1.1. *The HPSG framework*

The fundamental concept for linguistic representation in HPSG is the *sign*, generalizing de Saussure's idea of the minimal independent unit relating form and meaning in language. A sign in HPSG is a collection of multiple kinds of properties or information, including phonological, syntactic, semantic, and contextual constraints, represented as a typed attribute-value matrix (AVM) where each attribute's value is also assigned a type, potentially with additional constraints. Since such constraints on a type can introduce allowable or appropriate features for that type, they are called *appropriateness conditions*, or more generally *feature declarations*. Words are represented as signs, and phrases are also signs, with many but not all of the same attributes or features shared between words and phrases. Both words and phrases have at least the features PHON for phonology and SYNSEM for syntactic/semantic constraints. PHON has as its value a list of phonology descriptions, which we will in this chapter replace with the less precise but more convenient orthographic equivalents. SYNSEM has as its value another AVM structure of type *synsem*, with appropriate attributes LOC and SLASH, to distinguish constraints on local sisters in a phrase from constraints on unbounded dependency constructions. LOC again takes an AVM as value, with appropriate features CAT for syntactic (category) constraints and CONT for semantic content. CAT attributes include HEAD, identifying those properties of a sign which are shared with the phrase it heads; VALENCE for subcategorization constraints; and ARG-ST to represent the proper domain for several general constraints on argument structure. Finally, HEAD features are appropriately distinguished for the various types of lexical signs (e.g. VFORM for verbs or CASE for nouns), and the VALENCE attributes include SUBJ for subject, SPR for specifier, and COMPS for complements. This architecture for the sign is illustrated in Figure 1, showing the lexical entry for the base form of the verb *like*. The other two HEAD features AUX and INV illustrated here will be introduced and explained in section 5.

In this notation each attribute or feature, such as SUBJ, is shown in small caps, with the value to its right. List values are represented in angle brackets,[2] set values in curly brackets, and atomic values are given in italics. Notice here the use of boxed numbers in front of the values of certain attributes, to indicate that the structure for a given value is shared in two (or more) attributes. For example, the semantic index introduced by the subject noun phrase of *like* is coreferenced with the ARG1 attribute in the *like* relation in the CONT of the sign, expressing the linking of the syntactic subject's index with the appropriate thematic role in the semantics.

[2] The empty list is represented as $< \;\; >$, and a list with one member as $< [\,] >$.

$$
\begin{bmatrix}
\textit{phrase} & \\
\text{PHON} & \boxed{5} \oplus \boxed{6} \\
\text{SYNSEM} & \begin{bmatrix}
\text{LOC} & \begin{bmatrix}
\text{CAT} & \begin{bmatrix}
\text{HEAD} & \boxed{1} \\
\text{VAL} & \begin{bmatrix}
\text{SUBJ} & \langle\rangle \\
\text{SPR} & \langle\rangle \\
\text{COMPS} & \langle\rangle
\end{bmatrix}
\end{bmatrix} \\
\text{CONT} & \boxed{2}
\end{bmatrix} \\
\text{SLASH} & \boxed{3}
\end{bmatrix} \\
\text{NON-HD-DTR} & \begin{bmatrix}
\text{PHON} & \boxed{5} \\
\text{SYNSEM} & \boxed{4}
\end{bmatrix} \\
\text{HD-DTR} & \begin{bmatrix}
\text{PHON} & \boxed{6} \\
\text{SYNSEM} & \begin{bmatrix}
\text{LOC} & \begin{bmatrix}
\text{CAT} & \begin{bmatrix}
\text{HEAD} & \boxed{1} \\
\text{VAL} & \begin{bmatrix}
\text{SUBJ} \left\langle \boxed{4} \right\rangle \\
\text{SPR} \; \langle\rangle \\
\text{COMPS} \; \langle\rangle
\end{bmatrix}
\end{bmatrix} \\
\text{CONT} & \boxed{2}
\end{bmatrix} \\
\text{SLASH} & \boxed{3}
\end{bmatrix}
\end{bmatrix}
\end{bmatrix}
$$

Figure 2. HPSG Head-Subject rule schema

for the daughter nodes in the phrase, each of which is a sign (either word or phrase). In this example, the rule introduces the NON-HD-DTR and HD-DTR attributes for the subject daughter and the head daughter, respectively. What the rule requires is that the SYNSEM of the subject daughter be identified with the head daughter's SUBJECT's value; it also incorporates the Head-Feature Principle, and requires that the head daughter have picked up all of its complements before combining with its subject. As with lexical signs, phrases provide a specification of their syntactic and semantic properties in the SYNSEM attribute, using reentrancies (identities in attribute values) to relate properties of the phrase to those of its immediate daughters. The PHON feature of the phrase illustrates a further use of reentrancies, here encoding the constraint that the phonology of the phrase is the result of appending the phonology of the non-head daughter with that of the head daughter.

