1

What feature co-occurrence restrictions have to do with type signatures

Wiebke Petersen, James Kilbury [†]

Abstract

Computational linguistics of the last quarter century has seen the development of a hierarchical and lexicalist treatment of linguistic information. In the course of this development, formal constraints which were stated in GPSG in terms of feature co-occurrence restrictions came to be formulated within HPSG in terms of type hierarchies, but the relations between these descriptive devices has received little attention. Formal Concept Analysis now provides a framework within which these relations can be made explicit.

Keywords GPSG, HPSG, feature co-occurrence restriction, type signature, Formal Concept Analysis

1.1 Introduction

A rapid and remarkable development took place within computational linguistics in the years immediately following the introduction of unification-based models of language, in particular *Lexical Functional Grammar* (LFG) and *Generalized Phrase Structure Grammar* (GPSG), which employ feature structures to represent linguistic information. By the end of the 1980s a consensus had emerged, according to which the

[†]We wish to thank two anonymous reviewers for their comments on an earlier version of this paper. The work was carried out in the project "Modelling Subregularity in the Lexicon" (University of Düsseldorf, Sonderforschungsbereich 282 "Theory of the Lexicon", funded by the Deutsche Forschungsgesellschaft).

FG-MoL 2005.
Organizing Committee:, Gerhard Jaeger, Paola Monachesi, Gerald Penn, James Rogers, Shuly Wintner.

lexicon, which pairs word forms with feature structures, constitutes the main repository of information in a language. Furthermore, hierarchical structuring had come to be viewed as an essential aspect or perhaps even the most salient characteristic of the lexicon.

GPSG, as conceived in Gazdar and Pullum (1982) and even in Gazdar et al. (1985), still largely represents the older, dichotomous view of grammar versus lexicon. Here major aspects of linguistic structure were encoded in *syntactic* rules, many of which later came to be regarded as stating the possible complement structures of verbs, i.e. *lexical* information. While the question "How is a classification imposed on the content of the lexicon by the system of features" is raised (Gazdar et al., 1985, p. 13), the answer of GPSG does not explicitly model the hierarchical inheritance relations inherent in lexical classifications. Rather, these relations are captured in *logical constraints on feature structures* in the form of *feature co-occurrence restrictions* (FCRs) and *feature specification defaults* (FSDs), the latter of which are nonmonotonic.

GPSG uses FCRs to restrict the distribution of features and their values. A pair consisting of a feature and a feature value is called a *feature specification*. Whereas GPSG features are atomic symbols, feature values are either atomic symbols or categories,[1] i.e., sets of feature specifications (Gazdar et al., 1985, p. 22). FCRs are part of a grammatical theory and restrict the set of possible categories and their extensions in the theory. A typical FCR is [+INV] ⊃ [+AUX, FIN] (Gazdar et al., 1985, p. 28), which is [<INV,+>] ⊃ [<AUX,+>, <VFORM,FIN>] when written out fully.[2] The condition stated here is that in English the feature specification <INV,+> implies <AUX,+> and <VFORM,FIN>: if a verb occurs initially in a sentence containing a subject, then this verb must be a finite auxiliary.

From the start the *lexicalist* orientation was prominent in LFG (cf. Bresnan 1982, therein Kaplan and Bresnan 1982) and reached a peak in the radical lexicalism of Karttunen (1986), which uses the framework of categorial grammar to shift the entirety of linguistic description to the lexicon. The move toward the lexicalist view was independent of *hierarchical* modelling, which emerged in other work. In particular, Flickinger (1987) pioneered the explicit description of relations between English verb classes in terms of inheritance hierarchies. On a separate front, de Smedt (1984) initiated the use of inheritance-based representation formalisms to capture the structure of inflectional

[1] Categories of GPSG correspond to the untyped feature structures of other unification-based formalisms.

[2] The feature specification <INV,+> marks sentence-initial verbs, <AUX,+> marks auxiliary verbs, and <VFORM,FIN> specifies that the verb is finite.

classes. Practical advantages of hierarchical lexica quickly became apparent and include the economic representation, integrity, homogeneity, and updating of data. The grammar formalism PATR-II (cf. Shieber et al., 1983) provides *templates* as an indispensable formal device for stating inheritance relations in an inheritance hierarchy, the consequences of which are clear to the authors:

> But our notation does not only allow convenient abbreviations; it also plays an important role in the linguist's use of the formalism. [...] perhaps most importantly, grammar writers can use the notational tools to express generalizations they could not state in the "pure" unification notation of the formalism. (Shieber et al., 1983, p. 62)

Head-driven Phrase Structure Grammar (HPSG), as presented in Pollard and Sag (1987), carries over much of GPSG but restructures the model in terms of the hierarchical lexicalist framework and, in particular, encodes as lexical much of the information that GPSG represented with syntactic rules. The introduction of the *sign* as a uniform data structure for the representation of both lexical and phrasal information, together with the integration of all linguistic levels within the sign, provides the last means needed in order to state all information about a language within an inheritance hierarchy defining relations between signs. In contrast to the *nonmonotonic* inheritance hierarchies of de Smedt, Flickinger, and others, which allow exceptions and defaults, HPSG employs *monotonic* inheritance. The distinction will play no role in the rest of this paper.

