Monotonic Lexical Rules

Gosse Bouma
Vakgroep Alfa-informatica
Rijksuniversiteit Groningen
gosse@let.rug.nl

January 15, 1997

1 Introduction

Imagine a constraint-based, lexicalist, grammar formalism, i.e. Head-driven Phrase Structure Grammar, without lexical rules. In such a framework, all generalizations in the lexicon must be expressed by means of constraints which (subsets of) lexical entries must satisfy. In this paper, I will argue that such a formalism is simpler as well as linguistically more adequate than formalisms which use a combination of constraints and lexical rules.

Recent work in Head-driven Phrase Structure Grammar (Pollard and Sag, 1994) (1) has emphasized the role of argument structure (ARG-S) as a level of representation independent of valency (Manning and Sag, 1995), (2) has demonstrated that by imposing relational, recursive, constraints on lexical entries one may obtain accounts of linguistic phenomena which are superior to previous accounts (see, for instance, the work of Hinrichs and Nakazawa (1994) on verb complexes and complex predicates, of Sag (1995a) on unbounded dependency constructions, and Pollard and Yoo (1996) on quantifier scope), and (3) has used 'valency changing' (unary) lexical rules to describe lexical phenomena which interact in complex ways with syntax, such as clitization (Miller and Sag, 1996), or even phenomena which were previously considered to be a matter of syntax only (such as unbounded dependency constructions (Sag, 1995a) or the distribution and scope of adjuncts (Manning, Sag, and Iida, 1996; van Noord and Bouma, 1994).

Below, I argue that a combination of (1) and (2) makes (3) superfluous. In particular, the 'valency changing' lexical rules for complement extraction, introduction of adjuncts on COMPS, and clitization can be replaced by monotonic constraints on lexical entries which define the relationship between subcategorization requirements and argument structure, between argument structure and valency (including SLASH), and between argument structure and phonology.

The formal underpinnings of lexical rules in HPSG appear to be less clearly understood than other parts of the formalism. Pollard and Sag (1994, p. 395), for instance, point out that,while the procedural operation of lexical rules us well-understood, a declarative formalization remains to be provided:

> The fundamental difficulty here is that lexical rules must be seen as implicative relationships between lexical entries; but lexical entries themselves are constraints on feature structures (not feature structures themselves), so evidently a higher-order formalism must be developed within which such relationships can be expressed.

This difficulty by itself already suggests that there is something to be gained by eliminating lexical rules. There are other advantages, however. The elimination of unary, 'valency changing', lexical rules implies that the lexicon is to a large extent organized monotonically, i.e. there are no rules which apply to a single lexical item to derive a lexical item that is incompatible with the original item. Monotonicity implies order independence, which in its turn implies that a number of problems inherent to the use of (unary) lexical rules disappear. In particular, monotonicity and order independence provide an immediate solution for the following problems associated with the use of lexical rules:

Spurious ambiguity. Unordered lexical rules almost always imply that certain derived items can be obtained in more than one way. In the monotonic approach this is avoided, as there are no derivations and application of monotonic constraints is order independent.[1]

Default mechanisms. Lexical rules often appeal to default mechanism which ensures that information not mentioned in the rule is shared between input and output. The precise formulation of such a mechanism is difficult.

Integration with the lexical type hierarchy. The discussion of lexical rules in Pollard and Sag (1987) suggests that instead of defaults, one might use the type hierarchy to define the output in a non-redundant way. However, later work on lexical rules does not always seem to make this assumption.

Defining applicability. It has been a matter of debate whether lexical rules should be applicable whenever the input lexical entry is unifiable with the input conditions of the rule, or only when the input lexical entry subsumes the input conditions. Such discussions are irrelevant in a setting where there are no derivations.

Motivating lexical rules. It is often unclear why certain phenomena are modelled using a lexical rule, whereas others are modelled by means of a relational constraint. For (traceless) complement extraction, for instance, a complement extraction lexical rule can be formulated (as in Sag (1995a), but it seems there is no formal or empirical argument against an account which achieves the same effect by imposing constraints on (underspecified) lexical entries. (In fact, this is what shall be proposed below.) Complement inheritance (Hinrichs and Nakazawa, 1994), on the other hand, is presented as a constraint on valency, and not as the result of applying a lexical rule (i.e. something like the division rule of Categorial Grammar). If the use of constraints is consistently preferred over the use of lexical rules, the least one may hope for is a more uniform organization of lexical information.

A formalism which dispenses with unary lexical rules, and adopts monotonic inheritance of constraints as its only means for structuring the lexicon, has other advantages as well:

Increasing the scope of lexical constraints. In a theory with lexical rules, the scope of 'mapping principles' defining the relation between argument structure and valency is restricted to 'basic, underived' lexical entries. Rules such as the complement extraction lexical rule do not respect the canonical mapping relation. In the monotonic approach, all (major category, unexceptional) lexical entries must satisfy the same 'mapping constraints'. Thus, the scope of such constraints is widened and the motivation for an independent level of argument structure is strengthened.

Limiting the scope of variation for lexical entries. Adopting monotonicity implies that each constraint applying to a lexical entry must be satisfied by all instances of that entry. Thus, each constraint independently imposes a limit on the scope of variation to be found for a given lexical entry.

The structure of the rest of this paper is as follows. In the next section, a number of lexical rules are presented which play a crucial role in recent accounts of unbounded dependencies, the interaction between scope of adjuncts and complex predicates, and pronominal clitization in the French. The examples are chosen because they illustrate that lexical rules are an essential part of existing HPSG accounts of a diverse range of linguistic phenomena. At the same time, they illustrate most of the problematic aspects of lexical rules referred to above.

[1] Note that this particular problem can also be avoided by providing a declarative characterization of lexical rules (i.e. defining them as a kind of closure operator over lexical entries).

In the remainder of the paper, constraint-based alternatives for these lexical rules are developed. The starting point is a hierarchically organized lexicon (section 3), in which relational and possibly recursive constraints are used to express such principles as '(canonical) mapping' (of ARG-S onto valency features) and 'slash amalgamation'. The following sections introduce modifications of the lexical constraints, which account for complement extraction (section 4), addition of adjuncts on ARG-S (and COMPS) and its interaction with complement extraction (section 5), and clitization for non complement-inheritance verbs (section 6). Section 7 is devoted to complement inheritance. I argue that a sufficiently general statement of the mapping principle can capture the behaviour of complement inheritance verbs as well. This allows an account of clitization without lexical rules that extends to complement inheritance verbs.

2 Lexical Rules in HPSG

Lexical rules are of central importance in recent HPSG accounts of a rather diverse range of data. Below, I concentrate on lexical rules which have been proposed in accounts of complement extraction, the distribution and scope of adjuncts (in the context of complex predicates), and clitization. These rules illustrate very well what can be done with lexical rules. The complement extraction lexical rule, for instance, allows for an account of unbounded dependencies which does not employ traces. The lexical rule which adds adjuncts to COMPS can be used recursively, and thus gives rise to an infinite lexicon. The clitization lexical rules, finally, illustrate how a combination of lexical rules, complement inheritance, and the principles governing the value of SLASH can provide an account for a highly complex set of morphosyntactic data.

Although other valency changing lexical rules have been proposed,[2] the rules in this section can be considered to be representative for the way unary lexical rules are currently being used in HPSG.In the second half of this paper, each of these lexical rules will be eliminated in favour of one or more monotonic constraints which achieve the same effect as the corresponding lexical rule. I argue that this constraint-based perspective not only avoids some of the drawbacks of using lexical rules (in particular, it avoids the spurious ambiguity and default matching or type inheritance issues), but also leads to a clearer perspective on the interaction between these rules, and with other features of the grammar (such as complement inheritance).