While this rich architecture enables us to express precise, detailed properties of signs, and to express generalizations about these properties within phrases, the notation can be unnecessarily cumbersome. So in the remainder

2. Inheritance and the Hierarchical Lexicon

Inheritance is a knowledge representation technique familiar from work in artificial intelligence as well as (object-oriented) software engineering. Inheritance is particularly useful for storing concisely large amounts of knowledge related to a given set of objects. The objects are classified according to a hierarchically organized collection of *classes*. Each class comes with a number of properties. If class C_2 is a subclass of C_1, all properties of C_1 are inherited (by default) by C_2. Multiple inheritance allows a class to be a subclass of more than one superclass, in which case the class inherits the (non-conflicting) properties of all its super-classes. The power of inheritance lies in the fact that properties which hold for a class of objects need only be declared once. Apart from leading to more concise representations, inheritance, and the object-oriented approach that comes with it, also helps to ensure consistency and facilitates maintenance of large knowledge bases.

The use of inheritance as a technique to organize the lexicon has been a dominant trend in constraint-based grammar formalisms. The early work within HPSG on inheritance can be found in Flickinger et al. (1985), Pollard and Sag (1987; chpt. 8), and Flickinger (1987). Here, the lexicon is presented as a hierarchically organized collection of *frames* (or *classes*). In such a set-up, the lexical entry for a specific lexical item need only specify the idiosyncratic (unpredictable) properties of the item and the frame (or set of frames) from which it inherits.

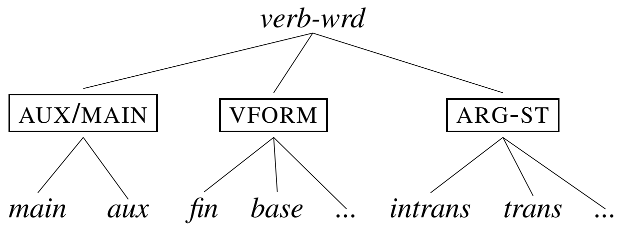
Consider, for instance, the AVM for the base form of the verb *like* presented in Figure 1. This AVM contains several pieces of information that are common to verbs, to transitive verbs, to base forms, etc. Using frames, we may therefore decompose this entry as shown in Figure 3 (leaving out redundant path-prefixes, and abbreviating the AVMs for NPs).

The base verb *like* inherits from the frames for *main*, *base* and *trans*, which all inherit from the more general frame *verb-wrd*, which itself is a specialization of *word*. The HEAD feature of a *verb-wrd* is required to be of type *verb*. For the type *verb* the features VFORM, INV, and AUX are appropriate. Furthermore, the constraints on *verb-wrd* restrict the value of the valence features SUBJ, SPR, and COMPS, in terms of ARG-ST. The frames *main*, *base*, and *trans* define *like* as a main verb,[3] a base form, and a transitive verb, respectively. The only information idiosyncratic to *like* is its phonology and its content. This seems reasonable, as the relationship between phonology

[3] We assume here that main verbs, as opposed to auxiliaries, always have one element on their SUBJ-list. See section 5 for details.

plicit in Figure 3. A specific lexical entry will always be either a main verb or an auxiliary, will belong to exactly one subcategorization type, and will have a specific morphological form. To make the structure of the hierarchy more explicit, Pollard and Sag (1987) introduce *partitions*. The idea is that whereas an entry may inherit from more than one frame, it will always inherit from exactly one frame in a given partition. In the inheritance hierarchy below, partitions appear in boxes:

(1)



The frame for *verb-wrd* has a number of subframes, which can be grouped in three partitions, AUX/MAIN, VFORM and ARG-ST. A given verb form will normally inherit from a member of each of these partitions, but not from both *main* and *aux*, for instance.

Lexical inheritance is naturally connected to the use of types. The type system in HPSG is defined hierarchically in terms of subsumption: if T' is a subtype of T, all features declared appropriate for T are appropriate for T' as well, and furthermore if the value of a feature f is required to be of type S on T, the value of f on T' can only be required to be of type S', where S' is a subtype of S. Since a subtype T' of T must satisfy all the constraints T satisfies, it is said that T subsumes T'. For example, given that *word* and *phrase* are both subtypes of *sign*, all features appropriate for *sign* are appropriate for *word* and *phrase* as well.