Crucially, HPSG adopts no obvious counterpart for the FCRs of GPSG, although the *conditional feature structures* (Pollard and Sag, 1987, p. 43) of HPSG could have been employed for this purpose. Instead, the HPSG strategy for avoiding lexical redundancy lies in the use of inheritance hierarchies: "Structuring the lexicon in terms of an inheritance hierarchy of types has made it possible to factor out information common to many lexical entries, thereby greatly reducing lexical redundancy" (Sag and Wasow, 1999, p. 202). The informational domain consists of typed feature structures. The types serve two functions: On the one hand they allow access to embedded feature structures appearing as values of features; this permits the formulation of generalizations about such substructures. On the other hand, the types bear appropriateness conditions which restrict the set of feature structures of this type; such statements are feature-type pairs. By ordering the types in an inheritance hierarchy, the so-called *type signature*, in which appropriateness conditions are inherited, further redundancies are avoided.

It was clear to linguists that HPSG had replaced the FCRs of GPSG

TABLE 1 Lexemes classified with respect to their feature specifications in Gazdar et al. (1985)

	v:+	v:−	n:+	n:−	vform:bse	vform:fin	vform:pas	aux:+	aux:−	inv:+	inv:−	nform:norm	nform:it	v:VAL	n:VAL	vform:VAL	aux:VAL	inv:VAL	nform:VAL	pform:with	pform:VAL
sing	×			×	×				×		×			×	×	×	×	×			
sings	×			×		×			×		×			×	×	×	×	×			
sung	×			×			×		×		×			×	×	×	×	×			
can1	×			×		×		×		×				×	×	×	×	×			
can2	×			×		×		×			×			×	×	×	×	×			
can3	×			×	×			×			×			×	×	×	×	×			
child		×	×									×		×	×				×		
it		×	×										×	×	×				×		
little	×		×											×	×						
with		×		×										×	×					×	×

with inheritance hierarchies of types, but the relations between these formal devices were misunderstood and hardly questioned. Clearly, the devices had to be related in some way, but no formal framework was available within which the relations could be made explicit. Gerdemann and King (1994, 1993) present a procedural method for transforming a type signature so that it expresses an FCR. With Formal Concept Analysis (FCA, cf. Ganter and Wille 1999) a general framework is now available which allows the equivalence of the devices to be explained in a transparent and declarative fashion, which we shall do after briefly introducing the FCA framework itself.

1.2 Basics of Formal Concept Analysis

FCA is a mathematical theory designed for data analysis. FCA starts with the definition of a *formal context* K as a triple (G, M, I) consisting of a set of *objects* G, a set of *attributes* M, and a binary *incidence relation* $I \subseteq G \times M$ between the two sets. Table 1 shows a formal context in the form of a *cross table*.[3] FCA associates with each formal context a lattice of formal concepts. A *formal concept* of a context is a pair consisting of a set of objects, its *extent*, and a set of attributes, its *intent*. The extent consists exactly of the objects in the context for which all the attributes of the intent apply; the intent consists correspondingly of all attributes of the context which the objects from the extent have in common. In order to formally express the strong connection between the extent and the intent of a formal concept given by the binary relation I, two derivational operators are defined between

[3]The content of this formal context will be explained in Section 1.3.