2.1 Complement Extraction

Chapter 9 of Pollard and Sag (1994) (a.k.a. HPSG 3) contains a first attempt at a *traceless* account of extraction in HPSG. Instead of traces, a COMPLEMENT EXTRACTION lexical rule is presented which has the effect of deleting an element from COMPS, and placing its LOCAL value on SLASH:[3]

(1) **Complement Extraction Lexical Rule (Pollard and Sag, 1994, p. 378):**

$$\begin{bmatrix} \text{SUBCAT} & \left\langle \ldots \boxed{3} \ldots \right\rangle \\ \text{COMPS} & \left\langle \ldots \boxed{3}\begin{bmatrix}\text{LOC} & \boxed{1}\end{bmatrix} \ldots \right\rangle \\ \text{INHER}\,|\,\text{SLASH} & \boxed{2} \end{bmatrix} \mapsto \begin{bmatrix} \text{SUBCAT} & \left\langle \ldots \boxed{4}\begin{bmatrix}\text{LOC} & \boxed{1} \\ \text{INHER}\,|\,\text{SLASH} & \left\{\boxed{1}\right\}\end{bmatrix} \ldots \right\rangle \\ \text{COMPS} & \left\langle \ldots\ldots \right\rangle \\ \text{INHER}\,|\,\text{SLASH} & \left\{\boxed{1}\right\} \cup \boxed{2} \end{bmatrix}$$

The diagram in (1) is only an informal characterization of the lexical rule for complement extraction. The relationship between the indices $\boxed{3}$ and $\boxed{4}$, for instance, is described in the accompanying text:

[2] for instance, for passive (Pollard and Sag, 1987), subject extraction, adjunct extraction, and *it*-extraposition (Pollard and Sag, 1994))

[3] The feature SUBCAT is renamed ARG-S (or ARG-ST) in more recent work. The distinction between TO-BIND and INHER(ited) nonlocal feature values is abandoned in Sag (1995a).

The intended interpretation of the lexical rule is that all occurrences of $\boxed{3}$ in the input (except for the one on the COMPS list, which is eliminated on the output) are to be replaced in the output by a specification $\boxed{4}$, which is exactly like $\boxed{3}$ except that it bears the additional specification $[\text{INHER}|\text{SLASH } \boxed{1}]$.

In Sag (1995a) and Sag (1995b) a simpler formulation of the COMPLEMENT EXTRACTION lexical rule is given. The simplification is a direct consequence of the head-driven account of *slash amalgamation.*

Previous nontransformational theories of extraction following the work of Gazdar (1981) have all assumed some version of the Nonlocal (or Foot) Feature Principle: The value of a nonlocal feature F on the mother of a phrase is (by default) the union of the value of this feature on the daughters. Sag (1995a) replaces this principle by a lexicalist, head-driven, alternative:

(2) **Slash Amalgamation:**
For a canonical lexical sign, the value of a nonlocal feature F is the (nonvacuous set) union of this value on all its ARG-S elements:

$$\begin{bmatrix} \text{ARG-S} & \left\langle \begin{bmatrix}\text{SLASH } \boxed{1}\end{bmatrix}, \ldots, \begin{bmatrix}\text{SLASH } \boxed{n}\end{bmatrix} \right\rangle \\ \text{SLASH} & \boxed{1} \uplus \ldots \uplus \boxed{n} \end{bmatrix}$$

Thus, the value of SLASH on a lexical head H is determined by the value of SLASH on its arguments. An immediate consequence of this assumption is that the Nonlocal Feature Principle can be abandoned. Instead, it is required that in HEAD-VALENCE phrases, the value of SLASH is identical on mother and head-daughter. Head-filler phrases are subject to the constraint that the local features of their filler daughter must unify with the single element on SLASH of the head-daughter.

Another consequence of *slash amalgamation* is that the CELR can be simplified. The new CELR only changes the value of COMPS (the $\bigcirc$ operator denotes sequence union, and is a formal rendering of the 'dots'-notation used in Pollard and Sag (1994)):

(3) **Complement Extraction Lexical Rule Sag (1995a):**

$${}_{word}\begin{bmatrix}\text{COMPS} & \boxed{1} \bigcirc \left\langle \begin{bmatrix}\text{LOC} & \boxed{2} \\ \text{SLASH} & \left\{\boxed{2}\right\}\end{bmatrix}\right\rangle\end{bmatrix} \mapsto \begin{bmatrix}\text{COMPS} & \boxed{1}\end{bmatrix}$$

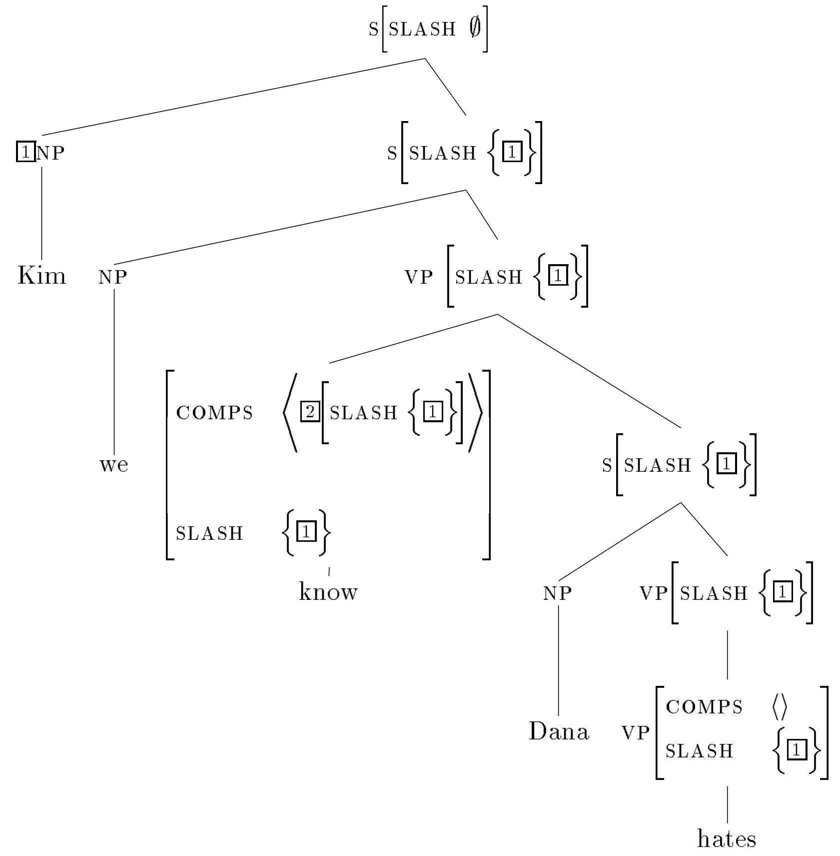
Since (canonical) lexical items no longer have a null-value for SLASH, but instead define their SLASH value in terms of the SLASH values of their ARG-S elements, removing an element with a non-empty SLASH value from COMPS will have the effect of instantiating the same element on ARG-S with a non-empty SLASH value as well, which in its turn (because of *slash amalgamation*) has the effect of ensuring a non-empty SLASH-value for the sign to which this rule applies. An example is given below:

(4)
$$\begin{bmatrix} \text{PHON} & \textit{hates} \\ \text{LOC} & \begin{bmatrix} \text{HEAD} & \begin{bmatrix}\textit{verb} \\ \textit{fin}\end{bmatrix} \\ \text{SUBJ} & \left\langle \boxed{1}\text{ NP} \right\rangle \\ \text{COMPS} & \left\langle \boxed{2}\text{NP}\begin{bmatrix}\text{LOC } \boxed{3}\end{bmatrix} \right\rangle \end{bmatrix} \\ \text{ARG-S} & \left\langle \boxed{1}\begin{bmatrix}\text{SLASH } \boxed{4}\end{bmatrix}, \boxed{2}\begin{bmatrix}\text{LOC} & \boxed{3} \\ \text{SLASH} & \boxed{5}\end{bmatrix} \right\rangle \\ \text{SLASH} & \boxed{4} \uplus \boxed{5} \end{bmatrix} \mapsto \begin{bmatrix} \text{PHON} & \textit{hates} \\ \text{LOC} & \begin{bmatrix} \text{HEAD} & \begin{bmatrix}\textit{verb} \\ \textit{fin}\end{bmatrix} \\ \text{SUBJ} & \left\langle \boxed{1}\text{ NP} \right\rangle \\ \text{COMPS} & \langle\rangle \end{bmatrix} \\ \text{ARG-S} & \left\langle \boxed{1}\begin{bmatrix}\text{SLASH } \boxed{4}\end{bmatrix}, \boxed{2}\begin{bmatrix}\text{LOC} & \boxed{3}[\textit{acc..}] \\ \text{SLASH} & \left\{\boxed{3}\right\}\end{bmatrix} \right\rangle \\ \text{SLASH} & \boxed{4} \uplus \left\{\boxed{3}\right\} \end{bmatrix}$$

Removing the object of *hates* from COMPS has the effect that the SLASH value of the verb must contain at least theLOCAL value of this NP as an element.

Together, these innovations allow for derivations such as the following:

(5)



A number of linguistic advantages of this analysis (which, apart from arguments against the use of traces, involve among others the analysis of *an easy man to please*, instances of WH-fronting for which there appears to be no non-fronted counterpart, and the analysis of clitization in Romance) are discussed in Sag and Fodor (1994), Sag (1995a), Sag (1995b), and Miller and Sag (1996).

