Inessential Features

Marcus Kracht [⋆]

II. Mathematisches Institut
Freie Universität Berlin
Arnimallee 3
D-14195 Berlin
kracht@math.fu-berlin.de

Abstract. If one converts surface filters into context free rules, one has to introduce new features. These features are strictly nonlexical, and their distribution is predictable from the distribution of the lexical features. Now, given a (feature based) context free grammar, we ask whether one can identify the nonlexical features. This is not possible; however, the notion of an *inessential feature* offers an approximation. To arrive at a descriptive theory of language one needs to eliminate all the inessential features. One can measure the complexity of a language by the complexity of the formulae needed to define the distribution of inessential features.

1 Introduction

According to the lexicalist doctrine, the features that get used in syntax must be exactly those that are used in the lexicon as well. That is to say, the lexicon already provides the necessary distinctive features on which syntax operates, and there is no need to introduce new ones. We will give two examples. In X–bar–syntax a distinction is drawn between (usually) three levels in the completion of a phrase: the lexical level, the intermediate level and the phrasal level. From a lexicalist perspective, although talk about the various levels is not meaningless, their use in syntax should in principle be eliminable. In other words the status of a constituent in a syntactic structure is not freely assigned by syntax, but follows exactly from the assignment of purely lexical features in the syntactic tree. If something is a phrase in a structure, it necessarily is a phrase in that structure. For example, in

(1.) *John is [very [proud of his son]].*

we have the adjective *proud*, and the adjectival phrase *very proud of his son*. That the latter is a phrase follows from the principles of X–bar–syntax. [1] There is an

[⋆] This research has been carried out in collaboration with the Innovationskolleg 'Formale Modelle kognitiver Komplexität' (INK 12/A3) at Potsdam University, funded by the DFG. I wish to thank Jens Michaelis for many useful discussions. Thanks also to Hans Leiß and James Rogers for raising interesting questions.

[1] Notice that strictly speaking, at least the distinction between lexical and phrasal level is justified for a lexicalist. In English, *John* acts like a noun phrase, not like

intuition that syntax should only talk about those features that are given by the lexicon. In other words, syntax and morphology share a pool of features, which we call lexical features. A syntactic theory (and a morphological theory) is not allowed to introduce new features, and computation should proceed using only the given lexical features. This is not a substantial criterion but a criterion that makes it possible to separate lexical from nonlexical features and to provide a complexity measure for the computational system (or syntax) alone. However, in the Minimalist Program it is actually assumed that the computational system performs computations only on the feature complexes of the lexical elements which get inserted from the lexicon. Consequently, Chomsky [2] has sought to eliminate the levels from X–bar–syntax (see also the discussion thereof in Kracht [6]).

The other example concerns the slash–feature of GPSG. [2] Here the intuition is somewhat clearer. The slash–feature was introduced to control the dependency between a filler and the corresponding gap. As an example we use simple question formation and topicalization. Consider the contrast between (2.), (3.) and (4.).

(2.) *Alfred is [stealing books].*
(3.) *What is Alfred [stealing]?*
(4.) *Books, Alfred is [stealing].*

Steal is a transitive verb, and it normally expects its object to the right. However, in questions and other constructions, this need not be so. Two solutions offer themselves. One is to supply the object in form of an empty element, and then discuss separately the distribution of such empty elements; another is to revise the context restriction of transitive verbs according to the facts just presented. The first has been implemented in transformational grammar, the latter in GPSG. In GPSG, a transitive verb may occur either in a constituent together with a direct object, or it may form a constituent of the form *transitive verb with an object missing*, which is coded with the help of the slash–feature roughly as tv[slash : np], where tv is the category of transitive verbs. The problem is that the addition of this new feature is inadmissible for a lexicalist. Any transitive verb is equally admissible in the contexts (2′.), (3′.) and (4′.). There is therefore no need to discriminate between words that can be used in one of the contexts and not the other. [3]

(2′.) *Alfred is ____ books.*
(3′.) *What is Alfred ____?*
(4′.) *Books, Alfred is ____.*

Thus in both cases we have an instance of features that are not motivated from the lexicon but only from syntax.

a noun, and there are a number of phrasal pro–forms with this property, too, e. g. *one*, *such*, *so*.

[2] Notice that what we call features in the sequel are booleans, not features in the sense of GPSG and feature logic.

[3] Actually, this is not quite right. (3′.) can be filled by *doing*, while (2′.) and (4′.) cannot. However, *do* is anyway syntactically distinct from a full verb. Hence, that it discriminates the given contexts need not invalidate our argument.

2 The Constituent Logic(s)

2.1 The Basic Structures

Some notation shall be fixed for convenience. With M a set, let $\wp(M)$ denote the powerset of M, and let M^* be the set of finite strings of elements from M. An element of M^* is denoted by bold face type, e. g. $\boldsymbol{x}$. Given binary relations R, S over M, put

$$R \circ S := \{\langle x, z\rangle : (\exists y)(\langle x, y\rangle \in R \text{ and } \langle y, z\rangle \in S)\}$$

Moreover, R^n is defined inductively by $R^0 := \{\langle x, x\rangle : x \in M\}$, $R^{n+1} := R \circ R^n$. Finally, $R^* := \bigcup_{n\geq 0} R^n$ and $R^+ := \bigcup_{n>0} R^n$.

Our original structures are ordered trees — possibly infinitely branching — and decorated with features from a given set F of features. F can in principle be any set (it may even even empty) but it is always required to be *finite*. Here, an *ordered labelled tree over* F is a quadruple $\mathfrak{O} = \langle T, <, L, \xi\rangle$, where $<$ is a tree ordering on T, $\xi : T \to \wp(F)$ is a function, called the *labelling function*, and L an ordering *compatible* with $<$, the *left–of–relation.* $<$ is a *tree ordering* on T if $<$ is a binary relation on T which is transitive, irreflexive, the sets $\{y : y > x\}$ are linear for all x, and there is an r such that $r > x$ for all $x \neq r$. A *leaf* is a node x such that for no y, $y < x$. We write $y \leq x$ if $y < x$ or $y = x$. L is *compatible with* $<$ if it satisfies the following postulates. (l) L is linear on the leaves, (c) xLy iff for all leaves u and v such that $u \leq x$ and $v \leq y$ it holds that uLv. Given a labelled ordered tree, we define two relations $\prec$ (child–of) and $\sqsubset$ (immediate left–sister–of).

$$\begin{array}{lll} x \prec y & \text{iff} & x < y \text{ and for no } z,\ x < z < y, \\ x \sqsubset y & \text{iff} & xLy, \text{ for no } v\text{: } xLvLy, \text{ and for some } z\text{: } x \prec z \text{ and } y \prec z. \end{array}$$

We put $\mathsf{T}(\mathfrak{O}) := \langle T, \prec, \sqsubset, \xi\rangle$. A quadruple $\mathfrak{T} = \langle T, \prec, \sqsubset, \xi\rangle$ where T is a set, $\prec$ and $\sqsubset$ binary relations on T and $\xi : T \to \wp(F)$ a function is called an *F–tree* if it is of the form $\mathsf{T}(\mathfrak{O})$ for some ordered labelled tree $\mathfrak{O}$. The relations $<$ and L can be recovered as follows: $< = \prec^+$, and $L = \prec^* \circ \sqsubset^+ \circ \succ^*$. Therefore, the class of F–trees can be characterized directly. Namely, $\mathfrak{T} = \langle T, \prec, \sqsubset, \xi\rangle$ is a finite F–tree iff $(\alpha) - (\delta)$ hold. (α) The transitive closure $\prec^+$ of $\prec$ is a tree ordering, (β) $\sqsubset^+$ and its converse are both irreflexive linear orders, (γ) If $x, y \prec z$ then $x \sqsubset^+ y$, $x = y$ or $y \sqsubset^+ x$, (δ) If $x \prec z$ and $x \sqsubset^+ y$ or $y \sqsubset^+ x$ then also $y \prec z$.

2.2 Modal Languages for Trees

With each $f \in F$ we associate a boolean constant f. (For simplicity, we also write f in place of f.) Our boolean connectives are $\top$, $\wedge$, $\neg$, the others being defined from them in the usual way. We denote by $\mathfrak{Tm}_{Boo}(F)$ the set of boolean terms over F that can be formed in this language. We denote members of $\mathfrak{Tm}_{Boo}(F)$ by a, b etc. We write $\mathsf{a} \leq \mathsf{b}$ if $\mathsf{a} \to \mathsf{b}$ is a boolean tautology. For a set $C \subseteq F$ we put

$$\chi(C) := \bigwedge_{f\in C} \mathsf{f} \wedge \bigwedge_{f\in F-C} \neg\mathsf{f}$$

Each boolean term over F is equivalent to a disjunction of formulae $\chi(C)$ for certain C (for example the disjunctive normal form).