The lexical hierarchy can be thought of similarly: if *verb-wrd* inherits from *word*, all constraints on *word* must hold for *verb-wrd* as well. Therefore, one can try to define the lexical hierarchy as a hierarchy of types. For instance, we could introduce a type *verb-wrd*, with *main, aux, fin, base, intrans, trans, etc.* as subtypes. Each of these subtypes is subsumed by *verb-wrd* and introduces additional constraints. If two types are elements of the same *partition*, i.e. *aux* and *main* are both elements of the AUX/MAIN partition, this means that they are incompatible (i.e. not unifiable). If two types belong to different partitions, they will in general be unifiable, and give rise to a uniquely typed most general unifier. *Aux* and *fin*, for instance, belong to different partitions, and thus their unification must give rise to something of type *aux-fin*. A procedure for translating the lexical hierarchy of Pollard and Sag (1987) into a type hierarchy is given in Carpenter (1992b). As lexical types frequently introduce inheritance, sometimes extended with more or less ad-hoc extensions to cope with some aspects of nonmonotonicity.

3. Lexical Rules

Lexical rules are often considered to be the principal mechanism for capturing generalizations within the lexicon. In HPSG, as in almost any other grammatical framework, lexical rules have been proposed to account for inflection and derivation (section 3.1), and valence alternations of various kinds (section 3.2). Some of the more problematic aspects of using lexical rules in a constraint-based setting are discussed in section 3.3.

3.1. *Inflection and Derivation*

Pollard and Sag (1987) propose an account of inflection based on lexical rules. Their rule for third person singular inflection in English is as follows.[6]

(3)
$$\begin{bmatrix} \textit{base} & \\ \textsc{phon} & \boxed{1} \\ \textsc{3rdsng} & \boxed{2} \\ \textsc{synsem} & \begin{bmatrix} \textsc{val} & \boxed{3} \\ \textsc{cont} & \boxed{4} \end{bmatrix} \end{bmatrix} \Rightarrow \begin{bmatrix} \textit{3rdsng} & \\ \textsc{phon} & f_{3rdsng}(\boxed{1},\boxed{2}) \\ \textsc{synsem} & \begin{bmatrix} \textsc{val} & \boxed{3} \\ \textsc{cont} & \boxed{4} \end{bmatrix} \end{bmatrix}$$

The function f_{3rdsng} produces the third person singular inflected phonological form of a given base form. It takes this base form as argument, as well as the value of the feature 3RDSNG. The latter is used to account for exceptional forms. For regular verbs, the value of this feature would be unspecified, in which case the inflected form is the base form + *-s*. If a value for 3RDSNG is provided (such as *has* for the base form *have*), this value is returned as result by f_{3rdsng}. The valence properties, as well as the semantics, are shared between input and output. The fact that the output must be specified as [VFORM *fin*] and selects for a third person singular subject is accounted for by assigning the output the type *3rdsng*. The constraints that come with this type are assumed to provide the relevant information.

[6] In this example, as well as in the examples of lexical rules given below, we assume the architecture of (lexical) signs given in section 1.1. We believe that the differences between our version of these rules and the original versions is of little or no importance for the issue at hand.

This lexical rule replaces the subject *that*-clause by an expletive *it*-NP, while appending the *that*-clause to COMPS.[7]

Also passive is often discussed under the rubric of valence alternation. The lexicalist analysis of the alternation in

(7) a. Kim will read the book

b. The book was read by Sandy

assumes that the passive participle *read* selects for a subject that corresponds to the object of the base form *read*, and that the *by*-phrase corresponds to the subject of the base form. This can be accounted for by a passive lexical rule, adopted from Borsley (1996; p. 197).[8]

$$(8)\quad \begin{bmatrix} \text{PHON} & \boxed{1} \\ \text{HEAD} & \begin{bmatrix}\text{VFORM} & \textit{bse}\end{bmatrix} \\ \text{SUBJ} & \langle\ \boxed{2}\ \text{NP}_i\ \rangle \\ \text{COMPS} & \langle\ \boxed{3}\ |\ \boxed{4}\ \rangle \\ \text{ARG-ST} & \langle\ \boxed{2}\ |\ \boxed{5}\ \rangle \end{bmatrix} \Rightarrow \begin{bmatrix} \text{PHON} & \textit{fpass}(\boxed{1}) \\ \text{HEAD} & \begin{bmatrix}\text{VFORM} & \textit{passp}\end{bmatrix} \\ \text{SUBJ} & \langle\ \boxed{3}\ \rangle \\ \text{COMPS} & \boxed{4}\ (\oplus \langle\ \boxed{6}\ \text{PP}[\textit{by}]_i\ \rangle\ ) \\ \text{ARG-ST} & \boxed{5}\ (\oplus \langle\ \boxed{6}\ \rangle\ ) \end{bmatrix}$$

Note that apart from making a change to the valence features SUBJ and COMPS, the rule also affects ARG-ST. This will account for the fact that binding possibilities differ for actives and passives. Furthermore, passive morphology is assumed to be added by the function *fpass*.