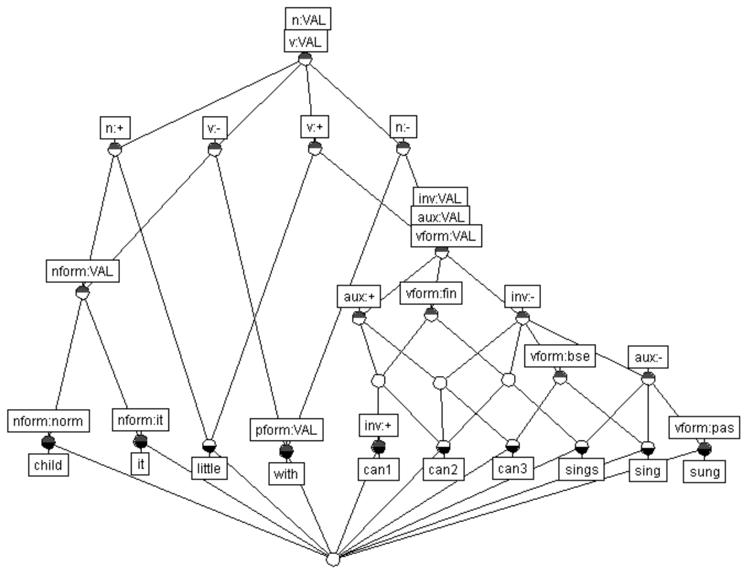

FIGURE 1 The concept lattice for the context of Table 1, which can be regarded as a type signature

the set of objects G and the attribute set M of a formal context: If $A \subseteq G$ is a set of objects, then the set of the common attributes of A is $A' := \{m \in M \,|\, \forall g \in A\colon (g,m) \in I\}$, and if, analogously, $B \subseteq M$ is a set of attributes, then the set of objects that have B in common is $B' := \{g \in G \,|\, \forall m \in B\colon (g,m) \in I\}$. A formal concept is thus a pair $(A,B) \subseteq G \times M$ with $A = B'$ and $B = A'$. Furthermore, the definition of the derivational operators guarantees that the set of all formal concepts of a formal context (G,M,I) equals $\{(A'',A') \,|\, A \subseteq G\}$ and $\{(B',B'') \,|\, B \subseteq M\}$.[4]

The subconcept-superconcept relation on the set of all formal concepts of a context defines a partial order:

$$(A_1,B_1) \leq (A_2,B_2) \Leftrightarrow A_1 \subseteq A_2 \Leftrightarrow B_1 \supseteq B_2$$

This order relation corresponds to our intuitive notion of super- and subconcepts. Superconcepts are more general, encompass more objects, and are characterized by fewer attributes. The main theorem of FCA states that the set of all formal concepts of a formal context (G,M,I) ordered with respect to the subconcept-superconcept relation constitutes a complete lattice, which is called the *formal concept lattice* of

[4]A detailed introduction to Formal Concept Analysis is given by Ganter and Wille (1999).

(G, M, I). Figure 1 shows the concept lattice corresponding to the formal context in Table 1. Formal concept lattices visualize structural connections between the data of a flat data table. Ganter and Wille (1998) stress:

> It is our belief that the potential of Formal Concept Analysis as a branch of Applied Mathematics is just beginning to show. A typical task that concept lattices are useful for is to unfold given data, making their conceptual structure visible and accessible, in order to find patterns, regularities, exceptions, etc.

The definition of concept lattices allows an especially economical labelling of their Hasse diagrams (see Figure 1): instead of labelling every concept with its complete extent and intent, only the object and attribute concepts are labelled. The *object concept* of an object g is the smallest concept whose extent includes g, i.e. (g'', g'). The *attribute concept* of an attribute m is analogously the largest concept whose intent includes m, i.e. (m', m''). In this way a concept lattice becomes an inheritance hierarchy. Any concept of the lattice inherits all objects which are labelled with subconcepts as its extent, and it inherits all attributes with which superconcepts are labelled as its intent. Inheritance hierarchies based on concept lattices are completely nonredundant, i.e., each attribute and each object appears exactly once in the hierarchy.

Besides the tabular form (as a formal context) and the hierarchical form (as a formal concept lattice) FCA provides a third device, namely attribute implications, to represent the mutual dependencies of the data to be analyzed. An *attribute implication* of a formal context is an implication of the form $\mu \to \nu$, where μ and ν are subsets of the attribute set (μ is called the premise and ν the conclusion of the implication). An attribute implication $\mu \to \nu$ is *valid* in a context if and only if every object which has all attributes in μ also has all attributes in ν; this is equivalent to the condition $\nu \subseteq \mu''$. The implications can be read off directly from the concept lattice: An implication $m_1 \wedge m_2 \wedge \ldots \wedge m_k \to m$ holds exactly when the greatest lower bound of the attribute concepts of $m_1, \ldots, m_k$ is a subconcept of the attribute concept of m. A set $\mathcal{I}$ of attribute implications of a context K which is complete[5] and non-redundant[6] is called an *implication basis* of the context. Since for every implication basis of a formal context the union of every premise and the maximal corresponding conclusion forms an intent of one of the concepts of the context, the structure of the concept lattice is determined by the set of valid attribute implications up

[5] Every valid attribute implication of K can be derived from the members of $\mathcal{I}$.
[6] No real subset of $\mathcal{I}$ forms a complete set of attribute implications of K.