There are a number of languages with which we will talk about these structures. The first three will be introduced in this section. The largest of them is called *Olt* (orientation language over trees). It is identical to propositional dynamic logic (**PDL**) over four basic programs, called *up*, *down*, *left* and *right*. Recall that **PDL** has two sorts of expressions, *programs* and *propositions*. There is a set *Var* of propositional variables, a set F of propositional constants, and a set Π_0 of program constants. Propositions are formed by using boolean connectives. The basic set is $\top$, $\neg$ and $\wedge$, all others are defined in the usual way. Moreover, if α is a program and φ a proposition, then $[\alpha]\varphi$ and $\langle\alpha\rangle\varphi$ are propositions. We assume that $[\alpha]\varphi$ is equivalent with $\neg\langle\alpha\rangle\neg\varphi$. Programs are formed in the following way. Members of Π_0 are programs, called *basic programs*. If α and β are programs, then $\alpha;\beta$, $\alpha\cup\beta$ and α^* are programs as well. We define $\alpha^+ := \alpha;\alpha^*$. Finally, if φ is a proposition, then $\varphi?$ is a program. In our case,

$$\Pi_0 = \{\mathit{left}, \mathit{right}, \mathit{up}, \mathit{down}\}$$

We write $\Diamond_{\uparrow}$ for $\langle \mathit{up}\rangle$, $\Box_{\uparrow}$ for $[\mathit{up}]$, and similar conventions are used for $\Diamond_{\downarrow}$, $\Box_{\downarrow}$, $\Diamond_{\rightarrow}$, $\Box_{\rightarrow}$, $\Diamond_{\leftarrow}$ and $\Box_{\leftarrow}$. Moreover, we write $\Diamond_{\uparrow}^{*}$ for $\langle \mathit{up}^*\rangle$, $\Box_{\uparrow}^{*}$ for $[\mathit{up}^*]$, $\Diamond_{\uparrow}^{+}$ for $\langle \mathit{up}^+\rangle$ and $\Box_{\uparrow}^{+}$ for $[\mathit{up}^+]$ (and likewise for *down*, *left* and *right*). There are two more languages which are of interest to us. The first is *BOlt(F)*, the *basic* orientation language, which is a 4–modal language based on the operators $\Box_{\downarrow}$, $\Box_{\uparrow}$, $\Box_{\leftarrow}$ and $\Box_{\rightarrow}$, with additional constants from F. The other language is what we call the *weak* orientation language *WOlt(F)* of Blackburn, de Rijke & Meyer–Viol [1]; it is a modal logic based on the primitive operators $\Box_{\uparrow}$, $\Box_{\uparrow}^{*}$, $\Box_{\downarrow}$, $\Box_{\downarrow}^{*}$, $\Box_{\leftarrow}$, $\Box_{\leftarrow}^{*}$, $\Box_{\rightarrow}$ and $\Box_{\rightarrow}^{*}$. [4] Both languages can be construed as fragments of **PDL**. Namely, *BOlt(F)* coincides with the $*$–free fragment of **PDL**, also known as **EPDL**. *WOlt(F)* coincides with the $*$–free fragment over Π_1 where

$$\Pi_1 := \Pi_0 \cup \{\mathit{up}^*, \mathit{down}^*, \mathit{left}^*, \mathit{right}^*\}$$

The structures for these logics are the same for all languages, namely generalized Kripke–structures $\mathfrak{T}$ together with a valuation function for the constants from F. Generalized Kripke–structures here are simply called *structures* and are quintuples $\langle T, \prec, \sqsubset, \mathbb{T}, \xi\rangle$ such that T is a non–empty set, $\prec$ and $\sqsubset$ binary relations over T, $\xi : T \to \wp(F)$ and $\mathbb{T} \subseteq \wp(T)$ a system of sets closed under relative complement, intersection and $[\alpha]$, where for $B \subseteq T$

$$[\alpha]B := \{x : (\forall y)(\langle x, y\rangle \in R(\alpha) \to y \in B)\} \ .$$

[4] This language has a different name in the quoted paper, but we decided to harmonize the terminology here.

Here α is a program of the language and $R(\alpha)$ the associated binary relation in T. $R(\alpha)$ is computed as follows.

$$\begin{array}{lcllcl}
R(\mathit{right}) & := & \sqsubset & R(\mathit{left}) & := & \sqsupset \\
R(\mathit{up}) & := & \prec & R(\mathit{down}) & := & \succ \\
R(\alpha;\beta) & := & R(\alpha) \circ R(\beta) & R(\alpha \cup \beta) & := & R(\alpha) \cup R(\beta) \\
R(\alpha^*) & := & R(\alpha)^* & & &
\end{array}$$

$\langle T, \prec, \sqsubset, \xi \rangle$ is a Kripke–structure if $\langle T, \prec, \sqsubset, \wp(T), \xi \rangle$ is a generalized Kripke–structure. So, the structures for these languages differ only with respect to the closure properties for the system $\mathbb{T}$. The Kripke–structures do not change. (Readers unfamiliar with generalized Kripke–structures may think of them as Kripke–structures instead. This may go at the expense of precision, but is more intuitive to begin with.)

A *model* is a triple $\mathfrak{M} = \langle \mathfrak{T}, \beta, x \rangle$, where $\mathfrak{T}$ is a generalized Kripke–structure, $\beta : \mathit{Var} \to 2^f$ an assignment, and $x \in f$. For a formula φ in $\mathit{Olt}(F)$ $\mathfrak{M} \models \varphi$ is defined by induction on φ.

$$\begin{array}{lcl}
\langle \mathfrak{T}, \beta, x \rangle \models \top & \Leftrightarrow & \text{true} \\
\langle \mathfrak{T}, \beta, x \rangle \models p & \Leftrightarrow & x \in \beta(p) \\
\langle \mathfrak{T}, \beta, x \rangle \models \mathsf{f} & \Leftrightarrow & \mathsf{f} \in \xi(x) \\
\langle \mathfrak{T}, \beta, x \rangle \models \neg\varphi & \Leftrightarrow & \langle \mathfrak{T}, \beta, x \rangle \not\models \varphi \\
\langle \mathfrak{T}, \beta, x \rangle \models \varphi \wedge \psi & \Leftrightarrow & \langle \mathfrak{T}, \beta, x \rangle \models \varphi; \psi \\
\langle \mathfrak{T}, \beta, x \rangle \models \langle \alpha \rangle \varphi & \Leftrightarrow & \text{there is } y \text{ such that } \langle x, y \rangle \in R(\alpha) \text{ and } \langle \mathfrak{T}, \beta, y \rangle \models \varphi
\end{array}$$

This means in detail that

$$\begin{array}{lcl}
\langle \mathfrak{T}, \beta, x \rangle \models \overset{\rightarrow}{\Diamond}\varphi & \Leftrightarrow & \text{there is } y \sqsupset x \text{ such that } \langle \mathfrak{T}, \beta, y \rangle \models \varphi \\
\langle \mathfrak{T}, \beta, x \rangle \models \overset{\leftarrow}{\Diamond}\varphi & \Leftrightarrow & \text{there is } y \sqsubset x \text{ such that } \langle \mathfrak{T}, \beta, y \rangle \models \varphi \\
\langle \mathfrak{T}, \beta, x \rangle \models \overset{\uparrow}{\Diamond}\varphi & \Leftrightarrow & \text{there is } y \prec x \text{ such that } \langle \mathfrak{T}, \beta, y \rangle \models \varphi \\
\langle \mathfrak{T}, \beta, x \rangle \models \overset{\downarrow}{\Diamond}\varphi & \Leftrightarrow & \text{there is } y \succ x \text{ such that } \langle \mathfrak{T}, \beta, y \rangle \models \varphi \\
\langle \mathfrak{T}, \beta, x \rangle \models [\alpha;\beta]\varphi & \Leftrightarrow & \langle \mathfrak{T}, \beta, x \rangle \models [\alpha][\beta]\varphi \\
\langle \mathfrak{T}, \beta, x \rangle \models [\alpha \cup \beta]\varphi & \Leftrightarrow & \langle \mathfrak{T}, \beta, x \rangle \models [\alpha]\varphi; [\beta]\varphi \\
\langle \mathfrak{T}, \beta, x \rangle \models [\alpha^*]\varphi & \Leftrightarrow & \langle \mathfrak{T}, \beta, x \rangle \models \varphi; [\alpha]\varphi; [\alpha^2]\varphi; \ldots
\end{array}$$

Furthermore, $\langle \mathfrak{T}, \beta \rangle \models \varphi$ if $\langle \mathfrak{T}, \beta, x \rangle \models \varphi$ for all $x \in T$, and $\mathfrak{T} \models \varphi$ if $\langle \mathfrak{T}, \beta \rangle \models \varphi$ for all valuations β. Given a class $\mathfrak{X}$ of generalized Kripke–structures,

$$\mathsf{Th}\,\mathfrak{X} := \{\varphi \in \mathit{Olt}(F) : (\forall \mathfrak{T} \in \mathfrak{X})(\mathfrak{T} \models \varphi)\}$$

The logic of F–trees is denoted by $\mathbf{CL}(F)$ ($\mathbf{BCL}(F)$, $\mathbf{WCL}(F)$). The reader should be aware of the fact that these logics may admit models which are not based on F–trees. Given a logic Λ, we denote by $\mathsf{Mod}(\Lambda)$ ($\mathsf{Krp}(\Lambda)$, $\mathsf{FKrp}(\Lambda)$) the set of model structures (Kripke–structures, finite Kripke–structures) for Λ. (Notice that generally we do not need to know from what language Λ is drawn.) Even though some of our definitions are parametric in the choice of the language, we will often suppress that dependency unless it is relevant. Also, we speak of an F–**logic** when we mean an extension of the logic of F–trees in one of the languages under consideration. The logics of n–ary branching trees are obtained by adding the (constant) axiom $\overset{\rightharpoonup}{\Box}^n\bot$.