While the use of lexical rules for phenomena such as extraposition and passive is relatively uncontroversial, there have also been proposals for using lexical rules to account for phenomena (such as extraction and the selection of adjuncts) that are traditionally accounted for outside the lexicon. For instance, following a proposal in Pollard and Sag (1994; p. 378), Sag and Fodor (1994) and Sag (1997) propose an account of extraction in which traces are eliminated in favour of the following lexical rule:

$$(9)\quad \begin{bmatrix} \text{COMPS} & \langle \ldots \begin{bmatrix}\text{LOC} & \boxed{3}\end{bmatrix} \ldots \rangle \\ \text{SLASH} & \boxed{2} \end{bmatrix} \Rightarrow \begin{bmatrix} \text{COMPS} & \langle \ldots\ \ldots \rangle \\ \text{SLASH} & \boxed{2} \cup \left\{\boxed{3}\right\} \end{bmatrix}$$

[7] See Keller (1995), Van Eynde (1996) and Bouma (1996), for alternative accounts, arguing that extraposition should be analyzed as a nonlocal dependency. The proposal of Van Eynde does not use lexical rules.

[8] $\langle H|T\rangle$ denotes the list with head H and tail T.

While most lexical rules have some effect on the phonological form of a word, there are also a number of rules where the phonological form of the input is simply identical to that of the output. This is true, for instance, for the extraposition (6) and extraction (9) lexical rules presented above. In many implemented grammar formalisms (e.g. ALE, Carpenter (1992a)) such lexical rules are treated as unary syntax rules, which happen to give rise to words, instead of phrases. Apart from questions about the linguistic adequacy of such an approach, one may question the effects it will have on processing. Unary rules may, if they can be applied recursively, give rise to non-branching derivations of arbitrary depth. Termination of the parse-process is typically not guaranteed in such cases.

In fact, Carpenter (1991) proves that even the simplest grammar formalism employing complex symbols for representing syntactic valence is in principle undecidable, if it includes a lexical rule component. His proof rests on the observation that linguistic adequacy suggests that lexical rules must be able to insert, delete, and permute elements of a valence-list such as COMPS, and furthermore, that lexical rules must be able to apply recursively to their own output. Taken together, these two assumptions lead to the conclusion that lexical rules can be used to perform arbitrary computations by manipulating a list-valued feature. A formalism containing such a rule component is therefore able to recognize all recursively enumerable languages, and thus is undecidable. Note that recursive rule application is necessary in any case for the adjunct lexical rule, and also for the extraction lexical rule, if multiple extractions are to be accounted for (as may be the case for the Scandinavian languages and for so-called *easy*-constructions in English).

Even if termination cannot be guaranteed in general, it is often possible for a given grammar and set of lexical rules to obtain a parser that will be able to deal with unary rule application. In van Noord and Bouma (1994), for instance, it is argued that a recursive lexical rule such as the adjuncts lexical rule can be dealt with if one employs a logic programming technique known as *delayed evaluation*. A variant of this approach is proposed in Meurers and Minnen (1997). They argue that standard lexical rules can be compiled automatically into a constraint on lexical entries, in which all possible rule interactions are encoded. Furthermore, information shared between rule inputs and outputs can be propagated 'upward', from input to output, before any of the actual rule applications are carried out.

As the result of (1991) illustrates, lexical rules are procedural devices not unlike transformations in transformational grammar. A lexical rule may perform arbitrary, unconstrained operations on feature structures, apply any number of times and in any order (often leading to spurious ambiguity), which gives rise to a number of formal and computational problems. Given the

(12) a. $r_{adj}(\boxed{1}, \begin{bmatrix} \text{NUM} & sg \\ \text{CASE} & gen \vee acc \\ \text{DECL} & weak \end{bmatrix}, \langle \boxed{1}, \texttt{en} \rangle)$

b. $r_{adj}(\boxed{1}, \begin{bmatrix} \text{NUM} & sg \\ \text{GEN} & fem \\ \text{CASE} & nom \vee acc \end{bmatrix}, \langle \boxed{1}, \texttt{e} \rangle)$

A given lexical entry such as (13) can now satisfy the constraint in (11), among others, as shown in (14a,b).