2.2 Adding Adjuncts

Following a proposal in Miller (1992), a number of authors (van Noord and Bouma, 1994; Iida et al., 1994; Manning, Sag, and Iida, 1996), have proposed a lexical rule which adds adjunct phrases to COMPS or argument structure (or both). The advantages of using a lexical rule of this kind is that it accounts for the distribution of adjuncts in languages such as Dutch and German, and for the scope of adjuncts in the context of 'complex predicates'. A striking feature of this type of rule is the fact that there appears to be no upper limit on the number of times it can apply to its own output (as phrases may contain an arbitrary number of adjuncts).

In its simplest form, a rule which introduces adjuncts as elements of ARG-S could be something like the following:[4]

[4] This formulation is derived from Iida et al. (1994), who do not actually give this rule, but present the following example of its application (to the Japanese verb *kaw* (*buy*)):

$$\begin{bmatrix} \text{PHON} & \textit{kaw} \\ \text{ARG-S} & \left\langle \text{NP}_i,\ \text{NP}_j \right\rangle \\ \text{CNTNT} & \text{buy}(i,j) \end{bmatrix} \mapsto \begin{bmatrix} \text{PHON} & \textit{kaw} \\ \text{ARG-S} & \left\langle \text{NP}_i,\ \text{NP}_j, \text{ADV} \right\rangle \\ \text{CNTNT} & \textit{adv}'(\text{buy}(i,j)) \end{bmatrix}$$

(6) $$\begin{bmatrix} \text{ARG-S} & \boxed{1} \\ \text{CNTNT} & \boxed{2} \end{bmatrix} \mapsto \begin{bmatrix} \text{ARG-S} & \boxed{1} \oplus \left\langle \text{ADV} \right\rangle \\ \text{CNTNT} & adv'(\boxed{2}) \end{bmatrix}$$

This (optional) lexical rule adds an adjunct to ARG-S, and modifies the semantics of the lexical entry to which it applies, in order to account for the semantic effect of adverbial modification.

A slightly different version of such a rule is given in van Noord and Bouma (1994). As there is no *a priori* upper limit on the number of adjuncts which must be added by means of a rule for adding adjuncts, they suggest a rule which, in one rule application, allows for an arbitrary number of adjuncts to be added to COMPS. Ignoring semantics, this proposal amounts to the following:

(7) $$\begin{bmatrix} \text{COMPS} & \boxed{1} \end{bmatrix} \mapsto \begin{bmatrix} \text{COMPS} & \boxed{1} \bigcirc \mathit{listof(adv)} \end{bmatrix}$$

That is, the output of this adjunct lexical rule adds an arbitrary number of adjuncts to COMPS, which are sequence unioned with the original COMPS-list. Note that this rule removes the need for recursive rule application, which is crucial for the computational treatment of such rules (see van Noord and Bouma (1994) for details).

The most important advantage of using a lexical rule for selecting of adjuncts is that it provides an account for the distribution as well as the scope of adjuncts. The problem is illustrated in the following example:

(8) ...dat Arie Bob de vrouwen met een telescoop zag bekijken
...that Arie Bob the women with a telescope saw look-at
that Arie saw Bob looking at the women with a telescope

Note that *zag* is a complement inheritance verb, i.e. it combines with the unsaturated verb *bekijken* and inherits its complements. The adjunct *met een telescoop* therefore occurs in a position between the head *zag* and its complements. This implies that the analysis of adjuncts as presented in Pollard and Sag (1994) cannot be maintained, as that analysis only allows adjuncts as sisters of saturated phrases. A solution for this word order problem has been proposed in Kasper (1994), but this still does not solve the second problem. The adjunct can take scope over the 'complex predicate' *zag bekijken* or over just the verb *bekijken*. Both the word order and scope issue are solved if adjuncts are introduced by lexical rule. If adjuncts can be anywhere on COMPS, they can be expected in all positions where complements can appear. The scope-ambiguity is accounted for by assuming that the adjunct can be introduced either on the COMPS list of *zag* or on the COMPS-list of *bekijken*.

In Iida et al. (1994), the introduction of a lexical rule for adding adjuncts is motivated by the fact that this accounts for a scope-ambiguity of adjuncts in the context of causative constructions. In Manning, Sag, and Iida (1996), which is a revision of Iida et al. (1994), a third version of a rule for adding causatives is presented. Rather than giving a traditional lexical rule, they propose the following 'zero derivation' stem type:

(9) $$_{\mathit{atr\text{-}stem}}\begin{bmatrix} \text{PHON} & \boxed{4} \\ \text{ARG-S} & \boxed{1} \oplus \left\langle \text{ADV} \begin{bmatrix} \text{CNTNT} & \boxed{3} \end{bmatrix} \right\rangle \\ \text{CNTNT} & \boxed{3}\begin{bmatrix} \text{ARG} & \boxed{2} \end{bmatrix} \\ \text{STEM} & \begin{bmatrix} \text{PHON} & \boxed{4} \\ \text{ARG-S} & \boxed{1} \\ \text{CNTNT} & \boxed{2} \end{bmatrix} \end{bmatrix}$$

Note that *atr-stem* is a lexical entry, not a rule. It has the effect of a lexical rule, however, since the attribute STEM must contain the value of a second lexical entry, and the values of the other attributes of *atr-stem* are defined in terms of this second entry. Below, I will discuss the question whether this is more than just a change in presentation.

2.3 French Clitic Movement

Pronouns in French may attach onto the verb which selects these pronouns, and sometimes onto higher, 'governing' verbs:

(10) a. Marie le voit
Marie him sees
Marie sees him

b. Marie le lui donne
Marie it her gives
Marie gives it to her

c. Marie l' a vu
Marie him has seen
Marie has seen him

d. Le livre lui a été donné
The book him has been given
The book has been given to him

Miller and Sag (1996) present a lexicalist analysis of this phenomenon. They argue that there is convincing evidence that the pronominal elements involved are not clitics (i.e. elements which attach to a host in syntax) but affixes (i.e. elements which combine with their host at the phonological or morphological level). The problem that some (auxiliary) verbs can combine with pronominal affixes which are not direct arguments of the verb, is not a counterargument for this lexicalist approach, given the assumption that the verbs which allow this are complement inheritance verbs.

Miller and Sag (1996) introduce the lexical rule in (11) to account for the phonological aspect of pronominal affixation as well as the fact that adding a pronominal affix to a verb reduces the valency of this verb (i.e. the complement corresponding to the pronominal affix must no longer be available in syntax).

(11) **Pronominal Affixation Inflection Rule:**

$$ {}_{\textit{word}}\begin{bmatrix} \text{PHON} & \boxed{0} \\ \text{SYNSEM} & \boxed{2}\begin{bmatrix} \text{HEAD} & \text{verb} \\ \text{COMPS} & \boxed{1} \bigcirc \text{nelist(aff)} \end{bmatrix} \end{bmatrix} \mapsto {}_{\textit{cl-word}}\begin{bmatrix} \text{PHON} & \textit{Fpraf}(\boxed{0},\boxed{2}) \\ \text{SYNSEM} & \begin{bmatrix} \text{COMPS} & \boxed{1}\ \text{list(canon)} \end{bmatrix} \end{bmatrix} $$

The input of the rule is a lexical verb whose COMPS-list is the sequence union ($\bigcirc$) of a number of canonical signs and a non-empty list of affixes. On the output, the COMPS list does no longer contain these affixes, while the phonology is the value of the function *Fpraf* applied to the input phonology and the SYNSEM-value of the input.

For example, given a lexical entry for the verb *donner* (to give) as in (12a), the forms in (12b) and (12c) can be derived:

(12) a.

$$ {}_{\textit{word}}\begin{bmatrix} \text{PHON} & \textit{donne} \\ \text{SUBJ} & \left\langle \boxed{1}\ \text{NP}_{1sg} \right\rangle \\ \text{COMPS} & \left\langle \boxed{2}\ \text{NP}[acc], \boxed{3}\ \text{NP}[\grave{a}] \right\rangle \\ \text{ARG-ST} & \left\langle \boxed{1}, \boxed{2}, \boxed{3} \right\rangle \end{bmatrix} $$

$$\text{b.}\quad {}_{\textit{cl-word}}\begin{bmatrix} \text{PHON} & \textit{lui-donne} \\ \text{SUBJ} & \left\langle \boxed{1}\ \text{NP}_{1sg} \right\rangle \\ \text{COMPS} & \left\langle \boxed{2}\ \text{NP}[\textit{acc}], \right\rangle \\ \text{ARG-ST} & \left\langle \boxed{1}, \boxed{2}, \text{NP}[\textit{aff, à}] \right\rangle \end{bmatrix}$$

$$\text{c.}\quad {}_{\textit{cl-word}}\begin{bmatrix} \text{PHON} & \textit{le-lui-donne} \\ \text{SUBJ} & \left\langle \boxed{1}\ \text{NP}_{1sg} \right\rangle \\ \text{COMPS} & \langle\rangle \\ \text{ARG-ST} & \left\langle \boxed{1}, \text{NP}[\textit{aff,acc}], \text{NP}[\textit{aff, à}] \right\rangle \end{bmatrix}$$

Note that the phonological forms of the derived words are computed by the function F_{PRAF}. This function, and the way it accounts for idiosyncratic constraints on the realization of pronominal affixes on inflected verb stems, is described in detail in Miller and Sag (1996).