With a modal logic Λ we associate the following consequence relations. $\Phi \Vdash_\Lambda \phi$ if ϕ can be deduced from Φ and the theorems of Λ by means of modus ponens and the rule $\chi/\Box\chi$, where $\Box$ is any box–like modal operator. (So, in $Olt(F)$ this rule takes the form $\chi/[\alpha]\chi$, where α is a program.) Furthermore, we write $\Phi \vdash_\Lambda \varphi$ if φ can be derived from Φ and the theorems of Λ by modus ponens alone. We call $\vdash_\Lambda$ the *local consequence relation* and $\Vdash_\Lambda$ the *global consequence relation* of Λ. The following holds. $\Phi \vdash_\Lambda \varphi$ iff for all $\mathfrak{T} \in \mathsf{Mod}(\Lambda)$, all valuations β and $x \in T$, if $\langle \mathfrak{T}, \beta, x\rangle \models \Phi$ then $\langle \mathfrak{T}, \beta, x\rangle \models \varphi$. $\Phi \Vdash_\Lambda \varphi$ if for all $\mathfrak{T} \in \mathsf{Mod}(\Lambda)$ and all valuations β, if $\langle \mathfrak{T}, \beta\rangle \models \Phi$ then $\langle \mathfrak{T}, \beta\rangle \models \varphi$. The local deducibility relation has a deduction theorem, that is, for all formulae φ, ψ and sets of formulae Φ

$$\Phi; \varphi \vdash_\Lambda \psi \qquad \Leftrightarrow \qquad \Phi \vdash_\Lambda \varphi \to \psi$$

A logic Λ is *complete* with respect to a class $\mathfrak{X}$ of frames if $\Lambda = \mathsf{Th}\,\mathfrak{X}$; that is, $\psi \in \Lambda$ exactly when $\mathfrak{T} \models \psi$ for all $\mathfrak{T} \in \mathfrak{X}$. Λ has the *finite model property* of $\Lambda = \mathsf{Th}(\mathsf{FKrp}(\Lambda))$. Λ is *decidable* if there is an algorithm solving for each φ the problem '$\varphi \in \Lambda$' (which is the same as '$\vdash_\Lambda \varphi$'). A logic that has the finite model property and is finitely axiomatizable is decidable.

Say that a logic Λ has the *global finite model property* if whenever $\Phi \nVdash_\Lambda \varphi$ for finite Φ then for some finite $\mathfrak{T} \in \mathsf{Mod}(\Lambda)$ and some valuation β on $\mathfrak{T}$ we have $\langle \mathfrak{T}, \beta\rangle \models \Phi$ but $\langle \mathfrak{T}, \beta\rangle \not\models \varphi$. And say that Λ is *globally decidable* if for finite Φ, the question '$\Phi \Vdash_\Lambda \varphi$' is decidable. If a logic has global finite model property (is globally decidable) then it has the finite model property (is decidable).

2.3 Axiomatizing the Logics of Trees

Let us turn now to the axiomatization of the logic of F–trees. Consider the logic $BCL(F)$, obtained by adding the following axioms to $\mathbf{K}_4$:

(a)	$p \to \overset{\rightharpoonup}{\Box}\overset{\leftharpoonup}{\Diamond}p$	(b)	$p \to \overset{\leftharpoonup}{\Box}\overset{\rightharpoonup}{\Diamond}p$
(c)	$p \to \overset{\uparrow}{\Box}\overset{\downarrow}{\Diamond}p$	(d)	$p \to \overset{\downarrow}{\Box}\overset{\uparrow}{\Diamond}p$
(e)	$\overset{\rightharpoonup}{\Diamond}p \to \overset{\rightharpoonup}{\Box}p$	(f)	$\overset{\leftharpoonup}{\Diamond}p \to \overset{\leftharpoonup}{\Box}p$
(g)	$\overset{\uparrow}{\Diamond}p \to \overset{\uparrow}{\Box}p$	(h)	$\overset{\downarrow}{\Diamond}\overset{\uparrow}{\Diamond}p \to p$
(i)	$\overset{\downarrow}{\Diamond}\overset{\rightharpoonup}{\Diamond}p \to \overset{\downarrow}{\Diamond}p$	(k)	$\overset{\downarrow}{\Diamond}\overset{\leftharpoonup}{\Diamond}p \to \overset{\downarrow}{\Diamond}p$
(l)	$\overset{\downarrow}{\Diamond}\top \to \overset{\downarrow}{\Diamond}\overset{\rightharpoonup}{\Box}\bot \wedge \overset{\downarrow}{\Diamond}\overset{\leftharpoonup}{\Box}\bot$	(m)	$\overset{\uparrow}{\Box}\bot \to \overset{\rightharpoonup}{\Box}\bot \wedge \overset{\leftharpoonup}{\Box}\bot$

We note first that the logic of the set of finite F–trees is not axiomatizable in $BOlt(F)$. This is so because the one point structure in which all relations are reflexive is a model for the theory of F–trees. Furthermore, $\mathbf{BCL}(F)$ is Sahlqvist, hence canonical and complete with respect to (possibly infinite) Kripke structures. However, it is already complete with respect to finite F–trees. For let $\varphi \notin \mathbf{BCL}(F)$. Then there is a $\mathfrak{T}$, a Kripke frame for $\mathbf{BCL}(F)$, a valuation β and a point x such that $\langle \mathfrak{T}, \beta, x\rangle \models \neg\varphi$. $\mathfrak{T}$ can be unravelled from x into a (possibly infinite) F–tree $\mathfrak{U}$. Let d be the modal depth of φ, and let y be a point such that x can be reached from y in exactly d steps, going down. It is not hard to show that we can select a finite subtree $\mathfrak{U}$ of $\mathfrak{T}$ of depth at most $2d$ and with root y such that $\mathfrak{U} \models \mathbf{BCL}(F)$ and $\langle \mathfrak{U}, \beta, x \rangle \models \neg\varphi$.

Proposition 1. *The logic* $\boldsymbol{BCL}(F)$ *is complete with respect to finite* F*–trees.*

This theorem is also a direct consequence of Theorem 1.

$\mathbf{WCL}(F)$ (and $\mathbf{CL}(F)$) is obtained from $\mathbf{BCL}(F)$ by adding the following axioms in addition to the axioms for the basic normal logic (here, $\Box^+ := \Box\Box^*$)

$$
\begin{array}{llll}
(n) & \boxup^+(\boxup^+ p \to p) \to \boxup^+ p & (o) & \boxdown^+(\boxdown^+ p \to p) \to \boxdown^+ p \\
(p) & \boxright^+(\boxright^+ p \to p) \to \boxright^+ p & (q) & \boxleft^+(\boxleft^+ p \to p) \to \boxleft^+ p \\
 & & & \\
(r) & \diamondup \diamonddown p \leftrightarrow \diamondright^* p \vee \diamondleft^* p & & \\
(t) & \diamondup^* \boxup \bot & (t) & \diamonddown^* \boxdown \bot \\
(u) & \diamondright^* \boxright \bot & (v) & \diamondleft^* \boxleft \bot
\end{array}
$$

This axiomatization is not independent.

Theorem 1 ((Blackburn & Meyer–Viol & de Rijke)). $\boldsymbol{WCL}$*(F) is complete with respect to finite* F*–trees.*

There are two consequences of this theorem that are worth noting. The global and the local consequence relation are closely interrelated, since it is possible in $\mathbf{WCL}(F)$ and $\mathbf{CL}(F)$ to define the universal modality of Goranky & Passy [3], denoted here by $\boxtimes$.

Proposition 2 ((Goranko & Passy)). *Define* $\boxtimes$ *by* $\boxtimes\varphi := \boxup^* \boxdown^* \varphi$. *Assume that* $\Lambda \supseteq$ ***(W)CL***(F). *Then*

$$\Phi \Vdash_\Lambda \phi \qquad \Leftrightarrow \qquad \boxtimes\Phi \vdash_\Lambda \phi$$

Hence if Λ *has the finite model property (is decidable) it has the global finite model property (is globally decidable).*

Proof. Let $\Phi \Vdash_\Lambda \varphi$. Assume that $\langle \mathfrak{T}, \beta, x \rangle \models \boxtimes\Phi$ for an F–tree $\mathfrak{T}$. Then for all $y \in T$, $\langle \mathfrak{T}, \beta, y \rangle \models \Phi$, and so $\langle \mathfrak{T}, \beta \rangle \models \Phi$, by definition. Hence $\langle \mathfrak{T}, \beta \rangle \models \varphi$ and so $\langle \mathfrak{T}, \beta, x \rangle \models \varphi$. This shows $\boxtimes\Phi \vdash_\Lambda \varphi$. Now assume that $\boxtimes\Phi \vdash_\Lambda \varphi$. Then obviously $\boxtimes\Phi \Vdash_\Lambda \varphi$. Moreover, $\Phi \Vdash_\Lambda \boxtimes\Phi$ and so $\Phi \Vdash_\Lambda \varphi$.

This means that the global consequence relation is reducible to the global consequence relation in the present context. The next theorem immediately follows.

