:247)
A term K attracts a feature F if F is the closest feature that can enter into a checking relation with $\int$(K).

(56) *Closeness* (cf. Chomsky 1995c:356)
If β c-commands α and τ is the target of raising, then β is closer to τ than α is to τ, unless β is in the same minimal domain as τ or α.

(57) *Conditions on Attract* (Chomsky 1995c:304)
Only the head of a chain CH enters into the operation Attract/Move. [...] Where CH is a a (possibly trivial) chain headed by α,

a. α can raise, leaving the trace t, a copy of α.

b. Formal features of the trace of A-movement are deleted and erased.

c. The head of CH can attract or be attracted by K, but traces cannot attract and their features can be attracted only under narrow conditions [...].

Sloppily paraphrasing (55), whenever an LI_i contains a feature $F_2(=\int(\mathrm{LI}_i))$, LI_i attracts the closest feature F_1 that can check F_2. F_1 then moves in order to enter a checking relation with F_2.

(58) *Checking Configuration*
F is in a checking configuration with $\int$(K), iff F is in $\mathbb{C}$(K).

(59) *Checking Relation*
F is in a checking relation with $\int$(K), iff F and $\int$(K) are in a checking configuration, and F and $\int$(K) match. If F and $\int$(K) mismatch, the derivation crashes.

A crucial point and at the same time the weakest one in (59) is the requirement that F and $\int$(K) match. As often before, Chomsky is imprecise when tackling issues that dwell below the level of syntactic terminals, and consequently feature matching remains a fuzzy concept. I shall try to improve on this.

In Chomsky (1995b) a distinction is entertained between *Match*, *Non-Match* and *Mismatch*. Chomsky states that two features match if they are identical and mismatch if they are incompatible. *Non-Match* refers to situations like an aspect feature and an accusative feature being in a checking configuration.

Ignoring for a second the technical aspects, we can safely assume a system along the lines of (60) to be a good replication of Chomsky's idea.

(60) *The Matching System* (Informal)
Two features F_1, F_2 stand in one of the following three relations to each other:

D_C contained only the complement of the current node, the checking domain of an adjoined head would be limited to its sister, yet its features would be sublabels of the relevant X^{0max} projection and could hence be checked by features belonging to syntactic objects in $\mathbb{C}(X^{0max})$.

a. *Match*
F_1, F_2 match iff they are identical.

b. *Mismatch*
F_1, F_2 mismatch iff they are not identical and in the same feature class.

c. *Non-Match*
F_1, F_2 non-match iff they are not identical and not in the same feature class.

Two aspects of (60) need clarification, identity and feature classes. Even though their precise characterization may be an empirical issue, it is tenable to provide simple definitions as preliminary approximations.

First we have to group all features into classes. Due to the lack of an extensive list of features, such a grouping will be anything but conclusive, wherefore the short compilation in (61) only serves the illustratory purposes.

(61) *Feature classes*
A feature F may belong to one of the following classes:
Case, Person, Number, Gender, Tense, Mood, Aspect, Voice, Clause Type, Category ...

I repeat, the overall number of classes as well as the categorization of specific features is an empirical matter. The intricacies of the latter are shown by attempts to reduce Case to tense (Pesetsky and Torrego 2001, 2007) or telicity (Kratzer 2004).[23]

Next we have to define identity.

(62) *Identity*
Two features F_1, F_2 are identical when replacing F_1 in LI_1 with F_2 affects neither LI_1's syntactic behavior nor its interpretation at the interfaces.

This principle is meant to be understood as a methodological guideline — obviously syntax isn't actually carrying out such calculations in order to determine whether two features are identical. The version of identity we entertain by adopting the definition is a very strong one, probably too strong for a recursive attribute-value system. Weaker versions may not require the feature values (if one assumes valued features) to be the same, but it is unknown in how far that might be empirically adequate.

With a more explicit concept of Match in place, (59) can be considered a viable basis for further definitions, which allows us to continue our exploration into the connection between feature checking and movement. According to Chomsky (1995b:269f), feature movement is constrained in various ways:

(63) *Constraints on Move F*

a. F is an unchecked feature.

b. F enters into a checking relation with $\int$(K) as a result of the operation.

[23] In an attribute-value feature system, the implementation of feature classes is straightforward. If one wants to use a privative or a binary feature system, it has to be stipulated that syntax knows which features belong to which classes. Explicitly encoding this knowledge would probably require some sort of typing for features.

c. FF[F], the set of formal features containing F, raises along with F.

d. A category a containing F moves along with F only as required for convergence.

e. Covert operations are pure feature raising.

(63a) and (63b) are a more precise formulation of Last Resort.

(64) *Last Resort* (cf. Chomsky 1995b:280)
Move F raises F to target K only if F enters into a checking relation with $\int$(K).

(63c)–(63e) are derived by Chomsky from bare output conditions, although the logic of the argument is debatable.

Interestingly, (63) does not prevent other features of FF[F_1] from checking further sublabels of the target. This has two non-trivial consequences: first, if $\int_1$(K) attracting F_1 is a strong feature, additional $\int_i$(K) can be checked by the corresponding $F_i \in$ FF[F_1] before Spell Out. Second, if $\int_j$(K) and $F_j \in FF[F_1]$ are mismatching features, the derivation will crash even though F_j is not attracted by $\int_j$(K).

Another defining property of checking in MP is its asymmetry, apparently an epiphenomenon of the overall layout.

(65) *The Asymmetry of Checking Theory* (cf. Chomsky 1995b:278)
$\int$(K) is always uninterpretable. The attracted F may be uninterpretable.

After having digested various definitions concerning checking domain and checking relation, the reader most certainly agrees with me that checking is strictly speaking not an operation, despite the derivational nature of the overall framework. Instead, it is a descriptive term for the effects of two features being in a certain structural configuration to each other. Yet I do not mean to insinuate that there are no operations involved in checking. Move evidently is an essential prerequisite. And ultimately, checking is about getting rid of features, which is accomplished by the feature deletion operations Delete(α) and Erasure. Those will be the main theme of the remaining sections.

2.6 Siamese Twins — Delete(α) and Erasure

It is commonly left implicit what ”checking a feature” actually means, as it is seldomly of any importance for the empirical analysis. I will do the very opposite and look at feature checking from various angles and define it in various ways. The general aim is to obtain as many sound formalizations of feature checking as possible, such that there is a broad spectrum of candidates to apply Reuland's proposals to.

Chomsky (1995b:280) bifurcates checking into two operations, such that:

(66) a. A checked feature is deleted when possible. (Delete(α))

b. Deleted α is erased when possible. (Erasure)

Two aspects of (66) have to be refined. It's neither apparent what the difference between deleting and erasing a feature is, nor do we know what the cryptic condition “when possible” is alluding to. The latter is clarified on page 280 of Chomsky (1995b) by the Principle of Recoverability of Deletion.

(67) *Principle of Recoverability of Deletion*
Interpretable features cannot delete, even if checked.[24]

It follows that uninterpretable features delete as soon as they are checked. However, (67) only covers Delete(α). Erasure of deleted features is subjected to different constraints which are in turn imposed by parametric variation and bare phrase structure. The former regulates how often an uninterpretable feature has to be deleted before it can be erased, whereas the latter disallows Erasure of the last formal feature, because this is said to be equivalent to erasing the whole term, which would corrupt the phrase structure.

The phrase structural ban against Erasure is quite odd. The underlying logic seems to be that a set is erased if it is empty. This conjecture is debatable and definitely incompatible with my assumptions on the structure of LIs and the status of bundles. Furthermore, the argument implies that the constraint is only relevant for LIs which contain nothing but uninterpretable features, that is no PF-features, no LF-features, nor any interpretable formal features. This description probably only fits covert expletives or overt expletives after Spell-Out, for only expletives may have an uninterpretable categorial feature. Irrespective of the empirical merits of the constraint, it simply cannot be derived from any phrase structural considerations. It is a stipulation, but that does not mean that we should not embrace it. To the contrary, it implies that we can easily adopt it without worrying about its soundness in our slightly different framework.

Chomsky's attempt to derive the constraint from phrase structure requirements is nevertheless illuminating, because it insinuates that Erasure of α is equivalent to removing α from syntax altogether. That erased material becomes inaccessible to syntax is additional evidence for this interpretation. Erasure thus represents the intuitive concept of deletion (not to be confused with Delete(α)).[25]

(68) *Erasure*
A checked feature F marked n times as invisible at LF, $n \in \mathbb{N}^+$ and subject to parametric variation, is eliminated from the syntactic phrase marker (and therefore inaccessible to further computation), unless F is the only remaining formal feature of its LI, save any F^S.

Hopefully, we will find an equally simple meaning for Delete(α). Chomsky seemingly offers us one when he claims that the operation will "leave the structure unaffected apart from an indication that α is not 'visible' at the interface" (Chomsky 1995c:250). That sure sounds simple: Delete(α) does nothing save marking a feature α as invisible at LF.[26] Now the conclusions of my discussion of feature strength in section 2.3 show further merits. Recall how I reimplemented feature strength with recourse to bundles, thereby evading any problems for $F \to F$. If we cling to Chomsky's claim that Delete(α) adds a property to features, namely invisibility, it is a troublesome operation for the very same reasons.

[24] The interpretability of features can be evaluated in syntax as discussed in section 2.4.

[25] For non-privative feature systems, we can also conclude that Erasure eliminates the whole feature, not only its value.

[26] Whether a feature is marked as invisible at PF too is immaterial here.

Building on our sharpened notion of feature bundles, we can rectify this shortcoming and define Delete(α) as an operation which creates an invisibility feature and bundles it with the checked feature.

This definition, however, is not the only conceivable one. In the case at hand, there are two ways to obtain the desired results. Either one proceeds as just proposed and bundles the checked feature F with an invisibility feature F^{INV}, or one stipulates that every FF comes with a subset containing a single F^{INV} to which a checked feature is copied before being erased. Both approaches have in common that the invisibility feature has to be interpretable, otherwise its deletion would be obligatory, which clearly would prevent it from serving its purpose at LF. At LF, the denotation of an (in)visibility feature is the identity function which has the feature it is bundled with both as input and output.[27]

(69) $[\![F^{INV}]\!] = \lambda f : f \in D_{\langle e,e \rangle}.f$

While being closer to Chomsky's wording, the first approach, according to which a deleted feature is bundled with F^{INV}, faces two severe threats. The first one is acknowledged by Chomsky himself when he points out that Delete(α) comes down to a violation of the Inclusiveness Condition (Chomsky 1995b:228). That is to say, adding a feature in the middle of the derivation isn't actually a valid operation. As features aren't syntactical nodes, they cannot reside in the numeration, waiting to be selected and merged. They have to come from somewhere else, which is forbidden by the Inclusiveness Condition.

The second problem is due to the way I chose to implement diacritic features. I assumed that a so-called "strong feature" is created in the lexicon by bundling F with a feature strength marker F^S which encodes that every feature which is a member of the same set as F^S is a strong feature. If this account is to be extended to Delete(α), we have to stipulate that narrow syntax is perfectly capable of creating new feature bundles. That is suspicious for various reasons. First, it isn't foreseeable in how far the expressive power of syntax would be affected by this move, second, it begs the question why we should stick to treating LIs as terminal nodes if syntax has no problem whatsoever to manipulate them by adding as well as deleting their structure and elements.

If one assumes a kind of "invisibility bag", on the other hand, those problems vanish immediately. No violation of the Inclusiveness Condition is induced, as $\{F^{INV}\}$ is an integral part of every LI and hence does not need to be added during the computation. The relevant copy mechanism is already an integral part of syntax, although up to now it could not copy single features. Reuland (2001), however, entertains this possibility anyhow, so there is no compelling reason why I should refrain from doing so. Erasure of features, finally, is needed independently, so it can easily be employed after the feature has been copied to $\{F^{INV}\}$. Just by making use of a more general version of the copy operation and an additional constraint on the structure of LIs, we therefore arrive at a more parsimonious system that does not require both a separate deletion mechanism and the concomitant bundling in syntax.

[27] The actual denotation may vary depending on whether features are bundled individually or simply copied into $\{F^{INV}\}$.

But there is still a third option waiting for us to be explored. Suppose that every feature in the lexicon was bundled with a visibility feature F^{VIS}, such that Delete(α) was simply Erasure of the respective visibility feature. Evidently, no violation of the Inclusiveness Condition is induced by Delete(α), as every F^{VIS} is already bundled with a corresponding feature in the lexicon. Nor is it necessary to allow any new bundling in syntax. Besides showing none of the conceptual issues of the first implementation, making use of visibility features is just as parsimonious as the second one, because it allows defining Delete(α) as the Erasure of F^{VIS}. Furthermore, the condition that a feature F has to be deleted n times, $n \in \mathbb{N}^+$, before being erased can easily be captured by bundling F with n instances of F^{VIS}. This is a very welcome result indeed.

Now that looks like a breath-taking victory for F^{VIS}. Unfortunately, the approach is not without its own acute problems. Its appeal lies in Delete(α) being defined as Erasure of F^{VIS}. For a feature to be erasable, it must not be interpretable. But if it isn't interpretable, by simple logic it is uninterpretable. If F^{VIS} were uninterpretable, it inevitably had to be erased before reaching LF, depriving all features of their visibility at LF and hence every sentence of its semantic interpretation. Now that is certainly something we do not want our theory to predict. But there is an easy, albeit sneaky, way out. Nowhere in Chomsky (1995b) does it say that interpretable features must not erase. Such a stipulation would be redundant, for it is already covered by two constraints, namely that only deleted features may be erased and that interpretable features cannot be deleted. But if we stipulate that F^{VIS} does not need to be deleted prior to its Erasure, restricting Erasure to uninterpretable features suddenly is a contentful condition on its own.

We are then faced with the simple decision, whether the PRD should carry over to Erasure as well or rather not. Conceptually, the former is the only reasonable move, thereby putting a sudden stop to any kind of F^{VIS}-approach. But as I already pointed out numerous times, this chapter isn't mainly about elegance but about establishing a testing ground for Reuland's theory. From this perspective it seems advisable to refrain from generalizing the PRD to Erasure for the sake of broadening the set of possible test scenarios, giving us a total of three different versions of Delete(α).[28]

(70) *Delete(α)* (Bundling version)
Deletion of F is equivalent to bundling F with an invisibility feature F^{INV}.

(71) *Delete(α)* (Bag version)

[28]Returning to Chomsky's newer account for feature strength, which assigns an EPP-feature a specific feature type as its value, we can now see why its generalization to feature checking is unfeasible. In the case of overt movement, it suffices to assign some Y's EPP-feature some value, e.g. Case, and locality conditions and checking requirements ensure that the right XP containing a Case feature winds up in Spec,YP. For marking the correct feature as invisible, though, we also need to know the value of said feature. Suppose LI_i contains two Case features, one specified for nominative, the other one for dative. Which one of those features is to be marked as invisible plays a crucial role, but Chomsky's account cannot be generalized in such a way that we are able to unambiguously specify the relevant feature without introducing a recursive attribute value system.

a. Every $FF \in LI$ contains a set $INV := \{F^{INV}\}$.

b. Delete(α) is copying of α to INV followed by the Erasure of the original α.

(72) *Delete(α)* (VIS version)

a. Every feature $F \epsilon LI$ is bundled with a visibility feature F^{VIS} in the lexicon.

b. Delete(α) is Erasure of $F_i^{VIS} \in \{\alpha, F_1^{VIS}, F_2^{VIS}, \dots, F_{i-1}^{VIS}, F_i^{VIS}\}$, $i \in \mathbb{N}^+$.

Now that we have all variants of Delete(α) in place, we must also slightly modify the definition of Erasure.

(73) *Erasure*

A checked feature F marked n times as invisible at LF, $n \in \mathbb{N}^+$ and subject to parametric variation, is eliminated from the syntactic phrase marker (and therefore inaccessible to further computation), unless F is the only remaining formal feature of its LI, save any F^S or F^{INV}.

2.7 On the Redundancy of Having the Same Twice

One question arises immediately when looking at (66), namely why do we actually need to split checking into two operations if the second one is totally dependent on the first one anyhow. From a minimalist perspective, it is even more puzzling that syntax — which is driven solely by bare output conditions — should bother to erase a feature after it has already been rendered invisible at LF and thus won't cause the derivation to crash. Obviously, a theory employing only one of the two, either Delete(α) or Erasure, would be more parsimonious.

Chomsky bolsters the division with empirical data, but the implications of his analysis are refuted by Nunes (2000), who asserts that Delete(α) alone can account for the data if a checked feature may not participate in further checking relations. Chomsky starts his argument for Erasure with a treatment of transitive expletive constructions (TEC) in Icelandic. Have a look at (74), taken from Jonas and Bobaljik (1993:76).

(74) Það lásu einhverjir stúdentar bókina.
There read some students book-the.
'Some students read the book'.

Chomsky suggests that the strong D-feature of T in Icelandic has to be deleted twice in order to be erased. First the subject *einhverjir stúdentar* moves from Spec,*v*P to Spec,TP and checks the strong feature of T, which is in turn deleted, but not erased. Then the expletive *θað* is merged with TP, checking the strong D-feature of TP for the second time and thereby inducing its Erasure. The Merge-over-Move condition wasn't introduced earlier than in Chomsky (1998), hence it isn't unreasonable to move the subject prior to merging the expletive.

Nunes (2000) notes that in the account for (74) it is the availability of two specifiers that matters, not how one creates them. Instead of stipulating the Delete(α)-Erasure-dichotomy and adding the proviso that in some languages features ought to be checked several times before they can be erased, one could also contend that Icelandic T has two strong, uninterpretable D-features, the first one being checked by the subject, the second one by the expletive. As Nunes (2000:418) puts it: "The point here is not to argue this is the proper way to handle TECs [...], but rather to show that it is possible to technically implement Chomsky's [...] analysis of TECs without resorting to [E]rasure." Concerning the validity of his claim, two non-trivial questions have to be sorted out: can an LI contain the same feature twice? And does a checking relation hold between exactly two features?

The answer to the first question is trivial from a set-theoretic as well as from a semantic perspective. We defined LIs and FF as multisets, so it is to be expected that sets contained in FF are multisets too. As a consequence, a feature may be present twice, in line with the view of features as building blocks. Even if those sets aren't multisets, a feature can be present more than once as long as it is a member of more than one set. As for semantics, I don't see how any issues could arise. If the feature is uninterpretable, it won't ever reach LF, and if it is interpretable, i.e. if it denotes the identity function and some presupposition, duplicating the presupposition does not hurt either.

What about the second question? Does a checking relation hold between exactly two features? Evidently this is a matter of definition which cannot be sorted out by recourse to economy conditions. Consider the case of Merge. On the one hand, it seems to be the least costly option to merge exactly two arguments at a time, on the other hand this increases the number of applications of Merge needed to create a converging derivation. Nor are methodological commitments of any help, as is again illustrated by Merge. Limiting Merge to two arguments appears to be the least powerful option, but it is definitely not the most parsimonious one, because the definition of Merge would be simpler without this additional stipulation. The situation is more or less the same for checking relations. That a checking relation holds between exactly two features is as plausible as that it does not. Let us assume the former.

After having settled the conceptual issues and found Nunes' account sound, we will now turn to expletive raising in English, another example provided by Chomsky and discussed by Nunes.

(75) [there$_i$ seems [t$_i$ to be a cat on the mat]]

According to Chomsky, *there* has neither Case nor ϕ-features, and its D-feature is uninterpretable. In other words, its set of formal features consists of nothing but its uninterpretable categorial feature. As a result, the categorial feature must not be erased after checking the strong feature of the embedded T in (75), because this would amount to erasing a term, which is forbidden. That allows *there* to raise and check the strong feature of matrix T, too. It's easy to figure out that the complexity of the analysis is due to *there* having an uninterpretable categorial feature. As soon as one assumes that the categorial feature is interpretable, the analysis is straightforward and does not need two different ways of checking a feature.

We see that Nunes' take on the empirical facts in (74) and (75) is tenable, but up to now his reanalysis only shows that there is no need for both Delete(α) and Erasure, not which one is dispensable. Things change as soon as additional data is considered.

Instead of looking at all phenomena presented in Nunes (2000), I will restrict myself to one illustrative example, the construction in (75) (Nunes 2000:419).

(76) $[_{TP}$ T seems $[_{CP}$ that it was told John $[_{CP}$ that he was fired]]]

Assuming, as Chomsky (1995c:287) does, that *it* has D-, Case-, and ϕ-features, the derivation could proceed with *it* checking its uninterpretable Case- and ϕ-features in Spec,TP of the embedded clause, such that both are first deleted and then erased. The uninterpretable D-feature, on the other hand, can't be erased, for reasons already mentioned twice, wherefore it raises further to Spec,TP of the matrix clause and checks the strong feature of the matrix T, yielding (77).

(77) $[_{TP}$ it$_i$ T seems $[_{CP}$ that t$_i$ was told John $[_{CP}$ that he was fired]]]

Consequently, FF[John] can be checked by adjoining covertly to matrix T. Intervention effects are unexpected, for the trace of the expletive contains only a categorial feature which does not block movement of FF[John]. If it did, (75) could not be well-formed under standard assumptions, as FF[cat] would not be able to move covertly to matrix T in order to be checked. This leads to the prediction that (77) is grammatical, contrary to the facts.

One might object that the categorial feature need not be checked after the Case- and ϕ-features, allowing the remaining Case- or ϕ-feature on the trace to create an intervention effect, whereby FF[John] cannot be checked and the derivation crashes. But this makes things even worse, as this would imply that the sentence is grammatical roughly every third time. The analysis can only be saved if the categorial feature always is the first feature to be checked — I have my doubts that such a generalization can be shown to be valid.

Nunes concludes that the best empirical results can be obtained with a system along the lines of (78).

(78) *The feature checking system of Nunes (2000)*

a. There is no Erasure.
b. All categorial features are interpretable.
c. Deleted features are invisible at LF and cannot participate in further checking relations.
d. An LI may contain the same feature more than once.
e. Exactly two features are checked at the same time.

One can now see how the deleted features on the trace of *it* could yield an intervention effect, thus leaving FF[John] unchecked and making the derivation crash.

Abandoning Erasure solves a conceptual issue too, namely why features that have already been erased in the overt cycle nevertheless show PF reflexes. Chomsky (1995b:fn.50) proposes reinterpreting overt feature deletion as conversion into phonological features, but this is difficult to implement, requires specific assumptions about the structure of PF and ultimately draws a division between processes in the overt and the covert cycle, thereby undermining a cornerstone of minimalist reasoning.

I agree with Nunes when it comes to the empirical superiority of his account, yet I feel obliged to emphasize that my goal in this chapter does not lie in adopting the empirically best minimalist theory but in finding as many plausible versions of Chomsky (1995b) as possible. Nunes' version is definitely a highly attractive one, conflating Delete(α) and Erasure. Yet one can't help but wonder whether a theory without Delete(α) isn't a plausible variant of Chomsky (1995b) too. If empirical considerations are neglected, it is for sure.

(79) *An Erasure based feature checking system*

a. There is no Delete(α).

b. All categorial features are interpretable.

c. An LI may contain one and the same feature more than once.

d. Exactly two features are checked at the same time.

Let us see how this reproduces the effects of standard minimalist feature checking. The analysis of (74) and (75) does not change at all, the former one is still accounted for by allowing identical features on the same LI, the latter one by (79b). The example sentence in (76) is still predicted to be grammatical. In this feature checking system, it does not even matter in which order the features are checked, the only feature remaining is the interpretable categorial feature. From there on, the analysis is the same, and so are the results.

2.8 Three Classes of Feature Checking Systems

Combining the results of the last two sections, we define three classes of feature checking systems. $\mathfrak{D}$-systems use an adapted version of Delete(α) and forgo Erasure entirely. $\mathfrak{E}$-systems depend solely on Erasure, either by assuming no deletion operation at all or defining Delete(α) as a special case of Erasure. $\mathfrak{DE}$-systems stick to the original assumptions of Chomsky (1995b) by using both Delete(α) and Erasure. For every feature checking class, we encountered at least one member of it in the last two chapters. The list here probably isn't exhaustive, but nonetheless it provides a sufficient basis for the analysis of Reuland (2001) in the next chapter.

(80) $\mathfrak{D}$-*system*

a. There is no Erasure.

b. Deletion of F is bundling F with F^{INV}.

c. $F \in \{F, F^{INV}\}$ cannot participate in further checking relations.

d. All categorial features are interpretable.

e. An LI may contain the same F twice.

f. Exactly two features are checked at the same time.

(81) $\mathfrak{E}$-*system* 1

a. There is no Delete(α).

b. A checked feature F is eliminated from the syntactic phrase marker (and therefore inaccessible to further computation), unless F is the only remaining formal feature of its LI, save any F^S.

c. All categorial features are interpretable.

d. An LI may contain the same F twice.

e. Exactly two features are checked at the same time.

(82) $\mathfrak{E}$-*system 2*

a. Every feature $F \epsilon LI$ is bundled with a visibility feature F^{VIS} in the lexicon.

b. Delete(α) is Erasure of $F_i^{VIS} \in \{\alpha, F_1^{VIS}, F_2^{VIS}, \ldots, F_{i-1}^{VIS}, F_i^{VIS}\}$, $i \in \mathbb{N}^+$.

c. Erasure eliminates $F_i \in \{F_i\}$, unless F_i is the only remaining formal feature of its LI, save any F^S.

(83) $\mathfrak{E}$-*system 3*

a. Every $FF \in LI$ contains a set $INV := \{F^{INV}\}$.

b. Delete(α) is copying of α to INV followed by the Erasure of the original α.

c. Erasure eliminates $F_i \in \{F^{INV}, \ldots, F_i\}$, unless F_i is the only remaining formal feature of its LI, save any F^S or F^{INV}.

(84) $\mathfrak{D}\mathfrak{E}$-*system*

a. Deletion of F is equivalent to bundling F with an invisibility feature F^{INV}.

b. Erasure of F is equivalent to eliminating the bundle consisting of F and at least one F^{INV} entirely from the structure.

c. F is erased when it is bundled with n F_{INV} , $n \in \mathbb{N}$ and subject to parametric variation, unless F is the last feature contained by FF, save any F^S or F^{INV}.

If a system employs both Erasure and some variant of Delete(α), the latter always precedes the former. Furthermore, the Principle of Recoverability of Deletion holds for every system.

Chapter 3

Reuland's "On Primitives of Binding"

The last chapter was devoted to the obligatory refinement of Chomsky's feature checking mechanism. Now I will introduce Reuland's theory of Binding and specifically its modified Principle of Recoverability of Feature Deletion, which, as one will soon be evident to the reader, either does not loosen the restrictions of Chomsky's feature checking mechanism that Reuland wants to omit, or maneuvers itself into a position such that only very specific stipulations at the expense of empirical coverage and conceptual elegance can stop it from overshooting the mark. This is especially unfortunate considering that Reuland's general assumptions and aims are very appealing from a Minimalist perspective.