TABLE 2 Basis of attribute implications of the formal context in Table 1

$\emptyset$	$\to$	$\{\mathrm{v} : \mathrm{VAL}, \mathrm{n} : \mathrm{VAL}\}$	(1.1)
$\{\mathrm{v} : -, \mathrm{n} : -\}$	$\leftrightarrow$	$\{\mathrm{pform} : \mathrm{VAL}\}$	(1.2)
$\{\mathrm{v} : -, \mathrm{n} : -\}$	$\leftrightarrow$	$\{\mathrm{pform} : \mathrm{with}\}$	(1.3)
$\{\mathrm{v} : -, \mathrm{n} : +\}$	$\leftrightarrow$	$\{\mathrm{nform} : \mathrm{VAL}\}$	(1.4)
$\{\mathrm{v} : +, \mathrm{n} : -\}$	$\leftrightarrow$	$\{\mathrm{vform} : \mathrm{VAL}\}$	(1.5)
$\{\mathrm{v} : +, \mathrm{n} : -\}$	$\leftrightarrow$	$\{\mathrm{aux} : \mathrm{VAL}\}$	(1.6)
$\{\mathrm{v} : +, \mathrm{n} : -\}$	$\leftrightarrow$	$\{\mathrm{inv} : \mathrm{VAL}\}$	(1.7)
$\{\mathrm{inv} : +\}$	$\to$	$\{\mathrm{vform} : \mathrm{fin}, \mathrm{aux} : +\}$	(1.8)
$\{\mathrm{aux} : -\}$	$\to$	$\{\mathrm{inv} : -\}$	(1.9)
$\{\mathrm{vform} : \mathrm{pas}\}$	$\to$	$\{\mathrm{aux} : -\}$	(1.10)
$\{\mathrm{vform} : \mathrm{bse}\}$	$\to$	$\{\mathrm{inv} : -\}$	(1.11)
$\{\mathrm{nform} : \mathrm{it}\}$	$\to$	$\{\mathrm{nform} : \mathrm{VAL}\}$	(1.12)
$\{\mathrm{nform} : \mathrm{nor}\}$	$\to$	$\{\mathrm{nform} : \mathrm{VAL}\}$	(1.13)
$\{\mathrm{inv} : -\}$	$\to$	$\{\mathrm{inv} : \mathrm{VAL}\}$	(1.14)
$\{\mathrm{aux} : +\}$	$\to$	$\{\mathrm{aux} : \mathrm{VAL}\}$	(1.15)
$\{\mathrm{vform} : \mathrm{fin}\}$	$\to$	$\{\mathrm{vform} : \mathrm{VAL}\}$	(1.16)
$\{\mathrm{n} : +, \mathrm{n} : -\}$	$\to$	$\perp$	(1.17)
$\{\mathrm{v} : +, \mathrm{v} : -\}$	$\to$	$\perp$	
$\vdots$		$\vdots$	
$\{\mathrm{vform} : \mathrm{bse}, \mathrm{vform} : \mathrm{pas}\}$	$\to$	$\perp$	

to isomorphism. Table 2 shows a basis of attribute implications of the context from Table 1.[7] Some abbreviations have been used in the table: $A \leftrightarrow B$ denotes the two implications $A \to B$ and $B \to A$ and $\perp$ is the (inconsistent) set of all attributes of the context of Table 1.

1.3 FCA relates FCRs and type signatures

The main idea for showing the convertability of FCRs and type signatures is to construct a suitable formal context in order to employ the methods of FCA. Figure 2 shows a lexical fragment consisting of 10 lexemes classified with respect to some of the features proposed in Gazdar et al. (1985).[8] The chosen features are exactly those which play a role

[7]Bases of formal contexts can be efficiently calculated with the help of the program *ConImp* (`http://www.mathematik.tu-darmstadt.de/ags/ag1/Software/DOS-Programme/Welcome_de.html`).