Corollary 1. $\boldsymbol{WCL}(F)$ *has the (global) finite model property and is (globally) decidable.*

We can use Theorem 1 to prove a theorem announced in Kracht [4].

Theorem 2. $\boldsymbol{CL}$*(F) is complete with respect to finite trees. Hence* $\boldsymbol{CL}$*(F) has the (global) finite model property and is (globally) decidable.*

Proof. It suffices to show that $\mathbf{CL}(F)$ has the finite model property. Moreover, we assume $F = \emptyset$. This simplifies the notation somewhat. Let ψ be a formula of $Olt(F)$. We associate a formula $\nabla(\psi)$ with ψ as follows. For each χ in the Fischer–Ladner–Closure of ψ we pick a new variable q_χ. Then $\nabla(\psi)$ is the conjunction of the following formulae

$$\begin{array}{llllll}
q_p & \leftrightarrow & p & q_{\neg\chi} & \leftrightarrow & \neg q_\chi \\
q_{\chi\vee\omega} & \leftrightarrow & q_\chi \vee q_\omega & q_{\chi\wedge\omega} & \leftrightarrow & q_\chi \wedge q_\omega \\
q_{\langle\chi?\rangle\omega} & \leftrightarrow & q_{\chi\wedge\omega} & q_{\langle\alpha\cup\beta\rangle\chi} & \leftrightarrow & q_{\langle\alpha\rangle\chi} \vee q_{\langle\beta\rangle\chi} \\
q_{\langle\alpha;\beta\rangle\chi} & \leftrightarrow & q_{\langle\alpha\rangle\langle\beta\rangle\chi} & q_{\langle\alpha^*\rangle\chi} & \leftrightarrow & q_\chi \vee q_{\langle\alpha;\alpha^*\rangle\chi} \\
 & & & & & \\
q_{\langle\alpha\rangle\chi} & \leftrightarrow & \langle\alpha\rangle q_\chi & \multicolumn{3}{l}{\alpha \in \{\mathit{up}, \mathit{down}, \mathit{left}, \mathit{right}\}}
\end{array}$$

Notice that $\nabla(\psi) \in$ *WOlt(F)*. It is not hard to see that for a model $\langle\mathfrak{T}, \beta\rangle$ based on an F–tree $\mathfrak{T}$, if $\langle\mathfrak{T}, \beta\rangle \models \nabla(\psi)$ then for any formula χ in the Fischer–Ladner–Closure of ψ, $\langle\mathfrak{T}, \beta\rangle \models \chi \leftrightarrow q_\chi$. Hence the following holds

$$\vdash_{\mathbf{CL}} \psi \qquad \Leftrightarrow \qquad \nabla(\psi) \Vdash_{\mathbf{CL}} q_\psi$$

Furthermore, for $\Phi \subseteq$ *WOlt(F)* and $\psi \in$ *WOlt(F)* we have

$$\Phi \Vdash_{\mathbf{CL}} \psi \qquad \Leftrightarrow \qquad \Phi \Vdash_{\mathbf{WCL}} \psi$$

From right to left is immediate, since $\mathbf{CL}$(F) extends $\mathbf{WCL}$(F). Moreover, suppose the right hand side fails. Then there is a model $\mathfrak{M} = \langle\mathfrak{T}, \beta\rangle$ based on a finite tree $\mathfrak{T}$ such that $\mathfrak{M} \models \Phi$ but $\mathfrak{M} \not\models \psi$. However, $\mathfrak{M}$ also is a model for $\mathbf{CL}(F)$, and so the left hand side fails as well. So, $\vdash_{\mathbf{CL}} \psi$ iff $\nabla(\psi) \Vdash_{\mathbf{WCL}} q_\psi$. Finally, by Proposition 2,

$$\nabla(\psi) \Vdash_{\mathbf{WCL}} q_\psi \qquad \Leftrightarrow \qquad \vdash_{\mathbf{WCL}} \boxtimes\nabla(\psi) \to q_\psi$$

Putting these three together we get that $\vdash_{\mathbf{CL}} \psi$ iff $\vdash_{\mathbf{WCL}} \boxtimes\nabla(\psi) \to q_\psi$. Now suppose that the last fails. Then there exists a finite model $\langle\mathfrak{T}, \beta, x\rangle \models \boxtimes\nabla(\psi); \neg\psi$. Then $\langle\mathfrak{T}, \beta, x\rangle \models \neg\psi$. Moreover, $\mathfrak{T}$ is a model for $\mathbf{CL}(F)$.

Theorem 3. *Let Φ be a finite set of variable–free formulae of Olt(F), and let $\Lambda := \boldsymbol{Olt}(F) \oplus \Phi$. Then Λ has the (global) finite model property and is (globally) decidable.*

Proof. (The argument is given in Kracht [4]. Therefore we will just sketch it.) We may work with the case $\Phi = \{\varphi\}$ (for example, let φ be the conjunction of all members of Φ). Now notice that the following holds for all ψ

$$\Vdash_\Lambda \psi \qquad \Leftrightarrow \qquad \varphi \Vdash_{\mathbf{CL(F)}} \psi \qquad \Leftrightarrow^{\ddagger} \qquad \vdash_{\mathbf{CL(F)}} \boxtimes\varphi \to \psi$$

(where ‡ follows from Proposition 2). Now the theorem follows from Theorem 1.

3 Syntactic Codes

3.1 Boolean Grammars

Contrary to the usual definition in the theory of formal languages we will distinguish between the grammar and the lexicon. A language is generated by a pair consisting of a grammar and a lexicon. Recall that a language over a set D is simply a subset of D^*.

Definition 1. *Let D and F be sets. An F–***lexicon** *for D is a pair $\langle D, \gamma\rangle$, where $\gamma : D \to \wp(F)$. D is called the* **dictionary** *of $\mathbb{L}$ and γ the* **class assignment function**. [5]

The concept of a (context free) grammar for F–trees is defined as follows.

Definition 2. *A* **context free F–grammar** *is a triple $\mathbb{G} = \langle \Sigma, \Omega, R\rangle$ where Σ and Ω are boolean F–terms, called the* **start term** *and the* **stop term**, *and $R \subset \mathfrak{Tm}_{Boo}(F) \times \mathfrak{Tm}_{Boo}(F)^+$ a finite set, the set of* **rules**. *It is required that every term in a rule from R is consistent. Rules are as usual written $\mathsf{x} \to \mathsf{y}_0 \ldots \mathsf{y}_{k-1}$. An F–tree $\mathfrak{T} = \langle T, \prec, \sqsubset, \xi\rangle$ is* **generated by** *$\mathbb{G}$, in symbols $\mathbb{G} \gg \mathfrak{T}$, if (i.) for the root x: $\chi(\xi(x)) \leq \Sigma$, (ii.) for all leaves y: $\chi(\xi(y)) \leq \Omega$, and (iii.) for all non–leaves z, if $z \succ y_i$, $i < k$, and $y_i \sqsubset y_j$ iff $i < j < k$, then there exists $\mathsf{a} \to \mathsf{b}_0 \ldots \mathsf{b}_{k-1} \in R$ such that $\chi(\xi(z)) \leq \mathsf{a}$ and $\chi(\xi(y_i)) \leq \mathsf{b}_i$ for all $i < k$.*

Boolean grammars manipulate only nonterminal symbols in the usual sense of the word. Hence, Ω should not be thought of as a set of nonterminals, but as a description of the lexical nodes. A *grammar* for an actual language is therefore a pair $\langle \mathbb{G}, \mathbb{L}\rangle$, where $\mathbb{G}$ is an F–grammar and $\mathbb{L}$ an F–lexicon.

This split into grammar and lexicon will turn out to be crucial. Given an F–tree $\mathfrak{T}$ and $\boldsymbol{a} = \langle a_i : i < k\rangle \in D^*$, let $\boldsymbol{a} \in \mathsf{Y}(\langle \mathfrak{T}, \mathbb{L}\rangle)$ if the leaves of $\mathfrak{T}$ are x_i, $i < k$, and $x_i L x_j$ whenever $i < j$, and for all $i < k$ we have $\gamma(a_i) = \xi(x_i)$. $\mathsf{Y}(\langle \mathfrak{T}, \mathbb{L}\rangle)$ is the set of strings modulo γ represented by $\mathfrak{T}$. For a set S of F–strings we put

$$\mathsf{Lang}(\langle S, \mathbb{L}\rangle) := \bigcup_{\mathfrak{T} \in S} \mathsf{Y}(\langle \mathfrak{T}, \mathbb{L}\rangle)$$

For a grammar $\mathbb{G}$, we let

$$\mathsf{Lang}(\langle \mathbb{G}, \mathbb{L}\rangle) := \bigcup_{\mathbb{G} \gg \mathfrak{T}} \mathsf{Y}(\langle \mathfrak{T}, \mathbb{L}\rangle)$$

We will give two examples of boolean grammars. Both will be needed later. In both cases we do not use the boolean nature of the labels explicitly in the notation, to keep matters simple. Distinct symbols denote distinct atoms of the free boolean algebra generated by the features.

[5] For technical convenenience we allow no lexical ambiguity. Also, each $a \in D$ has a unique syntactic category, and all categories are mutually exclusive. See also Section 5.