(13) $\begin{bmatrix} adj\text{-}word & \\ \text{STEM} & rot \end{bmatrix}$

(14) a. $\begin{bmatrix} adj\text{-}word & \\ \text{PHON} & \langle rot,en \rangle \\ \text{STEM} & rot \\ \text{HEAD} & \begin{bmatrix} \text{INFL} & \begin{bmatrix} \text{NUM} & sg \\ \text{CASE} & gen \vee acc \\ \text{DECL} & weak \end{bmatrix} \end{bmatrix} \end{bmatrix}$

b. $\begin{bmatrix} adj\text{-}word & \\ \text{PHON} & \langle rot,e \rangle \\ \text{STEM} & rot \\ \text{HEAD} & \begin{bmatrix} \text{INFL} & \begin{bmatrix} \text{NUM} & sg \\ \text{GEN} & fem \\ \text{CASE} & nom \vee acc \end{bmatrix} \end{bmatrix} \end{bmatrix}$

Krieger and Nerbonne argue that paradigmatic information can be incorporated into the lexical entries (following the inheritance-based approach to inflection presented in Evans and Gazdar (1989)), thus opening the possibility for dealing in a principled manner with exceptional cases. The constraint-based approach to inflection can also be extended to related phenomena. Miller and Sag (1997), for instance, present a constraint-based analysis of French cliticization.

ture corresponding to a word or lexeme, see Riehemann (1994), Manning and Sag (1995), and others. Yet, we believe there is an important difference. The approach of Kathol does not distinguish between stems, lexemes or roots, and derived or inflected words. Rather, his presentation suggests that the passive and active forms of a verb are specializations of a general type. Thus, even though one might argue that the *passive* type encodes the effect of rule-application, it is difficult to envisage something like successive, or even cyclic, rule application. The only way in which lexical rules could be 'applied' to the structures introduced by Kathol is by refining or specializing these structures, and thus, rule application essentially is restricted to monotonic application of constraints to a single structure. We believe that this is the hallmark of a declarative account.

Manning and Sag (1995) argue that argument structure (ARG-ST) is a level of representation that should be distinguished from valence (i.e. the features SUBJ, SPR and COMPS). In the default case, the relationship between the two may be transparent (i.e. ARG-ST is typically the `append` of SUBJ and COMPS, for verbs), but this is by no means the only possibility. They argue that passive and ergative constructions in a variety of languages can be analyzed in terms of a noncanonical relationship between ARG-ST and valence. Van Noord and Bouma (1997) argue that an account of reflexive binding in the context of argument composition verbs provides another argument for distinguishing between argument structure and valence.

Interestingly, the introduction of ARG-ST makes it possible to restate a number of lexical rules as constraints, more or less along the lines of Kathol's analysis of passive (where ARG-ST takes over the role of the *proto* features). Bouma (1997) and Bouma et al. (1997) argue that the adjuncts and extraction lexical rules may be eliminated. Instead, the effect of these rules is incorporated in the relational constraint which defines the mapping between valence and argument structure. For instance, adopting the lexicalist approach to unbounded dependencies outlined in Sag (1997), in which complement extraction simply amounts to realizing certain complements as *gaps*, Bouma et al. (1997) demonstrate that one may incorporate the effect of complement extraction as part of the constraint defining the mapping between argument structure and COMPS. The mapping constraint in (16) realizes the tail of ARG-ST as complements, with the exception of those arguments that are instantiated as *gaps*.

(16) $$\begin{bmatrix} \textit{verb-wrd} & \\ \textsc{subj} & \langle \boxed{1} \rangle \\ \textsc{comps} & \boxed{2} - \textit{list}(\textit{gap}) \\ \textsc{arg-st} & \langle \boxed{1} \rangle \oplus \boxed{2} \end{bmatrix}$$