Concerning the organization of this chapter, I decided against focusing entirely on the mechanics of feature checking. After all, criticizing an integral part of a theory should be done against the corresponding backdrop, even though this isn't indispensable for the intelligibility of the discussion. Therefore, I will first give a short survey of the status of binding theories in the Minimalist Program, and where Reuland (2001) is to be located within this heterogeneous group (3.1.1). Then I will present Reuland's theory in its entirety prior to the more refined discussion of its syntactic mechanisms (3.1.2 and 3.1.3, respectively). From there on it is just a small step to showing the technical fallibility of the proposal (3.2). I conclude with remarks on the general conclusion to draw from the miserable fate of Reuland's stimulating ideas, which will be picked up again in chapter 4.

3.1 Reconnoitering Reuland

3.1.1 A General Overview

Considering that Binding phenomena constitute one of the most studied aspects of natural language, it is not surprising that the set of proposed Binding theories is very heterogeneous. Even if one disregards the efforts of researchers in computational linguistics (Liddy 1990), Head-driven phrase structure grammar (Pollard and Sag 1994), lexical functional grammar (Bresnan 2000) and semantics (Bach and Partee 1980; Keenan 1988; Bonato

2005; Schlenker 2005), as I will do here, it is impossible to identify a non-trivial unifying axiom that all the remaining theories have in common. Even the assumption that pure anaphora (i.e. not long distance anaphora) have to be licensed by an antecedent is not shared by everyone (in particular, Reinhart and Reuland 1991, 1993 and Pollard and Sag 1992 argue against it).

As lively as the research on Binding still is, with the advent of Minimalism it took a serious hit. The Inclusiveness Condition put a stop to the use of indices, surface structure (SS) was dispensed with as a level, Binding conditions were considered a classic example for stipulative, non-derivable definitions without explanatory value, and Governing categories were a purely representational concept that did not fit a derivational framework like Minimalism.[1] The more Minimalism matured and got rid of its GB remnants, the less compatible it became with mainstream Binding theories.

As is to be expected, some efforts were directed towards rescuing as much of the old binding theory as possible. A recent exponent of this line of research is Lasnik and Hendrick (2003), who — for reasons entirely opaque to me — tout themselves minimalist but aim for nothing but recasting GB binding theory in minimalist terms. They accept the axiomatic status of binding conditions without a twinkle and even go as far as showing how one can emulate SS in Minimalism if one wants to capture binding data in a GB style.

Fortunately, there are more experimental approaches. They all have in common that their stance on what the size of the binding domain should be and how the data is to be partitioned is rather difficult to assess. The former is due to the derivational nature of minimalist analysis, which does not allow for easy paraphrasing in representational terms like domains. The latter is an immediate result of the comparative immaturity of most proposals. Usually, only a handful of English utterances is considered, hence it isn't clear at all in how far these approaches could be expanded to account for exempt anaphora or whether the authors even intend them to account for such phenomena. Currently, the focus seems to be on capturing the core cases.

But even when the researcher's attention is restricted to those core cases, he or she still has to decide a question that ultimately could not be solved in GB either, namely which levels are involved in Binding. Is binding restricted to LF, or does syntax contribute its fair share? Although Chomsky (1993) called the former the more minimalist assumption when he got rid of SS, there is neither conceptually nor empirically conclusive support that Binding is restricted to LF. Considering that the properties of LF are not well-known (even if one assumes that current research in semantics covers what syntacticians refer to as LF), decisive evidence is also lacking for claims that syntax has to be involved because of considerations of locality.

Still, most Minimalist theories attribute some importance to syntax, so just like the size of the binding domain and the partitioning of the data, this criterion does not allow

[1] Again it must be noted that Minimalism has its fair share of representational devices that make it more of a hybrid than a derivational system. While variants that were put forward after Chomsky (1995c) attempted to decrease representationality, Chomsky himself has moved more and more towards a less derivational system (Chomsky 2004, 2005a, 2007).

for fine-grained differentiations of Minimalist Binding theories. Fortunately, there is still another measuring rod, namely the set of syntactic mechanisms employed. Interestingly, while this gauge was the least useful one in distinguishing GB binding theories, it is the most promising one for telling apart minimalist binding theories.

There are four basic tools a Minimalist could put to good use in his binding theory: movement, phases, economy, and feature checking. Movement based accounts (Kayne 2002; Zwart 2002; Hornstein 2006)[2] contend that the pronominal is first merged with the antecedent as its specifier, yielding a DP which in turn is merged with the verb. The antecedent then moves further to check its features, leaving the rest of the DP, i.e. the pronominal, behind. Binding conditions are thus assumed to be derivable from lexical properties of DPs and restrictions on movement.

Since their introduction in Chomsky (2001), phases initiated a promising line of research too (Canac-Marquis 2005; Heinat 2006). Canac-Marquis (2005), for instance, postulates that anaphora contain a morphological feature which tells LF to bind the anaphor immediately after conversion into a variable. Hence the antecedent has to be in the same phase, for the use of phases entails that the interpretation of a sentence is split into a sequence of interpretations of multiple phases. Phase-based approaches bolster my claim that the size of the binding domain is difficult to evaluate for Minimalist approaches: even though it is evident that a phase is a binding domain in such theories, the size of a phase is not fixed and so it is impossible to determine the concrete size of the binding domain, it may be a DP (Svenonius 2004), a *v*P, or a CP.

Purely economy based accounts establish that some kind of pronominal is more economic than another one and then construct an algorithm to select the most economic form. A well-known exponent of this variety is Safir (2004), who adds the condition that a less economic form might be chosen for pragmatic reasons. This captures the insight of Reinhart (1983) (extended by Heim (1998)) that discourse factors may license violations of binding conditions. However, the downside of such global economy approaches is that different numerations have to be compared (with all the negative effects, cf. 1.4), unless pronominals aren't treated as LIs (cf. fn. 2).

The last group constitutes theories using feature checking as their main tool. To my knowledge, it comprises solely Richards (1996) and Reuland (2001, 2005). Both theories presuppose that covert feature movement establishes chains, which can transitively be connected with each other. Reuland (2001) further assumes that feature checking establishes feature chains, such that a checking relation between β and FF[α] entails a relation between β and α.

Compared to other Minimalist binding theories, Reuland (2001) is rather unique in

[2]Hornstein (2006) diverts significantly from other movement approaches. First of all, he also wants to account for the distribution of PRO, *pro* and resumptive pronouns. Second, his approach enhances syntax with certain repair strategies: pronouns are inserted by narrow syntax to license illicit movement, e.g. extraction out of an island. A very controversial consequence of this hypothesis concerns the status of pronouns. Given that the Inclusiveness Condition bans addition of material that is not part of the numeration and the numeration is the only way for LIs to enter syntax, pronouns must not be considered LIs but grammatical primitives.

various respects. First, it is one of the oldest genuine Minimalist binding theories. The actual publication date is definitely misleading here, for a previous version appeared in 1997 as an UiL OTS Working Paper, and the first related talk was already given in 1995 at GLOW in Tromsø. Due to this long development process, the paper's framework seems rather dated when compared to other publications that appeared at the same time.

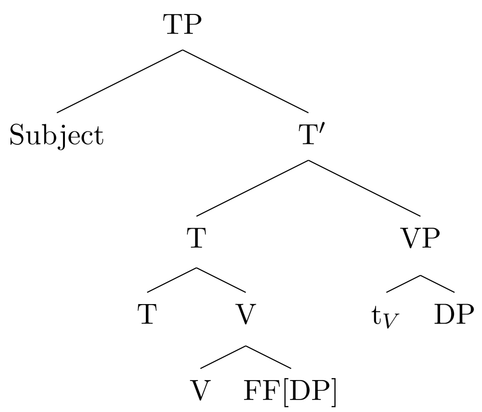
On the other hand, Reuland's general aims and assumptions are way more minimalist than those of many later works, e.g. Lasnik and Hendrick (2003) or Canac-Marquis (2005). Reuland claims that anaphora, pronouns and R-expression can be sufficiently distinguished by their set of ϕ-features, i.e. without recourse to the features [± anaphorical] or [± pronominal]. In the same vein, he does not stipulate any kind of binding principles and furthermore provides a ϕ-feature based account for the asymmetry between first and second person anaphora on the one hand and third person anaphora on the other. In addition, he predicts that SELF anaphora are structurally more complex than SE anaphora. Finally, his most Minimalist claim — although it was inherited from Reinhart and Reuland (1991, 1993) — is that binding itself does not show any parametric variation at all. Whatever apparent differences there might be between languages, they are just epiphenomena of other syntactic parameters.

One might argue that this position just shifts the locus of variation from binding theory to other areas and thus provides barely more than a noble façade, however, this misses the crucial point, namely that binding phenomena are predicted to co-occur with specific structural configurations. What exactly gives rise to those configurations is certainly difficult to evaluate and it probably needs more refined empirical tests than we currently have at our disposal, but this does not render the logic behind the argument vacuous. Reuland and Reinhart (1995) is an interesting example for how one could explain some differences in binding parameters in the Germanic languages by careful scrutiny of their case system.

Unfortunately, the conceptual appeal of Reuland's general take on binding theory does not carry over to its technical implementation. In particular his treatment of ϕ-features and feature checking leaves a lot to be desired. For the sake of conciseness, I will save my criticism on those issues for later on and closely follow Reuland's exposition for the remainder of this section and the first half of the following one.

In order to account for the diversity of binding phenomena, Reuland uses both narrow syntax, LF and pragmatics. Syntax is crucial for the distribution of anaphora and pronouns. Both are DPs that need to check their Case, wherefore their set of formal features has to undergo covert head-movement and adjoin to V (or v, depending on where one assumes accusative case is checked). V then moves further and adjoins to T, pied-piping FF[DP]. Both thus end up with the subject in their checking domain.

(85)



If V-to-T movement takes place in the overt cycle, V is already adjoined to T when FF[DP] adjoins covertly to V, however, the final result is the same if we follow Chomsky's assumption that copies cannot be subjected to further syntactic operations such that only the highest occurrence of V is a viable Case checker for FF[DP].

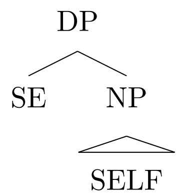
As soon as the subject is in $\mathbb{C}$(FF[DP]), a checking relation is established between the ϕ-features of the anaphor and the subject. In addition, if the DP is an anaphor, a checking chain is established as well. This *Chain* $CH :=$ (Subject, FF[DP]) is then linked with the *chain* CH := (FF[DP], DP), yielding the *CHAIN* $\mathcal{CH} :=$ (Subject, DP). $\mathcal{CH}$ encodes that the subject is the antecedent of the DP. In the case of pronouns, or R-expressions, no such Chain will be constructed because of special properties of some of their ϕ-features. This will be discussed in more detail later on. As a result, the distribution of anaphora, pronouns and R-expressions is an immediate result of the properties of movement and ϕ-features.

LF requirements account for the configurational differences between SE and SELF anaphora. Upon translation into a structure readable at LF,

> [...] within the framework of Chomsky 1995, only terms (minimal and maximal categories) remain visible. Intermediate stages of projection are not. Unless one stipulates order, given the definition of occurrence in Chomsky 2000:115 [Chomsky (1998:29f)] this entails that occurrences of arguments that can be distinguished in syntactic structure become literally indistinguishable at the interface. If so, notations like P(x,x) are in fact misleading. They effectively contain only one argument, to be interpreted by just one semantic object. (Reuland 2001:477)

If one accepts this line of reasoning, one has to come up with means to provide dyadic predicates with two identical arguments. In other words, if two occurrences of the same variable are collapsed into one, how is it possible that *John hates himself* is not translated as [hates(John,John)], which is indistinguishable from [hates(John)]? Reuland maintains that SELF anaphora are a structurally more complex type of anaphora, where SELF is a function that takes SE as its argument.

(86) a.



b. SELF(SE)

No denotation is supplied for f(x), but "it must approximate that of the first argument, x, without being formally indistinguishable from it." (Reuland 2001:481f) That way, *John hates himself* is translated as [hates(John,f(John)], which gives the same interpretation [hates(John,John)] would, if occurrences could be distinguished. If we further assume that verbs occurring with a SE anaphor as their second argument are inherently reflexive, i.e. monadic, but nevertheless select two DPs (in contrast to middle verbs), we have a simple account for the distribution of SE and SELF anaphora based on the assumptions of Chomsky (1995b). Monadic transitive verbs must select a SE anaphor, dyadic transitive verbs may select a SELF anaphor.[3]

Pragmatics, finally, determines the presence of coreferent readings and the interpretation of exempt anaphora. The former is regulated by Reinhart's well-known Rule I (Reinhart 1983; Grodzinksy and Reinhart 1993; Reinhart 2000; see also Heim 1998), while the latter is probably subject to some salience scale (Ariel 1990:cf.). Crucially, although free anaphora are obligatorily interpreted by pragmatics, there is no intrinsic need for anaphora to be bound, just like R-expressions do not need any special kind of licensing. This conjecture is in the spirit of Reinhart and Reuland (1991, 1993).

Nevertheless anaphora are subject to specific economy conditions regulating how they should be interpreted. Those economy conditions determine the distribution of computational processes over the three submodules, syntax, LF and pragmatics, but they do not interfere with the internal workings of the modules. The underlying intuition is that fixing the interpretation of an expression as soon as possible is the most economic choice. Syntactic computation precedes semantic computation, which in turn precedes pragmatic computation. Thus if the value of an anaphor is already established in syntax, it can be immediately translated as a bound variable. Otherwise, this job has to be done in semantics, where it is more costly because semantic computation is less automatic and more sensitive to interpretative differences. The same reasoning carries over to the relation between semantics and pragmatics.[4] Consequently, bound anaphora are cheaper than free anaphora, because their interpretation is fixed earlier. This economy hierarchy has far reaching consequences, for it entails that syntactic processes should always be preferred over semantic ones, and those over pragmatic ones. This is a bold assumption which, as far as I know, lacks conclusive support from other areas of research.

Reuland uses his economy hierarchy to derive three global economy conditions, Reinhart's Rule I and his Rule BV and Rule L.

(87) *Rule I: Intrasentential coreference*
NP A cannot corefer with NP B if replacing A with C, C a variable A-bound by B, yields an indistinguishable interpretation.

[3]The special behavior of anaphora in ECM constructions is more involved, cf. Reuland (2001:465).

[4]I am using the simpler presentation of Reuland (2005) here. Reuland (2001) actually derives the differences in cost in a more technical way based on his assumptions about the translation processes for the various kinds of chains.

(88) *Rule BV: Bound variable representation* (Reuland 2001:471)
T[5] may not translate an expression E′ in Sem′ with syntactically independent NPs A′ and B′ into an expression E in Sem in which A is A-bound by B, if there is an expression E″ resulting from replacing A′ in E′ with C′, C′ an NP such that B′ heads an A-CHAIN [a CHAIN connecting two A-positions; TG] tailed by C′ and T also translates E″ into E.

(89) *Rule L: Logophoric interpretation* (Reuland 2001:472)
NP A cannot be used logophorically if there is a B such that an A-CHAIN $\langle B, A\rangle$ can be formed.

No matter whether one likes the general architecture Reuland is sketching, it couldn't be any better for our inquiry. The economy conditions do not tinker with the internal mechanisms of the modules, so syntax is left unaffected by them. There is no condition that blocks a syntactic process just because it could eventually produce an unwanted interpretation. To the contrary, independent syntactic processes giving rise to binding effects is one of Reuland's main tenets. This means that we can leave aside major parts of his theory and devote our efforts entirely to its syntactic aspects without running danger of skewing the results of the investigation.

3.1.2 Syntax in Close-up

Remember that the job of syntax in Reuland's theory is to capture that pronouns must be free in their binding domain, in other words, that no interpretative dependency relation holds between a DP and a pronoun. Such a dependency relation is encoded via a CHAIN $\mathcal{CH}$, which is established by connecting a normal, movement induced chain $\mathrm{CH} := (\mathrm{FF}[\mathrm{DP}_i], \mathrm{DP}_i)$ with a Chain $CH := (\mathrm{DP}_j, \mathrm{FF}[\mathrm{DP}_i])$. A Chain is always a direct result of feature checking, i.e. $CH := (\mathrm{DP}_j, \mathrm{FF}[\mathrm{DP}])$ is established only if $\mathrm{DP}_j \in \mathbb{C}(\mathrm{FF}[\mathrm{DP}_i])$ and, crucially, DP_j can check all ϕ-features of $\mathrm{FF}[\mathrm{DP}_i]$. Consequently, we have to make sure that anaphora can get all of their ϕ-features checked, such that a dependency between them and their checker can be established, while pronouns cannot accomplish this.

Reuland's first step is to determine an anaphor's set of ϕ-features by considering its morphological shape.[6] Many Germanic anaphora like Icelandic *sig*, Dutch *zich* or German *sich* are solely specified for person. Hence, for each of those anaphora, FF contains only a categorial D-feature and a ϕ-feature bundle consisting of a single person feature. But person features are interpretable on nominal elements, so checking them should be barred

[5] Reuland (2001:471) defines T, Sem and Sem′ as follows:

> Chomsky defines a language L to be a device that generates expressions Exp = $\langle$Phon, Sem$\rangle$, where Phon provides the instructions for sensory-motor systems, and Sem for systems of thought. On the basis of this, $\mathrm{C_{HL}}$ can be defined as generating expressions Exp′ = $\langle$Phon′, Sem′$\rangle$ that obey the Inclusiveness Condition. The C-I interface, then, contains a translation procedure T, which maps Sem′ onto Sem [...].

[6] As already discussed in section 2.1, this is a questionable move.

by the Principle of Recoverability of Deletion (PRD) we encountered in (67) and which I repeat here for the reader's convenience.

(90) *Principle of Recoverability of Deletion*
Interpretable features cannot delete, even if checked.

Reuland argues that the PRD in this form is too strong, and that a less restrictive version actually was more natural. The main insight of the PRD, he asserts, is that the interpretation of the structure must not be endangered at any time by the loss of unrecoverable material. This does not entail that interpretable features are being prevented from getting deleted, but that such deletion processes are blocked if and only if the denotation of the feature isn't recoverable from somewhere else. Given a valid backup for a feature, deleting the feature should not affect interpretation at all, wherefore there is no need to protect it from being deleted and erased. This reasoning is summed up in (91).

(91) *Reuland's Principle of Recoverability of Deletion* (cf. Reuland 2001:456f)
Interpretable features can be deleted iff their interpretative contribution can be recovered after deletion.

Recovering a deleted feature is inseparably tied to the presence of an identical feature from which it can be copied back into its original position. Copies left behind by movement do not qualify as a backup, because they are inaccessible to syntactic processes.[7] The checking DP, though, does. Therefore, a DP can check and delete every interpretable feature for which it has a matching feature. After deletion, a copy of the checker's feature is created and inserted into the feature bundle of the checkee. If all ϕ-features of the checkee are replaced by copies in that way, $CH := (\mathrm{DP}_j, \mathrm{FF}[\mathrm{DP}_i])$ is constructed which can be further linked with $\mathrm{CH} := (\mathrm{FF}[\mathrm{DP}_i], \mathrm{DP}_i)$ to create $\mathcal{CH} := (\mathrm{DP}_j, \mathrm{DP}_i)$.

Replacing (90) with (91) thus provides us with the means to establish the sorely needed dependency between anaphora and their antecedents. But this achievement is nullified if we find no way to block pronouns from doing the same. In contrast to anaphora, pronouns are usually inflected for person, gender and number. There is an interesting bifurcation here: while person and gender features are inherent, number features are optional, they aren't added sooner than in the numeration. Furthermore, the semantic contribution of number features seems to behave differently from that of person or gender features. Consider (92), taken from Reuland (2001:458).

(92) The times were rough. Men were betraying men.

Obviously, the interpretation for the person and gender features of the two occurrences of *men* does not differ. In both cases the third person feature presupposes that the individuals are neither the speaker nor the hearer in the present context, and in both cases does the masculine gender feature presuppose that the individuals referred to are male. However, the impact of the number features differs, for there is no requirement

[7] As we already saw above, this stipulation is needed to correctly predict the behavior of anaphora in languages with covert V-to-T movement, too.

that both groups contain the same number of individuals. In fact, Reuland argues it is presupposed that the sets of individuals must not be identical.[8] If the behavior of number features differs indeed from that of other ϕ-features, it is plausible that the number feature of the checker cannot function as a backup for that of the checkee. Consequently, deletion of the checkee's number feature is blocked, not all of its ϕ-features can be checked, and hence no dependency relation ensues. This derives that neither pronouns nor R-expressions can be syntactically bound.

This result is only partially welcome, as it rules out binding of first and second person pronouns in all constructions where binding of a third person pronoun is ungrammatical, against the empirical facts in (93) (Reuland 2001:464).

(93) a. * Oscar voelde hem wegglijden.
Oscar felt him slide away

b. Ik voelde mij wegglijden.
I felt me slide away

c. Jij voelde je wegglijden.
You felt you slide away

Reuland (2001:464) argues that in first and second person pronouns, singular and plural pronouns do not stand in a grammatical number opposition to each other: "[*W*]*e* does not denote a plurality of I's (speakers)." Hence the number feature of a first or second person pronoun is special insofar as it is not determinded by grammar, but by pragmatics. The value of e.g. *we* depends on the current speech situation, the number of speakers and addressees. Consequently, Reuland claims, all occurrences of a first or second person pronoun within a single sentence have the same denotation for their number feature and thus can serve as a backup for each other, such that the number feature can be deleted and a CHAIN be established.[9] Therefore, the special semantic behavior of the number feature of first and second person pronoun explains why their distribution patterns differently from that of third person pronouns.

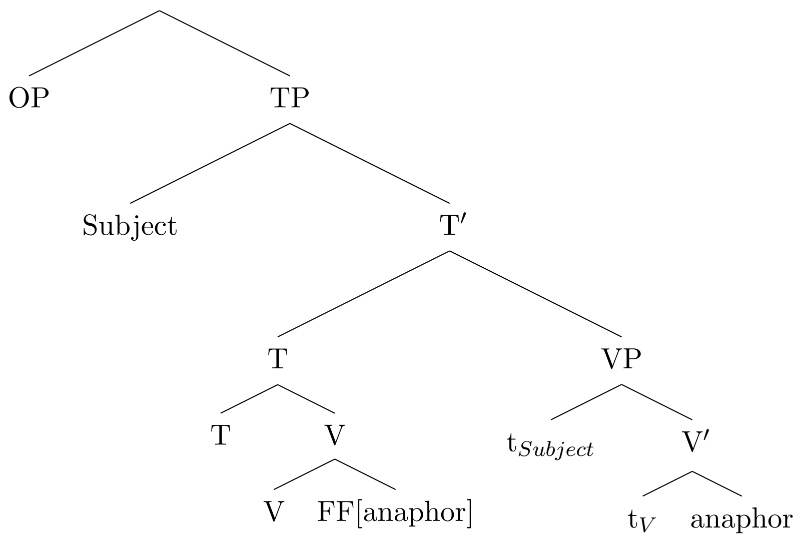
Reuland's checking approach to syntactic binding also has an interesting answer to why subjunctive licenses long-distance reflexives in Icelandic. Remember that according

[8]I have to object. At least to my intuition, things aren't that simple. First, speaker and hearer might belong to the set of betrayers or betrayed. This can easily be explained if we adopt a different denotation for third person features along the lines of Sauerland (2004). The second point, though, undermines Reuland's conjecture that number features have special denotations. To me it is perfectly sensible that both sets contain the same individuals, as long as no relation of self-betraying ensues. That is, A might well be a betrayer of B and likewise be betrayed by B, such that we get two identical sets $\{A, B\}$, but the relation mapping the first set to the second one must not map an individual to itself. This constraint arguably isn't connected to the contribution of the number feature, but to some kind of Condition C effect. In addition, the ban on coreference can be lifted for pragmatic reasons if A is not aware that he's actually committing an act of self-betraying. This behavior seems to be related to binding phenomena that arise in connection with *de se*/*de re* ambiguities, which could be taken as further support for my hypothesis that number does not play an exceptional role here.

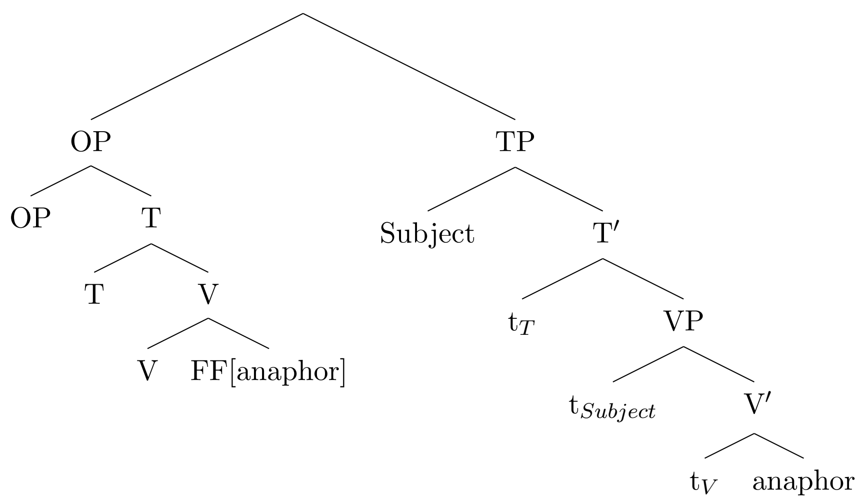
[9]The facts for second person pronouns are slightly more complicated than for first person pronouns, see Reuland (2001:465).

to Rule L no long-distance reading obtains if a bound reading is possible. This implies that Icelandic subjunctive somehow blocks the formation of CHAINs, such that the anaphor remains free until pragmatics, where it can obtain its value from discourse. Reuland (2001:466f) posits that this is due to a high subjunctive operator covertly attracting T, thereby forcing the subject out of $\mathbb{C}$(FF[DP]) and hence preventing the creation of a CHAIN.

(94) a.



b.



But why should CHAIN-formation be blocked in this case? At the earlier stage of the derivation, FF[anaphor] did not move yet, whence it is still accessible to syntactic computation when the Chain should be formed. This Chain could then be linked with CH := (FF[anaphor],anaphor) to yield $\mathcal{CH}$:= (Subject,anaphor).

Reuland apparently wants to delay Chain-formation, such that it does not take place immediately after checking but only after all covert instances of movement.[10] Binding would then be restricted to those cases where the antecedent does not leave the checking domain of the anaphor during the course of the derivation. In particular this applies to

[10]That seems to be the implicit assumption of his otherwise misguided argument on page 467: "Under a copying analysis the copy δ that the moved V/I left behind is frozen, hence inaccessible to a further process."

both subsequent movement of the anaphor and further movement of the antecedent. A similar result can be obtained by delaying CHAIN-formation instead of Chain-formation. Chain linking, no matter of what type the chains are, can only take place where a constituent is the head of a lower chain and the tail of a higher one. If FF[anaphor] moves further before a CHAIN can be established, this is subsequently rendered impossible because the head of CH := (FF[anaphor],anaphor) is not the tail of CH := (Subject,FF[anaphor]).