Following the analysis of French auxiliaries in Abeillé and Godard (1994), Miller and Sag propose the following type of lexical entry for *avoir* and *être*:

$$(13)\quad \begin{bmatrix} \text{PHON} & \textit{avoir/être} \\ \text{SUBJ} & \left\langle \boxed{1} \right\rangle \\ \text{COMPS} & \left\langle \boxed{2} \begin{bmatrix} \text{HEAD} & \textit{v[psp]} \\ \text{SUBJ} & \left\langle \boxed{1} \right\rangle \\ \text{COMPS} & \boxed{3} \end{bmatrix} \right\rangle \oplus \boxed{3} \\ \text{ARG-ST} & \left\langle \boxed{1}, \boxed{2} \right\rangle \oplus \boxed{3} \end{bmatrix}$$

Note that an auxiliary subcategorizes for a (possibly unsaturated) passive participle verb, and for all the complements on the COMPS list of this verb. Furthermore, these 'inherited' complements are added to ARG-ST. This analysis makes it straightforward to account for examples of 'clitic climbing'. Consider, for instance, the instantiation of COMPS of the (inflected) auxiliary *avons* (*have*, plural) in case it combines with the past participle *donné*:

$$(14)\quad \begin{bmatrix} \text{PHON} & \textit{avons} \\ \text{SUBJ} & \left\langle \boxed{1}\ \text{NP}_{pl} \right\rangle \\ \text{COMPS} & \left\langle \boxed{2} \begin{bmatrix} \text{HEAD} & \textit{v[psp]} \\ \text{SUBJ} & \left\langle \boxed{1} \right\rangle \\ \text{COMPS} & \left\langle \boxed{3}\ \text{NP}[\textit{acc}], \boxed{4}\ \text{NP}[\textit{à}] \right\rangle \end{bmatrix}, \boxed{3}, \boxed{4} \right\rangle \\ \text{ARG-ST} & \left\langle \boxed{1}, \boxed{2}, \boxed{3}, \boxed{4} \right\rangle \end{bmatrix}$$

This type of lexical entry can be input the the PRAF-LR, to produce forms such as *l'avons*, *le-lui-avons*, etc.

Not all types of 'clitic climbing' involve inheritance, however. Miller and Sag argue that in cases of *en*-affixation, where *en* is not a local argument of the host verb (as in example (15)), there is evidence on the one hand that this type of 'clitic climbing' is more restricted than cases involving inheritance verbs (in particular, only *en* can climb in this context), while on the other hand the 'climbing' is subject to constraints that are similar to the constraints that must be imposed on complement extraction.

(15) Marie en connaît la fin
Marie it knows the end
Marie knows the end of it

They therefore propose a second lexical rule, which, in combination with the revised account of unbounded dependencies in Sag (1995a) , allows an element on SLASH to become an element of ARG-ST, just in case it is a NP-affix with *de*-marking:

(16) **En-Affixation LR (EnLR):**

$$
{}_{lexeme}\left[\begin{array}{ll}
\text{HEAD} & \textit{verb} \\
\text{COMPS} & \boxed{1} \\
\text{ARG-S} & \boxed{2} \oplus \left\langle \text{NP}\ {}_{unm}\left[\begin{array}{ll} \text{LOC} & \boxed{3} \\ \text{SLASH} & \{\} \end{array}\right] \right\rangle \oplus \boxed{4}
\end{array}\right] \mapsto
$$

$$
\left[\begin{array}{ll}
\text{COMPS} & \boxed{1} \oplus \boxed{7} \\
\text{ARG-S} & \boxed{2} \oplus \left\langle {}_{canon}\left[\begin{array}{ll} \text{LOC} & \boxed{3} \\ \text{SLASH} & \left\{\boxed{6}\right\} \end{array}\right] \right\rangle \oplus \boxed{4} \oplus \boxed{7}\left\langle \text{NP}_{\textit{aff,de}}\left[\text{LOC} \quad \boxed{6}\right] \right\rangle
\end{array}\right]
$$

Consider, for instance, the basic lexical entry for *connaît* (17a). Applying the *en*-LR gives the result in (17b) (note that the SLASH-value $\boxed{4}$ is instantiated as the empty set in the input). This derived entry contains an affix on COMPS, which implies that the element cannot be selected in syntax (under the assumption that affixes are no be selected by means of a syntactic rule). If the verb is not selected by a complement-inheritance verb, this implies that the PRAF has to apply. This would give rise to the entry in (17c).

(17) a.
$$
\left[\begin{array}{ll}
\text{PHON} & \textit{connaît} \\
\text{SUBJ} & \left\langle \boxed{1} \right\rangle \\
\text{COMPS} & \left\langle \boxed{2} \right\rangle \\
\text{ARG-ST} & \left\langle \boxed{1}\ \text{NP}/\boxed{3},\ \boxed{2}\ \text{NP}/\boxed{4} \right\rangle \\
\text{SLASH} & \boxed{3} \uplus \boxed{4}
\end{array}\right]
$$

b.
$$
\left[\begin{array}{ll}
\text{PHON} & \textit{connaît} \\
\text{SUBJ} & \left\langle \boxed{1} \right\rangle \\
\text{COMPS} & \left\langle \boxed{2}\ , \boxed{6} \right\rangle \\
\text{ARG-ST} & \left\langle \boxed{1}\ \text{NP}/\boxed{3},\ \boxed{2}\ \text{NP}/\left\{\boxed{5}\right\},\ \boxed{6}\ {}_{\textit{aff}}\left[\text{LOC} \quad \boxed{5}\left[\text{HEAD} \quad \text{np[de]}\right]\right] \right\rangle \\
\text{SLASH} & \boxed{3}
\end{array}\right]
$$

c.
$$
\left[\begin{array}{ll}
\text{PHON} & \textit{en-connaît} \\
\text{SUBJ} & \left\langle \boxed{1} \right\rangle \\
\text{COMPS} & \left\langle \boxed{2} \right\rangle \\
\text{ARG-ST} & \left\langle \boxed{1}\ \text{NP}/\boxed{3},\ \boxed{2}\ \text{NP}/\left\{\boxed{5}\right\},\ {}_{\textit{aff}}\left[\text{LOC} \quad \boxed{5}\left[\text{HEAD} \quad \text{np[de]}\right]\right] \right\rangle \\
\text{SLASH} & \boxed{3}
\end{array}\right]
$$

3 A Lexical Hierarchy with Constraints

Below, a fragment of a hierarchically organized HPSG lexicon is presented. Instead of using type inheritance, I will assume that the lexicon is defined as the denotation of the unary predicate *lexical-entry*. This predicate can be defined as a disjunction of more specific predicates (such as *canonical-entry* or *verb*), conjoined with one or more relational constraints.

In HPSG, the lexicon is, formally, a set of signs of type *word*. Below, this set is defined by means of the unary predicate *lexical-entry*. That is, the lexicon is equivalent to the set of signs L which satisfy this unary predicate. In a non-hierarchically organized lexicon, *lexical-entry* would be defined by listing all its instances. However, it is more economically defined in terms of other unary, sign denoting, predicates (each of which denote a subset of the lexicon) and constraints. A fragment of such a definition of the lexicon is presented below.