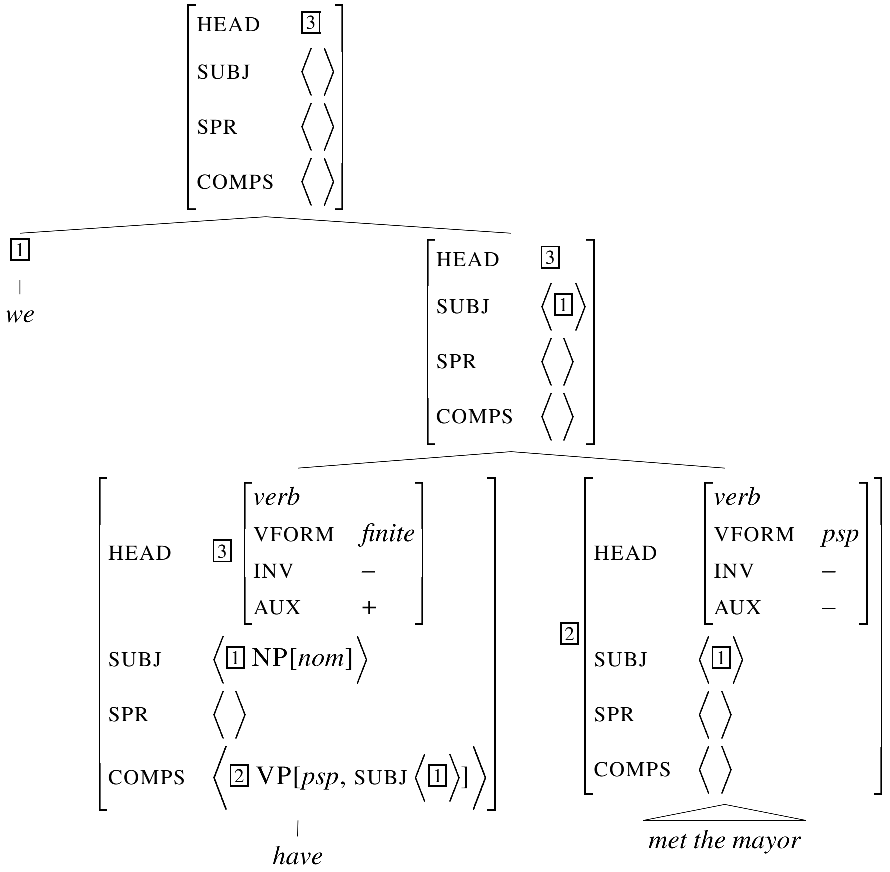


Figure 4. HPSG analysis of *We have met the mayor*

negation, inversion, contraction and ellipsis (the NICE properties). In this case study, it is the inversion property which we will focus on.

Roughly speaking, the auxiliaries are the only English verbs which can precede their subject. Compare, for instance, the auxiliary *have* with the main verb *put*.

(17) a. Where have you put your car?

b. * Where put you your car?

This observation, though, has to be qualified in two ways. First, we should make a distinction between Subject-Aux inversion and stylistic inversion, for the latter is also allowed for main verbs.

AUX	present	pres-neg	past	past-neg	subj	imp-neg
Central Modals	can	can't	could	couldn't		
	may	[mayn't]	might	mightn't		
	must	mustn't				
	shall	[shan't]	should	shouldn't		
	will	won't	would	wouldn't		
Marginal Modals	dare	daren't	dared	[daredn't]		
	need	needn't				
			ought	oughtn't		
			used	[use(d)n't]		
Do	do	don't	did	didn't		don't
	does	doesn't				
Have	have	haven't	had	hadn't	have	
	has	hasn't				
Copula	are	aren't	were	weren't	be	
	is	isn't	was	wasn't		
	am					

Figure 5. The invertible forms of the auxiliaries

As for the imperative forms, invertibility is harder to test, since they usually occur without subject. However, if there is one, it turns out that *don't* is the only form which may—and actually must—precede it.

(22) a. Don't you ever say that again!

b. *You don't ever say that again!

c. *Do you behave, please.

d. *Have you some patience, please.

e. *Be you quiet, please.

The sentences with *do*, *be* and *have* are grammatical if the subject *you* is dropped, but then we no longer have inversion, of course. Notice also that *be*