A Grammar for Movement This grammar generates the language $M := ab^* \cup b^*a$. In transformational terms this language can be generated by writing a right regular grammar that generates b^*a, and then having an optional movement process that 'topicalizes' a. However, there is a regular grammar that generates this language without movement. Put $\Sigma := \mathsf{x}$, $\Omega := \mathsf{a} \vee \mathsf{b}$ and let the set R of rules be

$$\begin{array}{lcll}
\mathsf{x} & \to & \mathsf{a} & \mathsf{y} \\
\mathsf{x} & \to & \mathsf{a} & \\
\mathsf{x} & \to & \mathsf{b} & \mathsf{z} \\
\mathsf{y} & \to & \mathsf{b} & \mathsf{y} \\
\mathsf{y} & \to & \mathsf{b} & \\
\mathsf{z} & \to & \mathsf{b} & \mathsf{z} \\
\mathsf{z} & \to & \mathsf{a} &
\end{array}$$

Now $\mathbb{M} := \langle \Sigma, \Omega, R \rangle$. The lexicon is $\langle \{a, b\}, \gamma \rangle$, where $\gamma(a) = \{\mathsf{a}\}$ and $\gamma(b) = \{\mathsf{b}\}$. For let the first rule apply. Then we generate $\mathsf{a} \cdot \mathsf{y}$. It is easy to see that $\mathsf{y} \to^* \mathsf{b}^+$. If the second rule applies, the string generated is a alone. Now assume that the third rule is applied. Then we get the string $\mathsf{b} \cdot \mathsf{z}$. Moreover, $\mathsf{z} \to^* \mathsf{b}^* \cdot \mathsf{a}$. Thus, the grammar generates the preterminal strings of the forms $\mathsf{a} \cdot \mathsf{b}^* \cup b^* \cdot \mathsf{a}$. Hence, with the lexicon as given, the language $ab^* \cup b^*a$ is generated, as promised.

A Grammar for Reflexives The next grammar is a simplified grammar for illustrating the behaviour of reflexives. It generates the language

$$R := \{a[d^* \cup c^+d(c \cup d)^* \cup d^+c(c \cup d)^*]\}^*a$$

This language can be more succinctly described as follows. It generates a string $\boldsymbol{x}$ exactly if (i) $\boldsymbol{x}$ begins and ends with a and (ii) for any occurrence of c, a d must also occur in between the occurrence of c and the next a either to the left or to the right. To phrase condition (ii) into linguistic terminology suppose that this languages is generated by a right regular grammar. Then we may say that each occurrence of a c must a–command an occurrence of d that a–commands the given c. Here, x *a–commands* y if for all nodes z properly dominating x and also dominating an a, z dominates y. To see the connection with reflexives, think of a as being a complementizer, of c as being a reflexives, and of d as an antecedent. With some generosity we see that what is encoded is the requirement that a reflexive only occurs in a sentence that also contains an antecedent. (We trust that reader can see this. A sufficiently realistic grammar would be too complicated for the present paper.) The set S of rules is the following.

$$\begin{array}{lcll}
\mathsf{u} & \to & \mathsf{a} & \\
\mathsf{u} & \to & \mathsf{a} & \mathsf{u} \vee \mathsf{q} \vee \mathsf{v} \vee \mathsf{w} \vee \mathsf{x} \\
\mathsf{q} & \to & \mathsf{a} & \mathsf{q} \vee \mathsf{u} \\
\mathsf{v} & \to & \mathsf{d} & \mathsf{v} \vee \mathsf{u}
\end{array}$$

$$\begin{array}{rcll@{\qquad\qquad}rcll}
\mathsf{w} \vee \mathsf{y} & \to & \mathsf{d} & \mathsf{y} & \mathsf{x} \vee \mathsf{r} & \to & \mathsf{c} & \mathsf{r} \\
\mathsf{y} & \to & \mathsf{c} & \mathsf{z} \vee \mathsf{u} & \mathsf{r} & \to & \mathsf{c} & \mathsf{s} \vee \mathsf{u} \\
\mathsf{z} & \to & \mathsf{c} \vee \mathsf{d} & \mathsf{z} \vee \mathsf{u} & \mathsf{s} & \to & \mathsf{c} \vee \mathsf{d} & \mathsf{s} \vee \mathsf{u}
\end{array}$$

$\Sigma := \mathsf{x}$, $\Omega := \mathsf{a} \vee \mathsf{c} \vee \mathsf{d}$. $\mathbb{R} := \langle \Sigma, \Omega, S \rangle$. The lexicon is $\langle \{a, c, d\}, \kappa \rangle$ where $\kappa(a) = \{\mathsf{a}\}$, $\kappa(c) = \{\mathsf{c}\}$ and $\kappa(d) = \{\mathsf{d}\}$. (To see that this grammar generates R, let U be the language produced by u. Observe that q produces $a^+ \cdot U$, v produces $d^+ \cdot U$, w produces $d^+c(c \cup d)^* \cdot U$ and x produces $c^+d(c \cup d)^* \cdot U$. Let $P := d^+ \cup d^+c(c \cup d)^* \cup c^+d(c \cup d)^*$ and $Z = a^+ \cup P$. Then we get $U = a \cup (a[Z \cup \{\epsilon\}])U$. So, $U = (a[Z \cup \{\epsilon\}])^* a$. We may actually rewrite this as $(aP)^* a$. So, $U = R$, as required.)

3.2 Quasi Context Free Sets

We say that a set S of finite F–trees is a context–free set if there exists an F–grammar $\mathbb{G}$ such that $S = \{\mathfrak{T} : \mathbb{G} \gg \mathfrak{T}\}$. Now, given a rule $\rho = \mathsf{x} \to \mathsf{y}_0 \ldots \mathsf{y}_{k-1}$ we define

$$\gamma(\rho) := \mathsf{x} \to \Diamond(\neg\Diamond\top \wedge \mathsf{y}_0 \wedge \Diamond(\mathsf{y}_1 \wedge \Diamond(\mathsf{y}_2 \wedge \ldots \wedge \Diamond(\mathsf{y}_{k-1} \wedge \neg\Diamond\top)\ldots)))$$

Given an F–grammar $\mathbb{G} = \langle \Sigma, \Omega, R \rangle$ let

$$\gamma(\mathbb{G}) := (\neg\Diamond\top \to \Sigma) \wedge (\neg\Diamond\top \to \Omega) \wedge \bigvee_{\rho \in R} \Diamond\top \to \gamma(\rho)$$

We call $\gamma(\mathbb{G})$ the **characteristic formula** of $\mathbb{G}$. It is not hard to see that for a finite F–tree $\mathfrak{T}$, $\mathbb{G} \gg \mathfrak{T}$ iff $\mathfrak{T} \models \gamma(\mathbb{G})$. The following is now immediate.

Proposition 3. *Let S be a context–free set of finite n-branching F–trees. Then $\mathsf{Th}\, S$ is axiomatizable over $\boldsymbol{CL}_n(F)$ by formula in $BOlt(F)$.*

The converse need not hold. For a characterization of the language in which only context–free sets can be defined see Rogers [9].

Let $G \subseteq F$ and $\mathfrak{T} = \langle T, \prec, \sqsubset, \xi, \mathbb{T} \rangle$. Define the **projection** $\mathfrak{T}_G$ of a $\mathfrak{T}$ onto G by $\mathfrak{T}_G := \langle T, \prec, \sqsubset, \zeta, \mathbb{T} \rangle$, where $\zeta(x) := \xi(x) \cap G$. Likewise, the projection S_G of a class S of F–structures is defined by $S_G := \{\mathfrak{T}_G : \mathfrak{T} \in S\}$.

Definition 3. *Let S be a set of finite F–trees. S is called* **quasi context–free** *if there exists a finite set G and a context–free set T of $F \cup G$–trees such that $S = T_F$.*

There is a characterization of quasi context–free sets of finite F–trees in Rogers [8] and Kracht [5]. This characterization is given in terms of logic.

Grammars define well–formed sets of structures. In our case we assume that these structures are finite F–trees, where F is an arbitrary but fixed set of features. It follows that a grammar may be viewed as a logic. This view has been defended in Kracht [4]. It is not entirely unproblematic for the reason that logics typically admit infinite structures while grammars do not generate such structures. We will see here that the logics we will be studying in connection with context–free grammars are complete with respect to their finite structures, so that the infinite structures — though not excluded by the logics — can be ignored in practice. This is similar to non–standard models of the real line. The difference between a logic and a grammar is roughly the following. A logical system is just a description of the legitimate objects with no indication of how to obtain one, while a generating device allows to obtain just the right set of structures with no indication of their characteristic properties apart from the immediate ones. For a generating device it is therefore not immediate how to recognize the structures that it produces, though in the case of context sensitive language such recognizing algorithms can easily be obtained (though they may not be efficient, but that is another matter). It would be preferrable if one had a method to mediate between generative systems on the one hand and descriptive systems on the other. Such a method has been proposed in Kracht [5]. It shows how to construct a grammar from a description. These descriptions may not contain variables, however, and the outcome is always a context–free grammar. Though that is known to be a restriction, since natural languages are not necessarily context–free, we will deal with that case throughout this paper.