(18) a. $\textit{lexical-entry}(\boxed{1}\begin{bmatrix}\textit{word}\end{bmatrix}) \leftarrow \textit{major}(\boxed{1}) \vee \textit{minor}(\boxed{1}) \vee \textit{exceptional}(\boxed{1})$

b. $\textit{major}(\boxed{1}\begin{bmatrix}\text{ARG-ST} & \boxed{2}\\ \text{SLASH} & \boxed{3}\end{bmatrix}) \leftarrow (\ \textit{pred}(\boxed{1}) \vee \textit{non-pred}(\boxed{1})\ ) \wedge \textit{amalgamate-slashes}(\boxed{2},\boxed{3})$

c. $\textit{pred}(\boxed{1}\begin{bmatrix}\text{ARG-ST} & \left\langle \boxed{2} | \boxed{3} \right\rangle\\ \text{SUBJ} & \left\langle \boxed{2} \right\rangle\\ \text{COMPS} & \boxed{4}\end{bmatrix}) \leftarrow (\textit{verb}(\boxed{1}) \vee \textit{pred-n}(\boxed{1}) \vee \textit{pred-a}(\boxed{1})) \wedge \textit{map-args}(\boxed{3},\boxed{4})$

d. $\textit{verb}(\begin{bmatrix}\text{PHON} & \boxed{1}\\ \text{HEAD} & v\\ \text{ARG-ST} & \boxed{2}\\ \text{CONT} & \boxed{3}\end{bmatrix}) \leftarrow \textit{verbal-lexeme}(\boxed{1},\boxed{2},\boxed{3})$

The first clause states that a feature structure of type *word* which is either major, minor, or exceptional, is a lexical entry. A stronger conclusion can be reached by making the *closed world assumption*: a sign can *only* be a lexical entry if it is of type *word*, and either major, minor, or exceptional. That is, all lexical entries are words, and either major, minor, or exceptional. By making the closed world assumption we can therefore conclude that the first clause does indeed *define* the set of lexical entries.

By the same reasoning, a major category lexical entry is either a predicative or non-predicative lexical entry, and furthermore, its values must satisfy the *amalgamate-slashes* relation, which is a constraint on the values of ARG-ST and SLASH. Predicative lexical entries have a subject, which corresponds to the first element on ARG-ST, and must satisfy the *map-args* relation, which is a constraint on the values of ARG-ST and COMPS. A verbal lexical entry, finally, must satisfy the *verbal-lexeme* relation, which is a constraint on the values of PHON, ARG-ST, and CONT.

I will now presente preliminary versions of the relational constraints used in (18). Slash amalgamation is accounted for by the *slash-amalgamation* constraint. It will be convenient to think of this constraint as a relation which determines for each element on ARG-ST how it contributes to the value SLASH. For now, I will assume, following Sag (1995a), that *slash-amalgamation* incorporates the SLASH value of each argument into the SLASH value of the sign as a whole:[5]

(19) a. `slash-amalgamation(listof(synsem),setof(loc))`

$\textit{slash-amalgamation}(\left\langle\ \right\rangle,\emptyset)$

$\textit{slash-amalgamation}(\left\langle \boxed{1} | \boxed{2} \right\rangle, \boxed{3} \uplus \boxed{4}) \leftarrow$

$\textit{amalgamate-slash}(\boxed{1},\boxed{3}) \quad \wedge \quad \textit{slash-amalgamation}(\boxed{2},\boxed{4})$

[5]Mainly for expository reasons, the types of the arguments of a relational constraint will be given below as `relation-name(type,...type)`.

b. `amalgamate-slash(synsem,setof(loc))`

$$\textit{amalgamate-slash}(\left[\textsc{slash}\quad \boxed{1}\right], \boxed{1})$$

The recursive part of the definition is handled by *slash-amalgamation*, which is defined in terms of a non-recursive constraint *amalgamate-slash*. The latter determines for each element on ARG-ST, how it contributes to SLASH. The recursive *slash-amalgamation* relation is defined by means of two clauses: the first ensures that the recursion stops if ARG-ST is empty (in which case the value of SLASH is the empty set), and the second clause defines the value of SLASH as the (non-vacuous) set-union of the SLASH value provided by *amalgamate-slash* (applied to the first element of ARG-ST) and the SLASH value obtained by calling *slash-amalgamation* for the tail of ARG-ST.[6]

The *map-args* constraint is similar to *slash-amalgamation* in that it determines for each element on ARG-ST how it contributes to the value of a second attribute, in this case COMPS. In the simplest case, each non-subject argument is added to COMPS:

(20) a. `map-args(listof(synsem),listof(synsem))`

$$\textit{map-args}(\langle\rangle,\langle\rangle)$$

$$\textit{map-args}(\langle \boxed{1} | \boxed{2} \rangle, \boxed{3} \oplus \boxed{4}) \leftarrow$$
$$\textit{map-arg}(\boxed{1},\boxed{3}) \quad \wedge \quad \textit{map-args}(\boxed{2},\boxed{4})$$

b. `map-arg(synsem,listof(synsem))`

$$\textit{map-arg}(\boxed{1}, \langle \boxed{1} \rangle)$$

Taking into account the effects of complement extraction, clitization, and *complement inheritance* will lead to redefinitions of the *map-arg* relation.

The *verbal-lexeme* constraint, finally, defines the values of PHON, ARG-ST, and CONT. Each clause can be thought of as specifying a subcategorization class. For each class, the argument structure and relationship between argument semantics and CONT is fixed. The value for PHON and the semantic relation introduced can be defined disjunctively.

(21) `verbal-lexeme(phon,listof(synsem),cont)`

$$\textit{verbal-lexeme}(\boxed{1}, \left\langle \begin{bmatrix} \textsc{head} & n \\ \textsc{cont} & \boxed{3} \end{bmatrix} \right\rangle, \boxed{2}\begin{bmatrix} \textsc{arg1} & \boxed{3} \end{bmatrix}) \leftarrow$$
$$(\ \boxed{1} = \textit{walk} \wedge \boxed{2} = \textit{walk}'\ ) \vee$$
$$(\ \boxed{1} = \textit{sleep} \wedge \boxed{2} = \textit{sleep}'\ ) \vee$$
$$\ldots$$

$$\textit{verbal-lexeme}(\boxed{1}, \left\langle \begin{bmatrix} \textsc{head} & n \\ \textsc{cont} & \boxed{3} \end{bmatrix}, \begin{bmatrix} \textsc{head} & n \\ \textsc{cont} & \boxed{4} \end{bmatrix}, \right\rangle, \boxed{2}\begin{bmatrix} \textsc{arg1} & \boxed{3} \\ \textsc{arg2} & \boxed{4} \end{bmatrix}) \leftarrow$$
$$(\ \boxed{1} = \textit{kiss} \wedge \boxed{2} = \textit{kiss}'\ ) \vee$$
$$(\ \boxed{1} = \textit{see} \wedge \boxed{2} = \textit{see}'\ ) \vee$$
$$\ldots$$

Note that the definition of *verb* strongly constrains the type of information that may be provided by an individual lexical item (i.e. a *verbal-lexeme*). It can only refer to the values of PHON, ARG-ST, and CONT, and thus, canonical verbal lexical items cannot impose specific constraints on the value of COMPS or SUBJ, or SLASH.

It is instructive to consider how the lexical entry for *kiss* could be obtained from this set of constraints. The fact (22) can be derived from the second clause given in (21):

(22) $$\textit{verbal-lexeme}(\textit{kiss}, \left\langle \mathrm{NP}_i, \mathrm{NP}_j \right\rangle, {}_{\textit{kiss}'}\begin{bmatrix} \textsc{arg1} & i \\ \textsc{arg2} & j \end{bmatrix})$$

[6] To account for *en*-clitization, I will propose a slightly more complex version below.

Resolution of (18d) with the fact above gives:

$$(23)\ \ \mathit{verb}(\left[\begin{array}{ll} \text{PHON} & \mathit{kiss} \\ \text{HEAD} & v \\ \text{ARG-ST} & \left\langle \text{NP}_i, \text{NP}_j \right\rangle \\ \text{CONT} & {}_{\text{kiss}'}\left[\begin{array}{ll} \text{ARG1} & \text{i} \\ \text{ARG2} & \text{j} \end{array}\right] \end{array}\right])$$

Resolving (23) with (18c) and (20) gives:

$$(24)\ \ \mathit{pred}(\left[\begin{array}{ll} \text{PHON} & \mathit{kiss} \\ \text{HEAD} & v \\ \text{ARG-ST} & \left\langle \boxed{1}, \boxed{2} \right\rangle \\ \text{SUBJ} & \left\langle \boxed{1}\ \text{NP}_i \right\rangle \\ \text{COMPS} & \left\langle \boxed{2}, \text{NP}_j \right\rangle \\ \text{CONT} & {}_{\text{kiss}'}\left[\begin{array}{ll} \text{ARG1} & \text{i} \\ \text{ARG2} & \text{j} \end{array}\right] \end{array}\right])$$