[8]The feature *vform* distinguishes parts of the verb paradigm (*bse*, base-form; *fin*, finite; *pas*, passive participle) and *nform* analogously distinguishes the special expletive pronoun *it* from normal nouns (*norm*).

$$\text{sing}: \begin{bmatrix} \text{v}: & + \\ \text{n}: & - \\ \text{vform}: & \text{bse} \\ \text{aux}: & - \\ \text{inv}: & - \end{bmatrix} \quad \text{sings}: \begin{bmatrix} \text{v}: & + \\ \text{n}: & - \\ \text{vform}: & \text{fin} \\ \text{aux}: & - \\ \text{inv}: & - \end{bmatrix} \quad \text{sung}: \begin{bmatrix} \text{v}: & + \\ \text{n}: & - \\ \text{vform}: & \text{pas} \\ \text{aux}: & - \\ \text{inv}: & - \end{bmatrix}$$

$$\text{can}: \begin{bmatrix} \text{v}: & + \\ \text{n}: & - \\ \text{vform}: & \text{fin} \\ \text{aux}: & + \\ \text{inv}: & + \end{bmatrix} \quad \text{can}: \begin{bmatrix} \text{v}: & + \\ \text{n}: & - \\ \text{vform}: & \text{fin} \\ \text{aux}: & + \\ \text{inv}: & - \end{bmatrix} \quad \text{can}: \begin{bmatrix} \text{v}: & + \\ \text{n}: & - \\ \text{vform}: & \text{bse} \\ \text{aux}: & + \\ \text{inv}: & - \end{bmatrix}$$

$$\text{child}: \begin{bmatrix} \text{v}: & - \\ \text{n}: & + \\ \text{nform}: & \text{norm} \end{bmatrix} \quad \text{it}: \begin{bmatrix} \text{v}: & - \\ \text{n}: & + \\ \text{nform}: & \text{it} \end{bmatrix} \quad \text{little}: \begin{bmatrix} \text{v}: & + \\ \text{n}: & + \end{bmatrix}$$

$$\text{with}: \begin{bmatrix} \text{v}: & - \\ \text{n}: & - \\ \text{pform}: & \text{with} \end{bmatrix}$$

FIGURE 2 Lexical fragment classified with respect to some features of Gazdar et al. (1985)

in the first 4 FCRs given in Gazdar et al. (1985); these FCRs regulate the distribution of the features *v*, *n*, *vform*, *nform*, *pform*, *inv*, *aux*, and their values (see Table 3). The formal context in Table 1 is formed by taking each feature-value pair as an independent attribute of the context. GPSG also allows FCRs of the form [VFORM] ⊃ [+V, −N] (see FCR 2), which encode not only restrictions on feature-value pairs but also on the admissibility of certain features in the first place. Because of this, further attributes of the form *feature:VAL* have been added in Table 1, where such an attribute applies to a word if there is some value *value* such that *feature:value* is an feature specification of the word. Feature structures with embedded structures can be represented in a formal context by *flattening* them and viewing the path-value pairs as attributes (cf. Sporleder, 2003, Petersen, 2004). We call such formal contexts obtained from feature structures *feature-structure contexts.*

To illustrate the relationship between FCRs and type signatures we will proceed as follows: First we show how a type signature can be derived from a feature structure context, and then we do the same for a system of FCRs. Finally, we demonstrate how a feature structure context can be constructed from either a type signature or a set of FCRs.

TABLE 3 The first 4 FCRs from Gazdar et al. (1985)

FCR 1 :	[+INV]	$\supset$	[+AUX, FIN]
FCR 2 :	[VFORM]	$\supset$	[+V, −N]
FCR 3 :	[NFORM]	$\supset$	[−V, +N]
FCR 4 :	[PFORM]	$\supset$	[−V, −N]

Figure 1 shows the concept lattice for the context in Table 1, which can be directly interpreted as a type signature of HPSG if, first, a unique type is assigned to each node of the lattice[9] and if, second, an additional type *VAL* with subtypes +, −, *it*, *norm*, *with*, *fin*, *bse*, and *pas* is added.[10] The feature labels then encode the appropriateness conditions associated with each type, and the subconcept relation corresponds to the subtype relation in the type signature. If one reads the hierarchy in Figure 1 as a type signature one sees that it makes extensive use of multiple inheritance. Due to the definition of formal concepts, however, it can never be the case that a type inherits incompatible information from its upper neighbors.[11]

As noted before, type signatures fulfill two tasks: they avoid redundancies by structuring the information in an inheritance hierarchy and they restrict the set of permissible feature structures. As Gerdemann and King (1993, 1994) note, feature structures must be well-typed and sort-resolved in order to be total models of linguistic objects and furthermore, type systems must be closed-world systems. In order to show that our type signature successfully restricts the set of permissible feature structures, we have to demonstrate that, on the one hand, no inadmissible structure will be sort-resolved and well-typed with respect to the type signature and, on the other hand, every permissible structure is well-typed and sort-resolved. The former follows from the fact that the minimal concept of a concept lattice derived from a feature-structure context never represents an object of the context, since the values of an attribute mutually exclude each other; hence, the extent of the minimal concept of such a context is always empty. It follows

[9] The lowest node is assigned the type $\perp$ for *bottom* and the highest gets $\top$ for *top*. The type $\perp$ expresses contradiction and ensures that unification never fails.