However, both proposals are highly stipulative and not in accordance with the general intuition about chains, i.e. that they are created immediately after Move has taken place. If they adhere to the laws of chain creation, both Chains and CHAINS should be created as soon as possible, too. Yet we know that Reuland wants his system to account for these empirical facts, so one of the stipulations has to be adopted. Unfortunately, it can't be decided on any principled grounds which one should be chosen. Hence we arrive at a cumbersome fork induced by Reulandian vagueness, forcing us to check each of the three different versions in later sections.

Another fork is induced by empirical considerations of a different kind. Have a look at the utterance in (95).

(95) Who$_i$ did he$_j$ ask himself$_j$ t$_i$ had t$_i$ killed himself$_i$?

Arguably *himself$_i$* is bound by *who$_i$*. If it was not, it would be an exempt anaphora and thus in free variation with pronouns. What more, it even would not need an antecedent given a suitable context, which clearly is a wrong prediction for English. It also does not qualify as a logophoric pronoun in the Reulandian sense, for it should then be possible for the anaphor to take *he$_j$* as its antecedent, again, contrary to the facts. We may thus conclude that *himself$_i$* is indeed bound by *who$_i$*, i.e. there is a CHAIN connecting both DPs. But given Reuland's assumptions, that's plainly impossible.

Let us assume that English *himself* behaves like a SE anaphor (nothing crucial hinges on that, an analysis with a SELF anaphor is just slightly more complex). In the overt cycle, *who* moves overtly into Spec,CP of the matrix clause with some intermediate landing sites, among them Spec,TP of the embedded clause. Then FF[himself$_i$] adjoins to the verb *kill* to check its Case feature, and [$_V$ V FF[himself$_i$]] in turn adjoins to T. Usually checking would take place now, but the subject *who* already moved further up the structure in the overt cycle, leaving behind a copy which is not accessible for syntactic processes any longer. Reuland stresses frequently that copies are inaccessible, because both his analysis of covert V-to-T movement and the stipulation that copies are no viable backups for feature restoral rely on it. How are we supposed to solve this unfortunate situation of conflicting empirical requirements?

We could speculate that the whole T-V-complex moves further after adjunction to T until it finally adjoins to T of the matrix clause. But then we face some puzzling situations. First of all, FF[himself$_i$] and FF[himself$_j$] are in each other's checking domain and should be perfectly capable of checking each other's entire ϕ-feature bundle at the same time, leading to the creation of both CH := (FF[himself$_i$],FF[himself$_j$]) and CH := (FF[himself$_j$], FF[himself$_i$]). After linking those Chains with the respective chains, the anaphora will be coreferent. Even if just one Chain could be created, that suffices to create a CHAIN and induce a coreferent reading. That is bad enough, but it is going to get even worse: obviously both FF[himself$_j$] and FF[himself$_i$] are in $\mathbb{C}$(he$_j$), and evidently the pronoun is perfectly capable of checking both anaphora's entire ϕ-feature bundles. So now we have three coreferent DPs, *he$_j$*, *himself$_i$* and *himself$_j$*. Yet we still have to somehow create a CHAIN between *who$_i$* and *himself$_i$*. The only option left is adjoining the entire V/T complex to C, giving us both $\mathcal{CH}_a :=$ (who,himself$_i$) and $\mathcal{CH}_b :=$ (who,himself$_j$). At this point, I really don't feel obliged to discuss whether *who$_i$* and *he$_j$* then are coreferent.

So extending the set of viable dislocation targets has proved to be a glaring failure, which leaves us with only one option, allowing overt and covert movement to intersperse, i.e. entertaining a single syntactic cycle instead of two. Now FF[*himself$_i$*] will raise to T first, followed by movement of the wh-word into Spec,T. $\mathcal{CH}$ is established, and we get the bound reading. In addition, we also get another fork, this time a binary one. The number of Reulandian variants hence increases to 6.

The number might increase even more, for there is still another issue waiting to be resolved. At no time does Reuland mention his most controversial assumption, namely that checking is asymmetric. One may wonder why this should be controversial, I claimed the very same thing myself in the preceding chapter. Yet I was not talking about the checking operations Delete(α) and Erasure, rather I was referring to the fact that only uninterpretable features can be attractors, which induces an obvious asymmetry in the set of possible checking configurations. This must not be confused with the properties of the checking operations themselves, whose symmetry is conceptually indispensable for checking Case features. Both Case features in a checking relation are uninterpretable and both get erased. If Delete(α) was not symmetric, only one feature could be deleted (and in turn erased), and the other one would be left behind without a suitable checker, hence leading to a crash at LF.

Reuland, on the other side, relies on asymmetric checking operations. Although his PRD allows every interpretable feature to be checked if an identical feature is present, it is always the features of the anaphor which are checked, while the features of the subject remain untouched such that they can function as a backup for the anaphor's checked features. Assuming that he probably does not want to change the way Case features are checked, this entails that the checking of uninterpretable features differs greatly from that of interpretable features. This is an unwanted result in any Minimalist theory, and it considerably reduces the soundness Reuland attributes to his generalization of checking.

A related question is why the subject represents the only viable backup. Why are no other DPs in the tree allowed to serve this function? In particular, consider the case where the numeration contains *him$_2$*, i.e. a pronoun ready to be merged twice. In such cases, one occurrence of *him* obviously would provide a perfect backup for the other one, such that local syntactic binding of a pronoun is permitted as long as the other identical pronoun is accessible. Showing that this prediction is borne out would be an earth-shattering result.

Both issues degrade the elegance of Reuland's theory, but note that the former can be derived from the latter. Assume that given a checking relation between interpretable features, only the features of the c-commanding constituent may be a viable backup. The asymmetry of such checking relations then becomes an artifact of global economy conditions if we turn checking into an "anything goes" process. When two features $F_1 \epsilon LI_{\text{Subject}}$, $F_2 \epsilon$ FF[*anaphor*] are in a checking relation, syntax has three options: either F_1 checks F_2, or F_2 checks F_1, or both check each other. If F_1 and F_2 are uninterpretable, only the derivation choosing the third option won't crash at LF. The other options will be carried out as well due to reference-set computation, but we will never see their effects in real-life because of their deficiency. If F_1 is interpretable and F_2 is uninterpretable, the first and the third option are indistinguishable, given that F_1 has no suitable backup. The second checking relation, however, will crash at LF and hence we will never get to see it. Finally, if both F_1 and F_2 are interpretable, the first option will be preferred by considerations of global economy because it allows syntactic binding of the anaphor, in contrast to the other options. So if we maintain that the c-commanding constituent in a checking relation provides the only viable backup, we can easily explain away the differences in the checking of interpretable and uninterpretable features.

The restriction to the subject can in turn be derived from other primitives, too. First, observe that chains are created by an algorithm which is triggered after the execution of Move(Copy(α),β). By analogy, we conclude that Chains are established by an algorithm which is triggered after a sufficient number of applications of Move(Copy(α), β), where $\alpha \in \mathcal{F}_{FF}$ and β some set of FF[XP]. This intuition is shared by Reuland (2001:457):

> Deletion of a feature F_α in DP_1 and recovery of F_α under identity with F_α in DP_2 is tantamount to treating F_α in DP_1 and F_α in DP_2 as copies, and in fact as occurrences of the same feature [...]. This operation is therefore conceptually similar to the operation Agree in [Chomsky (1998, 2001)], which copies feature values into feature matrices that are underspecified [...].

Moving features arguably differs from moving syntactic objects insofar as no additional sets are created and no labels projected — the copies are simply inserted into the targeted bundle as they are. Despite this minor difference, it is plausible that both variants of Move are subject to comparable economy conditions. The checking domain is the smallest of all domains, restricting the search space for backups to this domain thus reduces computational load.[11] If we now add the proviso that Chains can only be established if at least one constituent in the checking relation has all its ϕ-features checked (which is needed for independent reasons, as we saw in the preceding sections), it follows that the subject is the only possible backup, as its ϕ-feature bundle is a superset of the ϕ-feature bundle of the anaphor. Attempting to do the copying in the other direction, from the anaphor to the subject, won't allow all ϕ-features to be checked and no syntactic binding ensues, which is disfavored by global economy.

Let me sum up the argument. We independently need global economy conditions preferring syntactic binding, *plus* the stipulation that an entire ϕ-feature bundle has to

[11] It isn't necessarily more efficient, as this restriction can lead to cases where an anaphor has to remain free although it could have been bound if the search space for backups was bigger. This also shows that we need to maintain that this local constraint on search space is ranked higher than all global economy metrics.

be deleted and restored in order for a Chain to be created. We further stipulate that the restoration of features is a kind of Copy and Move process which is subject to local economy conditions restricting the search space for backups of features of LI_i to the smallest domain possible, $\mathbb{C}(LI_i)$. As the subject has more ϕ-features than an anaphor, the former can restore all ϕ-features of the latter, while the inverse does not hold true. Consequently, a Chain can only be established in case the subject checks the features of the anaphor and the inverse does not hold. Construction of a Chain implies syntactic binding of the anaphor, which is the most economical option. Consequently, no competing derivation where the subject did not restore the anaphor's features will be selected for further computation. This yields the apparent asymmetry of checking involving interpretable features.

The qualification that Chains are only established if an entire ϕ-feature bundle is checked is actually phrased in a different way by Reuland (2001:450):

> [...] ϕ-feature bundles, but not the individual ϕ-features, correspond to variables at the C-I interface. I will therefore assume that only ϕ-feature bundles can be manipulated by C_{HL} and enter syntactic dependencies that can be interpreted. [...] I argued that no dependency between *de jongens* 'the boys' and *hen* 'they' can be formed since establishing a CHAIN would violate the PRD owing to the nature of the number feature. Yet *de jongens* and *hen* do share the features of category, person, and gender. From the assumption that interpretable dependencies can only be established between entire bundles of ϕ-features, it follows that no such dependency can be established between *de jongens* and *hen*, since identifying $\phi_{dejongens}$ and ϕ_{hen} violated the PRD and **there are no identifiable linguistic objects in the structure corresponding to subparts of these bundles [my own emphasis; TG].**

This is a significantly stronger assumption than the one I introduced above. Taken literally, it denies that ϕ-features are syntactic objects on their own, which violates the fundamental intuition of features as building blocks from which LIs can be constructed in a purely compositional way. If we were to adopt this new perspective on ϕ-features, a lot of our definitions would need to be revised. First, bundles should be able to act as a kind of meta-feature that is assigned a checksum calculated from the features it contains, and if two bundles have the same checksum, they may check each other, unless the checksum indicates the presence of a number feature. Then we need to redefine our checking operations such that they can operate on bundles directly, and the relevant constraints on checking evidently have to be reformulated, too.

It is doubtful that this is what Reuland had in mind on a technical level, so I will simply assume that he just meant to point out that Chains only hold between syntactic objects as defined in (35).[12] But then the entire ϕ-feature bundle has to be checked in one derivational step, i.e. we have to allow multiple applications of Delete(α) and Erasure at once. Otherwise, we have no way to tell whether the ϕ-feature bundle of some LI_i was checked by a single LI_j exclusively or whether it was actually a result of multiple LIs

[12]Before anyone jumps to premature conclusions concerning the honesty of this move, though, let me note that the ultimate results of Reuland's theory would not improve even if we took the passage literally and implemented such a checksum approach.

being in $\mathbb{C}(LI_i)$, because there are no representationally encoded dependencies between individual features and syntax has no memories of earlier stages in the derivation (i.e. there is no lookback). But if there are neither such Chains nor lookback, checking whether all features have been checked and restored by the same LI becomes an intricate issue.

A simple account builds on the intuition that checking and restoral of features are part of a bigger operation, just like Move is actually a complex operation (see fn. 12 on page 10). This is almost trivial if we redefine checking as a complex operation comprising Delete(α), Erasure and restoral of features.[13] If we further maintain that such suboperations never induce new derivational steps, syntax just needs to keep track of whether all ϕ-features of some LI were checked and recovered during a single derivational step.[14]

This is a plausible assumption without any negative consequences for our checking systems. However, remember that we maintained that exactly two features are manipulated by one application of a checking operation, for empirical reasons and because it fit the underlying intuition of checking. It follows that anaphora must not contain the same ϕ-feature twice. If they did, their entire ϕ-feature bundle could only be checked in one derivational step if the checker had the same feature twice, too. Consequently, we would predict the existence of anaphora that can only be bound by specific LIs under totally idiosyncratic conditions. Evidently this behavior isn't encountered with anaphora, so we are forced to disallow doubled ϕ-features for anaphora. But as Reuland wants anaphora to be normal LIs without any special property except their poverty of ϕ-features, this constraint has to be generalized to all LIs. Therefore, no LI is allowed to have the same feature twice, although it is perfectly possible if we define LIs as multisets.[15]

3.1.3 Wrapping up

Just like I did at the end of chapter 2, I will now list the definitions of plausible versions of Reuland's theory of syntactic binding, which I will abbreviate as $\mathfrak{R}$ with optional parameters as sub- or superscripts. Please note that I leave aside semantic and pragmatic aspects as they are in no way relevant to our enterprise.

(96) *Condition of Feature Distinctiveness* (CFD)
For every LI_i, there are no features $F_i, F_j \in LI_i$, and $F_i, F_j \notin \{F^S, F^{INV}, F^{VIS}\}$, such that F_i and F_j are identical.

(97) *Principle of Induction by Morphological Paradigm*
The set of ϕ-features of a DP can be determined by investigation of the morphological alternations within the inflectional paradigm.

[13] The precise ordering of this suboperations is a non-trivial matter, as will be seen in section 3.2.

[14] The formal specification is a little cumbersome and will be relegated to 3.1.3.

[15] This stipulation conflicts with the definition of $\mathfrak{D}$-systems and $\mathfrak{E}$-system 1, which for empirical reasons need to assume that some LIs have two identical features. As often before, I will neglect the empirical aspects and just grant this assumption.

(98) *Unity of the ϕ-feature bundle*
For every LI_i, there is a smallest set B_ϕ containing all ϕ-features F^ϕ of LI_i such that every syntactic operation other than Delete(α), Erasure or Feature-Restoral has to be applied to B_ϕ instead of any F^ϕ.

(99) *Feature Restoral*
$F_i \epsilon LI_i$ can be copied into FF[LI_j] after deletion of $F_j \epsilon LI_j$ iff F_i and F_j have the same interpretative contribution and $LI_i \in \mathbb{C}(LI_j)$. If F_i has been copied into FF[LI_j], we say that the interpretative contribution of F_j was recovered from F_i.

(100) *Interpretative Contribution of Features*
With the exception of number features, all identical features have the same interpretative contribution. The occurrences of a number feature do not have the same interpretative contribution unless the number feature belongs to a first or second person pronoun.

(101) *Feature Checking*
For all F_i, F_j, if F_i and F_j are in a checking relation, they are checked according to one of the systems in section 2.8, where the original PRD is replaced by the new one in (102). Furthermore, (99) is a part of checking.

(102) *Principle of Recoverability of Deletion*
Interpretable features can be deleted iff their interpretative contribution can be recovered after deletion.

(103) *Derivation*
Let a derivation Δ be a set of representations on which a partial order is defined by syntactic operations.

(104) *Syntactic Operations*
An n-ary operation, $n \geq 1$, is a partial function mapping n representations R_i, $i \in [1, n]$, to exactly one representation R_{n+1}. We entertain an idiosyncratic distinction between complex operations Ω (Select, Merge, Move, Checking) and simplex operations ω (Select, Copy, Delete(α), Erasure, Feature Restoral, Label, Concatenate, ...).[16] Without loss of generality let ω_i and ω_j be unary operations. Then $\omega_i \circ \omega_j$ denotes the concatenation of simplex operations ω_i, ω_j, such that there exist representations R_k, R_l, R_m, and ω_i maps R_k to R_l, and ω_j maps R_l to R_m. A complex operation is of the general form $\Omega_i := \omega_1 \circ \omega_2 \circ \ldots \circ \omega_{n-1} \circ \omega_n$, $n \geq 1$, where ω and n depend on the specific definition of the respective complex operation.[17]

[16]I abstract away from the details of Delete(α) and Erasure, which are no simplex operations in the narrow sense according to the definitions in 2.8.

[17]For example, Merge is a complex function $\Omega_M := \omega_1 \circ \omega_2$, where $\omega_1 :=$ Concatenate and $\omega_2 :=$ Label. Select is defined as $\Omega_S := \omega_1$, where ω_1 is the simplex operation Select. Select is rather special insofar as it is the only operation that maps R_n to itself.

(105) *Formal Definition of Checking*
Let $\omega \in \{\text{Delete}(\alpha), \text{Erasure}, \text{Feature Restoral}\}$. Checking is a complex operation $\Omega_C := \omega_1 \circ \omega_2 \circ \ldots \circ \omega_{n-1} \circ \omega_n$, $n \geq 1$, where the respective simplex operations are partially ordered such that $\exists x[x \in \mathcal{F}_{FF} \wedge \text{Delete}(x) \wedge \text{Erasure}(x)] \rightarrow \ldots \circ \text{Delete}(x) \circ \ldots \circ \text{Erasure}(x) \circ \ldots$, and the conditions in (101) and (102) apply.[18]

(106) *Derivational steps*
A derivational step δ_i is a subset of some Δ_i such that the concatenation of the simplex operations defining a partial order on the members of δ_i yields a well-formed complex operation.

(107) *Condition of Unique Checking*
For every feature F_i, if F_i is operated on by Delete(α) or Erasure in δ_i, the same operation may not be applied again to F_i in δ_i.

(108) *Construction of Chains*
A Chain $CH := (\alpha, \beta)$, α and β syntactic objects, is constructed iff for every F_j in the set of ϕ-features of β there is an $F_i \; \epsilon \; \alpha \in \mathbb{C}(\beta)$ such that the interpretative contribution of F_j was recovered from F_i in one derivational step. The construction of CH takes place

a. immediately after the recovery of the interpretative contribution of F_j. ($\mathfrak{R}_\emptyset$ and $\mathfrak{R}_{\mathcal{CH}}$)
b. after all instances of movement. ($\mathfrak{R}_{CH}$)

(109) *Construction of CHAINs*
Given a Chain $CH := (\alpha, \gamma)$ and a chain CH $:= (\gamma, \beta)$, with α, β, γ syntactic objects, a CHAIN $\mathcal{CH} := (\alpha, \beta)$ is constructed

a. as soon as both CH and CH have been constructed. ($\mathfrak{R}_\emptyset$ and $\mathfrak{R}_{CH}$)
b. after all instances of movement. ($\mathfrak{R}_{\mathcal{CH}}$)

(110) *A-CHAINs*
If both α and β are A-positions, $\mathcal{CH} := (\alpha, \beta)$ is called an A-CHAIN.

(111) *Syntactic Cycle*

a. Syntax has only one syntactic cycle, with overt and covert movement interspersed. ($\mathfrak{R}^{1C}_\emptyset$, $\mathfrak{R}^{1C}_{CH}$, $\mathfrak{R}^{1C}_{\mathcal{CH}}$)
b. Syntax has two syntactic cycles, an overt and a covert one, separated by Spell Out. ($\mathfrak{R}^{2C}_\emptyset$, $\mathfrak{R}^{2C}_{CH}$, $\mathfrak{R}^{2C}_{\mathcal{CH}}$)

(112) *Rule BV: Bound Variable Representation*
T may not translate an expression E′ in Sem′ with syntactically independent NPs A′ and B′ into an expression E in Sem in which A is A-bound by B, if there is an expression E″ resulting from replacing A′ in E′ with C′, C′ an NP such that B′ heads an A-CHAIN tailed by C′ and T also translates E″ into E.

[18] The timing of Feature Restoral varies between different theories (see 3.2.2).

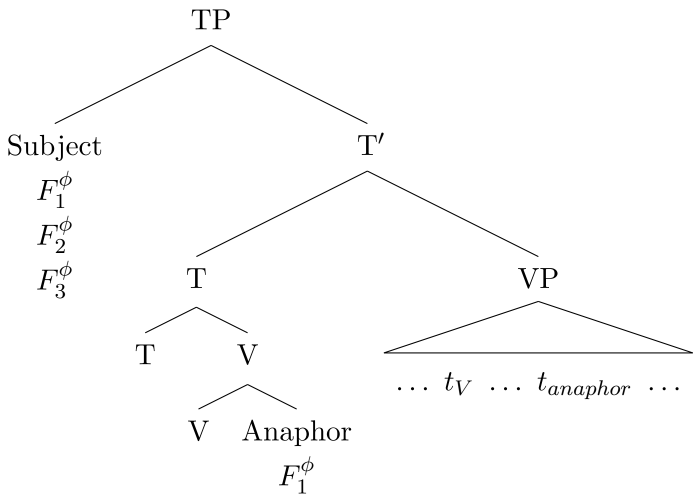
(113) *Rule L: Logophoric Interpretation*
NP A cannot be used logophorically if there is a B such that an A-CHAIN $\mathcal{CH} := (\mathrm{A}, \mathrm{B})$ can be formed.

3.2 Doomed to Crumble, Blessed to Bloom

3.2.1 $\mathfrak{D}$-systems

After a painstaking exegesis of both Chomsky (1995c) and Reuland (2001), we finally arrived at the point where the technical soundness of Reuland's proposal can be assessed.

Let us start our actual analysis with $\mathfrak{D}$-systems (henceforth, I will refer to the various feature checking systems by their alphanumerical designation alone). Just like for the rest of the chapter, I will reason from the canonical scenario depicted in (114), where ϕ-features are abbreviated as F_i^ϕ.

(114)


According to (108) and (99), F_1^ϕ of the anaphor (henceforth $F_1^{\phi,A}$) has to be deleted, such that the subject's F_1^ϕ (henceforth $F_1^{\phi,S}$) can be copied to the anaphor.[19] In addition, the CFD blocks copying $F_1^{\phi,S}$ onto the anaphor as long as $F_1^{\phi,A}$ is still present, as the anaphor would then contain two identical features, which we had to rule out previously in order to get the theory flying.

[19]If we were to follow Reuland as strictly as possible, the anaphor's entire ϕ-feature bundle had to be deleted and replaced by a new set created as a subset of the subject's ϕ-feature bundle. From a formal perspective, it does not matter whether we abstract away from Reuland's stipulation or not, it is just a matter of definitions and leaving aside vacuous steps. But during the discussion of feature checking systems I did already mention that I do not really like the idea of allowing syntax to rebundle features, so I see no good reason to take this narrow stance. Interestingly, the checksum approach remains neutral on this issue. Overwriting a string representing $B_1 := \{F_1^\phi\}$ with an identical string still requires that syntax determined that such a string could in fact be generated from the string of $B_2 := \{F_1^\phi, F_2^\phi, F_3^\phi\}$ and then does so. But whether this represents deletion and restoral of F_1^ϕ or whether this is actually rebundling in a less set-theoretic guise is undecidable.

Now remember how $\mathfrak{D}$, inspired by Nunes (2000), is supposed to work. When a feature is in a checking relation, Delete(α) is triggered, just as usual. The features are marked as invisible at LF by bundling them with F^{INV}. They are not erased, though, for $\mathfrak{D}$-systems have no operation corresponding to Erasure. Instead, it is stipulated that deleted features fail to participate in further checking relations. But this implies that the number of features of an LI can only grow (by adding additional F^{INV}), under no circumstances will a feature be lost. Yet if we are not able to get rid of $F_1^{\phi,A}$, copying of $F_1^{\phi,S}$ will always be blocked by the CFD and no Chain will be constructed.

If we want to circumvent this undesired result, we somehow have to weaken the CFD. There are two conceivable options: either the CFD is a lexical principle that does not apply in syntax, or the condition is restricted to active, i.e. non-deleted features. The former looks plausible at first sight, but loses its appeal when subjected to closer scrutiny. The reason we introduced the CFD in the first place was to ensure that all morphologically compatible DPs can in principle function as the antecedent of the anaphor — obviously this is a purely syntactic issue which should not bother the lexicon at all. And now we aim to weaken the CFD, again for purely syntactic reasons which the lexicon couldn't care less about. Note that we cannot claim that the CFD is "virtually conceptually necessary", because it does not affect any crucial part of language. If the lexicon did not adhere to the CFD, the behavior of syntactic binding would sometimes be rather odd, but otherwise everything would be fine. Thus turning the CFD into a completely unmotivated lexical principle is equivalent to viewing the regularity of syntactic binding as a mere coincidence, which is not particularly satisfying from a Minimalist perspective.

So let us see whether exempting deleted features from the CFD fares any better. For $\mathfrak{D}$, the answer is a resounding yes. We already stipulated that deleted features can't be checked several times, although they themselves aren't special in any way, they are just sharing a bundle with an F^{INV}. That is, syntax treats deleted features differently from active ones in $\mathfrak{D}$-systems. Stipulating that deleted features are also exempted from certain syntactic conditions like the CFD is just a natural extension of this line of reasoning. So we can indeed weaken the CFD as desired and allow the copy procedure to take place. $F_1^{\phi,S}$ is copied onto the anaphor after $F_1^{\phi,A}$ has been bundled with F^{INV} and a Chain is created as desired.

Let us abbreviate theories using the lexical CFD as $\mathfrak{R}^L$, those using the syntactic CFD as $\mathfrak{R}^S$, and those employing the stricter, original CFD as $\mathfrak{R}^O$. Further, I represent the combination of a Reulandian variant $\mathfrak{R}$ and a feature checking system $\mathfrak{F}$ as a tuple $\langle \mathfrak{R}, \mathfrak{F} \rangle$. Summing up what we do already know, $\left\langle \mathfrak{R}^L, \mathfrak{D} \right\rangle$ works flawlessly and is conceptually sounder than $\left\langle \mathfrak{R}^S, \mathfrak{D} \right\rangle$. $\left\langle \mathfrak{R}^O, \mathfrak{D} \right\rangle$ on the other hand, fails miserably because no copying can take place. Whether we assume one or two cycles has purely empirical consequences, allowing us to account for binding by wh-words. Consequently, $\left\langle \mathfrak{R}^{S/L,1C}, \mathfrak{D} \right\rangle$ and $\left\langle \mathfrak{R}^{S/L,2C}, \mathfrak{D} \right\rangle$ fare equally on conceptual grounds.