From this, we can conclude that:

$$(25)\ \ \mathit{major}(\left[\begin{array}{ll} \text{PHON} & \mathit{kiss} \\ \text{HEAD} & v \\ \text{ARG-S} & \left\langle \boxed{1}, \boxed{2} \right\rangle \\ \text{SUBJ} & \left\langle \boxed{1}\ \text{NP}_i \left[\text{SLASH}\ \boxed{3}\right] \right\rangle \\ \text{COMPS} & \left\langle \boxed{2}\ \text{NP}_j \left[\text{SLASH}\ \boxed{4}\right] \right\rangle \\ \text{CNTNT} & {}_{\text{kiss}'}\left[\begin{array}{ll} \text{ARG1} & \text{i} \\ \text{ARG2} & \text{j} \end{array}\right] \\ \text{SLASH} & \boxed{3} \uplus \boxed{4} \end{array}\right])$$

which in turn leads to the conclusion that:

$$(26)\ \ \mathit{lexical\text{-}entry}({}_{\mathit{word}}\left[\begin{array}{ll} \text{PHON} & \mathit{kiss} \\ \text{HEAD} & v \\ \text{ARG-S} & \left\langle \boxed{1}, \boxed{2} \right\rangle \\ \text{SUBJ} & \left\langle \boxed{1}\ \text{NP}_i \left[\text{SLASH}\ \boxed{3}\right] \right\rangle \\ \text{COMPS} & \left\langle \boxed{2}\ \text{NP}_j \left[\text{SLASH}\ \boxed{4}\right] \right\rangle \\ \text{CNTNT} & {}_{\text{kiss}'}\left[\begin{array}{ll} \text{ARG1} & \text{i} \\ \text{ARG2} & \text{j} \end{array}\right] \\ \text{SLASH} & \boxed{3} \uplus \boxed{4} \end{array}\right])$$

This bottom-up resolution proof illustrates how a full-blown lexical entry can be derived starting with a *verbal lexeme* triple containing only a phonological form, a subcategorization frame, and a semantics.

4 Complement Extraction without Lexical Rules

Sag's revised COMPLEMENT EXTRACTION LR (27), discussed in section 2.1, removes a *gap* from COMPS:

(27) **Complement Extraction Lexical Rule:**

$$\begin{bmatrix}\textsc{comps} & \boxed{1} \bigcirc \left\langle \begin{bmatrix}\textsc{loc} & \boxed{2} \\ \textsc{slash} & \left\{\boxed{2}\right\}\end{bmatrix} \right\rangle\end{bmatrix}_{word} \mapsto \begin{bmatrix}\textsc{comps} & \boxed{1}\end{bmatrix}$$

As was pointed out already, COMPS is in fact the only feature that is (nonmonotonically) changed in the current version of this rule.

Given the lexical hierarchy just introduced, there is a straightforward alternative for the CELR. Remember that the value of COMPS on predicative lexical entries is determined by the *map-args* constraint. In the simplest case, this constraint simply unifies COMPS with the tail of ARG-ST. The CELR removes elements of type *gap*'s from COMPS. The effect of this rule can therefore be incorporated into *map-args* by providing a second possibility for the non-recursive *map-arg* constraint (the definition of *map-args* is repeated below for convenience):

(28) a. `map-args(listof(synsem),listof(synsem))`

$$map\text{-}args(\langle\rangle,\langle\rangle)$$

$$map\text{-}args(\langle \boxed{1} \mid \boxed{2} \rangle, \boxed{3} \oplus \boxed{4}) \leftarrow map\text{-}arg(\boxed{1},\boxed{2}) \quad \wedge \quad map\text{-}args(\boxed{3},\boxed{4})$$

b. `map-arg(synsem,listof(synsem))`

$$map\text{-}arg(\boxed{1}, \langle \boxed{1} \rangle)$$

$$map\text{-}arg(\begin{bmatrix}\textsc{loc} & \boxed{1} \\ \textsc{slash} & \left\{\boxed{1}\right\}\end{bmatrix}_{gap}, \langle\, \rangle)$$

The non-subject elements on ARG-S are now either on COMPS, or instantiated as *gaps*. This is precisely what is achieved by means of the CELR otherwise.

For example, the lexeme *kiss* gives rise to the following instantiation of *verb*:

$$(29)\quad verb(\begin{bmatrix}\textsc{phon} & kiss \\ \textsc{head} & v \\ \textsc{arg-st} & \langle \mathrm{NP}_i, \mathrm{NP}_j \rangle \\ \textsc{cont} & kiss'(i,j)\end{bmatrix})$$

Resolution of (29) with the definition of *pred* gises rise to:

$$(30)\quad pred(\begin{bmatrix}\textsc{phon} & kiss \\ \textsc{head} & v \\ \textsc{arg-st} & \langle \boxed{1}, \boxed{2}\ \mathrm{NP}_j \rangle \\ \textsc{subj} & \boxed{2}\ np_i \\ \textsc{comps} & \boxed{3} \\ \textsc{cont} & kiss'(i,j)\end{bmatrix}) \leftarrow map\text{-}args(\langle \boxed{2} \rangle, \boxed{3})$$

which an be resolved in two different ways. The second element of ARG-ST can be instantiated either on COMPS or as a *gap*. In the second case, COMPS would be instantiated as the empty list, and the SLASH-value of the second element on ARG-ST would be a singleton list:

$$\text{(31)}\quad \textit{pred}(\begin{bmatrix} \text{PHON} & \textit{kiss} \\ \text{HEAD} & v \\ \text{ARG-S} & \left\langle \boxed{1}, \begin{bmatrix} \text{LOC} & \boxed{2}\ np_j \\ \text{SLASH} & \left\{\boxed{2}\right\} \end{bmatrix} \right\rangle \\ \text{SUBJ} & \left\langle \boxed{1}\ \text{NP}_i \right\rangle \\ \text{COMPS} & \langle\rangle \\ \text{CONT} & \textit{kiss}'(i,j) \end{bmatrix})$$

As in Sag's proposal, *slash-amalgamation* will ensure that this singleton is also included in the SLASH value of the lexical entry itself:

$$\text{(32)}\quad \textit{major}(\begin{bmatrix} \text{PHON} & \textit{kiss} \\ \text{HEAD} & v \\ \text{ARG-S} & \left\langle \boxed{1}, \begin{bmatrix} \text{LOC} & \boxed{2}\ np_j \\ \text{SLASH} & \left\{\boxed{2}\right\} \end{bmatrix} \right\rangle \\ \text{SUBJ} & \left\langle \boxed{1}\ \text{NP}_i \begin{bmatrix} \text{SLASH} & \boxed{3} \end{bmatrix} \right\rangle \\ \text{COMPS} & \langle\rangle \\ \text{CONT} & \textit{kiss}'(i,j) \\ \text{SLASH} & \boxed{3} \uplus \left\{\boxed{2}\right\} \end{bmatrix})$$

Note that by eliminating the CELR the spurious ambiguity, which resulted from the fact that two complements could in principle be removed in either order, is removed. The *map-args* constraint allows an arbitrary number of elements on ARG-ST to be realized as *gap*'s, and thus can account for cases of multiple extraction. However, since it is a relational constraint, defining the relationship between two attributes, and not a rule mapping one feature structure into another, there are no derivations in the linguistic sense.[7]

5 Adding Adjuncts Without Lexical Rules

In this section, I will propose a reformulation of the rule which introduces adjuncts lexically as arguments of the heads they modify. Instead of using a lexical rule, the definition of *verbal* lexical entries will be modified so as to allow for the optional presence of adjuncts on ARG-ST.

As in the proposals of Iida et al. (1994) and Manning, Sag, and Iida (1996) adjuncts are added not only to COMPS, but also to ARG-ST. There are two reasons for the latter assumption: First, adjuncts may contain reflexives, pronouns, and full NP's, to which the binding conditions apply much in the same way as they apply to subcategorized-for arguments. Second, the fact that adjuncts can be extracted is most easily accounted for (in the lexicalist, head-driven, approach to extraction) if they are members of ARG-ST. I will come back to the second point below.[8]

The lexical introduction of adjuncts has an effect on ARG-ST (and thus, indirectly, on valency) and on semantics. This implies that the argument structure and semantics of a verb can no longer be fully determined by the *verbal-lexeme* constraint. Instead, this constraint only provides the

[7] Of course, the spuriousity which arises for the CELR if multiple complements can be deleted, can also be avoided by reformulating the CELR. If the CELR states that the value of COMPS on the output is the value of COMPS *minus* one or more elements which are instantiated as *gaps*, one application of the CELR would suffice for cases of multiple extraction. Note however, that this still means that the CELR is a *rule*, relating two lexical entries, whereas in the present proposal it is a constraint on lexical entries.