[10] Normally, one would add extra types *boolean*, *vform*, *pform*, and *nform* between *VAL* and its subtypes in order to explicitly state the value ranges of the attributes. For further details on the automatic construction of type signatures from formal concept lattices see Petersen (2004).

[11] The principle of *unique feature introduction* is never violated because, by adding attributes of the form *attribute:VAL* to the formal context, the features are introduced on their own, unique attribute concepts.

that every *atom* (i.e. direct upper neighbor of the minimal concept) is an object concept and thus represents a permissible feature structure. The latter follows from the fact that all object concepts of the linguistic objects of Table 1 are atoms of the lattice. This property is not a necessary consequence of using a feature-structure context; it rather follows from the principle of choosing sort-resolved feature structures as total models of linguistic objects. This principle would be violated, e.g., if the example data were classified by attributes like *non-auxiliary verb*, *auxiliary verb*, and *inverted auxiliary verb*. In that case, the inverted auxiliary verb *can1* would belong to a subclass of the class of non-inverted auxiliary verbs like *can2*. By introducing the Boolean-valued attribute *inv*, the property of being non-inverted is marked explicitly and each linguistic object is described by an atom of the concept lattice.

The hierarchy in Figure 1 encodes so much information about the distribution of atomic values that it suffices to pair phonological forms with types in the lexicon in order to obtain adequate feature structures. For realistic lexica such a procedure leads to enormous type signatures since every lexical feature structure must have its own type.[12] Moreover, it conflicts with the idea of types when they, e.g. in the case of *with*, cover only a single object. On this issue Pollard and Sag (1987, p. 192) state:

> [...] lexical information is organized on the basis of relatively few — perhaps several dozen — word types arranged in cross-cutting hierarchies which serve to classify all words on the basis of shared syntactic, semantic, and morphological properties. By factoring out information about words which can be predicted from their membership in types (whose properties can be stated in a single place once and for all), the amount of idiosyncratic information that needs to be stipulated in individual lexical signs is dramatically reduced.

Having constructed type signatures from formal concept lattices, we will now show how to derive FCRs from a feature-structure context as given in Table 1. The FCRs of GPSG are nothing other than implications that are compatible with the data of Table 1. In order to restrict the set of admissible categories sufficiently, the FCRs must reflect the mutual dependencies between the data of the context. Hence, the data of the context must respect the FCRs and the object concepts of the context must be derivable from the FCRs. The latter ensures that the FCRs license no feature structure with features which do not co-occur in the input structures.

From what we have seen in Section 1.2 it is obvious that the set of

[12] Petersen (2004) presents a *folding strategy* to reduce the number of types.

attribute implications from Table 2 is already an adequate set of FCRs for the input structures of Figure 2 if one interprets $\perp$ as contradiction. In order to explain why our automatically derived set of implications contains so many more elements than the four given by Gazdar et al. (1985), we will compare the two sets:

It is apparent that all four FCRs from Table 3 can be derived from the attribute implications in the implication basis of Table 2: FCR 1 corresponds to implication 1.8; FCR 2 is contained in equivalence 1.5, FCR 3 in equivalence 1.4, and FCR 4 in equivalence 1.2.

However, Table 2 includes further implications, some of which bear no real information in the sense of GPSG since they either result from the sparse input data (cf. equivalence 1.3), from the special role of the feature value *VAL* (cf. implications 1.12–1.16), or from the fact that no knowledge about the exclusivity of features is implemented in FCA (cf. implication 1.17). Implication 1.9 follows from FCR 1 on the condition that, first, whenever *inv* is specified then *aux* is also specified and vice versa (equivalence 1.7) and, second, *inv* is restricted to the values $+$ and $-$; implication 1.11 follows analogously from FCR 1.

The implications 1.1, 1.6, and 1.7 are missing in the FCRs and moreover, only one direction of the equivalences 1.2, 1.4, and 1.5 is stated in the FCRs. These implications regulate when categories necessarily must be specified with respect to certain features without saying anything about the concrete feature values. A very surprising gap in the list of FCRs is evident in implication 1.10, according to which passive verbs are not auxiliaries. This fact cannot be derived from the FCRs in Table 3. The GPSG grammar given in Gazdar et al. (1985) thus allows the feature-specification bundle $\{[PAS], [+AUX], [-INV]\}$, which encodes a non-inverted auxiliary in passive. This demonstrates that the automatically extracted basis of attribute implications is more explicit than the manually formulated FCRs, which arise from linguistic intuition and miss statements which probably were too obvious for the investigators. Hence, FCA can be a powerful tool for tracking down gaps in intellectually constructed theories.