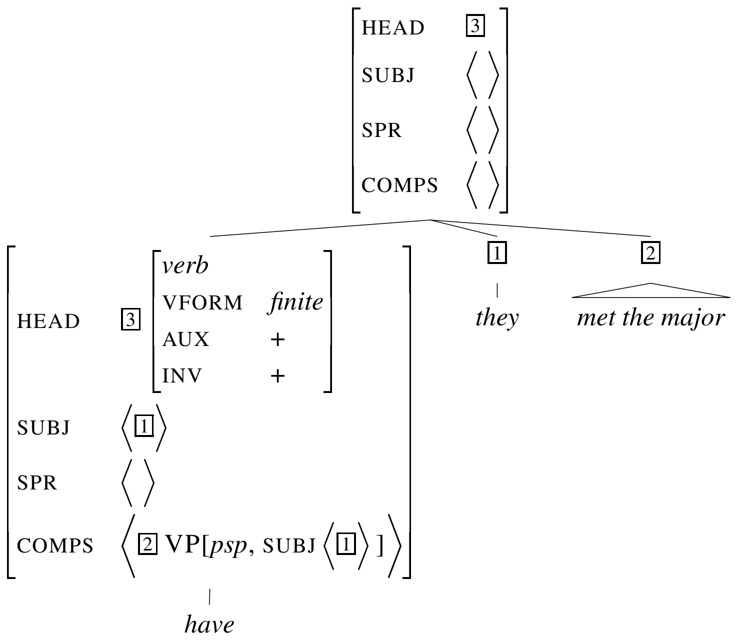


Figure 6. HPSG analysis of *Have they met the mayor?*

(23) a. Aren't I pretty?

b. * I aren't pretty.

Another concerns the meaning of *shall* in

(24) a. I shall go downtown.

b. Shall I go downtown?

While the [–INV] *shall* expresses futurity, its [+INV] counterpart "has a deontic sense, essentially equivalent to that of *should*." (o.c., p. 611)

A third example concerns the scope of the negation in *mightn't*.

(25) a. Mightn't Kim go?

b. Kim mightn't go.

Whereas the negation in the [+INV] *mightn't* has wide scope, the one of its [–INV] counterpart has narrow scope. Not mentioned in Gazdar et al. (1982), but equally telling is the case of *mustn't*.

At first sight, this approach looks better than the one of underspecification, for since it assigns separate AVMs to the inverted auxiliaries and their non-inverted counterparts, it is –at least in principle– better equipped to deal with their discrepancies. In practice, though, it does not fare any better, for since the information which is not explicitly mentioned in the lexical rule is considered to be identical for both the input and the output (see section 3.3), the prediction is that the inverted auxiliaries share all properties of the [–INV] ones, apart from the values of INV, SUBJ and COMPS. As a consequence, the rule incorrectly predicts that the inverted *aren't* is only used in the plural and the second person, that *shall I XP* has the same meaning as *I shall XP*, and that the inverted *mustn't*, *shouldn't* and *mightn't* have narrow scope negation.

These deficiencies can be repaired, of course: it is always possible to add whatever stipulations are needed to fit the facts, but then it is worth looking at the nature of those stipulations. The one which is needed for the negated modals will have to undo the constraint that the negation has narrow scope, and the one which is needed for *shall* will have to change its meaning; both of these stipulations are nonmonotonic. Moreover, the one which is needed for *aren't* involves the addition of a separate entry for the first person singular use, so that *aren't* is assigned three AVMs (the [–INV] one, the derived [+INV] one and the non-derived [+INV] one), which is one more than would be needed in a treatment which simply acknowledges that [+INV] and [–INV] *aren't* are different, equally basic entries.

What this suggests, is a radically different view on the relation between inverted and non-inverted auxiliaries. Instead of taking their similarities as the norm, and resorting to ad-hoc stipulations whenever there is a discrepancy, it would be preferable to take their differences as the starting point and to leave the capturing of their similarities to some specific constraints. This is the approach which will be adopted in the next paragraph.

5.4. *Constraint-based inheritance*

In contrast to the other methods, which could be exemplified with existing proposals, this method has not yet been applied to the phenomenon of Subject-Aux inversion. The following is, hence, novel.

For a start, let us assume the (partial) hierarchy of lexical types in (27).

As pointed out in section 2, the lexical items of type *verb* belong to the part of speech *verb* and have an ARG-ST value which equals the append of their SUBJ and COMPS values (the Argument Realization Principle).

The ordinary verbs include both the main verbs, which are [– AUX], and the non-inverted auxiliaries, which are [+ AUX].[16]

Given this hierarchy, most of the auxiliaries have two AVMs, i.e. one of type *operator* and one of type *aux*. Since these AVMs are not related by lexical rule, but equally basic, there is no implication—neither explicit nor implicit—that they have the same properties. On the contrary, the initial assumption is that their properties are different. This does not lead to a high degree of redundancy, though, since the properties which they have in common can be captured in terms of constraints and multiple inheritance. The fact that the inverted *shouldn't* has the same argument structure as its non-inverted homonym, for instance, can be stated independently of the question of whether the first argument is realized as a subject or as a complement.