How does $\langle \mathfrak{R}_{CH}, \mathfrak{D} \rangle$ behave? The main question we have to ask ourselves is: if the formation of Chains is delayed until all syntactic covert movement has taken place, as is the case with $\mathfrak{R}_{CH}$, can the Chain be established by syntax later on by examination of the antecedent and the anaphor. Maintaining that no LI may start with deleted features (which is one of Chomsky's many implicit assumptions), it follows from the CFD that if LI_i has two identical features F_i and F_j, the latter already deleted, then F_i is a copy of a feature of some DP in $\mathbb{C}(LI_i)$. If the DP has moved further after checking the features of the anaphor, the copy has to be investigated instead of the DP. This requires syntax to be able to operate on copies, which conflicts with both Chomsky's and Reuland's claims. But of course we can always stipulate that copies are accessible to syntax when Chains have to be established. It isn't sensible, it is highly unattractive, but technically it works.

In cases where $\mathbb{C}(LI_i)$ contains multiple LIs (i.e. more than some heads and one DP), syntax has to look at the ϕ-features of each syntactic object. If several are suitable antecedents, multiple representations with different Chains and hence different CHAINS are sent to LF, giving rise to ambiguity.[20] Ambiguity itself is not a problem, it also ensues when Chains are created immediately if we allow copied features to be checked again such that another Chain can be constructed later on. But if overt and covert movement are separated by Spell Out, $\mathfrak{R}_{CH}$ will possibly overgenerate when more than one DP is in $\mathbb{C}$(anaphor). Just think of the wh-construction we discussed. When the anaphor reached T, only a copy was left in the specifier. However, if Chain formation is delayed until the end of the derivation, there is no way syntax could know that there never was an accessible DP in $\mathbb{C}$(anaphor). In situations where the anaphor raises to a T-node with two specifiers, one a copy, the other one still the head of its chain, the anaphor will get its features checked by the accessible DP, but syntax will eventually create two Chains, a correct and a false one, the latter connecting the anaphor and the copy. This situation does not arise if overt and covert movement intersperse, for then it can never be the case that the anaphor is moved to a head whose specifier is already filled.

We conclude that $\left\langle \mathfrak{R}_{CH}^{S/L,1C}, \mathfrak{D} \right\rangle$ is conceptually highly unattractive, but at least it does not overgenerate, in contrast to $\left\langle \mathfrak{R}_{CH}^{S/L,2C}, \mathfrak{D} \right\rangle$. $\left\langle \mathfrak{R}_{CH}^{O}, \mathfrak{D} \right\rangle$, of course, is the same failure $\left\langle \mathfrak{R}_{\emptyset}^{O}, \mathfrak{D} \right\rangle$ was.

For $\mathfrak{D}$, we are only left with the task of assessing the behavior of $\mathfrak{R}_{\mathcal{CH}}$. This is a matter of a few seconds at most, as it is perfectly obvious that $\mathfrak{R}_{\mathcal{CH}}$ behaves identical to $\mathfrak{R}_{\emptyset}$ when it comes to checking. The only difference is an empirical one, as some CHAINs established by $\mathfrak{R}_{\emptyset}$ would not be constructed by $\mathfrak{R}_{\mathcal{CH}}$ (namely in those cases where the anaphor moves further after its features have been checked). For the remainder of our investigation, we can thus conflate $\mathfrak{R}_{\emptyset}$ and $\mathfrak{R}_{\mathcal{CH}}$ into $\mathfrak{R}_{\emptyset/\mathcal{CH}}$, saving us a lot of work.

3.2.2 $(\mathfrak{D})\mathfrak{E}$-systems

For $\mathfrak{D}$, we have just seen that all variants of Reuland's theory initially fail to work, but they can be made compatible if we extend the special status of deleted features and hence weaken the CFD. Do the good results of the weakened CFD hold for $\mathfrak{E}$ and $\mathfrak{DE}$? This

[20]Allowing the interfaces to take more than one representation as their input does not have any negative consequences, as long as we do not allow any kind of comparisons between the representations. Still it isn't a particularly attractive solution.

depends on when we want the recovery of features to take place. In the definition given in (99), I intentionally used the vague term "deletion", which does not distinguish between Delete(α) and Erasure. According to Reuland (2001:455), it does not matter whether interpretable features are deleted or erased before their restoral. This is a fallacy, as we will see immediately.

If we let the restoral of interpretable features take place after Delete(α) but before Erasure (denoted $\mathfrak{R}_D$), nothing changes in comparison to $\mathfrak{D}$, except that the deleted features of the anaphor are now also erased after the copy procedure has applied. In the case of $\mathfrak{R}_{CH,D}$, a new problem is introduced, because now syntax cannot look at the deleted features in order to assess which features are copies and which belonged to the LI from the very beginning. The algorithm for such systems thus becomes more complicated.

(115) a. Let $\Phi[\text{XP}] := \left\{F^\phi : F^\phi \text{ is a } \phi\text{-feature of XP}\right\}$. For any syntactic object SO_i heading a chain and not analyzed yet by the algorithm, check whether there is a $SO_j \in \mathbb{C}(SO_i)$ heading or tailing a chain such that $\Phi[SO_i] \subseteq \Phi[SO_j]$. If so, proceed with the algorithm, else if there is another syntactic object heading a chain that has not already been analyzed, repeat this step for it, else quit the algorithm.

b. For every $F_i^\phi \in \Phi[SO_i]$, check whether $F_i^\phi \in \Phi[SO_j]$ has the same interpretative contribution. If this is the case for all $F_i^\phi \in \Phi[SO_i]$, create $CH := (SO_j, SO_i)$. In any case start the algorithm again for the next syntactic object.

As we can see, the algorithm basically duplicates all the work that has to be done before $F_1^{\phi,S}$ can be copied onto the anaphor and furthermore deprives Reuland's theory of any chance to meaningfully reduce Chains to feature copying targeting subparts of LIs, rather than syntactic structure. In comparison to $\mathfrak{R}_{\emptyset/\mathcal{CH},D}$, $\mathfrak{R}_{CH,D}$ boosts higher complexity and less conceptual appeal. And just like $\left\langle \mathfrak{R}^{2C}_{CH}, \mathfrak{D} \right\rangle$, $\left\langle \mathfrak{R}^{2C}_{CH,D}, \mathfrak{F}' \right\rangle$, $\mathfrak{F}' \in \{\mathfrak{D}\mathfrak{E}, \mathfrak{E}1, \mathfrak{E}2, \mathfrak{E}3\}$, is at risk of overgenerating, too.

If copying never happens sooner than after Erasure of the relevant features, we are faced with the peculiar problem that as soon as the features have been erased, syntax has no clue which features the LI contained just one minute before. This leaves us with two possibilities. First, no Chain is created, because syntax does not even know that it erased interpretable features and hence does not copy any ϕ-features. Obviously no version of Reuland's theory can work with this option, so let us look at the second one.

We may assume that if the LI is a DP, syntax still knows that it has to copy some ϕ-features into the LI (but not which one), because every DP contains some interpretable ϕ-features,[21] so if a DP has no ϕ-features any more, some interpretable features have been deleted and now have to be restored. If no information must be lost, but one does not know which features have to be recovered in order to rescue the derivation, the only viable

[21]Expletives are the only exception to this rule, but their feature composition is special in any regard.

option is to copy all ϕ-features of the checker onto the DP. The result is a Chain and everything is fine, it seems.

However, we now have a DP probably containing more ϕ-features than before. So instead of just being specified for person, the anaphor *zich* may also be specified for gender or number. Granted, this does not interfere with interpretation, because the bound variable and the antecedent will have the same ϕ-features and thus will trigger the same presuppositions. Similarly, we do not expect the anaphor to change its phonological form, unless we assume that phonological information is added post-syntactically like in Distributed Morphology. The system I developed in the preceding chapter assigns every LI a fixed phonetic content that might, but need not relate to ϕ-features, so it is agnostic on this issue. Still, conjecturing that various instances of *zich* may differ in regard to their ϕ-features is at odds with (97), the Principle of Induction by Morphological Paradigms which Reuland needs in order to assess the ϕ-feature composition of DPs, unless we interpret it as a purely lexical principle determining the initial ϕ-feature configuration, not the final one.

This is a considerable weakening of the principle, and apparently Reuland did not intend it to be understood this way. On page 475, he claims that his system actually employs no means of global economy except Rule I, because anaphora can be constructed locally from pronouns by Erasure of any ϕ-features except person. We need not concern us with technical details here (Reuland does not provide them anyway), but two premises are indispensable for this argument to go through: first, at least some LIs have no fixed phonetic content, and second, the Principle of Induction by Morphological Paradigms is not restricted to the lexicon. So we arrive at a contradiction. In order to make assumptions on the internals of DPs, we introduce a principle that relates ϕ-features of DPs with their number of different phonetic exponents in a morphological paradigm. This is the starting point of our theory. But when we proceed with our theory, an artifact of the checking mechanism forces us to contradict ourselves by denying that an LI's ϕ-feature composition is necessarily reflected by its phonetic form. From this we conclude that every system $\langle \mathfrak{R}_E, \mathfrak{F}' \rangle$ is inherently contradictory, given certain further assumptions of Reuland (2001).

Summing up, we saw that $\mathfrak{D}$-systems fail with the strict CFD but can easily accommodate a weaker version which excludes deleted features. As a consequence, $\left\langle \mathfrak{R}^{S,1C/2C}_{\emptyset/\mathcal{CH}}, \mathfrak{D} \right\rangle$ remains functional, while $\left\langle \mathfrak{R}^{S,1C}_{CH}, \mathfrak{D} \right\rangle$ overgenerates and $\left\langle \mathfrak{R}^{S,2C}_{CH}, \mathfrak{D} \right\rangle$ lacks conceptual soundness and is computationally less efficient than the variants with $\mathfrak{R}_{\emptyset/\mathcal{CH}}$. For systems comprising Erasure, assuming the weaker CFD only does any good for $\langle \mathfrak{R}_D, \mathfrak{F}' \rangle$, which works like $\langle \mathfrak{R}, \mathfrak{D} \rangle$ (with the exception of $\mathfrak{R}_{CH}$, which then faces even more problems). $\langle \mathfrak{R}_E, \mathfrak{F}' \rangle$, on the other hand, becomes contradictory.

Evidently we are still lacking results for $\mathfrak{F}'$ with the strict CFD. But those results can be easily deduced by pure logic. If the strict CFD holds, no feature can be copied before Erasure has taken place, and thus $\langle \mathfrak{R}^O, \mathfrak{F}' \rangle$ and $\langle \mathfrak{R}_E, \mathfrak{F}' \rangle$ are equivalent (and $\langle \mathfrak{R}^O_D, \mathfrak{F} \rangle$ always a total failure).

A relevant question I left aside is whether the weaker CFD fits the Erasure systems as nicely as $\mathfrak{D}$. Making the CFD a lexical principle obviously does not become any more logical by the introduction of an additional checking operation. The soundness of exempting deleted features from it, on the other hand, is affected by this enlargement of the set of syntactic operations, but in a negative way. As I already pointed out, deleted features in $\mathfrak{D}$ are given a special treatment that goes beyond simply regarding them as members of a set containing a F^{INV}. In Erasure based systems, however, deleted features are just normal features that happen to be invisible at LF, which isn't of any relevance for syntactic processes. To syntax, deleted features are normal features. That they are erased when they are in a checking relation is a peculiar property of checking that is primarily conditioned by other factors, mainly the presence of some F^{INV} or F^{VIS}. Hence there is no good reason why deleted features should be treated in any special way when it comes to the CFD. The only exception is $\mathfrak{C}3$, which needs the weaker CFD in order to allow for rendering Delete(α) as copying of α to $INV := \{F^{INV}\}$ *plus* Erasure of the original α.[22]

3.2.3 Evaluating the Results

The results we obtained are rather complex, although it could have been much worse. We had two binary and two ternary parameters for Reuland's theory and in addition five different feature checking systems, so there were 180 logically possible theories to be evaluated. Fortunately we only needed to distinguish two feature checking systems for most of the parameter settings, so that the workload was reduced dramatically. Still, it is anything but a sign of scientific preciseness when that many interpretations of a single theory are indeed imaginable.

The table below aims to summarize the results as clearly as possible. Working theories are indicated by ✓, non-working theories by ✕, overgenerating theories by O, contradictory theories by ↯, and implausible theories by U. If a feature checking system is incompatible with some parameter of the respective Reulandian variant, the field is left empty. Parameter values are only specified where they make any meaningful difference for the results. So if a theory is contradictory in general, but furthermore implausible with some specific parameter value, this will be ignored because being contradictory is already bad enough for the theory to be ruled out.

$\mathfrak{R}_E$ is contradictory under Reuland's reading of (97), else it would probably be an empirically inadequate theory, depending on whether there are any situations where an anaphor enters new checking relations after it has acquired additional ϕ-features. If LIs do not have a fixed phonetic exponent, it overgenerates as pronouns are predicted to behave like anaphora.

This leaves us with variants of $\mathfrak{R}_D$. $\mathfrak{R}_D^O$ does not work for technical reasons, because no features can be copied onto the anaphor as long as the original features are still present, which is a given if there is no Erasure. Any variant of $\mathfrak{R}_{CH}$ is conceptually implausible, be-

[22]One could argue that copying being a subpart of Delete(α) is different from copying as a separate simplex operation, especially considering that Delete(α) and Erasure do not induce new derivational steps. However, I don't think that this point is important enough to merit further discussion. Therefore, I will just assume that the syntactic CFD and $\mathfrak{C}3$ fit conceptually, although it is disadvantageous to my enterprise.

	$\mathfrak{D}$	$\mathfrak{DE}/\mathfrak{E}1/\mathfrak{E}2$	$\mathfrak{E}3$
$\mathfrak{R}^{S}_{\emptyset/\mathcal{CH},D}$	✓	U/✓	✓
$\mathfrak{R}^{L}_{\emptyset/\mathcal{CH},D}$	U/✓	U/✓	U/✓
$\mathfrak{R}^{S/L,1C}_{CH,D}$	U/✓	U/✓	U/✓
$\mathfrak{R}^{S/L,2C}_{CH,D}$	U/O	U/O	U/O
$\mathfrak{R}^{O}_{D}$	×	×	×
$\mathfrak{R}_{E}$		↯	↯

Table 3.1: Feasibility of Reulandian variants

cause it requires that copies are suddenly available to syntactic computation after all covert operations have taken place. $\mathfrak{R}^{2C}_{CH}$ also overgenerates because some copies in $\mathbb{C}$(anaphor) could be mistaken for actual antecedents. Any variant of $\mathfrak{R}^{L}$ is conceptually implausible, too, as it restricts a syntactically needed principle to the lexicon. Similarly $\langle \mathfrak{R}^{S}, \mathfrak{F}'' \rangle$, $\mathfrak{F}'' := \{\mathfrak{DE}, \mathfrak{E}1, \mathfrak{E}2\}$, is implausible because those feature checking systems provide no motivation to treat deleted features different from undeleted ones.

If we keep in mind that assuming a single cycle allowed us to account for binding by wh-words, in contrast to systems with two cycles, and that $\mathfrak{R}_{\mathcal{CH}}$ can account for Icelandic long-distance binding, whereas $\mathfrak{R}_{\emptyset}$ can't, it follows that the optimal variants are $\left\langle \mathfrak{R}^{S,1C}_{\mathcal{CH},D}, \mathfrak{D} \right\rangle$ and $\left\langle \mathfrak{R}^{S,1C}_{\mathcal{CH},D}, \mathfrak{E}3 \right\rangle$. That is, out of 180 possibilities, 2 are conceptually and empirically sound. If we focus on conceptual soundness alone, the number increases to a whopping 8, i.e. less than 5 percent.

Ironically, no such system is used in Reuland (2001). Taken literally, he uses the variant $\left\langle \mathfrak{R}^{O,2C}_{CH,D/E}, \mathfrak{DE} \right\rangle$, which evidently is a total failure. If we grant him the use of the weakened CFD, as the issue does not really become apparent if one simply adopts Chomsky (1995b) without further questioning, the result is either $\left\langle \mathfrak{R}^{S,2C}_{CH,D}, \mathfrak{DE} \right\rangle$ or $\left\langle \mathfrak{R}^{S,2C}_{CH,E}, \mathfrak{DE} \right\rangle$. Reuland claims that those theories are equivalent, but evidently this is not the case. The former is implausible and overgenerates, while the latter is in addition contradictory. So not only do both fail, Reuland also mistakenly asserts that they should fail in the same way. Even if $\mathfrak{R}_{CH}$ is replaced by $\mathfrak{R}_{\mathcal{CH}}$ the doubtful status of this conjecture does not improve. But at least this move gives us a working, although implausible and empirically less than stellar theory, $\left\langle \mathfrak{R}^{S,2C}_{\mathcal{CH},D}, \mathfrak{DE} \right\rangle$. Delaying the moment of feature copying after Erasure, on the other hand, results in $\left\langle \mathfrak{R}^{S,2C}_{\mathcal{CH},E}, \mathfrak{DE} \right\rangle$, which is still contradictory.

The status of $\mathfrak{R}_{E}$ would be equivalent to $\mathfrak{R}_{D}$ if we allowed syntax to browse the derivational history, i.e. to use lookback, for it could then determine which features were present prior to Erasure, thereby preventing additional features to be copied. We also would not need to worry about how syntax can determine whether all ϕ-features have been checked by the same LI, and in turn there would be no reason to conjecture that features must not be present twice on the same LI, with obvious positive repercussions for the entire proposal. Adopting lookback, though, would remove the last bit of elegance from Reuland's theory. Methodologically, supplying a derivational theory with lookback is against the basic idea of a derivational system, namely that the input of a transformation isn't available later on, because the output of every operation is atomic. It is bad enough that Minimalism is a weakly representational theory (Brody 2002) that allows parts of the tree to be visible and available for further computation even after they been manipulated by syntax. But extending the representationality beyond representations to total derivations is even more conceptually misguided than it is computationally burdensome.

Attempts to accommodate lookback through global economy are misguided, too. While lookback is a possible implementation of global economy, it is not the only one. We could easily assume that every derivation is assigned a score which is increased depending on the operation carried out. As a consequence, we still arrive at a total order of derivations from which we can pick the most economical one (which is the one with the lowest score), but in no way is it possible to deduce how this derivation has become the most economical one.[23] Not so surprisingly, the problem of overgeneration could also be solved by an additional transderivational contraint which when supplied with converging derivations containing the same CHAINs picks the one with the least number of features. But remember that we already settled in section 1.4 that the inclusion of global economy never was a good idea to begin with. Loading it with more and more work is technically possible, and I actually did this to keep the number of hidden premises in Reuland's article artificially low, but it is not a wise move if we want the result to be compatible with other theories.

The last point highlights a crucial fact we up to now did not pay enough attention to, namely how does the theory perform if we consider that there are myriads of approaches to empirical phenomena which we might want to use concomitant with Reuland's binding theory, each of them coming with its own set of premises which might not line up with those we used in this chapter.[24] Though it might not be technically trivial, it definitely is possible to get every analysis flying if one introduces a sufficient number of additional premises and modifies the old ones until they fit. To some extent, that is what I had to do in this chapter in order to do any meaningful comparisons. I also tried to derive assumptions wherever possible, because I do not want to be criticized as unfairly biased against Reuland. I accounted for the asymmetry of checking from global economy, and I reduced the restriction to subjects as only viable backup to economy considerations (as already pointed out in footnote 11 I am not particularly happy with this move), and I worried about the feature composition of ϕ-feature bundles. In my variant, Reuland is uncomfortably dependend on global economy, in the original version he entertains an awful lot of hidden assumptions.

[23]The mechanism only captures quantifiable economy, such as the Smallest Derivation Principle, but keep in mind that the only qualitative global economy metric used in Chomsky (1995b) is the doubtful and apparently superfluous Has an Effect on Output Condition. Even if we inseparably tie Reuland's theory to Tanya Reinhart's Rule I we are not forced to use lookback. In fact, employing lookback allows us to state Rule I as a local constraint (Graf 2007), which should be taken as evidence that lookback is an incredibly powerful device.

[24]Just consider the following example: $\mathfrak{D}$ with the CFD (no matter whether weak or strong) is now incompatible with the checking system of Nunes (2000) which it was actually modeled after.

Both options inevitably lead to the problem noted above, decreased compatibility with other theories. For instance, turning checking into an anything-goes option such that the asymmetry of feature copying can be derived minimizes the set of compatible theories to those which assume global economy (or at least do not condemn it explicitly). There is no way to circumvent this than by adding the asymmetry as another stipulation to the already long list of definitions in section 3.1.3, and this in turn rules out any theories that rely on symmetric checking in any case. And with every new stipulation, every single one out of the 180 Reulandian variants becomes less and less attractive, even those 8 that could be made to work.

Now assume that those other theories we want to combine with Reuland (2001) have their fair share of vagueness, too. Even if there were only three ambiguous theories and each of them had exactly two conceivable interpretations, the set of possible variants would increase to 1440 (which is way better than the usual case of having a vague theory for which the number of possible interpretations cannot be determinded easily). Sure, many of those tuples of theories would probably group into classes, reducing the amount of work needed to assess their soundness, but even then we would face the problem that we could try to redeem invalid classes by adding new premises that do not conflict with the other ones. As there are no formal measures for what constitutes a good Minimalist assumption, this task has to be done by humans. At the same time, those new assumptions have to be reevaluated every time a new theory is added. Even if we decide to prohibit the introduction of new postulates, there remains the task of determining all premises of a given theory, also the implicit ones. At this point, it should be clear that doing so for a bigger number of papers is a herculean task. I am pretty sure that nobody wants valuable ressources to be wasted on the exegesis of massive amounts of vague phrasing, and vagueness, ambiguities, logical holes and contradictions are bound to arise if one isn't extremely careful when adopting and combining the ideas of others.

Chapter 4

Employing Formal Tools

This chapter will be concerned with familiarizing the reader with recent developments in the realm of logic and formal grammar and how they can be used to make linguistic theories more precise. I start with a discussion of the implications of Reuland's failure, and to which degree it is typical of other Minimalist articles. In connection with this I speculate on the reasons for why there seems to be so little concern about this issue in contemporary linguistics. Afterwards, I demonstrate how logic can be used by linguists to decrease vagueness and how it could simplify the evaluation of theories. Section 4.3 focuses on formal grammar, especially Minimalist Grammars (Stabler 1997) as a current area of research that produces many valuable results relevant to syntacticians. Crucially, I do not want to insinuate that formal approaches were superior to mainstream linguistics, I merely wish to demonstrate how generative grammar can benefit from adopting logic as a metalanguage and paying attention to results from more mathematically inclined areas. In contrast to the occasionally very aggressive critique coming from outside the P&P-community, I am not striving for a replacement of Minimalism, I am merely offering ways to enhance Minimalist research.

4.1 Some Personal Reflections on Vagueness in P&P

My careful examination of Reuland (2001) demonstrated the negative impact of vague definitions on scientific theories, with only 2 out of 180 possible interpretations being somewhat conceptually and empirically tenable, albeit not particularly attractive (and those 2 are unlikely to represent the system Reuland had in mind). This situation did not arise from ambiguous definitions exclusively; contradictions and holes in the theory contributed their notable share, too. This indicates that vagueness is an obstacle to both the reader and the author. While the former has to do considerable additional work to get more out of an article than a purely intuitive grasp of the main ideas, the latter is prone to be tricked into overlooking weak points in his arguments. When this becomes a chronic problem of a scientific community, the integrity of its framework is in danger. That is not to say that a community has to agree on a canonic theory which must not be altered, but when scientists can't be sure whether they are actually working with the same parameters, doing research becomes a solipsistic activity where one can't build on the work of others without risking contamination with hidden yet serious errors.

Hence the central question we have to ask ourselves is: does this danger exist for Minimalism? In other words, do other Minimalist articles share Reuland's shortcomings? A statistically informed reply to this question is outside the scope of this thesis, nevertheless I think that the answer is positive, unfortunately. I do not contend that the greatest part of Minimalist research was dubious from a formal perspective, or that nobody was interested in a coherent and elegant theory. Evidently there are linguists whose inquiries were invaluable for the maturation and enhancement of recent generative grammar, just think of Brody (1995), Collins (1996), Chametzky (2000), and particularly Gärtner (2002), to name but a few. Still, many articles show a tendency for sporadic handwaving and omission of non-trivial parts, and consequently a considerable number of holes and inconsistencies is introduced into the system. I shall refer to such deficits as *gaps*.

A distinction can be drawn between isolated and propagating gaps, depending on whether the assumptions in the article are adopted by other scientists or not. Obviously propagating gaps are the real danger, while the negative impact on isolated gaps is restricted to a single paper and hence only affect the author. Yet isolated gaps are the initial state of propagating gaps and should therefore be avoided at any price for more altruistic reasons, too. Chomsky evidently is the main source of propagating gaps, due to his reputation as a leading figure in the community. He himself is the least likely person to appreciate this "follow the leader"-attitude, judging from how often he has called Minimalism a framework, which implies that he expects other people to join the debate with their own projects instead of simply adopting or slightly modifying his most recent proposal. I do not want to speculate on the reasons why Chomsky's main ideas[1] usually find rapid adoption, whereas alternative proposals by other linguists struggle to build up a considerable following, but it is a fact of the history of the P&P framework.

Let us look at some examples for both kinds of gaps, starting with isolated gaps. They are of course less interesting than propagating gaps due to their non-pervasiveness, but keep in mind that many problems of Reuland (2001) arose because of the interaction of propagating and isolated gaps. Epstein et al. (1998) provide us with a good example for an isolated gap when they conjecture that a node might be the daughter of two distinct branching nodes (a comparable proposal is put forward in Hornstein and Nunes 2006). Without further qualifications, this is simply impossible provided that phrase structure corresponds to sets, because it would require that one element could belong to two distinct sets at the same time. I see only one tenable way to get the intended result, namely stipulating that a sentence may consist of multiple non-disjoint sets, with the occurrences of identical elements as their joints. So the sets $\{\underline{a}, \{a, b\}\}$ and $\{\underline{c}, \{b, c\}\}$ would describe a tree with two roots and three leaves, a, b, and c, with b acting as joint. But apparently that's not what Epstein et al. (1998) had in mind, for they do not discuss this implemen-

[1] By 'main ideas' I am referring to anything save very specific assumptions he makes for the analysis of some empirical phenomenon, e.g. that Icelandic Case features need to be deleted twice.

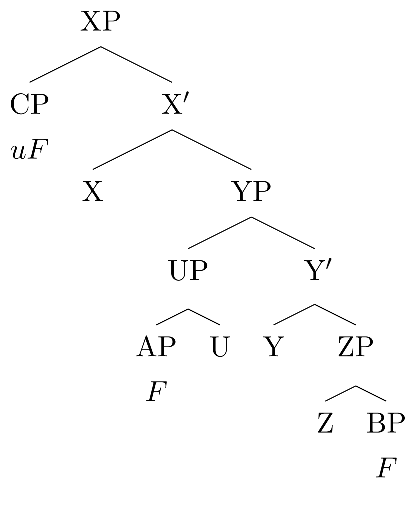
tation anywhere. This isolated gap, though, affects only peripheral details of the theory, fortunately. Such problems hence might decrease the value of a proposal, but they do not endanger it in the way we encountered in the previous chapter. Only if isolated gaps affect core parts in a negative way and cannot be wiped out easily because of certain restrictions imposed by other theories does a Reulandian scenario arise. However, the likelyhood of such a situation increases with the number of premises one adopts from other articles, trivially because their inherent gaps are imported, too, and the interaction of different premises might create new gaps. I am pretty sure every reader has encountered a considerable number of papers where, just like above, either certain points are missing or specific premises collide with each other. In any case, the details just do not fall neatly in place.