The manually constructed FCRs of GPSG theories are not always as easy to read off a basis of attribute implications as in the example. Each attribute implication $A \rightarrow B$ encodes a *cumulated Horn clause* of the form $\bigwedge A \rightarrow \bigwedge B$ or $\bigwedge A \rightarrow \perp$ if B'' equals the set of all attributes. But FCRs are not necessarily cumulated Horn clauses: some of the FCRs in Gazdar et al. (1985), p. 246, make use of negation and disjunction. The following consideration shows that such FCRs are implicitly encoded in the concept lattice of a feature-structure context, too. It is a well-known fact from propositional logics that each theory can be generated by conditionals, so called *observational statements* which use only disjunction and conjunction, but not negation. An FCR with a disjunctive premise can be directly transformed into a set of Horn clauses:

$$a \vee b \rightarrow c \quad \Leftrightarrow \quad (a \rightarrow c) \wedge (b \rightarrow c)$$

Hence, we can focus on the derivation of conditionals with disjunctive conclusions from concept lattices.

Let (G, M, I) be a formal context. A theory of observational statements over M is said to be complete if every conditional which is compatible with the data of the context is entailed by the theory; we call such a theory a *complete observational theory* of the context. The *information domain* of such a complete observational theory, i.e. the set of all maximal consistent subsets of M w.r.t. the theory, consists exactly of the intents of the object concepts.[13] These considerations open a way to derive a complete observational theory and hence a complete set of FCRs from a concept lattice:

Each formal concept (A, B) of the lattice which is not an object concept corresponds to an observational statement whose premise is the conjunction of the elements of the intent B and whose conclusion is the disjunction of the conjunctions of its subconcept intents minus B. Adding the corresponding statement of a concept to the theory amounts to removing the concept intent from the informational domain of the theory. For example, the statement corresponding to the attribute concept of *nform:val* is

$$\begin{multline}\mathrm{n:VAL} \wedge \mathrm{v:VAL} \wedge \mathrm{v:-} \wedge \mathrm{n:+} \wedge \mathrm{nform:VAL}\\ \rightarrow \mathrm{nform:norm} \vee \mathrm{nform:it}, \tag{1.18}\end{multline}$$

which can be simplified by Table 2 to

$$\mathrm{nform:VAL} \rightarrow \mathrm{nform:norm} \vee \mathrm{nform:it}.$$

This states that if an object bears the feature *nform*, then the latter must be specified for the value *norm* or *it*. As a second example, we look at the concept with the extent $\{\mathrm{can2}, \mathrm{can3}\}$: here the simplified corresponding observational statement is

$$\mathrm{aux:+} \wedge \mathrm{inv:-} \rightarrow \mathrm{vform:fin} \vee \mathrm{vform:bse}.$$

Ganter and Krausse (1999) present an alternative procedure for systematically constructing a complete observational theory of a formal context. Depending on the concrete task for which the complete theory

[13]The information domain of a complete Horn theory of a formal context, i.e. a complete theory consisting only of Horn clauses, equals the set of the intents of the formal concepts.

of a formal context is used, it often makes sense to restrict the theory class by dispensing with the disjunctive or conjunctive operators (Osswald and Petersen, 2002, 2003).

It remains to show how feature structure contexts can be derived either from a type signature or from FCRs. Given a type signature, the adequate feature structure context can be directly constructed from the set of totally well-typed and sort-resolved feature structures. A detailed description of the construction method (even for type signatures with co-references) is given in Petersen (2004, 2005). Given a set of FCRs, the corresponding formal context is equivalent to the information domain of the FCRs (see Osswald and Petersen, 2003).

1.4 Conclusion

We have seen that FCA provides three different ways to display data: in tabular form, as a hierarchy, and as a set of implications. None of the three reduce the structure of the given data, and that is why it is possible to switch from one representation to another without loosing information. Hence, FCA is superior to many alternative approaches to induction inasmuch as it is neutral with respect to the analyzed data and describes them completely.[14] The relations between formal contexts and the corresponding concept lattices is absolutely transparent since there is exactly one of the latter for each context.

FCA allows us to capture the relationship between FCRs and type signatures explicitly. The prerequisite for this is the construction of a suitable feature-structure context.

It is remarkable that FCA has not yet been adopted as a standard tool in linguistics. Until now only a few linguists are employing FCA (for an overview see Priss, 2003); most of them explore the utility of FCA in lexical semantics and wordnets (e.g. Janssen, 2002). Our work shows that FCA can play an important methodological role for linguists in that it helps them to discover generalizations missed with intuitive procedures and because it facilitates a better understanding of the relationships between different formal theories. We hope that FCA will soon come to take on a more important role in linguistics.