[8] The lexicalist account of quantifier scope developed in Pollard and Yoo (1996) might provide a third argument for adding adjuncts to ARG-ST.

arguments which the verb subcategorizes for. The *adverb* constraint, introduced below, provides the value for the list of additional ARG-ST members, and also provides the value of CONT (based on the semantics provided by a *verbal-lexeme*).

The redefinition of the lexical class *verb* (assuming for simplicity that adjuncts are only added to verbal lexical entries) is given in (33):

$$\text{(33)}\quad verb(\begin{bmatrix} \text{PHON} & \boxed{1} \\ \text{HEAD} & v \\ \text{ARG-S} & \boxed{2} \oplus \boxed{2a} \\ \text{CONT} & \boxed{3} \end{bmatrix}) \leftarrow verbal\text{-}lexeme(\boxed{1},\boxed{2},\boxed{3a}) \wedge adjuncts(\boxed{4},\boxed{3a},\boxed{3})$$

Note that ARG-ST is now defined as the `append` ($\oplus$) of the arguments specified by subcategorization (i.e. these are the arguments for which the verb truly subcategorizes) and a list of arguments supplied by the *adjuncts* constraint. Furthermore, the semantics of the lexeme is 'input' to this constraint, which produces the actual semantics as 'output'.

The *adjuncts* constraint is a recursive constraint which instantiates its first argument as a (possibly empty) list of adverbial phrases:

$$\text{(34)}\quad adjuncts(\langle\rangle,\boxed{1},\boxed{1})$$

$$adjuncts(\left\langle \begin{bmatrix} \text{LOC} & \begin{bmatrix} \text{HEAD} & adv \\ \text{CNTNT} & \boxed{1b}\begin{bmatrix}\text{ARG} & \boxed{1a}\end{bmatrix} \end{bmatrix} \end{bmatrix} \mid \boxed{2} \right\rangle,\boxed{1a},\boxed{1}) \leftarrow adjuncts(\boxed{2},\boxed{1b},\boxed{1})$$

Note that the second argument of *adjuncts* represents the semantics as it was constructed so far. This serves as the argument of the semantics of the adverb, and the resulting semantics is passed on to the next invocation of *additional-args*. The recursion stops if no adjuncts need to be added any more, in which case the 'intermediate' semantics can be unified with the third argument, the 'resultant' semantics, which is also the value of the CONT attribute of the verb.

For example, resolving the *verbal-lexeme* for *kiss* with the new definition of *verb* gives:

$$\text{(35)}\quad verb(\begin{bmatrix} \text{PHON} & kiss \\ \text{HEAD} & v \\ \text{ARG-S} & \langle \text{NP}_i,\text{NP}_j \rangle \oplus \boxed{1} \\ \text{CONT} & \boxed{2} \end{bmatrix}) \leftarrow adjuncts(\boxed{1},\ _{kiss}\begin{bmatrix}\text{ARG1} & i \\ \text{ARG2} & j\end{bmatrix},\boxed{2})$$

This clause can be be resolved with (34a), which would give rise to a unit clause for *verb* in which $\boxed{1}$ would be the empty list and CONT would be the semantics of the lexeme *kiss* as given. However, we can also resolve this clause with (34b), which would lead to the following:

$$\text{(36)}\quad verb(\begin{bmatrix} \text{PHON} & kiss \\ \text{HEAD} & v \\ \text{ARG-S} & \left\langle \text{NP}_i,\text{NP}_j, \begin{bmatrix} \text{HEAD} & adv \\ \text{CNTNT} & \boxed{3a}\begin{bmatrix}\text{ARG} & _{kiss}\begin{bmatrix}\text{ARG1} & i \\ \text{ARG2} & j\end{bmatrix}\end{bmatrix} \end{bmatrix} \right\rangle \oplus \boxed{1a} \\ \text{CONT} & \boxed{2} \end{bmatrix}) \leftarrow adjuncts(\boxed{1a},\boxed{2a},\boxed{2})$$

Note that by resolving with (34b), the tail of ARG-ST (i.e. $\boxed{1}$ in (??)) is instantiated as a non-empty list, whose first element is an adverb. Thus, ARG-ST must at least contain three elements, and the new tail $\boxed{1a}$ is recursively passed on to the *adjuncts* constraint again. If we now choose to resolve (36) with (34a), $\boxed{1a}$ is instantiated as the empty list and $\boxed{2a}$ would be unified with $\boxed{2}$. This would be the required *sign* for *kiss* in case it combines with one adverb. More adverbs can be introduced by resolving with (34b) again.

The introduction of adjuncts on ARG-ST by means of a constraint eliminates spurious redundancy. In a grammar with lexical rules for complement extraction as well as the introduction of adjuncts, these two rules can be applied in either order, often giving identical results. If adjuncts are introduced by means of a constraint, the rule for complement extraction is replaced by a constraint on the relation between argument structure and valency, such spuriousity is eliminated. Since both constraints apply to the same argument structure, there is no longer a sense in which they are ordered with respect to each other.

The reformulation of the adjuncts lexical rule as a constraint has an important additional benefit in that it accounts for the interaction with extraction. As is discussed at length in Hukari and Levine (1994), adjuncts can be extracted, and can be extracted out of. By calling the adjuncts constraint recursively, one or more adjuncts may be introduced on ARG-ST. The *map-args* constraint allows one or more of these adjuncts to be *gaps*. The *amalgamate-slashes* constraint, finally, will ensure that if one of the adjuncts is instantiated as a gap, its SLASH value will be unioned with the SLASH value of the verb itself. Thus, the possibility of extracting adjuncts is accounted for, without having to introduce additional rules or constraints.

It should be noted that the fact that *map-args, adjuncts*, and slash amalgamation are constraints applying to the *same* argument structure is essential here. If adjuncts are introduced by means of a rule, it is tempting to define the value of SLASH on the output as being identical to the SLASH value on the input. Note, however, that this would make the newly introduced adjunct *invisible* to slash amalgamation, and thus applying the CELR to remove the adjunct from COMPS would not have the 'side-effect' of including it in SLASH. The alternative would be to define SLASH on the output as the union of the SLASH-value on the input and the SLASH-value of the adjunct. While this would lead to the correct result in case the adjunct is removed by complement extraction, it has the disadvantage that it must define explicitly the values of a feature, SLASH, which, intuitively, is not directly relevant for the introduction of adjuncts. In the constraint-based approach, such problems do not arise.[9]

6 French Clitic Movement Without Lexical Rules

In this section, an analysis fo French clitics or pronominal affixes is presented in which the effect of the two lexical rules assumed by Miller and Sag (1996) is obtained by introducing a constraint which defines the relation between ARG-ST and PHON. Furthermore, the *map-args* constraint is redefined. First, clitization of 'local' complements is considered, next the analysis is extended to cltici climbing, and finally, *en*-clitization is dealt with.

6.1 Clitization of local complements

The PRONOMINAL AFFIXATION lexical rule of Miller and Sag, repeated below, removes certain elements from COMPS and realizes these as pronominal affixes.

(37) **Pronominal Affixation Inflection Rule:**

$$_{word}\begin{bmatrix}\textsc{phon} & \boxed{0} \\ \textsc{synsem} & \boxed{2}\begin{bmatrix}\textsc{head} & \text{verb} \\ \textsc{comps} & \boxed{1} \bigcirc \text{nelist(aff)}\end{bmatrix}\end{bmatrix} \mapsto {}_{cl\text{-}word}\begin{bmatrix}\textsc{phon} & \mathit{Fpraf}(\boxed{0},\boxed{2}) \\ \textsc{synsem} & \begin{bmatrix}\textsc{comps} & \boxed{1}\ \text{list(canon)}\end{bmatrix}\end{bmatrix}$$

This rule is similar to the COMPLEMENT EXTRACTION lexical rule in that it removes elements from COMPS. The effect of this rule can therefore be obtained in a similar manner: instead of explicitly

[9] Note also that one cannot simply assume that SLASH-amalgamation must be 'recomputed' each time an element is added to ARG-ST. For their *En* lexical rule to operate correctly, for instance, Miller and Sag (1996) must assume that the SLASH value on the output is identical to that of the input, eventhough an element has been added to ARG-ST.

removing elements from COMPS, the *map-args* constraint will be redefined so as to allow *affixes* to be not realized on COMPS. The second aspect of the PRAF-LR is that it changes the phonology of the input and supplements it with pronominal affixes corresponding to the *affix synsems* found on ARG-ST. This part of the rule can be captured by a constraint which defines the phonology of a verb in terms of the phonology provided by a verbal lexeme and argument structure.