As for propagating gaps, their impact on the overall quality of an article varies, too, but more often than not the consequences are more severe by virtue of the fact that propagating gaps usually concern central assumptions, which is the reason why they did not remain isolated gaps in the first place. Let us look at a gap of intermediate relevance, closeness as it is defined in Chomsky (1995b). We already encountered the relevant definition in section 2.5, but I repeat it here for the reader's convenience.

(116) *Closeness* (cf. Chomsky 1995c:356)
If β c-commands α and τ is the target of raising, then β is closer to τ than α unless β is in the same minimal domain as τ or α.

Cases where α and β do not share the same minimal domain and where no relation of c-command holds between them are not covered by this definition. Such a structural configuration is depicted in (117).

(117)



Please note that it does not matter whether we ban proper extraction out of specifiers such that AP is frozen in place. Suppose that the closeness condition applies even if movement of the closer constituent is blocked, which seems to be the case in many languages. Then we still fail to predict whether the frozen AP could induce intervention effects preventing BP from moving to CP, because we do not know whether AP is closer to CP than BP.

Further propagating gaps in Chomsky's work include the constructions of numerations, the definition of Select, adjunction in BPS (cf. Chametzky 2003), the structure of

LIs, feature percolation, or the properties of transderivational constraints. Epstein and Seely (2006) show that chain formation is suspicious too, because chains apparently are not induced by movement but by a special algorithm connecting the landing position with the position where the moved element was introduced into the derivation. Moreover, the addressing system is flawed and does not allow unequivocal identification of chain positions with the respective landing sites. A minor, yet amusing quirk is introduced by the weakened version of the phase impenetrability condition, which states that Spell-Out of a phase XP takes place as soon as the next higher phase YP has been assembled. Without further modifications, this won't allow the highest phase in a sentence to be spelled out.

I conclude this list with an almost anecdotal example drawn from Chomsky (2005a). First, consider Chomsky's comment on remerge and the copy theory of movement (Chomsky 2005a:fn.16).

> There has been much misunderstandings since the copy theory was proposed in Chomsky (1993), modifying earlier conceptions of movement by eliminating trace and indexing in favor of the N[on] T[ampering] C[ondition]: that is, leaving the moved element unaffected instead of replacing it by an indexed trace (indexing is now superfluous, under identity). It has sometimes been supposed that a new "copy" is created, then inserted in the position of the moved element — all unnecessary — and an alternative has been proposed in terms of "remerge," [sic] which is simply the copy theory as originally formulated.

Now read what Chomsky has to say a few pages later (Chomsky 2005a:11).

> There must be some way to identify internally-merged α with its copy, but not with other items that have the same feature composition: to distinguish, say, "John killed John" or "John sold John to John" (with syntactically unrelated occurrences of *John*), from "John was killed John" (with two copies of the same LI *John*). That is straightforward, satisfying the inclusiveness condition, if within a phase each selection of an LI from the lexicon is a distinct item, **so that all relevant identical items are copies** [my own emphasis; TG].

Although the second quote is in principle compatible with a remerge approach, this is clearly not the favored reading. Besides, all the confusion could haven been avoided if the copy theory would have been stated explicitly in Chomsky (1993). Instead, it is introduced in the context of generalized transformations and the technical details remain vague. Interestingly, Chomsky (1995b) leaves their details completely implicit, despite the change from $\overline{\text{X}}$-structure and generalized transformations to BPS.

From the examples given above we see that the formal sloppiness that led to Reuland's failure is also common in other Minimalist articles, although the consequences seem to be usually less severe (considering the effort that was needed to give a useful analysis of Reuland (2001), it could just as well be the case that I am simply underestimating their severity). A crucial point to observe is that all the tools in the examples I discussed are lacking in departments which cannot be decided on purely empirical grounds, so leaving them undefined does not do any good. In the worst case, it will even hide that there is a problem waiting to be solved.

Now that we assured ourselves that we aren't engaging in a phantom discussion based on a single non-representative example, the natural question is how this problem is currently being dealt with. The inconsistencies of "Chomskyan theories" have frequently been pointed out before by advocates of different frameworks, with the infamous LLJ-controversy (see Lappin et al. 2000b,a, 2001 and the respective Minimalist replies) as the latest exponent. Even in the few cases where such critique actually provoked any responses, the results were meager at best. Most of the time the underlying reasons for the dissent stem from different views on what a linguistic theory has to account for, which is especially apparent in the 2005 Linguist-List discussion evolving around a dare by Shalom Lappin and Richard Sproat. They claimed that nobody could develop a tractable P&P-based parser until May 2008 which would be able to achieve results in the league of parsers in use in natural language processing at the time of their writing.[2] The underlying assumption, of course, is that a linguistic theory not only has to concern itself with issues of computability, but also has to be efficiently computable. Few linguists in the P&P-tradition would accept the second assertion without qualifications. A quote by Sean Fulop captures their view most eloquently.[3]

> Worrying about tractability in P&P is like denigrating relativity theory because it makes it needlessly harder to calculate artillery ballistics.

However, both confrontations also reveal a certain reluctance of the P&P-community to accept or even perceive criticism from the outside. That might be partially due to the ranting subcontext that frequently accompanies the technical and methodological criticism, as Martha McGinnis laments in one of her replies:

> Remember, Sproat and Lappin began the discussion, not by offering to play together, but by declaring that Minimalism/P&P can't be taken seriously until it produces a large-scale trainable parser. It's not surprising that many people's first reaction was to refute this assertion. If the suggestion had been to play together, the response would have been very different.[4]

Comments related to the LLJ-controversy, however, indicate that at least some advocates of P&P do not appreciate criticism from outsiders in general. On the HPSG-l mailing list, Lappin, Levine and Johnson cite an infuriated passage from a mail by Andrew Carnie, consulting editor at Linguist-List.[5]

> Any good scientist attempts to be aware of the real problems in their own

[2]The entire mail correspondence is available in Linguist-List's archive under http://linguistlist.org/issues/indices/Disc2005.html [verified on 02-June-2007].

[3]See LinguistList for what else he has to say on this issue: http://listserv.linguistlist.org/cgi-bin/wa?A2=ind0505a&L=linguist&P=8624 [verified on 02-June-2007].

[4]The full reply is available at the following URL: http://linguistlist.org/issues/16/16-1580.html#2 [verified on 02-June-2007].

[5]Their message is available at: http://listserv.linguistlist.org/cgi-bin/wa?A2=ind0406&L=hpsg-l&D=0&P=3573 [verified on 02-June-2007].

> work, even if they choose to temporarily ignore them for practical reasons. The MP world does not need HPSG critics, it has its own.

Irrespective of the ultimate reasons for this polemical uproar (considering the rhetoric of the addressees it didn't come entirely unasked for), the reaction indicates a certain dislike for unasked contributions from other frameworks. Although it is merely a speculation, I think that this attitude towards competing theories could have had a negative impact on the status of mathematics and computational sciences in mainstream linguistics. After all, the people voicing the critique are often working with more formally or descriptively oriented theories, so calls for a step in this direction might be met with a feeling of repulsion. Even if there was a glimpse of truth in my surmise, one must not overrate its importance. Its explanatory value is restricted, because it covers only the status of mathematical tools, yet preciseness does not depend on mathematics, although it can profit enormously from it. The general lack of formal rigor thus has to have other underlying causes, the most likely one being simply the intuition that the system is already precise enough with respect to what is to be accomplished.

No matter what ultimately led to the ubiquity of vagueness, the fact remains that it is currently an undesirable trait of the P&P framework. I already showed that we are a far cry away from a well-defined theory, but as I do come from a Minimalist background and tried to avoid any inflammatory formulations, I am confident that nobody will be overly offended by this conclusion, nor by my proposals how the situation could be improved. Some readers will certainly disagree with me on whether there's any actual benefit to following my recommendations, but hopefully no one will perceive my critique as simplistic Minimalism bashing.

Those who do share my point of view, though, will probably raise another issue: while it is evident that precision is a core tenet of contemporary science, it is a delicate question at which level the style of exposition can be considered sufficiently precise. This has direct implications for what qualifies as a means to reach this level. I can't deny that formal accuracy does not rely on any special notation or mathematical devices, plain English is in principle sufficient — under the proviso that it is used carefully. However, mathematical tools are less likely to be used in a sloppy way and thus grant a higher chance to produce unambiguous statements (although they might still be nonsensical, of course). In addition, a formal description language might have extra advantages like allowing us to test the consistency of a theory in a purely automatic fashion or suggesting natural ways to extend our proposals. In the remaining sections, I will present some expedient formal tools, starting with logic and moving into the realm of formal language theory later on. Owing to the general aim of this chapter, namely demonstrating the usefulness of a more mathematically informed approach to constructing theories, I will try to give an accurate assessment of the effort it would take to integrate the respective tools into current research methodology. I hope that after reading this introductory survey, at least some people will indeed be eager to push the use of those powerful tools.

4.2 Using Logic

4.2.1 Logic as Metalanguage

If some idea is presented in a sloppy fashion, it is usually plagued by one of two issues. Either the wording is ambiguous or some crucial parts are left undefined. The precision of logic provides a solution to both problems. Firstly, a well-formed logical formula cannot be ambiguous,[6] and secondly, one can easily figure out whether all occurring primitives have already been defined. Consider the claim of Chomsky (1995b) that Delete(α) marks a feature as invisible at LF. Because this marking procedure never plays a distinctive role in any analysis, just as the distinction between Delete(α) and Erasure is very unlikely to be relevant for most empirical research, many readers probably won't waste too much time on this issue. But we saw in chapter 2 that how a feature is marked as invisible is anything but a trivial issue. Even worse, the implications for the whole framework might be devastating. Now look at the definition in (118).

(118) a. $\forall x \forall y \exists z \exists z' [(F(x) \wedge F(y) \wedge z = LI_x \wedge z' = LI_y \wedge x \epsilon LI_x \wedge y \epsilon LI_y \wedge (LI_y \in \mathbb{C}(\mathrm{FF}[LI_x]) \vee \mathrm{FF}[LI_y] \in \mathbb{C}(\mathrm{FF}[LI_x])) \wedge match(x, y)) \rightarrow ((unint(x) \rightarrow Delete(x)) \wedge (unint(y) \rightarrow Delete(y)))]$

b. $\forall x [Delete(x) \rightarrow invisible(x)]$

The formula in (118a) states the conditions under which Delete(α) takes place. Suppose that all predicates except *Delete* have already been logically defined, such that F designates a formal feature, $\mathbb{C}(LI_x)$ the checking domain of LI_x, $\mathrm{FF}[LI_x]$ the set of formal features of LI_x, $match(x, y)$ is true iff x and y are identical and in the same feature class, $unint(x)$ only holds true of uninterpretable x, and $\in$ and ϵ have the familiar denotation. The formula, albeit convoluted, tells us immediately that *Delete* still has to be defined. So we proceed to (118b), which defines the operation itself. There we encounter another new predicate, *invisible*, which if left undefined will deprive (118b) and hence (118a) of any relevant meaning. Now three possible situations have to be considered: *invisible* is not defined anywhere, or it is defined logically, or it is defined in plain English. The negative implications of the first option are evident, and so is the positive upshot of the second option. A possible formalization of *invisible* is given in (119) below, based on the $\mathfrak{DE}$-system in (84).

(119) $\forall x [invisible(x) \leftrightarrow \exists y \exists z [F^{INV}(y) \wedge Bundle(z) \wedge y \in z \wedge x \in z]]$

Closer scrutiny of (119) raises some issues that are intimately related to the third option above, defining predicates in natural language, for, at some point, this is the only option left. As we are lacking an articulated theory of LF, there is no way we could meaningfully define F^{INV} in our logical language without recourse to the predicate *invisible*, which would be circular. That is to say, if we want to say more about F^{INV} than that

[6] Ambiguities can of course arise if the logical definition of module X does not match the actual use of X in the article. But if this happens to be the case, it is already indicative of overall low quality.

it is a distinguished element required for well-formedness of syntactic representations, we either have to define its effects on LF, or we have to fall back to intuitive terms to explain why F^{INV} should be relevant for well-formedness. The more ground of our linguistic theory is covered by the logical description, the more predicates and elements can be defined *insightfully* in logic. Adapting some logical description language hence does not eradicate the need for non-formal motivation of the definitions. In some cases, a basic predicate might even be defined in purely intuitive terms if its effects apply outside the domain that is covered by our logical formalization. Consider again the predicate *invisible*. If we opted to define it in plain English, rather than in logic, because we do not want to use any F^{INV} or a comparable method that could be described in our logic, it is a sound move to define the predicate and its motivation in plain English. However, the attention of the reader would be immediately focused on this non-formal definition, and he or she would critically assess its soundness, simply because non-formal definitions are suspect in a setting where everything is defined formally (in the very case of *invisible*, the readers would probably conclude that a logical definition of the marking mechanism but not its impact at LF should be found).

Obviously this can also be achieved with natural language definitions if one is careful to keep them very precise and to set them apart from the rest of the article. For instance, the definitions of checking domain in Chomsky (1995b) are sufficient from a formal perspective. Nevertheless logic does a better job at directing the attention of the reader, and it is not prone to ambiguity, in contrast to natural language. Considering how little effort it takes to write definitions in logic, I see no profound reason to refrain from using it as a metalanguage for our linguistic theories, provided that it is adopted by the entire community. In this case it even carries the prospect of easing the integration of foreign proposals, strengthening collaborative research.

The most important point to observe here is that we are talking about using logic as a metalanguage, just like we're using natural language as a metalanguage to talk about language. In no way is this proposal related to integrating logic into syntactic theory, which inevitably requires a major rethinking of our conception of syntactic processes and constraints. That is not to say that logical syntax wasn't a plausible theory, nor that Minimalism can't be recast in such a framework (cf. Vermaat 1999; Lecomte 2003). In fact, categorial grammar (Buszkowski et al. 1988; Steedman 1996) offers such a logical approach to syntax, and it has proven itself over decades as an insightful theory intimately relating syntax, semantics, parsing and computability. Categorial grammar assumes that category labels are complex and regulate the way they can combine with each other. The combinatorial rules are based on the rules of logical calculi, such that the assembly of syntactic structure is likened to logical reasoning. This idea has been coined 'parsing as deduction'.

Consider the following simplified example. Suppose that labels can be recursively built from the types np and s, our atoms, and three binary type-forming operations $\bullet$, $/$ and $\backslash$, such that if A and B are well-formed formulas, then $A \bullet B$, A/B and $A\backslash B$ are well-formed formulas. Suppose further that our grammar contains the theorems given below.

$$\frac{\Gamma \vdash A/B \qquad \Delta \vdash B}{\Gamma\Delta \vdash A}\ [/\mathrm{E}] \qquad\qquad \frac{\Gamma \vdash B \qquad \Delta \vdash B\backslash A}{\Gamma\Delta \vdash A}\ [\backslash E]$$

Then the sentence *He kisses her* is to be derived as follows.

$$\cfrac{\cfrac{\cfrac{he}{n}\ \text{Lexicon} \qquad \cfrac{kisses}{n\backslash(s/n)}\ \text{Lexicon}}{s/n}\ [\backslash E] \qquad \cfrac{her}{n}\ \text{Lexicon}}{s}\ [/\mathrm{E}]$$

Obviously this trivial example does not do full justice to the power and elegance of categorial grammar (see Moortgat 1996 and Steedman and Baldridge 2003 for in-depth introductions), but it is more than sufficient for illustrating the difference between using logic in syntactic theory and using logic in the description of syntactic theory.

4.2.2 Automatic Theorem Proving

Besides the psychological reasons for adopting logic as a tool of theory construction, there are also more technical ones, but it takes additional efforts to harvest them. One advantage revolves around automatic theorem provers. As indicated by their name, those are programs that decide the validity of theorems, and they have become an essential tool in both mathematics and computational sciences. Their potential benefits for linguistics are enormous, since automated theorem proving allows us to check our theories for logical gaps, provided they are fully formalized. But please note that this verification proceeds in purely algorithmic terms, the semantics of the formulas cannot be tested, for obvious reasons. Consequently automated theorem proving does not help us when it comes to picking useful axioms. In the case of the predicate *invisible*, for instance, we still have to decide on our own what its denotation is, and whether it can and should be stated in more primitive notions, based on our understanding of the relevant syntactic theory.

Stabler (1992) uses an automatic theorem prover to verify his formalization of Chomsky (1986), giving stellar results concerning the interaction of various submodules. Recall that GB is a generate-and-filter approach, where the initial structure is determined by Deep Structure from which Move α generates all logically plausible permutations. Those are in turn filtered out by various syntactic submodules enforcing specific constraints at SS or LF. How those submodules interact and depend on each other wasn't understood in full detail prior to Stabler's work. Chomsky (1986), for example, asserts that barriers depend on subjacency, so any theory deprived of subjacency should not be a valid base for the barriers approach. Stabler, on the other hand, shows that subjacency is no critical prerequisite, whereas the Empty Category Principle (ECP) is. The appeal of this result is twofold: we gain a new understanding of our theory and how it really works, and we obtain new means to assess the prospects of certain research programs. In the specific case of Chomsky (1986), we can rule out the possibility of removing the ECP and still retaining barriers. Without Stabler's work, we would need to resort to mainly empirical arguments to determine whether the ECP could be dispensed with, which is a lengthy process.

In the ideal case, automatic theorem provers could be used by any linguist to see whether their proposals make unexpected predictions, whether they are compatible with the original theory, and if not, which parts ought to be altered. This scenario, though, is utopian. First the standard theory (or variants of it) needs to be fully described in logical terms which are explicit enough for formal proving. Then those formulas have to be rewritten in a language that the theorem prover understands. The only straight-forward way to do this is a logical programming language (Stabler uses Prolog), but programming skills are rare among linguists. If a linguist has some knowledge in a programming language, it is usually a scripting language like Perl, Python or Ruby, which are ideal for working with large text corpora but have virtually nothing in common with logical programming.

So there are only two possibilities to bring automatic theorem provers to the masses: teaching students logical programming, or developing user-friendly software that automatically translates logical definitions into programming code and then does the theorem proving. The first option requires additional courses and resources which hardly any linguistic department can afford. The second option is too expensive and time-consuming to realize. We also can't expect a commercial software vendor to produce a kind of Maple for linguistics any time soon. Maple is a commercially distributed computer algebra system widely used in mathematics. It comes with both its own programming language and a graphical user interface for entering formulas directly in mathematical notation. Obviously the creation of such a piece of software does not pay off if the market is too small, which is the case for linguistics. Because of all these additional hurdles, automatic theorem proving is most likely to remain restricted to a few linguists only. However, as long as those are enough to produce a continuous stream of research papers, the analytic insight that comes with automatic theorem proving could have a positive effect on the development of linguistics as a whole.

4.2.3 Tree Logics

At the beginning of the nineties, Barker and Pullum (1990) sprouted new interest in the mathematical properties of command relations on trees. Their general definition of command relations can be rephrased as in (120), where $\triangleleft^*$ is the reflexive transitive closure of the *mother-of* relation $\triangleleft$, such that $k \triangleleft^* w$ if and only if k (reflexively) dominates w.

(120) *Command relations on trees* (cf. Blackburn and Meyer-Viol 1994:8)
Let T be a tree and P be a unary relation on the nodes of T. The P-command relation C_P on T is defined as $\{\langle w, v\rangle : \forall k[k \neq w \wedge k \triangleleft^* w \wedge k \in P \rightarrow k \triangleleft^* v]\}$.

The definition is pretty abstract, so let us look at a practical example, c-command, which holds of two nodes w and v if and only if w does not (reflexively) dominate v and the first branching node k irreflexively dominating w also dominates v. These conditions are a superset of those expressed in (120), if we define P as the unary relation holding only of branching nodes. If, however, we decide to define P as the unary relation holding only of maximal projections of w, we get m-command (Aoun and Sportiche 1983) by altering the first condition of c-command such that v does not (reflexively) dominate w. Finally, if we define P as the relation holding of any VP and add no further conditions, we arrive at VP-command which holds between any two nodes dominated by the same VPs.

Inspired by the approach of Barker and Pullum (1990), Kracht (1993), Blackburn et al. (1993), and Blackburn and Meyer-Viol (1994) started their own investigations of command-relations using modal logic. Although the relevance of the outcome was negatively affected by the mathematically uninteresting properties of command-relations,[7] they demonstrated the applicability of modal logic concerning the description of trees and relations defined on them. Rogers (1998) improved on this with his inception of $L^2_{K,P}$, a monadic second-order logic. Monadic logics allow only one argument per predicate, and second-order logics allow quantification over both individuals and sets of individuals. The power of $L^2_{K,P}$ makes it particularly easy for linguists to state well-formedness constraints on trees because those tree structures can be included directly in the logical statements. In order to achieve this, we use the basic connectives in (121) to define the predicate CHILDREN in (122).[8]

(121) a. Classical logical connectives: $\wedge, \vee, \rightarrow, \leftrightarrow, \neg$

b. $\triangleleft$ = immediately dominates

c. $\prec$ = immediately precedes

d. $\approx$ = is equal to

(122) CHILDREN$(x, y_1, \cdots, y_n) \equiv$

$$\bigwedge_{y_i:1\leq i\leq n} [x \triangleleft y_i] \wedge \bigwedge_{i\neq j}[\neg[y_i \approx y_j]] \wedge \forall z[x \triangleleft z \rightarrow \bigvee_{y_i:1\leq i\leq n} [z \approx y_i]]$$

The big connectives in (122) have to be read as operators like $\sum$ or $\prod$. So if our syntactic theory allows only ternary branching, n is restricted to 2 and (122) can be written as in (123).

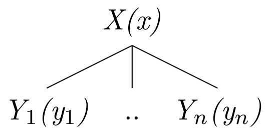
(123) CHILDREN$(x, y_1, y_2) \equiv$

$$[x \triangleleft y_1] \wedge [x \triangleleft y_2] \wedge [\neg[y_1 \approx y_2]] \wedge \forall z[x \triangleleft z \rightarrow [[z \approx y_1] \vee [z \approx y_2]]]$$

The definition itself should be easy to understand, it simply states that the set of children of x comprises all distinct nodes immediately dominated by x and nothing else. With the new predicate in place, we can use trees as graphical representations of formulas as indicated below.

[7]Take c-command, which is a transitive and irreflexive, but non-symmetric relation. The last property renders it uninteresting from a mathematical perspective, as there isn't a lot one can say about non-symmetric relations. As its name implies, asymmetric c-command isn't non-symmetric but asymmetric, and hence it defines a strict partial order, making it more appealing to a mathematician. M-command has the same properties as normal c-command.

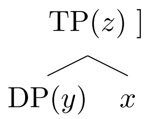
[8]The definitions to follow are taken from Potts (2001).

(124) 

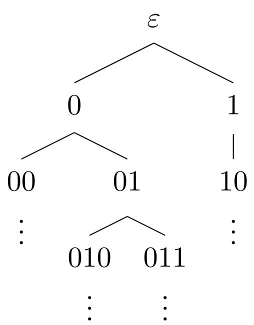
$$\equiv \text{CHILDREN}(x, y_1, ..., y_n) \wedge X(x) \wedge \bigwedge_{y_i:1\leq i\leq n} [Y_i(y_i)]$$

The number of daughters in the tree is captured by the CHILDREN predicate, $X(x)$ specifies that node x has the label X and the last subformula assigns every daughter y_n the label Y_n. We thus have a direct way to use trees in our logical statements, which makes them easier to read. For illustratory purposes, consider the equivalent definitions of the EPP in (125). For the sake of easy exposition, the EPP has been simplified so that only DPs are valid subjects.

(125) a. $\forall x[\mathrm{T}(x) \rightarrow \exists y, z[\mathrm{TP}(z) \wedge \mathrm{DP}(y) \wedge z \triangleleft x \wedge z \triangleleft y \wedge y \prec x \wedge \forall v[z \triangleleft v \rightarrow [y \approx v \vee x \approx v]]]]$

b. $\forall x[\mathrm{T}(x) \rightarrow \exists y, z[$  $]]$

The isomorphism between trees and logic evidently simplifies the formulation of constraints and thus makes $L^2_{K,P}$ a very attractive logic for linguists. But besides being easy to handle, $L^2_{K,P}$ has a further advantage over classical first order logic, namely its restricted expressiveness. $L^2_{K,P}$ is equivalent to $SnS_{n\leq\omega}$, the the monadic second-order theory of multiple successor functions. This logic basically talks about successors relations between integers, which can be interpreted as structural conditions on tree domains Gorn (1967). A tree domain is a subset of the set of all sequences of non-negative integers that is prefix-closed and left-sibling-closed. The first condition says that if there is a sequence s of integers that belongs to some tree domain τ, then for all subsequences v such that s is composed from v and some subsequence w of length at least 1, v is also in τ. So if the sequence 0110 belongs to τ, then so do 0, 01, and 011. Left-sibling closed requires that for all sequences s which end in $j > 0$, there are sequences s' identical to s except that their last integer is less than j. So $01 \in \tau$ implies $00 \in \tau$. Those two conditions allow us to represent tree domains as trees.

(126) 

Without going into further formal details, it should now be intuitively plausible why $SnS_{n\leq\omega}$ is equivalent to $L^2_{K,P}$. This equivalence is so important to us because the properties of $SnS_{n\leq\omega}$ have been investigated for a long time. In particular, there are many predicates which are known not to be definable in $SnS_{n\leq\omega}$, e.g. subtree-isomorphism or the equal-level predicate, that connects nodes with the same depth of embedding in a tree. From this it follows that we can't use those in $L^2_{K,P}$ either. The restrictiveness of $L^2_{K,P}$ thus gives us a very basic classification for which parts of our theory are more powerful than others. Those that cannot be defined are strictly more powerful than those that can be captured. Free indexation, for instance, has been shown by Rogers (1998) to be such an unstateable device, which is rather surprising considering how trivial it seems to be from an intuitive point of view. This is a good example for how apparently simple premises can turn out to be very complex and demanding.

Unfortunately, though, $L^2_{K,P}$ does not establish the bifurcation where we would like to see it, because some constructions we encounter in natural language are undefinable in it, in particular long-distance extraction in Swedish and cross-serial dependencies in Swiss German (cf. 4.3.1). Those constructions are said to be mildly context-sensitive. $L^2_{K,P}$, however, is equivalent to $SnS_{n\leq\omega}$ which is in turn equivalent to $SnS_{n=2}$. Definability in $SnS_{n=2}$ characterizes so-called context-free constructions, which are less complex than the mildly context-sensitive ones (we will talk about these issues in more detail later on, for now those basics suffice). Consequently, $L^2_{K,P}$ can only describe context-free constructions. But if it is only capable of defining theories that describe context-free languages, it follows that certain aspects of natural language like crossing dependencies can't be captured by a theory fully formalized in this logic. In the worst case, this means that indefinability in $L^2_{K,P}$ could lead us to dispense with theories that are perfectly acceptable accounts for natural language.