References

Bresnan, Joan. 1982. *The Mental Representation of Grammatical Relations*.

[14]Ganter and Wille (1998) on the disadvantages of FCA: "Nevertheless it may be exponential in size, compared to the formal context. Complexity therefore is, of course, a problem, even though there are efficient algorithms and advanced program systems. A formal context of moderate size may have more concepts than one would like to see individually."

Cambridge, MA: The MIT Press.

de Smedt, Koenraad. 1984. Using object-oriented knowledge representation techniques in morphology and syntax programming. In *Proceedings of the European Conference on Artificial Intelligence 1984*, pages 181–184.

Flickinger, Dan. 1987. *Lexical Rules in the Hierarchical Lexicon*. Ph.D. thesis, Stanford University.

Ganter, Bernhard and R. Krausse. 1999. Pseudo models and propositional horn inference. Tech. rep., MATH-AL-15-1999, Technische Universität Dresden, Germany.

Ganter, Bernhard and Rudolf Wille. 1998. Applied lattice theory: Formal concept analysis. In G. Grätzer, ed., *General Lattice Theory*, pages 591–605. Basel: Birkhäuser Verlag.

Ganter, Bernhard and Rudolf Wille. 1999. *Formal Concept Analysis. Mathematical Foundations*. Berlin: Springer.

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

Gazdar, Gerald and Geoffrey K. Pullum. 1982. *Generalized phrase structure grammar: a theoretical synopsis*. Bloomington, Indiana: Indiana University Linguistics Club.

Gerdemann, Dale and Paul John King. 1993. Typed feature structures for expressing and computationally implementing feature cooccurence restrictions. In *Proceedings of 4. Fachtagung der Sektion Computerlinguistik der Deutschen Gesellschaft für Sprachwissenschaft*, pages 33–39.

Gerdemann, Dale and Paul John King. 1994. The Correct and Efficient Implementation of Appropriateness Specifications for Typed Feature Structures. In *Proceedings of the 15th Conference on Computational Linguistics (COLING-94)*, pages 956–960. Kyoto, Japan.

Janssen, Maarten. 2002. *SIMuLLDA – a Multilingual Lexical Database Application using a Structural Interlingua*. Ph.D. thesis, Universiteit Utrecht.

Kaplan, Ronald M. and Joan Bresnan. 1982. Introduction: grammars as mental representations of language. In Bresnan (1982), pages xvii–lii.

Karttunen, Lauri. 1986. Radical lexicalism. Tech. Rep. CSLI-86-68, CSLI, Stanford. Reprinted in M. Baltin and A. Koch, eds., *Alternative Conceptions of Phrase Structure*, pages 43–65. Chicago: CUP. 1989.

Osswald, Rainer and Wiebke Petersen. 2002. Induction of classifications from linguistic data. In *Proceedings of the ECAI-Workshop on Advances in Formal Concept Analysis for Knowledge Discovery in Databases*. Lyon.

Osswald, Rainer and Wiebke Petersen. 2003. A logical approach to data-driven classification. *Lecture Notes in Computer Science* 2821:267–281.

Petersen, Wiebke. 2004. Automatic induction of type signatures. Unpublished manuscript.

Petersen, Wiebke. 2005. *Induktion von Vererbungshierarchien mit Mitteln der formalen Begriffsanalyse*. Ph.D. thesis, Heinrich-Heine-Universtität Düsseldorf. (in progress).

Pollard, Carl and Ivan A. Sag. 1987. *Information-Based Syntax and Semantics*. Stanford, CA: CSLI Lecture Notes.

Priss, Uta. 2003. Linguistic applications of formal concept analysis. In *Proceedings of ICFCA 2003*. (to appear).

Sag, Ivan and Thomas A. Wasow. 1999. *Syntactic Theory: A Formal Introduction*. Stanford, CA: CSLI.

Shieber, Stuart, Hans Uszkoreit, Fernando Pereira, Jane Robinson, and Mabry Tyson. 1983. The formalism and implementation of PATR-II. In B. J. Grosz and M. Stickel, eds., *Research on Interactive Acquisition and Use of Knowledge*, techreport 4, pages 39–79. Menlo Park, CA: SRI International. Final report for SRI Project 1894.

Sporleder, Caroline. 2003. *Discovering Lexical Generalisations. A Supervised Machine Learning Approach to Inheritance Hierarchy Construction*. Ph.D. thesis, Institute for Communicationg and Collaborative Systems. School of Informatics. University of Edinburgh.