Theorem 4 (Coding Theorem). *Let Λ be an axiomatic expansion of $\boldsymbol{CL}_n(F)$. Λ is the logic of a quasi context–free set iff $\Lambda = \boldsymbol{CL}_n(F) \oplus \Phi$ for a finite set of variable–free formulae from $Olt(F)$. Moreover, there is an algorithm which, given Φ, computes a set G, formulae $\varphi_g \in Olt(F)$ for $g \in G$, and an $F \cup G$–grammar $\mathbb{G}$ such that (i) the set of finite structures of Λ is exactly the projection to F of the set of $F \cup G$–trees generated by $\mathbb{G}$ and (ii) a $F \cup G$–tree $\mathfrak{T}$ is generated by Λ iff $\mathfrak{T}_F$ is a Λ–structure and $\mathfrak{T} \models \mathsf{g} \leftrightarrow \varphi_g$ for each $g \in G$.*

The theorem expresses not only that given a logic axiomatized by constant formulae there exists a context free grammar over an expanded set of features that generates a set T of which the set of finite Λ–models is a projection. It also says that we can compute formulae that explicitly tell us how the additional features are distributed with respect to the set of original features.

4 Inessential and Eliminable Features

4.1 Inessential Features

In intuitive terms, a feature (by which we mean a boolean constant in this connection) is called *inessential* if its distribution is fixed by the other features. To give an example, [slash : np] is inessential, for it can be reconstructed from a tree in which it is not present. (This claim, however, is not trivial and depends on assumptions on movement in syntax.)

Definition 4. *Let S be a set of F–trees and $G \subseteq F$. Put $H := F - G$. G is* **inessential in** *S if for every $\mathfrak{T} \in S_H$ there exists exactly one $\mathfrak{U} \in S$ such that $\mathfrak{T} = \mathfrak{U}_H$.*

(This definition can obviously be generalized to general structures.) Thus, given an arbitrary set of F–trees, and a set G of features, S_H is the projection of S onto the complement of G in F. If $\mathfrak{T} \in S_H$ then, by definition of S_H, there exists at least one $\mathfrak{U}$ of which $\mathfrak{T}$ is the projection. G is inessential in S if there exists no two F–trees of which $\mathfrak{T}$ is the projection, for all $\mathfrak{T} \in S_H$. This means that the features of H alone suffice to identify a tree in S. However, it is by no means clear that given the projection $\mathfrak{T}_H$ of $\mathfrak{T}$ onto H we can produce $\mathfrak{T}$ from $\mathfrak{T}_H$ by some algorithm. In the general case this is impossible. However, in the present discussion we are interested in sets of trees definable by means of axioms. The reader is asked to verify that a set G is inessential in S iff all $\mathsf{g} \in G$ are inessential. Thus, we will often specialize without warning to the case of a single feature rather than a set.

Definition 5. *Let Λ be an F–logic Λ and $G \subseteq F$ a set of features. We say that G is* **inessential** *in Λ if it is inessential in* $\mathsf{Mod}(\Lambda)$.

The notion of being inessential can be rephrased in logical terms using the notion of an *implicit definition.*

Definition 6. *Let Λ be a logic extending $\mathbf{CL}_n(F)$ and let $\varphi(p, \boldsymbol{q})$ be a formula. $\varphi(p, \boldsymbol{q})$ is called a* **global implicit definition** *of p if*

$$\varphi(p, \boldsymbol{q}); \varphi(r, \boldsymbol{q}) \Vdash_\Lambda p \leftrightarrow r$$

In this definition, φ may also contain the constants from F. The next theorem considers the case where p replaces the occurrences of a given boolean $\mathsf{f} \in F$. For the purpose of that theorem $\varphi[p/\mathsf{f}]$ is the result of uniformly replacing each occurrence of f by p.

Proposition 4. *Let $\Lambda = \boldsymbol{CL}(F) \oplus \varphi$ be an F–logic. f is inessential in Λ iff $\varphi[p/\mathsf{f}]$ is a global implicit definition of p in Λ. Moreover, if φ is a constant formula, f is inessential in Λ iff it is inessential in* $\mathsf{FKrp}(\Lambda)$.

Proof. Let f be inessential in Λ. Assume that $\langle \mathfrak{T}, \beta \rangle \models \varphi[p/\mathsf{f}]; \varphi[q/\mathsf{f}]$. Then $\langle \mathfrak{T}, \beta, x \rangle \models p$ exactly if $\langle \mathfrak{T}, \beta, x \rangle \models \mathsf{f}$. Hence $\mathfrak{T} \models \varphi$, which implies that $\mathfrak{T} \in \mathsf{Krp}(\Lambda)$. Furthermore, also $\langle \mathfrak{T}, \beta, x \rangle \models q$ iff $\langle \mathfrak{T}, \beta, x \rangle \models \mathsf{f}$, and so $\langle \mathfrak{T}, \beta \rangle \models p \leftrightarrow q$. This shows that $\varphi[p/\mathsf{f}]$ implicitly defines p. The converse is immediate. For the second claim notice that Λ has the finite model property. This implies among other that $\varphi[p/\mathsf{f}]; \varphi[p/\mathsf{f}] \Vdash_\Lambda p \leftrightarrow q$ iff for every finite model $\mathfrak{T}$ and valuation β, if $\langle \mathfrak{T}, \beta \rangle \models \varphi[p/\mathsf{f}]; \varphi[q/\mathsf{f}]$ then $\beta(p) = \beta(q)$.

Theorem 5. *Let $\Lambda = \boldsymbol{CL}(F) \oplus \varphi$ for a constant formula φ and let $\mathsf{f} \in F$. Then it is decidable whether or not f is essential in Λ.*

Proof. f is essential iff $\varphi[p/\mathsf{f}]; \varphi[q/\mathsf{f}] \Vdash_\Lambda p \leftrightarrow q$. By Proposition 2 this is equivalent to

$$\vdash_\Lambda \boxtimes\varphi[p/\mathsf{f}] \wedge \boxtimes\varphi[q/\mathsf{f}]. \to .(p \leftrightarrow q)$$

Now, Λ is decidable by Theorem 3 and this establishes the claim.

4.2 Eliminable Features

Let Λ be a logic (= grammar) and f an inessential feature. We know then that the distribution of that feature is fixed by the distribution of the other features. Nevertheless, it may not be possible to know in what way it must be distributed.

Definition 7. *Let S be a set of F–structures, and $f \in F$. f is* **eliminable** *in S if there is a formula $\chi(p)$ such that for all $\mathfrak{T} \in S$*

$$\langle \mathfrak{T}, \beta \rangle \models \chi(p) \Leftrightarrow \langle \mathfrak{T}, \beta \rangle \models p \leftrightarrow \mathsf{f}$$

Definition 8. *Let Λ be a logic and $\varphi(p, \boldsymbol{q})$ an implicit definition of p. $\psi(\boldsymbol{q})$ is called a* **(corresponding) explicit definition** *of p in Λ if $\varphi(p, \boldsymbol{q}) \Vdash_\Lambda p \leftrightarrow \psi(\boldsymbol{q})$.*

Definition 9. *An inessential feature f of $\Lambda \supseteq \mathbf{CL}(F)$ is called* **eliminable** *if there exists an explicit definition in Λ.*

Now assume that f is eliminable in Λ. Then it is inessential, and we have

$$\begin{array}{rcl} \Lambda & = & \mathbf{CL}(F) \oplus \varphi \\ & = & \mathbf{CL}(F) \oplus \varphi \oplus \mathsf{f} \leftrightarrow \psi \\ & = & \mathbf{CL}(F) \oplus \varphi[\psi/\mathsf{f}] \oplus \mathsf{f} \leftrightarrow \psi \end{array}$$

Thus, an axiomatization of Λ can be given that uses the structural axiom $\varphi[\psi/\mathsf{f}]$, in which f does no longer occur, plus an explicit axiom $\mathsf{f} \leftrightarrow \psi$ defining the distribution of f. In that case we may simply pass to the language over the set $H := F - \{\mathsf{f}\}$. Define

$$\Lambda^{-f} := \mathbf{CL}(F - \{\mathsf{f}\}) \oplus \varphi[\psi/\mathsf{f}]$$

Then Λ^{-f} axiomatizes the logic of the structures $\mathsf{Mod}(\Lambda)_H$.

Proposition 5. *Let $\Lambda = \boldsymbol{CL}(F) \oplus \varphi$ be an F–logic, and $\mathsf{f} \in F$. Put $H := F - \{\mathsf{f}\}$. Suppose that f is eliminable with explicit definition ψ. Then*

$$\mathsf{Th}(\mathsf{Mod}(\Lambda)_H) = \mathbf{CL}(F) \oplus \varphi[\psi/\mathsf{f}]$$

A good illustration of these concepts is Chomsky [2]. Here it is argued that the additional features distinguishing levels in X–bar syntax are inessential since they can be deduced from the structure with the categorial labels alone. It is not hard to see that the level attributes are also eliminable. However, eliminability is not always guaranteed.

Theorem 6. *There exists a logic Λ axiomatized by constant formulae and an inessential feature which is not eliminable.*

For a proof take the example of Kracht [5] of a logic $\Lambda \supseteq \mathbf{CL}_3(\{\mathsf{f}, \mathsf{g}\})$ which axiomatizes the logic of ternary branching trees such that g is is true along the branches of a binary branching subtree, while f holds at those leaves exactly where g holds. Then g is inessential. For let $\mathfrak{T} = \langle T, \prec, \sqsubset, \xi \rangle$ be an $\{\mathsf{f}\}$–tree.