On the other side, this overrestrictiveness can at least function as an additional measuring rod for our empirical data, dividing it into context-free and mildly context-sensitive constructions. For example, English (or at least the fragment of English studied in GB) can be fully defined in $L^2_{K,P}$, which conforms with the fact that there are no known mildly context-sensitive constructions in English. Combined with the indefinability result for free indexation, this gives the very surprising result that free indexation is not needed for English. In fact, it is not needed for any context-free construction. Free indexation is only required if some language has constructions which allow an in principle unbounded number of chains to overlap, as in the Swiss German cross-serial dependencies. From this, Rogers deduces the theorem that all languages where the number of overlapping chains is bounded are context-free. In some sense, this can be seen as a typological distinction like the one between inflecting and isolating languages, but its implications bear considerably more weight for syntactic analysis. Therefore, it is anything but a trivial result.

Still, if certain, albeit few, parts of natural language are beyond our reach if we use $L^2_{K,P}$, why don't we define a new logic based on it that is restricted to mildly context-sensitive instead of context-free languages? The main reason is that pushing $L^2_{K,P}$ beyond the border of context-freeness means losing its decidability. A logic is decidable if it can be determined by an algorithm in a finite number of steps whether a given formula is valid. Propositional logic, for example, is decidable because every formula can be evaluated by constructing a truth table. $L^2_{K,P}$ is a monadic second-order logic, and those logics are the most powerful decidable logics we know, so increasing the power of $L^2_{K,P}$ will inevitably make it undecidable. While it isn't *a priori* clear that a logic for natural language needs to be decidable, it is a useful property that should not be given up too readily. After all, we have a strong interest in knowing the properties of our theory, especially its predictions. If we use an undecidable logic, we can no longer be sure that all the theories definable in it can be reasoned about adequately. Their behavior becomes unpredictable to a certain degree.

Actually, we need not push $L^2_{K,P}$ beyond context-freeness. Kolb (1999), extending results by Mönnich (1999), has shown that $L^2_{K,P}$ based theories can indeed assign correct structures to the relevant phenomena by the use of a homomorphic lifting operation. These structures, however, look fairly exotic and do not resemble the structures linguists are used to. Rogers (2003) tries another method by extending his logic to n-dimensional trees. Being mildly context-sensitive then corresponds to being definable in 3-dimensional binary branching trees that are definable in $L^2_{K,P}$. This approach furthermore can be connected to the control language hierarchy of Weir (1992), and thus we obtain another way to measure the complexity of our theories. However, it is evident that most current linguistic theories do not lend themselves easily to higher-dimensional formalization. From this perspective it looks as if $L^2_{K,P}$ can be rather cumbersome at times, but in fact these issues hardly matter for everyday research, where $L^2_{K,P}$'s inherent elegance shows its full merits.

Another disadvantage is of greater significance and is a result of the very idea of defining trees via logical constraints. This approach originates from mathematical model-theory, which concerns itself with the semantic side of logic. Non-technically speaking, a model is an instantiation of a *theory*, where *theory* is understood as a logical system with a set of axioms and all the theorems that can be derived from them. (Partee et al. 1990:200) sum it up nicely:

> Finding a model for a theory requires finding some abstract or concrete structured domain and an interpretation for all of the primitive expressions of the theory in that domain such that on that interpretation, all of the statements in the theory come out *true* for that model on that interpretation. If a theory has an axiomatic characterization, something is a model for that theory iff it is a model for the axioms.

A tree is well-formed if and only if it is a model of our theory. This perspective eschews any kind of derivations, it is strictly non-procedural. While that is no big problem for theories like GB or HPSG, derivational theories like classical transformational grammar or Minimalism do not fit easily into this scheme, although there have been attempts to recast derivational theories in representational terms. The earliest attempts date back to McCawley (1968) who proposed to view rewrite rules as admissibility conditions. So instead of interpreting S $\rightarrow$ NP VP as a process replacing the start symbol S with two symbols NP VP, we can also look at it as a constraint on trees such that a tree is well-formed if and only if every node S dominates an NP and a VP. While it makes no difference for grammars generating context-free languages which point of view we adopt, that does not carry over to context-sensitive grammars, which are higher in the hierarchy of generative capacity than mildly context-sensitive grammars. For context-sensitive grammars it has been shown that their expressive power varies with the interpretation one assigns to their rules (Peters and Ritchie 1969). While a derivational perspective retains their generative capacity, they are restricted to the power of context-free grammars under a representational interpretation as node admissibility conditions. We should take this as a warning that one must not implement a derivational theory by purely representational means and expect it to behave exactly the same. The equivalence has to be proven first. History also tells us that rethinking of derivational theories in model-theoretic terms is anything but a trivial task. Lakoff (1971) already proposed to think of a derivation representationally as a tuple of trees connected by transformations, but his formalization had some problems that caused it not to behave in line with what he wanted it to do (Soames 1974).

The current attempts to look at Minimalism from a representational perspective look promising (see Cornell 1997; Potts 2001, 2002), but it has yet to be seen whether they will be able to evade all pitfalls. Potts proposes defining a derivation as a partially ordered set of representations with operations like Move and Merge acting as relations between the elements of this set, picking up Lakoff's ideas. He also provides a logical implementation of transderivational constraints that could turn out to be very useful for further examinations of this ill-understood yet incredibly powerful tool.

4.2.4 Summary

We have seen that logic can be easily put to good use as a metalanguage in linguistics. The requirements are rather low, and there are huge rewards to reap. Stabler (1992:322) wraps this up concisely.

> Of course, the formalization does not add anything to the theories. Anyone who understood them well enough might be able to draw the same conclusions without the aid of the formalism, but the theories are rather complex. The formalism is a valuable aid to specify unambiguously the theory about which claims are being made, and the exercise of discovering a proof is a valuable check on whether we have forgotten any cases that would be exceptions of our claims. Two kinds of claims about a formalized theory are of interest. First, we want to claim that the formalism is faithful to linguists' intentions. The formal definitions are not always trivial, as we saw [...] The second kind of claim that we make about our formalization is much clearer: claims about consequences of the formal theory. When we claim that some formal proposition follows from $FB \cup SEQ_{FB}$, this is a claim with a definite truth value, a claim that can be checked with standard proof methods.

If we choose our logic carefully, the benefits will be even bigger, as illustrated by $L^2_{K,P}$, which is a very flexible logic for describing trees and at the same time it is constrained enough to tell us something about the power of our formalism. Therefore it both directs our attention towards logical gaps in our theories and has the capability to highlight aspects of our framework that have not surfaced prominently in research yet.

We also saw that logic isn't completely ready yet for the treatment of derivational theories, although efforts are currently undertaken to restate such theories from a model-theoretic perspective. It was also explained that certain advantages of logic are difficult to harvest considering the current state of linguistics, the lack of familiarity with computational science being the main cause for that. This unfortunate verdict will be even more accurate of the methods presented in the next section, which deals with formal language theory, which is better suited for evaluating derivational theories. I don't expect those methods to ever become mainstream in linguistics — whereas I definitely see this potential for logic — so the purpose of the next section mainly is to make linguists get to know the basics of this field and provide them with good reasons to keep an eye on its ongoing research. Let me reiterate the distinction between the current and the next section: the former presents tools that are very powerful yet easy to use, the latter is focused on research that is of immediate relevance to linguistics but too difficult for the average linguist to do it himself.[9]

4.3 Minimalist Grammars

4.3.1 Formal Languages

In order to be able to appreciate what is to follow, we first need to establish certain terms and concepts frequently encountered in formal language theory. Most of them should be at least remotely familiar, as they date back to the heydays of classic transformational grammar, when a lot of research was done on the inherent complexity of specific natural language phenomena and how powerful a grammar formalism ought to be in order to describe them. Research of this kind, i.e. on weak generative capacity, does not enjoy the best of reputations in today's linguistics, but before rashly dismissing the issue, give me the benefit of the doubt until 4.3.3, where I will discuss the relevance of those topics for linguistic research. Until then, let us put the methodological issues aside and restrict ourselves to the formal aspects.

In the early days of modern linguistics, i.e. shortly after Chomsky (1957), a lot of research was done on the properties of string rewriting systems, which were inspired by deductive systems of reasoning. There are two equivalent definitions for strings. The more intuitive one defines them as a finite sequence of occurrences of elements of a set A, usually called the alphabet. The more formal oriented one defines a string of length n over the alphabet A as a mapping from the first elements in $\mathbb{N}^+$ into A, where the integers indicate the position of the respective element in A. The string *ababc* is then equivalent to the relation $R := \{\langle 1, a\rangle, \langle 2, b\rangle, \langle 3, a\rangle, \langle 4, b\rangle, \langle 5, c\rangle\}$. Starting from a single axiom, the initial symbol S, strings are deduced by applying a finite number of rules of inference, which are more aptly called rewriting rules in this context. The rule $a \rightarrow ab$ is then to be read as 'replace a by ab. Usually a distinction is drawn between terminal symbols on the one hand

[9]Of course this is true just in case we're talking about simply using logic, not proving its decidability or related issues that should be left to logicians.

(denoted a, $b, \ldots$) and non-terminal symbols (denoted A, $B, \ldots$) on the other. Only the former are allowed to show up in the final string. It is also commonly assumed that every rewrite rules contains at least one non-terminal symbol on its left side.[10]

There is of course a more formal way to describe the general properties of a rewriting system. We define a rewriting system as a quadruple $\langle V_T, V_N, S, R\rangle$, where V_T is the terminal and V_N the non-terminal alphabet such that $V_T \cap V_N = \emptyset$. $S \in V_N$ is the initial symbol and R is a set of ordered pairs in $\Sigma^* V_N \Sigma^* \times \Sigma^*$, where $\Sigma = V_N \cup V_T$ and Σ^* denotes the set of all strings of symbols in Σ of non-negative length. A derivation is a sequence of strings $s_1, s_2, \ldots, s_n$, $n \in \mathbb{N}^+$, such that $s_1 = S$ and every s_i, $2 \leq i \leq n$, is derived from s_{i-1} by a single application of some rule in R. Finally, a grammar G is said to generate a string $s \in V_T^*$ if G derives s in a finite amount of steps and no further rule applications take place. We call the set of all strings generated by G the language generated by G, denoted $L(G)$.

Depending on which additional constraints we impose on G's rewriting rules, G belongs to a special class of grammars on the so-called Chomsky Hierarchy (Chomsky 1963).

(127) *The Chomsky Hierarchy*
Let $A, B \in V_N$, $x \in V_T$, $r \in R$, and α, β, ω arbitrary strings over $V_T \cup V_N$.
a. Type 0: every r is of the form $\alpha A \beta \to \omega$.
b. Type 1: every r is of the form $\alpha A \beta \to \alpha \omega \beta$, where $\omega \neq \varepsilon$, the empty string.[11]
c. Type 2: every r is of the form $A \to \omega$.
d. Type 3: every r is of the form $A \to xB$ or $A \to x$.

Those classes are often referred to as unrestricted, context-sensitive, context-free and regular, respectively. Type 1 grammars are properly included by Type 0 grammars, and Type 3 grammars are properly included by Type 2 grammars. Consequently, every regular grammar is also a context-free grammar, and every context-sensitive grammar belongs to the class of unrestricted grammars. Note though that the inclusion of Type 2 in Type 1 grammars is not proper, for Type 2 grammars allow ω to contain ε, while Type 1 grammars explicitly ban this option.

The most interesting question for a linguist obviously is where natural language is situated on this hierarchy. For a long time it was believed that natural language was context-free, but this was refuted by Shieber (1985) based on cross-serial dependency structures in Swiss German like the one in (128).

(128)

Jan	säit	das	mer	d'chind	em Hans	es	huus	haend	wele	
John	said	that	we	the children.ACC	Hans.DAT	the	house.ACC	have	wanted	
laa	hälfe	aastriiche								
let	help	paint								

[10] This requirement is sometimes suspended for Type-0 grammars. In (127), I will nevertheless stick with it.

[11] The so-called empty string is a terminal symbol that can be added to a string without altering it. Adding it to *abc*, for instance, still gives *abc*. Thus a string may in principle contain an arbitrary, though finite number of occurrences of ε. From an algebraic perspective, ε is the identity element under string concatenation such that $a^\frown \varepsilon = \varepsilon^\frown a = a$.

'... we have wanted to let the children help Hans paint the house.'

The total number of NPs and VPs is unrestricted, due to the recursiveness of natural language. But as the reader can see for himself, the first NP depends on the first VP, the second NP on the second VP, and so forth. That is to say, the number of NPs and VPs has to be identical, and they have to match in case.

Shieber then defines a regular language $L_1^R = wa^*b^*xc^*d^*y$, where z^* denotes that the number of occurrences of z can be any non-negative integer. An isomorphic variant of L_1^R is L_2^R where $w =$ 'Jan säit das mer', $a =$ 'd'chind', $b =$ 'em Hans', $x =$ 'es huus haend wele', $c =$ 'laa', $d =$ 'hälfe', and $y =$ 'aastriiche'. Crucially, there are no dependencies between NPs and VPs in L_2^R, the number of occurrences of a does not need to be equal to the numbers of c. In Swiss German, however, this is the case, so intersecting L_2^R with Swiss German gives us sentences of the form in (129).

(129) $L^{R\cap S} := wa^m b^n xc^m d^n y \equiv$ Jan säit das mer (d'chind)^m (em Hans)^n es huus haend wele (laa)^m (hälfe)^n aastriiche

It can be shown that $L^{r\cap S}$ is neither a regular nor a context-free language. It is a well-known fact that if two language L_i and L_j are regular, then their intersection is regular too. If one of those languages is context-free, their intersection is also context-free. Based on this we conclude that if $L^{r\cap S}$ cannot be generated by a context-free grammar, then either L_2^R or Swiss German cannot be generated by a context-free grammar. We can be sure that L_2^R is a regular language. It follows that Swiss German is not context-free.

Shieber was the first to give an irrefutable proof that, *contra* the conjecture of Gazdar et al. (1985), we need to go beyond context-freeness if we want to describe all constructions of natural language syntax. Previous proofs committed two fallacies (see Pullum and Gazdar 1982 for an overview of such flawed proofs). Often they based their arguments on empirically dubious data that involved semantic and pragmatic aspects and thus could be skewed. More frequently it was mistakenly assumed that it suffices to show that some subset of a language is not context-free to prove that the entire language is not context-free. This is trivially false. Consider the language $a^n b^* c^n$, $n \geq 1$, which is generated by the following grammar.

(130)
a. S $\rightarrow$ aSc
b. S $\rightarrow$ aTc
c. T $\rightarrow$ bT
d. T $\rightarrow$ ε

Evidently this grammar can by sheer coincidence generate the subset $a^n b^n c^n$, but this does not imply that it is not context-free. Shieber, however, didn't just pick some arbitrary subset of Swiss German, he generated the relevant subset by intersecting Swiss German with a regular language. And as context-freeness is preserved under intersection, the subset should be context-free if Swiss German were context-free. The rest of the argument is already familiar from above.

Shieber (1985) thus instantiated that the generative capacity of natural language syntax goes beyond context-freeness, but we do not know its upper bound yet. Context-sensitive grammars are too powerful, for they can generate languages like a^n, n a prime number. Arguably we do not encounter such structures in natural language. Joshi (1985) proposes that all natural language phenomena can be accounted for by so-called mildly context-sensitive grammars (MCSGs), i.e. that natural language belongs to the class of mildly context-sensitive languages (MCSLs). MCSGs have the following properties. First, context-free languages are properly contained in the MCSLs (while they are not properly contained in the context-sensitive languages). Second, mildly context-sensitive languages can be parsed in polynomial time. Third, MCSGs capture only a very restricted range of dependencies, primarily nested dependencies and certain kinds of crossing dependencies. Fourth, strings of mildly context-sensitive languages only grow linearly in size, i.e. there is no mildly context-sensitive language like $L := \left\{a^{2^n} : n \in \mathbb{N}\right\}$. We immediately see that the strings of L grow exponentially in size, rather than linearly. For $n = 0$ the string length is 1, for $n = 1$ it is 2, for $n = 2$ it is 4, for $n = 3$ it is 8, and so forth. In contrast to L, $L' := \{a^n : n \in \mathbb{N}\}$ and $L'' := \{a^n(bb)^n ccc : n \in \mathbb{N}\}$ have the linear growth property.

Many grammar formalisms have been shown to belong to the class of MCSGs (see Weir 1988; Joshi et al. 1991; Vijay-Shanker and Weir 1994), among them indexed grammar (Aho 1968; Gazdar 1985), tree adjoining grammars (Joshi et al. 1975; Joshi 1985; Joshi and Schabes 1997), head grammars (Pollard 1985; Roach 1987), and combinatory categorial grammar (Steedman 1996). In recent years, another grammar formalism could be proven to be a member of this class by Michaelis (1998, 2001) and Harkema (2001a), Minimalist grammars (Stabler 1997; Stabler and Keenan 2003). Those will be our next topic.

4.3.2 Minimalist Grammars

Minimalist grammars (MG) have first been defined in Stabler (1997), and since then they have sprouted considerable research in the properties of formal tools frequently used by Minimalist syntacticians. Owing to the mathematical prerequisites of most articles, I will restrict myself to a short demonstration of one variant of MGs, what kind of results have already been obtained with them and why they grant us an improved understanding of our technical machinery. The interested reader is referred to Gärtner and Michaelis (2007) for an in depth presentation of those topics.[12] I won't touch on the relation between MGs and categorial grammars (Lecomte 1998, 2003), nor on recent work on parsing MGs (Harkema 2001b) or attempts to enrich them with a semantic interface (Amblard et al. 2003; Kobele 2006).

I start with the classic definition of MGs given in Stabler (1997), but be aware that I will choose to replace some of the symbols so that they are in line with my own notation. Let $V := \mathcal{F}_{PF} \cup \mathcal{F}_{LF}$ be the set of non-syntactic features and $Cat := base \cup select \cup licensors \cup licensees$ be the set of syntactic features, where the respective sets are defined as follows:

[12] Frankly, I can't recommend this article highly enough, for it combines lucid, non-technical writing with a comprehensive list of definitions of several variants of MGs.

(131) a. *base* $:= \{c, t, d, v, n, p, \ldots\}$ is the set of categorial features.

b. *select* := {=x, =X, X= : x ∈ *base*}, where =x indicates the selection of an x-phrase, and =X and X= indicate selection of an x-phrase *plus* suffixation (=X) or prefixation (X=) of the phonetic features of the x-phrase to the selector.

c. *licensees* $:= \{-f : f \in \mathcal{F}_{FF} - base\}$ is the set of attractable features.

d. *licensors* $:= \{+f, +F : f \in \mathcal{F}_{FF} - base\}$ is the set of attracting features, where $+F$ denotes a feature inducing overt movement.

Further, we define an *expression* as a finite, binary, labeled, ordered tree $\tau := \langle N_\tau, \triangleleft^*_\tau, \prec^*_\tau, <^*_\tau, Label_\tau \rangle$, where N_τ is the set of nodes of τ, and $\triangleleft^*_\tau$, $\prec^*_\tau$, $<^*_\tau$ are the respective reflexive transitive closures of the immediate-dominance relation $\triangleleft_\tau$, the immediate-precedence relation $\prec_\tau$ and the immediately-project-over relation $<_\tau$ (henceforth, I will omit τ in my notation).

We define the following notions on τ:

(132) a. *Root*

The set of roots of a (sub)tree τ is a singleton set $R_\tau := \{x : \neg\exists y[y \triangleleft x]\}$.

b. *Leaves*

Let $L_\tau := \{x : \neg\exists y[x \triangleleft y]\}$ be the set of leaves of a tree τ.

c. *Head*

For any $x, y \in N_\tau$, x is a head of y if either $x = y \in L_\tau$ or $\exists z[y \triangleleft z \wedge \forall w[y \triangleleft w \rightarrow z <^* w] \wedge x$ is a head of $z]$.

d. *Maximal projection*

Some node y is a maximal projection of a node x iff $y \in A := \{z \in N :$ x is the head of $z\}$ such that there is no $w \in A$ that properly dominates y.

e. *Specifier*

Some node x is a specifier of a head y iff

i. x is a maximal projection, and

ii. $z \triangleleft x \rightarrow y$ is the head of z, and

iii. x properly precedes y.

f. *Complement*

Some node x is a complement of a head y iff

i. x is a maximal projection, and

ii. $z \triangleleft x \rightarrow y$ is the head of z, and

iii. y properly precedes x.

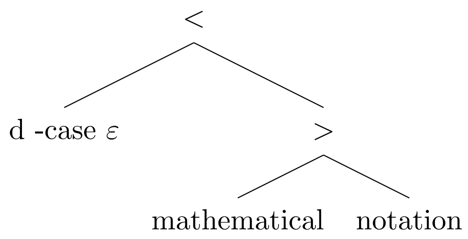
We further maintain that for every mother node m, there is exactly one daughter node d that immediately projects over all other daughters of m. Finally, let *Label* be a map from leaves into a regular set $select^*(licensors)select^*(base)licensees^*\mathcal{F}^*_{PF}\mathcal{F}^*_{LF}$. Then a minimalist grammar is a 4-tuple $\langle V, Cat, Lex, \mathcal{S}\rangle$, where V and Cat are defined as above, *Lex* is a finite set of expressions built from V and Cat as indicated above, and $\mathcal{S} := \{merge, move\}$ is the set of our syntactic operations.

I think enough formal grounds have already been covered, so I will turn to some examples and leave aside the definitions of *merge* and *move* — curious readers can look it up in the appendix of Stabler (1997). Consider a very simple sentence like *Every linguist loves mathematical notation.* To generate this sentence, we need the LIs listed in (133). Each of those entries is actually a tree with a single node adhering to the definition of τ above. This affects in particular the ordering of features, which is due to the definition of *Label*.

(133)
a. =n d -case ε
b. =v +CASE t ε
c. =t c ε
d. =n d -case every
e. n linguist
f. =d +case =d v loves
g. a mathematical
h. =a n notation

The assembly of the utterance then proceeds in a very familiar way, with the minor deviation that the peculiar order of features determines the course of the derivation. First, *mathematical* and *notation* are drawn from the lexicon and merged. The categorial feature of the adjective is deleted, and so is the selection-feature =a of *notation.* Now the first feature of the noun is n, wherefore it can be merged with the covert determiner listed under (133a). We now have assembled our first DP, depicted below. Note that we do not use phrase structure labels but rather indicate the projecting node by $<$ and $>$, depending on whether it precedes or follows its complement, respectively.

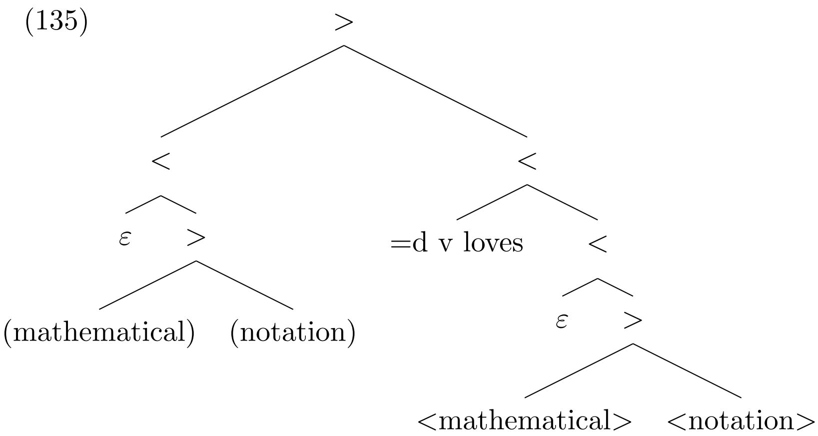
(134)



The first feature of the determiner now is its categorial feature, so it can merge with the verb, again resulting in deletion of the relevant features.[13] As a consequence, the determiner has only one feature left, -case, which allows it to move to a position where the feature can be checked. The verb's first feature is +case, so the DP can move to get the feature deleted. However, as the verb only has a weak feature, not a strong +CASE feature licensing overt movement, the operation has to apply covertly, yielding (135).

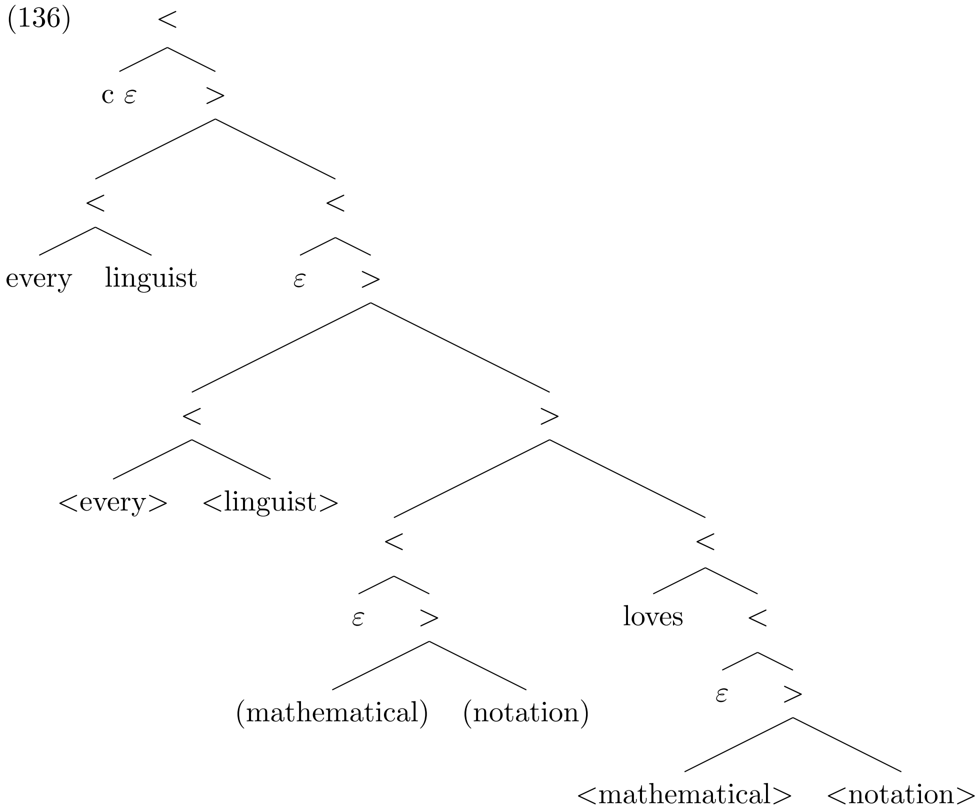
[13] If the verb had =D or D= instead of =d, the determiner would have also undergone head-movement and been adjoined as a suffix or prefix.