An unusual property of the present treatment is that it predicts more commonalities between the [– INV] auxiliaries and the main verbs than between the [– INV] auxiliaries and the operator verbs. As such, it is diametrically opposed to the predictions which are made in the underspecification treatment. Strange as this may seem at first, this might well be an asset, since it provides a natural explanation for the fact that a number of the auxilaries which are commonly used in non-inverted clauses, are hardly ever used as operators, especially in American English. Gazdar et al. (1985, p. 65) mentions *ought* and *might*, but the list can be extended with *dared* and *used*, and with the contracted negatives *mustn't, mightn't, needn't, daren't* and *oughtn't*. In the underspecification treatment, this discrepancy would have to be interpreted as a loss of generality, but within the present treatment, it is naturally interpreted as a gain of generality, since it amounts to the elimination of an ambiguity in the lexicon.

In sum, for the treatment of Subject-Auxiliary inversion in English, the combined use of multiple inheritance and implicational constraints appears to yield better results than either underspecification or lexical rules. As pointed out at the beginning of the section, this does not mean that this approach is intrinsically better than the other two. On the contrary, as a means for capturing lexical generalizations, underspecification will always be the most straightforward method, but there are cases in which it leads to overgeneralization, and in such cases the addition of finer-grained distinctions in the lexical hierarchy appears to yield better results.

[16] The non-inverted auxiliaries can still precede the subject, but only as the result of stylistic inversion, as in *By 'strategy' is meant the basic planning of the whole operation*, cf. Quirk et al. (1985, p. 1380-1).

Flickinger, D. P., C. Pollard, and T. Wasow: 1985, 'Structure Sharing in Lexical Representation'. In: *23th Annual Meeting of the Association for Computational Linguistics*.

Gazdar, G.: 1981, 'Unbounded Dependencies and Coordinate Structure'. *Linguistic Inquiry* **12**, 155–84.

Gazdar, G., E. Klein, G. Pullum, and I. Sag: 1985, *Generalized Phrase Structure Grammar*. Oxford: Basil Blackwell.

Gazdar, G., G. Pullum, and I. Sag: 1982, 'Auxiliaries and related phenomena in a restrictive theory of grammar'. *Language* **58**, 591–638.

Götz, T. and W. D. Meurers: 1997, 'Interleaving universal principles and relational constraints over typed feature logic'. In: *Proceedings of the 35th Annual Meeting of the ACL and the 8th Conference of the EACL*. Madrid, Spain.

Huddleston, R.: 1974, 'Further remarks on the analysis of auxiliaries as main verbs'. *Foundations of Language* **11**, 215–229.

Jackendoff, R.: 1975, 'Morphological and semantic regularities in the lexicon'. *Language* **51**, 639–671.

Kathol, A.: 1994, 'Passive without Lexical Rules'. In: J. Nerbonne, K. Netter, and C. Pollard (eds.): *German in Head-driven Phrase Structure Grammar*. Stanford, pp. 237–272.

Keller, F.: 1995, 'Towards an Account of Extraposition in HPSG'. In: *Proceedings of the EACL*. Dublin.

Kim, J.-B. and I. A. Sag: 1995, 'The Parametric Variation of English and French Negation'. In: *Proceedings of the West-Coast Conference on Formal Grammar 1995*.

Krieger, H.-U. and J. Nerbonne: 1993, 'Feature-Based Inheritance Networks for Computational Lexicons'. In: T. Briscoe, V. de Paiva, and A. Copestake (eds.): *Inheritance, Defaults, and the Lexicon*. Cambridge: Cambridge University Press, pp. 90–136.

Lascarides, A., T. Briscoe, N. Asher, and A. Copestake: 1996, 'Order Independent and Persistent Typed Default Unification'. *Linguistics and Philosophy* **19**(1), 1–89.

Manning, C. and I. Sag: 1995, 'Dissociations between Argument Structure and Grammatical Relations'. Draft, Stanford University, July 1995.

Manning, C., I. Sag, and M. Iida: 1997, 'The Lexical Integrity of Japanese Causatives'. In: R. Levine and G. Green (eds.): *Readings in HPSG*. Cambridge: Cambridge University Press. to appear.

Mastroianni, M. and B. Carpenter: 1994, 'Constraint-based Morpho-Phonology'. In: *Proceedings of the first meeting of the ACL-SIG in Computational Phonology*. Las Cruces, NM, pp. 13–24.

Meurers, W. D. and G. Minnen: 1997, 'A computational treatment of lexical rules in HPSG as covariation in lexical entries'. *Computational Linguistics* **23**(4), 543–568.

Miller, P.: 1992, *Clitics and Constituents in Phrase Structure Grammar*. New York: Garland.

Miller, P. and I. Sag: 1997, 'French Clitic Movement without Clitics or Movement'. *Natural Language and Linguistic Theory*. to appear.