Then there is at most one way to turn $\mathfrak{T}$ into an $\{\mathsf{f}, \mathsf{g}\}$–tree. Namely, let $\mathfrak{U} := \langle T, \prec, \sqsubset, \zeta\rangle$ such that $\mathsf{f} \in \zeta(x)$ iff $\mathsf{f} \in \xi(x)$ and $\mathsf{g} \in \zeta(x)$ iff there exists a leaf $y \prec^{+} x$ such that $\mathsf{f} \in \xi(y)$. Then if $\mathfrak{T}$ is the projection of a $\{\mathsf{f}, \mathsf{g}\}$–tree $\mathfrak{V}$, $\mathfrak{V} = \mathfrak{U}$. So, g is indeed inessential. It it not hard to come up with a formula $\varphi(p, q)$ which implicitly defines p in the way prescribed. But g is not eliminable. A proof can be found in Kracht [7].

5 On the Descriptive Complexity of Language

5.1 Naturalizing the Feature System

We are going to exemplify the usefulness of these concepts by illustrating how they allow to determine the complexity of languages. This complexity is not measured in terms of time or space complexity bounds for the recognition problem (or other problems) but rather in terms how of complicated it is to describe the facts of the language. We have considered four languages which we will discuss now: the language of boolean expressions, the basic language, the weak language and the (full) language for F–trees. Suppose that the language is given to us a subset of D^*, D the dictionary. We need to introduce a set F of features to begin with. Already here some assumptions must be made.

Definition 10. *Let $S \subseteq D^*$ be a language over D, and let $a \in D$. Put $C_S(a) := \{\langle \boldsymbol{x}, \boldsymbol{y}\rangle \in D^* \times D^* : \boldsymbol{x}\cdot a \cdot \boldsymbol{y} \in S\}$. Call a and b* **syntactically indistinguishable** *if $C_S(a) = C_S(b)$.*

Let F be a set of features. A *class* (*over F*) is a subset of F. A *class assignment function* on D is a function $\gamma : D \to \wp(F)$. A class assignment is *proper* if $C_S(a) = C_S(b)$ implies $\gamma(a) = \gamma(b)$; it is *minimal* if $\gamma(a) = \gamma(b)$ implies $C_S(a) = C_S(b)$. We are interested in proper and minimal class assignments. Proper assignments are such that they assign the same class to syntactically indistinguishable elements. Minimal assignments put all indistinguishables in one class. Given γ and a class $C \subseteq F$, let $E_\gamma(C) = \{a : \gamma(a) = C\}$ and call it the *lexical extension* of the class C. If the lexical extension of C is not empty, we call C a *lexical class.* Now, if there are for example three classes, there must be at least two features. But then we have four classes, so one of the classes has empty lexical extension and is therefore not lexical. Roughly, the *theory* of the lexicon $\langle D, \gamma\rangle$ is the set of all lexical classes. Formally, we put

$$\mathsf{M}(\gamma) := \bigvee_{E_\gamma(C) \neq \emptyset} \chi(C)$$

Given an F–lexicon $\mathbb{L} = \langle D, \gamma\rangle$ and a set $G \subseteq F$, put $\gamma_G(a) := \wp(a) \cap G$ and $\mathbb{L}_G := \langle D, \gamma_G\rangle$.

Definition 11. *Let D be a dictionary, $S \subseteq D^*$ a languange over D and $\mathbb{L} = \langle D, \gamma\rangle$ an F–lexicon. G is a* **natural subset** *of F if $\mathbb{L}_G$ is a proper and minimal class assignment. If G is natural, $\mathbb{L}_G$ is called the* **naturalization** *of $\mathbb{L}$ with respect to S. If $\mathbb{L} = \mathbb{L}_G$, $\mathbb{L}$ is called* **natural** *with respect to S.*

A lexicon may possess different naturalizations with respect to one and the same language. To see this, take two features, f_1 and f_2, and assume that only the classes $\{\mathsf{f}_1\}$ and $\{\mathsf{f}_2\}$ are lexical. This is the case with $S = \{ab\}$, and $\gamma(a) = \{\mathsf{f}_1\}$, $\gamma(b) = \{\mathsf{f}_2\}$. Then both $\{\mathsf{f}_1\}$ and $\{\mathsf{f}_2\}$ are natural subsets. (One can also show that natural subsets need even not be equal size. Namely, the boolean algebra of subsets of $\{a, b, c, d\}$ is generated by $\{\{a\}, \{b\}, \{c\}\}$ and also by $\{\{a, b\}, \{b, c\}\}$.) However, take two natural sets G and H. Then each member of H can be expressed as a boolean term over G, and each member of G by a boolean term over H. Since we generally care about definitions only up to interdefinability, we allow ourselves to speak about *the* natural subset and in particular about *the* naturalization of a lexicon.

5.2 Descriptive Complexity

We are now approaching the definition of a complexity hierarchy for languages. The idea is very simply put the following. We require that the language S be the language of a set U of F–trees. We now try to eliminate all features that are not in the natural set. The complexity of the language is measured in terms of the complexity of the defining formulae for the nonnatural features. To make this absolutely restrictive we require that each constituent has a class that is also lexical. This forbids the features to be used in combinations in which they do not occur in the lexicon.

Definition 12. *Let S be a set of F–trees, $\mathbb{L} := \langle D, \gamma\rangle$ a lexicon and $L :=$ $\mathsf{Lang}(\langle S, \gamma\rangle)$. S is a* **natural set of F–trees with respect to** $\mathbb{L}$*, if $\mathbb{L}$ is a proper and minimal lexicon with respect to L and for all $\langle T, <, \sqsubset, \xi\rangle \in S$ and all nodes x, $\xi(x)$ is a lexical class.*

Definition 13. *Let S be a set of F–trees and $\mathbb{L}$ a lexicon. Let G be a subset of F such that S_G is natural with respect to $\mathbb{L}$. Then $T := S_G$ is the* **naturalization of** S. *Let $\varphi \in Olt(G)$ be such that*

$$\mathsf{Th}\, T = \mathbf{CL}(G) \oplus \mathsf{M}(\gamma) \oplus \varphi$$

Then $\mathbf{CL}(G) \oplus \mathsf{M}(\gamma) \oplus \varphi$ is called the **natural theory** *of S.*

Of course, φ is not uniquely determined. However, it is clear from the definition that the logic $\mathbf{CL}(G) \oplus \mathsf{M}(\gamma) \oplus \varphi$ is uniquely determined by G. Now, if a different natural set is chosen, we get a logic $\mathbf{CL}(H) \oplus \mathsf{M}(\delta) \oplus \psi$, and formulae $\chi_g \in \mathfrak{Tm}_{Boo}(H)$, $g \in G$, $\omega_h \in \mathfrak{Tm}_{Boo}(G)$, $h \in H$, such that

1. $\mathsf{M}(\delta)[\omega_h/\mathsf{h} : h \in H]$ is derivable in $\mathbf{CL}(G) \oplus \mathsf{M}(\gamma)$,
2. $\mathsf{M}(\gamma)[\chi_g/\mathsf{g} : g \in G]$ is derivable in $\mathbf{CL}(H) \oplus \mathsf{M}(\delta)$,
3. $\psi[\omega_h/\mathsf{h} : h \in H]$ is derivable in $\mathbf{CL}(G) \oplus \mathsf{M}(\gamma) \oplus \varphi$,
4. $\varphi[\chi_g/\mathsf{g} : g \in G]$ is derivable in $\mathbf{CL}(H) \oplus \mathsf{M}(\delta) \oplus \psi$.

The two logics are interpretable in each other modulo boolean expressions, and hence the natural theory is unique up to boolean interpretation.

Definition 14. *Let S be a set of F–trees, $\mathbb{L}$ a lexicon and G a natural subset. The* **descriptive complexity** *of S is defined as follows. S has complexity* **pc** *(***b**, **w**, **pdl***) if for a nonnatural feature f, there exists a formula $\varphi(p)$ from $\mathfrak{Tm}_{Boo}(G)$ ($BOlt(G)$, $WOlt(G)$, $Olt(G)$) such that*

$$S_G \models \mathsf{f} \leftrightarrow \varphi[\mathsf{f}/p]$$

The reader may verify that the complexity class does not depend on the choice of G. Moreover, if S is quasi context free and has complexity α where $\alpha \neq pc$, then $\mathsf{Th}\, S$ can be axiomatized by a sentence of complexity α as well. The only exception is *pc*. Here, the sentence is of complexity *b*. For take the charactistic axiom of the context free grammar defined in Section 3.1). Now replace in it the unnatural features by their explicit definitions. This returns a formula of identical complexity. In a last step we define the complexity class of a language.

Definition 15. *Let D be a dictionary, and $L \subseteq D^*$ a language. L has* **complexity** *α if there exists a set of features F, a set S of F–trees, and an F–lexicon $\mathbb{L}$ such that $L = \mathsf{Lang}(\langle S, \gamma\rangle)$ and S has complexity α.*

5.3 Examples

We are giving examples of languages which are of different complexity class. The languages we choose are simplified languages displaying certain phenomena of natural language. Looking at these examples we will demonstrate that natural language has at least complexity *pdl*.