(135)



Now the subject DP *every linguist* can be assembled and merged with the verb in an analogous fashion. The head of the DP, *every*, still has a -case feature to discharge, while the verb needs to get its category feature deleted, which is accomplished by merging it with the T-node. This introduces a strong +CASE feature into the derivation and causes overt dislocation of the subject DP. After this step, the C-node is selected and merged to check the remaining categorial feature of T. As c is a licit root node and no more lexical items are available for further computation, the derivation converges.

(136)



Some readers are presumably scratching their heads right now, wondering how such a coarse implementation of Chomsky (1995c) could be useful for linguists. All intricacies of checking are omitted by the exclusive use of Erasure as a part of Merge, features have to be linearly ordered, there is no reference-set computation, no θ-roles, no binding, no control, no labels in the traditional sense, no adjunction, no scrambling. Even ϕ-features are completely left out of the picture.

The simple fact is that MGs are not meant to emulate a full-blown syntactic theory. Rather they provide us with an ideal testing ground for the inspection of the properties of any kind of syntactic module. A simple MG is a barebone implementation of Minimalism, and because of its simple nature its behavior is well understood. If we add some module M to our basic MG, denoted $\mathfrak{MG}$, and carefully examine the properties of the new $\mathfrak{MG}^M$, we can effectively determine the impact of M.Obviously we can also test how M behaves in interaction with other modules, and from all those experiments we gain an understanding of the internal workings of our theory we would not reach otherwise. The elicitation of telling results, however, is a very demanding task that requires a lot of knowledge in automata theory and familiarity with formal proof methods. Consequently the average linguist won't be able to accomplish this, and he or she probably also won't care how any of those results came about in detail. But the results themselves definitely are worth the attention.

Concerning operations, the main focus has been on head movement (Stabler 2001, 2003) and remnant movement (Stabler 1999), the copy theory of movement (Kobele 2006), affix hopping (Stabler 2001), and adjunction and scrambling (Frey and Gärtner 2002; Gärtner and Michealis 2003; Gärtner and Michaelis 2005, 2007). The work by Gärtner and Michaelis is a perfect example for the kind of results linguists can expect from MGs. They set out to determine whether adding late adjunction and scrambling to an MG increases the expressiveness of said grammar beyond mild context-sensitivity. The question arises because late adjunction can evade violations of the shortest move constraint (SMC).[14] They establish that generative capacity is preserved iff and only if extraction out of adjuncts is blocked. This condition is known as the Adjunct Island Constraint (AIC). These results, however, obtain just in case the SMC holds too, otherwise the AIC shows no restricting effects and the power of the grammar increases. The results for an MG with late adjunction and scrambling are summarized in (137), where we apply the parameter notation already familiar from the previous chapter, and $a < b$ indicates that a is less powerful than b:

(137) *Behavior of MGs with late adjunction and scrambling*
$\mathfrak{MG}^{+SMC}_{+AIC} < \mathfrak{MG}^{+SMC}_{-AIC} < \mathfrak{MG}^{-SMC}$

Comparable results have also been presented in Kobele and Michaelis (2005) for MGs without adjunction but with the Specifier Island Constraint (SPIC), which blocks proper extraction from inside a specifier. While $\mathfrak{MG}^{+SMC}_{+SPIC}$ is slightly weaker than $\mathfrak{MG}^{+SMC}_{-SPIC}$, as is to be expected, $\mathfrak{MG}^{-SMC}_{+SPIC}$ can emulate so-called 2-counter automata whose power is equivalent to that of a type 0 grammar. As we already saw in the previous section on formal languages, type 0 grammars are higher up the Chomsky hierarchy than mildly context-sensitive grammars, which are a superset of the class of grammars $\mathfrak{MG}^{+SMC}_{+SPIC}$ belongs to. Suspension of the SMC thus causes a sudden increase in generative capacity, contrary to the linguist's intuition that the constricting effect of the SPIC should remain the same independent of other constraints. This is a very profound result of immediate relevance to any syntactician.

[14]The shortest move constraint in MGs has a very special implementation, as it enforces that for every x, τ contains at most one feature -x at stage n of a derivation.

Another variant of MGs equivalent to a type 0 grammar is $\mathfrak{MG}^{+perc}$, an MG with feature percolation. Feature percolation is the process whereby the features of a specifier percolate into the node dominating it, and it is often used to account for pied-piping phenomena. In the question *Whose books did you read*, for instace, the wh-word is the specifier of the noun, yet the whole DP undergoes wh-movement. Forcing the wh-feature of *whose* to percolate into the DP accounts for this behavior straightforwardly. Kobele (2005) shows that a grammar comprising feature percolation can compute any task done by an infinite abacus, another type of automata, if LIs may contain one and the same feature more than once. Again we see how a small change in the overall design has far-reaching consequences for the power of a grammar. But does this have any consequences besides showing us how submodules interact? In other words, does generative capacity matter to linguistics?

4.3.3 A Note on Generative Capacity

Before we can address the question of the relevance of generative capacity, the distinction between weak generative capacity (WGC) and strong generative capacity (SGC) (Chomsky 1963) needs to be introduced. The WGC of some grammar G_i is the language it generates, i.e. a set of strings. Consequently, the WGC of some class of grammars is the union of all languages generated by those grammars. WGC thus is based on strings, i.e. the output of grammars. SGC, on the other hand, includes the structure assigned to those strings. The SGC of some grammar G_i is the set of structural descriptions assigned by the grammar to the strings it produces.

Naturally, SGC is of greater importance to linguistics than WGC, but unfortunately it has been neglected for a long time because its original definition rendered it inutile. According to the original formulation, two context-free grammars, for instance, are strongly equivalent if and only if they are identical. Distinct theories can't be compared either, because their structural primitives usually differ significantly. Things have improved in the last decade. Miller (1999) redefined SGC as a model-theoretic semantics for linguistic theories. Metaphorically speaking, he set up a theory-neutral space into which any grammar can be mapped. The final results of this mapping can then be compared despite the conceptual differences between the original theories as those primitives have all been mapped to the same entities in Miller's theory-neutral space. Another approach is based on the work of Rogers (1998), which we already discussed in 4.2.3 (so there is yet another reason for linguists to embrace logic). Those efforts show great potential, but it will definitely take some time until attention has completely shifted from WGC to SGC.

The relevance of WGC has often been issue to debate, and increasingly often linguists have concluded that they cannot benefit from research on WGC. Chomsky (1986) even maintains that practically no notion imported directly from the computational sciences can be fruitfully applied to natural language. His conjecture must not be confused with a refusal of mathematical methods *per se*, Chomsky merely states that these concepts fail to capture the characteristics a linguist cares about. Consider the case of complexity theory, which is a field of computational science that is concerned with the investigation of the computational load posed by a problem. Crucially though, it does not analyze the complexity of some problem instance, but of the problem in general, and in particular it is interested in asymptotic worst-case complexity, that is, the hardest case that might arise given infinite space and time. The complexity results for chess, for example, apply to playing chess on an infinite chess board, and it has been shown that there is no efficient algorithm to compute this problem. Of course this does not imply that it is impossible to build a good chess computer, as has already been proven.

Given its premises, complexity theory obviously does not line up perfectly with cognitivist concerns. Human cognition operates in finite space and time, and worst-case complexity isn't very important either. After all, some rare linguistic constructions might be computationally demanding or cause the parser to crash, but linguists are interested in the processing cost of common sentences, not a few occasional exceptions. Nevertheless I disagree with Chomsky. Linguistics can profit from such research, no matter whether it is formal language theory or complexity theory or some other field of the computational sciences we are talking about. But the results have to be interpreted very carefully, because the relation between the mathematical concept and its concrete instantiation is subtle and indirect.

With respect to WGC, I allege that it is useful in two respects. If a certain theory can be shown to be as powerful as a type 0 grammar, we have good reason to believe that it needs to be revised. First, it massively overgenerates. If some grammar is of type 1, one could still argue that this was due to the slight inadequacy of the methods applied. In the case of type 0, however, the leap in power is just too big to be an artifact of our proof technique. This is also confirmed by the fact that none of today's syntactic theories, no matter how advanced, is a type 0 grammar.[15] Second, such an expressive grammar is suspect for epistemological reasons, too. Type 0 grammars are equivalent to Turing machines, which are the current model for algorithmic computability. That is to say, Turing machines compute anything that can be computed in an algorithmic way, so if a grammar is equivalent to a Turing machine, it can more or less compute anything. But if a theory can explain anything, what does it actually explain?

The other practical aspect of WGC is its use as a measuring rod, as we already saw in the previous section, where it allowed us to trace the impact of certain modules and how they interact with each other. As long as there is at least an indirect relation between weak and strong generative capacity, this application alone justifies our interest in WGC.

4.4 So What are the Ramifications?

In this chapter, I set out to investigate the origins of vagueness in linguistics, and whether anything could be done about it. I could highlight only a few points that might be involved

[15]Occasionally the opinion is voiced in the literature that a grammar has to be mildly context-sensitive if it is to be explanatorily adequate. This normative stance, however, ignores the fact that an overgenerating syntax could be constrained by other factors like filtering at the interfaces or certain traits of the linguistic parser.

in the first issue, due to its complexity. I guess it would take a group of sociologists to determine the dynamics of the linguistic community that give rise to the unfortunate state of affairs. I proposed that things could be improved in various respects by adopting logic as a metalanguage. Logical notation guides our expression towards holes and contradictions in arguments and definitions, while there is a greater chance that those things will be missed if phrased in natural language, especially if they are directly woven into the text. Logic also allows us to get a better understanding of the innards of our theory, as we saw in 4.2.2 and 4.2.3. To this end, linguists should also keep an eye on the developments in formal grammar theory, especially Stabler's Minimalist Grammars.

Advantageous as logic and formal grammar might be, they can't solve the underlying problem in and of themselves. I emphasized this point several times throughout the chapter, because it is so easy to confuse these issues: being precise does not necessitate being formal, and being formal does not imply being precise. One can write a perfectly sound and lucid article in plain English, and likewise one can produce a mathematically flawless paper that still fails to define its basic terms and resists any concrete interpretation. Nonetheless, as soon as one is really devoted to the creation of an unambiguous and well-defined theory, those formal tools can be of great help, and if adopted by a considerable number of linguists, their benefits will be even greater in the long run. Of course it will take a collective effort to push current Minimalism in this direction, but I am confident that this move will happen sooner or later. After all, the number of linguists who are aware of this problem and consider it an obstacle to the progress of linguistics seems to be growing. I consider my thesis an attempt to cause a further increase in this number.

Conclusion

Throughout the entire thesis, I constantly tried to connect two separate realms, a technical and a methodological one. The former one enjoyed the greater share of my attention, but it is the latter one that supplied the motivation for the whole enterprise, namely demonstrating that vagueness is a more pressing problem of contemporary mainstream linguistics than commonly perceived.

Nevertheless the purely formal part of my work is interesting in and of itself: I pondered the status of features, feature bundles and lexical structure in Minimalism, and the results thereby obtained could be fruitfully applied to feature strength and checking theory. The checking operations Delete(α) and Erasure were implemented in five different ways, each of them significantly more explicit than Chomsky's own proposal. Finally, careful scrutiny of Reuland (2001) revealed that in total there are 180 variants of his theory, only 2 of which can be considered acceptable, although they still need a fair number of stipulations to work reliably.

While I was not concerned about the validity of Reuland's theory in particular — it just served as an illustrative example of the general problem — its doubtful performance is a worrying result, for two specific reasons. First, it shows that alterations of minor details can quickly make any given tool break, without anybody noticing. Thus the tendency to adopt premises from other lines of research has a very high potential to negatively affect the soundness and consistency of a theory. The second motive for critique stems from the fact that I had to employ rather unorthodox and cumbersome methods in order to arrive at valid results and exclude any artifacts from skewing the analysis. While comparable investigations of a theory's properties are straightforward for other frameworks, I was forced to carefully construct coherent incarnations of Chomsky (1995c) to which Reuland's proposal could be applied.

On the other hand, the investigation of Reuland (2001) also showed that most versions of Chomsky's feature checking mechanism behave in the same way, which begs the question whether they are just notational variants of each other. If so, this would support the objection that the level of accuracy I demand of syntactic theory is nothing more than a pointless technical exercise. This critique, however, misses the crucial point that we can't determine *a priori* whether the equivalence of the checking systems will be preserved in any case. A different modification might cause them to cluster in a completely unexpected new way. Furthermore, if such low-level details of the machinery had indeed no impact on the theory, how come that it is commonly argued that they can't be defined because of the paucity of empirical evidence and thus should be left to future research? This seems slightly contradictory to me.

In the same way, it is sometimes proposed that we should not care too much about formal aspects since linguistics is still a very young and immature field, wherefore our tools are too coarse to allow for anything but broad generalizations. This argument is flawed in various respects. First of all, it ignores the fact that many linguists actually trust their theory and are not afraid of making very specific technical assumptions which resist theory-neutral generalization. Secondly, I have to wonder why research done in other frameworks is mostly neglected by Minimalists although it is generalizations about language we are allegedly interested in. In addition, it is highly dangerous to rely on generalizations which arise from an ill-understood theory, for they might just be the result of a technical quirk. Consider Principle B, which also fulfilled the duty of regulating the distribution of PRO in GB. If it turns out that PRO does not exist, our knowledge about Principle B has to be reevaluated. Now suppose that other aspects of Principle B dependend on various factors, too, but we never realized that. If those hidden parameters did not carry over to our new theory, sticking to our generalizations about Principle B would become an obstacle to scientific progress. Finally, to link the level of due technical sophistication to our empirical coverage is to artificially restrict their mutual fertilization, and it also poses the unanswerable question how many empirical generalizations we have to discover before it makes sense to do more formally inclined research.

I think most of these objections stem from the misconception that I was leading a crusade against vagueness in general. To the contrary, I tried to look at the issue in a differentiated way, and I was only concerned with the narrowly restricted notion of vagueness of theoretical primitives and definitions. At no point did I touch on vagueness in empirical analysis, which is a given. The very application of our technical notions to linguistic material already includes a step of abstraction that invariably induces a certain degree of imprecision. Trivially, natural language itself contains its fair share of fuzziness. And as in the other empirical sciences, we frequently reach the limit of our understanding and simply have to guess what might be responsible for the phenomenon observed. Regarding formal obscurity, I also conceded that its negative impact varies from article to article, and that it often affects only peripheral aspects or can be solved rather easily. Still, even when those cases are disregarded during the dissection of vagueness, it is a worryingly prevalent trait.

Deutsche Zusammenfassung

In der vorliegenden Arbeit beschäftige ich mich mit einem Vorwurf, mit dem sich der Prinzipien-und-Parameter Ansatz (P&P) häufig konfrontiert sieht, nämlich dass ein Gutteil seiner Terminologie nicht oder bloß unzureichend definiert sei. Ich bestätige dieses Verdikt anhand eines Fallbeispiels, der Modifikation des klassischen minimalistischen Feature-Checking-Systems (Chomsky 1995b) in Reuland (2001).

Ich zeige, dass bereits Chomskys Definitionen fünf unterschiedliche Implementierungen von Feature Checking erlauben. Durch Reulands Eingriffe steigt diese Anzahl auf 180, doch nur zwei der vorgeschlagenen Systeme sind akzeptabel, dabei gleichzeitig aber höchst stipulativ. Die scheinbare minimalistische Eleganz von Reulands Ideen scheitert somit angesichts unüberwindbarer technischer Hürden. Die Evaluierung meiner Resultate zeigt auf, dass anderen minimalistischen Ideen ähnliches widerfahren kann, weswegen ich eine vermehrte Hinwendung zu formaler Logik als Hilfsmittel zur Beschreibung linguistischer Theorien vorschlage. Zwar lässt sich Vagheit dadurch nicht ausschließen, doch sind unvollständige Definitionen in dieser formalen Form leichter als solche zu erkennen, für Leser wie für Autor. Darüber hinaus eröffnen sich durch den konsequenten Gebrauch von Logiken neue Arbeitsweisen, beispielsweise der Gebrauch von automatischen Beweisern. Auch wird das Verständnis der Mächtigkeit der linguistischen Formalismen gefördert. Aus dem selben Grund sollte generell der rezenten Forschung aus dem Bereich der mathematischen Linguistik mehr Aufmerksamkeit zuteil werden, insbesondere den in Stabler (1997) definierten Minimalist Grammars.

Aufgrund ihres breit angelegten Ziels, dem Aufzeigen bestehender methodologisch-formaler Schwächen in P&P, ist diese Arbeit für Linguisten aus allen Bereichen interessant, die sich für die Validität minimalistischer Theorien interessieren. Gleichzeitig finden sich technisch exakte Besprechungen der Struktur lexikalischer Einträge, Stärke und Interpretierbarkeit von Features, Delete(α), Erasure, Ketten und globaler Ökonomie, sodass die Arbeit auch für jene von Wert ist, die sich nicht für das Vagheitsproblem interessieren oder überhaupt keines erkennen können.

Symbols and Abbreviations

This section is divided in two tables. The first one lists common mathematical symbols, whereas the second one captures notation specific to the subjects treated in this thesis. Regarding the ordering, symbols precede letters, and letters are ordered alphabetically and furthermore typographically as follows: $\mathrm{f} \prec f \prec \mathrm{F} \prec F \prec \mathcal{F} \prec \mathbb{F} \prec \mathfrak{F}$.

Mathematical symbols

$\emptyset$	The empty set
$\in$	Set membership relation
$\notin$	Complement relation of $\in$
$\cap$	Set intersection
$\cup$	Set union
$\subset$	Proper subset relation
$\subseteq$	Subset relation
$-$	As in $B - A$; denotes the relative complement of A in B
$\times$	As in $A \times B$; Cartesian product
$\langle x_1, \ldots, x_n \rangle$	An n-Tuple of x_i, $i \in [1, n]$
$\neg$	Negation
$\wedge$	Logical conjunction
$\vee$	Logical disjunction
$\rightarrow$	Logical implication
$\leftrightarrow$	Logical equivalence
$\exists$	Existential quantifier
$\forall$	Universal quantifier
$\mathbb{N}$	The set of non-negative integers
$\mathbb{N}^+$	The set of positive integers

Linguistic symbols and abbreviations

$\approx$	Equivalence

$\lhd$	Immediately dominates
$\lhd^*$	The reflexive transitive closure of $\lhd$
$\prec$	Immediately precedes
$\prec^*$	The reflexive transitive closure of $\prec$
$<$	Immediately-projects-over
$<^*$	The reflexive transitive closure of $<$
$\langle$XP$\rangle$	A copy of XP
$\int$(K)	The sublabel of K, see (54)
ϵ	The transitive closure of the set-membership relation, , page 25
ε	The empty string
Φ[X]	The set of all ϕ-features of XP (possibly a superset of the ϕ-feature bundle of X), see (115)
Σ	A derivational stage, cf. fn.5 on page 5 and (4) *or* the alphabet of some formal grammar ($\Sigma := V_T \cup V_N$)
Σ^*	The set of all strings over Σ
$\langle\mathfrak{R},\mathfrak{F}\rangle$	A combination of some Reulandian syntactic framework and some feature checking system
AIC	Adjunct Island Constraint
BOC	Bare Output Conditions
BPS	Bare Phrase Structure
CFD	Condition of Feature Distinctiveness, see (96)
$\mathrm{C_{HL}}$	Language faculty (abbreviation for “computational system of human language”)
CH	A chain, induced by Move, page 53
CH	A Chain, induced by feature checking, page 53
$\mathcal{CH}$	A CHAIN, established by connecting a chain with a Chain, page 53
$\mathbb{C}(\alpha)$	The checking domain of α, see (50)
$\mathrm{D}(\alpha)$	The domain of α, see (45)
$\mathrm{D}_C(\alpha)$	The complement domain of α, see (46)
$\mathrm{D}_I(\alpha)$	The internal domain of α, see (49)
$\mathrm{D}_M(\alpha)$	The minimal domain of α, see (48)
$\mathrm{D}_R(\alpha)$	The residue of α, see (47)

$\mathfrak{D}$	A feature checking system using only Delete(α), see (80)
$\mathfrak{DE}$	A feature checking system using both Delete(α) and Erasure, see (84)
ECP	Empty Category Principle
$\mathfrak{E}$	A feature checking system using only Erasure (and maybe emulating Delete(α) through it, see (81)–(83)
f	Some feature
F→F	One-to-one relation between features and functions, see (19)
F↔F	One-to-one correspondence between features and functions, see (20)
FF[X]	The set of formal features of X ($FF \in$ X); if X isn't a set, FF[X] denotes the set of formal features *containing* X
F	Some formal feature, i.e. some $f \in \mathcal{F}_{FF}$
F^{ϕ}	Some ϕ-feature
$F^{\phi,A}$	Some ϕ-feature of the relevant anaphor
$F^{\phi,S}$	Some ϕ-feature of the subject
FF	A subset of $\mathcal{F}_{FF}$ belonging to some lexical item
F^{INV}	A privative feature marking invisibility of features, page 42
F^{S}	A privative feature encoding feature strength, see (32)
F^{VIS}	A privative feature marking visibility of features, page 43
$\mathcal{F}$	The set of all features, see (22)
$\mathcal{F}_{FF}$	The set of all formal features
$\mathcal{F}_{LF}$	The set of all LF-features
$\mathcal{F}_{PF}$	The set of all PF-features
$\mathfrak{F}$	Some unspecified feature checking system, or a set of feature checking systems
GB	Government and Binding Theory
HEOC	Has an Effect on Output Condition, see (17c)
LCA	Linear Correspondence Axiom of Kayne (1994)
LF	Logical Form
LI	Lexical item; a multiset licensed by the definition in (24)
LF	A subset of $\mathcal{F}_{LF}$ belonging to some lexical item
LI	Some lexical item
Max(α)	The lowest maximal projection dominating α, see (41)
MCSG	Mildly-context sensitive grammar

MCSL	Mildly context-sensitive language
MG	Minimalist Grammar, see section 4.3.2
$\mathfrak{MG}^i_j$	A Minimalist Grammar with two parameters, where $i \in \{+SMC, -SMC\}$ expresses the presence of the Shortest Move Constraint, and the subscript $j \in \{+AIC, -AIC, +SPIC, -SPIC\}$ expresses the presence of the Adjunct Island Constraint or the Specifier Island Constraint
o	Occurrence index of a lexical item in the numeration
P&P	Principles-and-Parameters framework
PF	Phonological Form (sometimes called Physical Form)
PRD	Principle of Recoverability of Deletion, see (90) and (91)
PF	A subset of $\mathcal{F}_{PF}$ belonging to some lexical item
$\mathfrak{R}^{i,j}_{m,n}$	A minimalist framework modified as proposed in Reuland (2001) with the following parameters: $i \in \{S, L, O\}$ specifies the version of the CFD (see page 67), $j \in \{1C, 2C\}$ specifies the number of syntactic cycles (see (111)), $m \in \{\emptyset, CH, \mathcal{CH}\}$ specifies the timing of Chain- and CHAIN-construction (see (108) and (109)), and $n \in \{D, E\}$ specifies the timing of feature copying (see page 69)
SDP	Smallest Derivation Principle, see (17b)
SGC	Strong generative capacity
SMC	Shortest Move Constraint
SPIC	Specifier Island Constraint
SS	Surface Structure
S	Startsymbol
TEC	Transitive expletive construction, page 44
V_N	Set of non-terminal symbols
V_T	Set of terminal symbols
WGC	Weak generative capacity
X^{0max}	The minimal projection contained by no other minimal projection, see (40)

Bibliography

Abels, Klaus. 2003. *Successive cyclicity, anti-locality, and adposition stranding*. Doctoral Dissertation, University of Conneticut.

Adger, David. 2003. *Core syntax: A minimalist approach*. Oxford: Oxford University Press.

Adger, David. 2006. Remarks on minimalist feature theory and Move. *Journal of Linguistics* 42:663–673.

Adger, David, and S. Béjar, ed. to appear. *Phi theory: Phi features across interfaces and modules*. Oxford: Oxford University Press.

Adger, David, and Daniel Harbour. to appear. Why phi? In *Phi theory: Phi features across interfaces and modules*, ed. David Adger and S. Béjar. Oxford: Oxford University Press.

Aho, Alfred. 1968. Indexed grammars: An extension of context-free grammars. *Journal of the ACM* 15:647–671.

Amblard, Maxime, Alain Lecomte, and Christian Retoré. 2003. Syntax and semantics interacting in a minimalist theory. In *Prospects and Advances in the Syntax/Semantics Interface*, ed. Denys Duchier, Lorraine-Saarland Workshop Series, 17–22. Nancy.

Aoun, Joseph, and Dominique Sportiche. 1983. On the formal theory of government. *Linguistic Review* 2:211–236.

Ariel, Mira. 1990. *Accessing noun-phrase antecedents*. London: Croom Helm.

Asudeh, Ash, and Christopher Potts. 2004. Honorific marking: Interpreted and interpretable. Paper presented (by Chris Potts) at the Phi Workshop, McGill University, August 28, 2004, McGill University.

Asudeh, Ash, and Ida Toivonen. 2006. Response to David Adger's 'Remarks on minimalist feature theory and Move'. *Journal of Linguistics* 42:675–686.

Bach, E., and Barbara Partee. 1980. Anaphora and semantic structure. In *Papers from the parasession on pronouns and anaphora*, ed. Jody Kreiman and Almerindo E. Ojeda, 1–28. Chicago: Chicago Linguistic Society.

Barker, Chris, and Geoffrey K. Pullum. 1990. A theory of command relations. *Linguistics and Philosophy* 13:1–34.

Blackburn, Patrick, Claire Gardent, and Wilfried Meyer-Viol. 1993. Talking about trees. In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics*, 30–36.

Blackburn, Patrick, and Wilfried Meyer-Viol. 1994. Linguistics, logics, and finite trees. *Logic Journal of the IGPL* 2:3–29.

Bobaljik, Jonathan D., and Dianne Jonas. 1996. The subject positions and the role of TP. *Linguistic Inquiry* 27:195–236.

Bobaljik, Jonathan David. 2003. Realizing Germanic inflection: Why morphology does not drive syntax. *Journal of Comparative Germanic Linguistics* 6:129–167.

Boeckx, Cedric. 2006. *Linguistic minimalism: Origins, concepts, methods, and aims*. Oxford: Oxford University Press.

Boeckx, Cedric, and Kleanthes K. Grohmann. 2004. Barriers and phases: Forward to the past? Paper presented at Tools in Linguistic Theory 2004 (TiLT), Institute of Linguistics of the Hungarian Academy of Sciences (May 16-18), May 2004.

Bonato, Roberto. 2005. Towards a computational treatment of binding theory. In *Logical Aspects of Computational Linguistics, 5th International Conference, LACL 2005, Bordeaux, France, April 28-30, 2005*.