Languages of Complexity pc. Let us begin with the lowest complexity class. By definition, if L is a context free language of complexity *pc*, the defining formulae for the nonnatural features are booleans. That means, nonnatural features serve no structural purpose, and we are in effect dealing with a class of grammars that we call *natural*. They are defined as follows.

Definition 16. *Let $\langle \mathbb{G}, \mathbb{L}\rangle$ be a grammar for the language $S \subseteq D^*$. We say that $\langle \mathbb{G}, \mathbb{L}\rangle$ is a* **natural context free grammar** *if the assignment of the lexicon is proper and minimal, and for the theory of the lexicon $\mathsf{M}(\gamma)$ we have $\Sigma \leq \mathsf{M}(\gamma)$, $\Omega \leq \mathsf{M}(\gamma)$ and for every rule $\mathsf{a} \to \mathsf{b}_0 \ldots \mathsf{b}_{k-1}$, $\mathsf{a} \leq \mathsf{M}(\gamma)$ and $\mathsf{b}_i \leq \mathsf{M}(\gamma)$ for all $i < k$.*

The second part of the definition is equivalent to the requirement that $\mathbb{G} \gg \mathfrak{T}$ only if for every node x, $\xi(x) = C_S(a)$ for some $a \in D$.

Consider the case where $D = \{a\}$. Then there are no natural features. In a natural grammar the rules have the form $\top \to \top \ldots \top$. Hence, a natural context free grammar over the empty set of features is uniquely identifiable by the set of branching numbers for its rules. The reader may verify that the language $\{a\}$ is generated by a natural context free grammar, while $\{a^n\}$ is not, for every $n > 1$. Natural grammars look like a very restricted class of grammars. But they are not. Natural languages are to a large part generable by a natural context free grammar. Simply observe that in many cases a constituent can be replaced in a structure by a single lexical element. The ‘nonnaturalness’ of natural languages is largely induced by longdistance effects (movement, binding) as well as coordination.

Languages of Complexity b. Languages of complexity *b* allow elements to exercise influence over elements that are a specified number of nodes apart. This is the case for example in case assignment or selection of multiple arguments. It can be shown that all finite languages are of complexity *b*; as we have seen, not all of them are of complexity *pc*. To take a more interesting case, let us discuss the language $M := ab^* \cup b^*a$. We know it can be generated by the grammar $\mathbb{M}$ above. However, only the sets $\{\mathsf{a}\}$ and $\{\mathsf{b}\}$ are natural. Let us take $G := \{\mathsf{a}\}$. We have $\mathsf{M}(\gamma) = (\mathsf{a} \wedge -\mathsf{b}) \vee (\mathsf{b} \wedge -\mathsf{a})$. We have $\mathsf{x} \leftrightarrow \Box\bot$, so x is definable in $\mathbf{BCL}(G)$. However, y and z are not definable, as one can at least intuitively see. It does not follow, though, that the language M is not of complexity *b*. To see this, take the following grammar.

$$\begin{array}{lllll}
\mathsf{x} & \rightarrow & \mathsf{a} & \mathsf{b} \\
\mathsf{x} & \rightarrow & \mathsf{b} & \mathsf{a} \\
\mathsf{x} & \rightarrow & \mathsf{a} & \\
\mathsf{b} & \rightarrow & \mathsf{b} & \mathsf{b}
\end{array}$$

$\Sigma := \mathsf{x}$, $\Omega := \mathsf{a} \vee \mathsf{b}$. This grammar generates M, and we have $\mathsf{x} \leftrightarrow \Box\bot$. So M is of complexity *b*, but not of complexity *pc*.

Languages of Complexity w. It is not difficult to see that the language $\{b^n ac^n : n \in \omega\} \cup \{ab^n c^n : n \in \omega\}$ is not of complexity *b*. Take the grammar

$$\begin{array}{llllll}
\mathsf{x} & \rightarrow & \mathsf{a} & & \\
\mathsf{x} & \rightarrow & \mathsf{b} & \mathsf{x} & \mathsf{c} \\
\mathsf{y} & \rightarrow & \mathsf{a} & \mathsf{z} & \\
\mathsf{z} & \rightarrow & \mathsf{b} & \mathsf{z} & \mathsf{c} \\
\mathsf{z} & \rightarrow & \mathsf{b} & \mathsf{c} &
\end{array}$$

$\Sigma := \mathsf{x} \vee \mathsf{y}$, $\Omega := \mathsf{a} \vee \mathsf{b} \vee \mathsf{c}$. Then $\mathsf{z} \leftrightarrow \boxdot^* \neg\mathsf{a}$, $\mathsf{y} \leftrightarrow \Box\top \wedge \Diamond\mathsf{z}$, and $\mathsf{x} \leftrightarrow \Box\bot \wedge \neg\mathsf{y}$. Thus movement in this case can be defined by formulae of complexity *w*. Indeed, we conjecture that movement is in general of complexity *w*, as long as it is into c–commanding position.

Languages of Complexity pdl. Finally, we look at the complexity *pdl*. Take the language *R* of reflexives. This language is regular. To express the distribution of the additional features we claim that formulae of complexity *w* are not sufficient. We sketch the argument. An elementary formula α is modally definable if it is equivalent in predicate calculus to a formula that is composed from positive atomic formulae using conjunction, disjunction and restricted quantifiers. Moreover, the formula can be rewritten in such a way that each subformula contains exactly two free variables (see for example Kracht [7]). To verify that α holds in a structure, one can start a Fraissé–Ehrenfeucht game. Since at every stage of such a game the subformula under consideration contains only two free variables, we can actually check α using a game in which the players play with two pebbles that are placed on the structure and may be moved one at a time in the game along a relation. (Thus the memory is restricted from infinitely many to just two variables.) Now, it is easy to see that for conditions of the form *in between two occurrences of x, if there is a y then there is a z* no winning strategy can be formulated in such a game. For as soon as we have fixed our occurrences of x, we have exhausted our storage capacity. This argument is independent of any structural analysis we assume for the language R. Thus, reflexives require the expressive power of *pdl*.

6 Conclusion

In Rogers [8] and subsequent work, James Rogers has advocated the use of monadic second order logic as a tool in the analysis of language. This gives rise to yet another language to talk about F–trees, which we denote here by $\mathcal{L}_2(F)$. The $\mathcal{L}_2(F)$–logic of the finite trees is denoted by $\mathbf{MSOlt}(F)$. This gives us the following hierarchy of languages

$$BOlt(F) \subset WOlt(F) \subset Olt(F) \subset \mathcal{L}_2(F)$$

$\mathcal{L}_2(F)$ is expressively sufficient if we assume that for some extension $\mathcal{L}_2(F \cup G)$ by nonlexical features all facts can be expressed. Namely, suppose there exists a G and a φ such that the set S of well–formed F–trees is the projection of a $\mathcal{L}_2(F \cup G)$–definable set T. Then $S = \{\mathfrak{T} : \mathfrak{T} \models (\exists \boldsymbol{x})\varphi[\boldsymbol{x}/\boldsymbol{g}])\}$. Hence, the elimination of a feature (whether it be essential or not) is a trivial matter. This is bought at a price, though. We no longer need to know exactly how the features are distributed with respect to the other features in order to know that they are eliminable. Moreover we contend that the language $Olt(F)$ is sufficient in all respects. To that end we note that all relevant locality domains can be expressed in $Olt(F)$. This of course is far away from being a proof. To turn this into a real argument, one needs to investigate quite closely the role of movement in syntax. This has been done in Rogers [8] and Kracht [5]. Both have given explicit reductions to $Olt(F)$ of some theories. Also, in Kracht [6] it is shown how phrasal levels can be eliminated along the lines requested by Chomsky [2]. We need to warn the reader, however, that the preceding discussion makes sense only with the assumption that natural languages are context free. If not, matters are more complex. In order to be able to deal with natural language in its full complexity, we need to assume different classes of structures, more general than F–trees. The notions developed here can be extended to the general case, and we believe that the results also carry over. This, however, awaits further investigation.

References

1. Patrick Blackburn, Wilfried Meyer-Viol, and Maarten de Rijke. A Proof System for Finite Trees. In H. Kleine Büning, editor, *Computer Science Logic '95*, number 1092 in Lecture Notes in Computer Science, pages 86 – 105. Springer, 1996.
2. Noam Chomsky. Bare Phrase Structure. In Gert Webelhuth, editor, *Government and Binding Theory and the Minimalist Program*, pages 385 – 439. Blackwell, 1995.
3. Valentin Goranko and Solomon Passy. Using the universal modality: Gains and Questions. *Journal of Logic and Computation*, 2:5 – 30, 1992.
4. Marcus Kracht. Is there a genuine modal perspective on feature structures? *Linguistics and Philosophy*, 18:401 – 458, 1995.
5. Marcus Kracht. Syntactic Codes and Grammar Refinement. *Journal of Logic, Language and Information*, 1995.
6. Marcus Kracht. On Reducing Principles to Rules. to appear, 1996.
7. Marcus Kracht. Tools and Techniques in Modal Logic. Habilitationsschrift, 1997.
8. James Rogers. *Studies in the Logic of Trees with Applications to Grammar Formalisms.* PhD thesis, Department of Computer and Information Sciences, University of Delaware, 1994.
9. James Rogers. *Strict LT2: regular – Local: recognizable.* this volume, 1997.