Borer, Hagit. 1984. *Parametric syntax*. Dordrecht: Foris.

Bresnan, Joan. 2000. *Lexical-functional syntax*. Oxford: Blackwell.

Brody, Michael. 1995. *Lexico-logical form: A radically Minimalist theory*. Cambridge, Mass.: MIT Press.

Brody, Michael. 2002. On the status of representations and derivations. In *Derivation and explanation in the minimalist program*, ed. Samuel D. Epstein and Daniel T. Seely, 19–41. Oxford: Blackwell.

Broekhuis, Hans. 2000. Against feature strength: The case of Scandinavian object shift. *Natural Language and Linguistic Theory* 18:673–721.

Buszkowski, Wojciech, Witold Marciszewski, and Johan van Benthem, ed. 1988. *Categorial grammar*. Amsterdam: Benjamins.

Canac-Marquis, Réjean. 2005. Phases and binding of reflexives and pronouns in English. In *Proceedings of the HPSG05 conference*, ed. Stefan Müller, 482–502. Stanford: CSLI Publications.

Chametzky, Robert A. 2000. *Phrase structure: From GB to minimalism*. Oxford: Blackwell.

Chametzky, Robert A. 2003. Phrase structure. In *Minimalist syntax*, ed. Randall Hendrick, 192–225. Oxford: Blackwell.

Chomsky, Noam. 1957. *Syntactic structures*. The Hague: Mouton.

Chomsky, Noam. 1963. Formal properties of grammar. In *Handbook of mathematical psychology, Volume II*, ed. R.Duncan Luce, Robert R. Bush, and Eugene Galanter, 323–418. New York: Wiley.

Chomsky, Noam. 1981. *Lectures on government and binding: The Pisa lectures*. Dordrecht: Foris.

Chomsky, Noam. 1986. *Barriers*. Cambridge, Mass.: MIT Press.

Chomsky, Noam. 1993. A minimalist program for linguistic theory. In *The view from building 20*, ed. Kenneth Hale and Samuel Jay Keyser, 1–52. Cambridge, Mass.: MIT Press.

Chomsky, Noam. 1995a. Bare phrase structure. In *Government and binding theory and the minimalist program*, ed. Gert Webelhuth, 383–440. Oxford: Blackwell.

Chomsky, Noam. 1995b. Categories and transformations. In *The minimalist program*, 219–394. Cambridge, Mass.: MIT Press.

Chomsky, Noam. 1995c. *The minimalist program*. Cambridge, Mass.: MIT Press.

Chomsky, Noam. 1998. *Minimalist inquiries: The framework*, volume 15 of *MIT Occasional Papers in Linguistics*. Cambridge, Mass.: MIT Press.

Chomsky, Noam. 2001. Derivation by phase. In *Ken Hale: A life in language*, ed. Michael J. Kenstowicz, 1–52. Cambridge, Mass.: MIT Press.

Chomsky, Noam. 2002. *On nature and language*. Cambridge, UK: Cambridge University Press.

Chomsky, Noam. 2004. Beyond explanatory adequacy. In *Structures and beyond: The cartography of syntactic structures volume 3*, ed. Adriana Belletti, 104–131. Oxford: Oxford University Press.

Chomsky, Noam. 2005a. On phases. Ms., MIT.

Chomsky, Noam. 2005b. Three factors in language design. *Linguistic Inquiry* 36:1–22.

Chomsky, Noam. 2007. Approaching UG from below. In *Interfaces + recursion = language?: Chomsky's minimalism and the view from syntax-semantics*, ed. Uli Sauerland and Hans-Martin Gärtner, 1–30. Berlin: Mouton de Gruyter.

Chomsky, Noam, and Morris Halle. 1968. *The sound pattern of English*. New York: Evanston.

Chomsky, Noam, and Howard Lasnik. 1993. The theory of principles and parameters. In *Syntax: An international handbook of contemporary research*, ed. Joachim Jacobs, Arnim von Stechow, and Wolfgang Sternefeld, 506–569. Berlin: Walter de Gruyter.

Citko, Barbara. 2006. Missing labels. Ms., University of Washington.

Collins, Chris. 1996. *Local economy*. Cambridge, Mass.: MIT Press.

Collins, Chris. 2002. Eliminating labels. In *Derivation and explanation in the minimalist program*, ed. Samuel D. Epstein and Daniel T. Seely, 42–64. Oxford: Blackwell.

Collins, Chris, and Höskuldur Þráinsson. 1996. VP-internal structure and object shift in Icelandic. *Linguistic Inquiry* 27:391–444.

Cornell, Thomas L. 1997. Representational minimalism. Technical report, University of Tübingen. Arbeitspapiere des Sonderforschungsbereichs 340, Technical Report 83.

Epstein, Samuel D., Erich M. Groat, Ruriko Kawashima, and Hisatsugu Kitahara. 1998. *A derivational approach to syntactic relations*. Oxford: Oxford University Press.

Epstein, Samuel D., and Daniel T. Seely. 2002. Rule applications as cycles in a level-free syntax. In *Derivation and explanation in the minimalist program*, ed. Samuel D. Epstein and Daniel T. Seely, 65–89. Oxford: Blackwell.

Epstein, Samuel D., and Daniel T. Seely. 2006. *Derivations in minimalism*. Cambridge, Mass.: Cambridge University Press.

Fox, Danny. 1995. Economy and scope. *Natural Language Semantics* 3:283–341.

Frampton, John, Sam Gutmann, Julie Legate, and Charles Yang. 2000. Remarks on "Derivation by phase": Feature valuation, agreement, and intervention. Ms., available online: http://www.math.neu.edu/ling/pdffiles/remdbp.pdf [2007-09-01].

Freidin, Robert, and Jean-Roger Vergnaud. 2001. Exquisite connections: Some remarks on the evolution of linguistic theory. *Lingua* 111:639–666.

Frey, Werner, and Hans-Martin Gärtner. 2002. On the treatment of scrambling and adjunction in minimalist grammars. In *Proceedings of the Conference on Formal Grammar (FGTrento)*, 41–52. Trento.

Gazdar, Gerald. 1985. Applicability of indexed grammars to natural languages. Technical Report CSLI-85-34, Stanford University, Stanford, CA, USA, Center for Study of Language and Information.

Gazdar, Gerald, Ewan Klein, Geoffrey K. Pullum, and Ivan A. Sag. 1985. *Generalized phrase structure grammar*. Oxford: Blackwell.

Gorn, Saul. 1967. Explicit definitions and linguistic dominoes. In *Systems and Computer Science, Proceedings of the Conference held at University of Western Ontario, 1965*. Toronto: University of Toronto Press.

Graf, Thomas. 2007. So you are saying Rule I is transderivational? Ms., University of Vienna.

Grodzinksy, Yosef, and Tanja Reinhart. 1993. The innateness of binding and coreference. *Linguistic Inquiry* 24:69–102.

Grohmann, Kleanthes K. 2003. *Prolific domains: On the anti-locality of movement dependencies*. Amsterdam: John Benjamins.

Gärtner, Hans-Martin. 2002. *Generalized transformations and beyond: Reflections on minimalist syntax*. Berlin: Akademie-Verlag.

Gärtner, Hans-Martin, and Jens Michaelis. 2005. A note on the complexity of contraint interaction. In *Logical aspects of computational linguistics (lacl'05)*, ed. P. Blache, E. Stabler, and J. Busquets, number 3492 in Lecture Notes in Artificial Intelligence, 114–130. Berlin: Springer.

Gärtner, Hans-Martin, and Jens Michaelis. 2007. Some remarks on locality conditions and Minimalist Grammars. In *Interfaces + recursion = language? Chomsky's minimalism and the view from syntax-semantics*, ed. Uli Sauerland and Hans-Martin Gärtner, 161–196. Berlin: Mouton de Gruyter.

Gärtner, Hans-Martin, and Jens Michealis. 2003. A note on countercyclicity and minimalist grammars. In *Proceedings of the Conference on Formal Grammar (FGVienna)*, 103–114. Vienna.

Halle, Morris, and Alec Marantz. 1993. Distributed morphology and the pieces of inflection. In *The view from building 20*, ed. Ken Hale and Samuel J. Keyser, 111–176. Cambridge, Mass.: MIT Press.

Harkema, Henk. 2001a. A characterization of minimalist languages. In *Logical aspects of computational linguistics (lacl'01)*, ed. Philippe de Groote, Glyn Morrill, and Christian Retoré, volume 2099 of *Lecure Notes in Artificial Intelligence*, 193–211. Berlin: Springer.

Harkema, Henk. 2001b. *Parsing minimalist languages*. Doctoral Dissertation, University of California.

Harley, Heidi, and Elizabeth Ritter. 2002a. Person and number in pronouns: A feature-geometric analysis. *Language* 78:482–526.

Harley, Heidi, and Elizabeth Ritter. 2002b. Structuring the bundle: A universal morphosyntactic feature geometry. In *Pronouns — grammar and representation*, ed. Horst J. Simon and Heike Wiese, 23–39. Amsterdam: John Benjamins.

Hauser, Marc, Noam Chomsky, and W. Tecumseh Fitch. 2002. The faculty of language: What is it, who has it, and how did it evolve? *Science* 298:1569–1579.

Heim, Irene. 1998. Anaphora and semantic interpretation: A reinterpretation of Reinhart's approach. In *The interpretive tract*, ed. Uli Sauerland and O. Percus, volume 25 of *MIT Working Papers in Linguistics*, 205–246. Cambridge, Mass.: MIT Press.

Heim, Irene. 2005. Features on bound pronouns. Ms., MIT, Cambridge, Mass.

Heim, Irene, and Angelika Kratzer. 1998. *Semantics in generative grammar*. Oxford: Blackwell.

Heinat, Fredrik. 2006. *Probes, pronouns and binding in the minimalist program*. Doctoral Dissertation, University of Lund.

Hornstein, Norbert. 2006. Pronouns in a minimalist setting. In *University of Maryland working papers in linguistics*, ed. N. Kazanina, U. Minai, P. Monahan, and H. Taylor, 14, 47–80. Maryland: University of Maryland.

Hornstein, Norbert, and Jairo Nunes. 2006. Some thoughts on adjunction. Ms., University of Maryland.

Hornstein, Norbert, Jairo Nunes, and Kleanthes K. Grohmann. 2005. *Understanding minimalism*. Cambridge: Cambridge University Press.

Johnson, David, and Shalom Lappin. 1997. A critique of the minimalist program. *Linguistics and Philosophy* 20:273–333.

Jonas, Dianne, and Jonathan David Bobaljik. 1993. Specs for subjects: The role of TP in Icelandic. *MIT Working Papers in Linguistics* 18:59–98.

Joshi, Aravind. 1985. Tree-adjoining grammars: How much context sensitivity is required to provide reasonable structural descriptions? In *Natural language parsing*, ed. David Dowty, Lauri Karttunen, and Arnold Zwicky, 206–250. Cambridge: Cambridge University Press.

Joshi, Aravind, L. S. Levy, and M. Takahashi. 1975. Tree adjunct grammars. *Journal of Computer and System Sciences* 10:136–163.

Joshi, Aravind, and Y. Schabes. 1997. Tree-adjoining grammars. In *Handbook of formal languages*, ed. Grzegorz Rosenberg and Arto Salomaa, 69–123. Berlin: Springer.

Joshi, Aravind, K. Vijay-Shanker, and David Weir. 1991. The convergence of mildly context-sensitive grammar formalisms. In *Foundational issues in natural language processing*, ed. P. Sells, Shieber S. M., and T. Warsow, 31–81. Cambridge, MA, USA: MIT Press.

Kaye, Jonathan, Jean Lowenstamm, and Jean-Roger Vergnaud. 1985. The internal structure of phonological representations: a theory of charm and government. *Phonology Yearbook* 2:305–328.

Kaye, Jonathan, Jean Lowenstamm, and Jean-Roger Vergnaud. 1990. Constituent structure and government in phonology. *Phonology Yearbook* 7:193–231.

Kayne, Richard S. 1994. *The antisymmetry of syntax*. Cambridge, Mass.: MIT Press.

Kayne, Richard S. 2002. Pronouns and their antecedents. In *Derivation and explanation in the minimalist program*, ed. Samuel D. Epstein, 133–166. Oxford: Blackwell.

Keenan, Edward. 1988. On semantics and the binding theory. In *Explaining language universals*, ed. J. A. Hawkins, 105–144. Oxford: Blackwell.

Kobele, Gregory. 2006. *Generating copies: An investigation into structural identity in language and grammar*. Doctoral Dissertation, UCLA.

Kobele, Gregory M. 2005. Features moving madly: A formal perspective on feature percolation in the minimalist program. *Research on Language and Computation* 3:391–410.

Kobele, Gregory M., and Jens Michaelis. 2005. Two type-0 variants of minimalist grammars. In *FG-MoL 2005. The 10th conference on Formal Grammar and the 9th Meeting on Mathematics of Language*. Edinburgh.

Kolb, Hans-Peter. 1999. Macros for minimalism: Towards weak descriptions of strong structures. In *Mathematics of syntactic structure*, ed. Hans-Peter Kolb and Uwe Mönnich, 232–259. Berlin: Walter de Gruyter.

Kracht, Marcus. 1993. Mathematical aspects of command relations. In *Proceedings of the European Meeting of the Association for Computational Linguistics*, 240–249.

Kratzer, Angelika. 1998. More structural analogies between pronouns and tenses. In *Proceedings from Semantics and Linguistic Theory VIII*, ed. Devon Strolovitch and Aaron Lawson, 92–110. New York: Cornell University, CLC Publications.

Kratzer, Angelika. 2004. Telicity and the meaning of objective case. In *The syntax of time*, ed. Jacqueline Guéron and Jacqueline Lecarme, 389–423. Cambridge, Mass.: MIT Press.

Kratzer, Angelika. 2006. Minimal pronouns. fake indexical as windows into the properties of bound variable pronouns. To appear in *Linguistic Inquiriy*, Cambridge, Mass.

Lakoff, George. 1971. On generative semantics. In *Semantics: An interdisciplinary reader in philosophy, linguistics and psychology*, ed. Danny D. Steinberg and Leon A. Jakobovits, 232–296. Cambridge: Cambridge University Press.

Lappin, Shalom, Robert Levine, and David E. Johnson. 2000a. The revolution confused: A reply to our critics. *Natural Language and Linguistic Theory* 18:873–890.

Lappin, Shalom, Robert Levine, and David E. Johnson. 2000b. The structure of unscientific revolutions. *Natural Language and Linguistic Theory* 18:765–771.

Lappin, Shalom, Robert Levine, and David E. Johnson. 2001. The revolution maximally confused. *Natural Language and Linguistic Theory* 19:901–919.

Lasnik, Howard. 1999. On feature strength: Three minimalist approaches to overt movement. *Linguistic Inquiry* 30:197–217.

Lasnik, Howard, and Randall Hendrick. 2003. Steps toward a minimal theory of anaphora. In *Minimalist syntax*, ed. Randall Hendrick, 124–151. Oxford: Blackwell.

Lecomte, Alain. 1998. Categorial minimalism. In *Logical aspects of computational linguistics (lacl'98)*, ed. Michael Moortgat, number 2014 in Lecture Notes in Artificial Intelligence, 179–198. Berlin: Springer.

Lecomte, Alain. 2003. Categorial grammar for minimalism. In *Language and grammar: Studies in mathematical linguistics and natural language*, ed. Claudio Casadio, 143–158. Stanford: CSLI Publications.

Liddy, E. Du Ross. 1990. Anaphora in natural language processing and information retrieval. *Information Processing and Management: An International Journal* 26:39–52.

Lotfi, Ahmad R. 2002. Minimalist program revisited: Chomsky's strength to trigger movement. In *Proceedings of the 34th colloquium of linguistics*, ed. R. Rapp and Peter Lang.

McCawley, James D. 1968. Concerning the base component of a transformational grammar. *Foundations of Language* 4:243–269.

Michaelis, Jens. 1998. Derivational minimalism is mildly context-sensitive. *Lecture Notes in Artificial Intelligence* 2014:179–198.

Michaelis, Jens. 2001. Transforming linear context-free rewriting systems into minimalist grammars. *Lecture Notes in Artifiicial Intelligence* 2099:228–244.

Miller, Philip. 1999. *Strong generative capacity: The semantics of linguistic formalism*. Stanford: CSLI Publications.

Moortgat, Michael. 1996. Categorial type logics. In *Handbook of logic and language*, ed. Johan van Benthem and Alice ter Meulen, 93–177. Amsterdam and Cambridge, Mass.: Elsevier and MIT Press.

Mönnich, Uwe. 1999. On cloning context-freeness. In *Mathematics of syntactic structure*, ed. Hans-Peter Kolb and Uwe Mönnich, 195–231. Berlin: Walter de Gruyter.

Müller, Gereon. 1999. Imperfect checking. *The Linguistic Review* 16:359–404.

Nunes, Jairo. 2000. Erasing erasure. *D.E.L.T.A.* 16:415–429.

Nunes, Jairo, and Ellen Thompson. 1998. Appendix. In *Rhyme and reason*, Juan Uriagereka, 497–521. Cambridge, Mass.: MIT Press.

Partee, Barbara, Alice ter Meulen, and Robert Wall. 1990. *Mathematical methods in linguistics*. Dordrecht: Kluwer.

Pesetsky, David, and Esther Torrego. 2001. T to C movement: Causes and consequences. In *Ken hale: A life in language.*, ed. Michael J. Kenstowicz, 355–426. Cambridge, Mass.: MIT Press.

Pesetsky, David, and Esther Torrego. 2007. The syntax of valuation and the uninterpretability of features. In *Phrasal and clausal architectur*, ed. Simin Karimi, Samiian Samiian, and Wendy K. Wilkins, 262–294. Amsterdam: Benjamins.

Peters, P. Stanley, and Robert W. Ritchie. 1969. Context-sensitive immediate constituent analysis — Context-free languages revisited. In *Proceedings of the first annual ACM symposium on Theory of computing*. Republished in *Mathematical Systems Theory* 6, 324-333.

Pollard, Carl. 1985. *Generalized phrase structure grammars, head grammars and natural language*. Doctoral Dissertation, Stanford University.

Pollard, Carl, and Ian Sag. 1992. Anaphors in English and the scope of binding theory. *Linguistic Inquiry* 23:261–303.

Pollard, Carl, and Ivan Sag. 1994. *Head-driven phrase structure grammar*. Stanford and Chicago: CSLI and The University of Chicago Press.

Potts, Christopher. 2001. Three kinds of transderivational constraint. In *Syntax at santa cruz*, ed. Séamas Mac Bhloscaidh, volume 3, 21–40. Santa Cruz: Linguistics Department, UC Santa Cruz.

Potts, Christopher. 2002. Comparative economy conditions in natural language syntax. Paper presented at the North American Summer School in Logic, Language, and Information 1, Workshop on Model-Theoretic Syntax, Stanford University (June 28), June 2002.

Pullum, Geoffrey K., and Gerald Gazdar. 1982. Natural languages and context free languages. *Linguistics and Philosophy* 4:471–504.

Reinhart, Tanya. 1983. *Anaphora and semantic interpretation*. Chicago University Press: Croon-Helm.

Reinhart, Tanya. 2000. Strategies of anaphora resolution. In *Interface strategies*, ed. Hans Bennis, Martin Everaert, and Eric Reuland, 295–325. Amsterdam: Royal Academie of Arts and Sciences.

Reinhart, Tanya, and Eric Reuland. 1991. Anaphors and logophors: An argument structure perspective. In *Long-distance anaphora*, ed. Jan Koster and Eric Reuland, 283–321. Cambridge: Cambridge University Press.

Reinhart, Tanya, and Eric Reuland. 1993. Reflexivity. *Linguistic Inquiry* 24:657–720.

Reuland, Eric. 2001. Primitivs of binding. *Linguistic Inquiry* 32:439–492.

Reuland, Eric. 2005. Binding conditions: How are they derived? In *Proceedings of the hpsg05 conference*, ed. Stefan Müller, 578–594. Stanford: CSLI Publications.

Reuland, Eric, and Tanya Reinhart. 1995. Pronouns, anaphors and case. In *Studies in comparative Germanic syntax*, ed. Hubert Haider, Susan Olsen, and Sten Vikner, 241–269. Dordrecht: Kluwer.

Richards, Norvin. 1996. Toward a feature-movement theory of long-distance anaphora. Unpublished Ms., MIT.

Roach, Kelly. 1987. Formal properties of Head Grammars. In *Mathematics of language*, ed. Alexis Manaster-Ramer, 293–347. Amsterdam/Philadephia: John Benjamins Publishing Company.

Rogers, James. 1998. *A descriptive approach to language-theoretic complexity*. Stanford: CSLI.

Rogers, James. 2003. Syntactic structures as multi-dimensional trees. *Research on Language and Computation* 1:265–305.

Rohrbacher, Bernhard. 1999. *Morphology-driven syntax: A theory of V to I raising and* pro-*drop*. Amsterdam: John Benjamins.

Safir, Kenneth J. 2004. *The syntax of anaphora*. Oxford: Oxford University Press.

Sauerland, Uli. 1995. Early features. In *Papers in minimalist syntax*, ed. Robert Pensalfini and Hiroyuki Ura, volume 27 of *MIT Working Papers In Linguistics*, 223–242. Cambridge, Mass.: MIT Press.

Sauerland, Uli. 2003. A new semantics for number. In *SALT 13*, ed. R. B. Youn and Y. Zhou. Ithaca: CLC publications.

Sauerland, Uli. 2004. A comprehensive semantics for agreement. Paper presented at the Phi-Workshop, McGill University, Montreal, Canada. August 2004.

Schlenker, Philippe. 2005. Non-redudancy: Towards a semantic reinterpretation of binding theory. *Natural Language Semantics* 13:1–92.

Shieber, Stuart M. 1985. Evidence against the context-freeness of natural language. *Linguistics and Philosophy* 8:333–345.

Sigurðsson, Halldór Ármann. 2003. Case: abstract vs. morphological. In *New perspectives on Case theory*, ed. Ellen Brandner and Heike Zinzmeister, 223–268. Stanford: CSLI Publications.

Sigurðsson, Halldór Ármann. 2006a. The Nom/Acc alternation in Germanic. In *Comparative studies in Germanic syntax*, ed. Jutta Hartmann and Laszlo Molnarfi, 13–50. Amsterdam: John Benjamins.

Sigurðsson, Halldór Ármann. 2006b. The nominative puzzle and the low nominative hypothesis. *Linguistic Inquiry* 37:289–308.

Soames, Scott. 1974. Rule orderings, obligatory transformations, and derivational constraints. *Theoretical Linguistics* 1:116–138.

Stabler, Edward P. 1992. *The logical approach to syntax: Foundations, specifications and implementations of theories of government and binding*. Cambridge, Mass.: MIT Press.

Stabler, Edward P. 1997. Derivational minimalism. In *Logical aspects of computational linguistics: First international conference, LACL '96, Nancy, France, September 23-25, 1996. selected papers*, ed. Christian Retoré, 68–95. Berlin: Springer.

Stabler, Edward P. 1999. Remnant movement and complexity. In *Constraints and resources in natural language syntax and semantics*, ed. Gosse Bouma, Geert-Jan M. Kruijff, Erhard Hinrichs, and Richard T. Oehrle, 299–326. Stanford, CA: CSLI Publications.

Stabler, Edward P. 2001. Recognizing head movement. In *Logical aspects of computational linguistics (LACL'01)*, ed. Philippe de Groote, Glyn Morrill, and Christian Retoré, volume 2099 of *Lecture Notes in Artificial Intelligence*, 245–260. Berlin: Springer.

Stabler, Edward P. 2003. Comparing 3 perspectives on head movement. In *Syntax at sunset 3: Head movement and syntactic theory*, ed. A. Mahajan, volume 10 of *UCLA Working Papers in Linguistics*, 178–198. Los Angeles, CA: UCLA.

Stabler, Edward P., and Edward Keenan. 2003. Structural similarity. *Theoretical Computer Science* 293:345–363.

Stechow, Arnim von. 2003a. Feature deletion under semantic binding: Tense, person, and mood under verbal quantifiers. Ms., Universität Tübingen.

Stechow, Arnim von. 2003b. Postscript to "Feature deletion under semantic binding": A note on (Kratzer, 1998). Ms., Universität Tübingen.

Steedman, Mark. 1996. *Surface structure and interpretation*. Cambridge, Mass.: MIT Press.

Steedman, Mark, and Jason Baldridge. 2003. Combinatory categorial grammar. Unpublished tutorial paper.

Sternefeld, Wolfgang. 1996. Comparing reference-sets. In *The role of economy principles in linguistic theory*, ed. Chris Wilder, Hans-Martin Gärtner, and Manfred Bierwisch, 81–114. Berlin: Akademie Verlag.

Svenonius, Peter. 2002. Icelandic case and the structure of events. *Journal of Comparative Germanic Linguistics* 5:197–225.

Svenonius, Peter. 2004. On the edge. In *Peripheries: Syntactic edges and their effects*, ed. David Adger, Cécile de Cat, and George Tsoulas, 259–287. Dordrecht: Kluwer.

Uriagereka, Juan. 1998. *Rhyme and reason*. Cambridge, Mass.: MIT Press.

Uriagereka, Juan. 2000. Some thoughts on economy within linguistics. *D.E.L.T.A.* 16:221–243.

Vermaat, Willemijn. 1999. *Controlling movement: Minimalism in a deductive perspective*. Master's thesis, Utrecht University.

Vijay-Shanker, K., and David J. Weir. 1994. The equivalence of four extensions of context-free grammars. *Mathematical Systems Theory* 27:511–545.

Vikner, Sten. 1997. V^0-to-I^0 movement and inflection for person in all tenses. In *The new comparative syntax*, ed. Liliane Haegeman, 189–213. London: Longman.

Weir, David. 1988. *Characterizing mildly context-sensitive grammar formalisms*. Doctoral Dissertation, University of Pennsylvania. Available as Technical Report MS-CIS-88-74 of the Department of Computer and Information Sciences, University of Pennsylvania.

Weir, David. 1992. A geometric hierarchy beyond context-free languages. *Theoretical Computer Science* 104:235–261.

Williams, Edwin. 1994. *Thematic structure in syntax*. Cambridge, Mass.: MIT Press.

Zeijlstra, Hedde. 2006. Functional structure, formal features and parametric variation: Consequences of conflicting interface conditions. Ms., Universiteit van Amsterdam.

Zwart, Jan-Wouter. 2002. Issues relating to a derivational theory of binding. In *Derivation and explanation in the minimalist program*, ed. Samuel D. Epstein, 269–304. Oxford: Blackwell.

Þráinsson, Höskuldur. 2003. Syntactic variation, historical development, and minimalism. In *Minimalist syntax*, ed. Randal Hendrick, 152–191. Oxford: Blackwell.