The phonological part of the rule is considered first. In the definition of *verb* presented in section 3, the phonology of a lexeme was simply equated with the phonology of the sign which is the argument of *verb*. Affixation requires that the relationship between these two values is more indirect. In particular, the potential presence of affixes on ARG-ST must be taken into account.[10] This leads to the following redefinition of *verb*:[11]

(38) $verb(\begin{bmatrix} \text{PHON} & \boxed{1} \\ \text{HEAD} & v \\ \text{ARG-ST} & \boxed{2} \\ \text{CONT} & \boxed{3} \end{bmatrix}) \leftarrow verbal\text{-}lexeme(\boxed{1a},\boxed{2},\boxed{3}) \wedge praf(\boxed{1a},\boxed{2},\boxed{1})$

The *praf* constraint must distinguish two cases. First, if no clitics are realized phonologically, it can be satisfied by simpy unifying the two phonological values (39a). The second clause (39b) applies if one or more affixes are realized phonologically. The *affixes* constraint identifies the affixes present on ARG-ST.

(39) a. $praf(\boxed{1}non\text{-}cl\text{-}phon,\_,\boxed{1})$

b. $praf(\boxed{1a},\boxed{2},\boxed{1}cl\text{-}phon) \leftarrow affixes(\boxed{2},\boxed{2a}nelist) \wedge affixation(\boxed{1a},\boxed{2a},\boxed{1})$

(40) a. **affixes(listof(synsem),listof(aff))**

b. $affixes(\langle\rangle,\langle\rangle)$

c. $affixes(\left\langle \boxed{1}aff|\boxed{2} \right\rangle,\left\langle \boxed{1}|\boxed{3} \right\rangle) \leftarrow affixes(\boxed{2},\boxed{3})$

d. $affixes(\left\langle non\text{-}aff|\boxed{2} \right\rangle,\boxed{3}) \leftarrow affixes(\boxed{2},\boxed{3})$

The *affixation* constraint defines the relation between the phonology of the verbal lexeme, and the phonology of the verbal sign, taking the affixes into account. A few examples of clauses satisfying this constraint are given below. An exhaustive list can be found in Miller and Sag (1996).

(41) a. $affixation(\boxed{1},\left\langle \text{NP}[\mathit{3rd},\ sg,\ \grave{a}] \right\rangle, lui + \boxed{1})$

b. $affixation(\boxed{1},\left\langle \text{NP}[\mathit{3rd},\ sg,\ acc],\text{NP}[\mathit{3rd},\ sg,\ \grave{a}] \right\rangle, \text{'}le\ lui\text{'} + \boxed{1})$

c. ...

For instance, assume we have the follwing lexeme for the third person singular form of *donner* (*to give*):

(42) $verbal\text{-}lexeme(donne,\left\langle \text{NP}[nom],\text{NP}[acc],\text{NP}[\grave{a}] \right\rangle, give\text{-}rel)$

Resolution of this clause with (38) gives rise to the following resolvent (ignoring CONT):

[10] Miller and Sag use the whole of SYNSEM as parameter of the function F_{praf}. Thus, the head feature MOOD can have an effect as well. In particular, the question whether affixes are prefixed or suffixed depends on the mood of the verb. This complication is ignored below, although it would be straightforward to include the value of MOOD as an additional argument of the *praf* constraint.

[11] For simplicity, I ignore the adjuncts constraint here. It should be noted, however, that the inclusion of this constraint is fully compatible with the revisions introduced below.

$$(43)\quad verb\left(\begin{bmatrix}\text{PHON} & \boxed{1}\\ \text{HEAD} & v\\ \text{ARG-ST} & \boxed{2}\left\langle \text{NP}[nom],\text{NP}[acc],\ \text{NP}[\grave{a}]\right\rangle\end{bmatrix}\right) \leftarrow praf(donne,\boxed{2},\boxed{1})$$

Resolution of this clause with (39b) requires that the list of arguments is divided into a list of non-affixes and a non-empty list of affixes. As the lexeme for *donne* does not specify the *synsem*-subtype of its arguments, among others, the following two resolvents can be obtained:

(44)

$$verb\left(\begin{bmatrix}\text{PHON} & \boxed{1}\\ \text{HEAD} & v\\ \text{ARG-S} & \left\langle \text{NP}[non\text{-}aff,\ nom],\boxed{2}\ \text{NP}[aff,\ acc],\text{NP}[non\text{-}aff,\ \grave{a}]\right\rangle\end{bmatrix}\right) \leftarrow affixation(donne,\left\langle\boxed{2}\right\rangle,\ \boxed{1})$$

$$verb\left(\begin{bmatrix}\text{PHON} & \boxed{1}\\ \text{HEAD} & v\\ \text{ARG-ST} & \left\langle \text{NP}[non\text{-}aff,\ nom],\boxed{2}\ \text{NP}[aff,\ acc],\boxed{3}\ \text{NP}[aff,\ \grave{a}]\right\rangle\end{bmatrix}\right) \leftarrow affixation(donne,\left\langle\boxed{2},\ \boxed{3}\right\rangle,\ \boxed{1})$$

Resolving these with the two example cases of *affixation* given above, leads to the following two results:

(45)

$$verb\left(\begin{bmatrix}\text{PHON} & \text{'}le\ donne\text{'}\\ \text{HEAD} & v\\ \text{ARG-S} & \left\langle \text{NP}[non\text{-}aff,\ nom],\ \text{NP}[aff,\ acc],\text{NP}[non\text{-}aff,\ \grave{a}]\right\rangle\end{bmatrix}\right)$$

$$verb\left(\begin{bmatrix}\text{PHON} & \text{'}le\ lui\ donne\text{'}\\ \text{HEAD} & v\\ \text{ARG-S} & \left\langle \text{NP}[non\text{-}aff,\ nom],\ \text{NP}[aff,\ acc],\ \text{NP}[aff,\ \grave{a}]\right\rangle\end{bmatrix}\right)$$

To complete this account of pronominal affixation, the effect of affixation on valency must be accounted for. The basic idea is that elements on ARG-ST which are realized in phonology as affixes, should not be realized on COMPS. One way to achieve this, would be to modify the *map-args* constraint so that it would only realize *non-affix synsem*'s on COMPS. Such a solution is too restrictive, as it would effectively block the possibility of affixes ever occurring on COMPS. The analysis of clitic climbing in terms of *inheritance*, however, requires *affix synsem*'s to be on COMPS of the 'lower' verb. Thus, blocking of affixes on COMPS must only apply if the *praf* constraint has applied non-trivially. The subtyping of the phonological value was introduced primarily for this purpose.[12]

Since *map-args* is now sensitive to the phonological type of *sign*, this constraint must now take three arguments:

$$(46)\quad pred\left(\boxed{0}\begin{bmatrix}\text{PHON} & \boxed{1}\\ \text{ARG-ST} & \left\langle\boxed{2}\ |\boxed{3}\right\rangle\\ \text{SUBJ} & \boxed{2}\\ \text{COMPS} & \boxed{4}\end{bmatrix}\right) \leftarrow (\ verb(\boxed{0}) \vee pred\text{-}n(\boxed{0})\ ) \wedge map\text{-}args(\boxed{1},\boxed{3},\boxed{4})$$

[12] Note, however, that it can also be used to ensure, for instance, that past participles do not combine with clitics. The constraint for inflection which produces a past partiple only needs to add the information that the phonology is *canon-phon* in that case.

If no phonological pronominal affixes are present, the definition of *map-args* is as it was defined in the section on complement extraction, i.e. COMPS consists of the non-subject elements on ARG-ST, minus the elements that are realized as *gap*'s. If pronominal affixes are present and phonologically realized, COMPS consists of the non-subject ARG-ST elements, minus the non-canonical *synsems* (i.e. the gaps and affixes). Note also that since affixes are never selected syntactically, the SLASH values of the corresponding elements on ARG-ST must be the empty set, in order to ensure that slash amalgamation works as intended.

(47) a. `map-args(phon,listof(synsem),listof(synsem))`

b. $\textit{map-args}(\_,\langle\rangle,\langle\rangle)$

$$\textit{map-args}(\boxed{Ph},\left\langle \boxed{Arg} \mid \boxed{Args} \right\rangle, \boxed{Cs0} \oplus \boxed{Cs1}) \leftarrow$$
$$\textit{map-arg}(\boxed{Ph},\boxed{Arg},\boxed{Cs0}) \quad \wedge \quad \textit{map-args}(\boxed{Ph},\boxed{Args},\boxed{Cs1})$$

c. `map-arg(phon,synsem,listof(synsem))`

d. $\textit{map-arg}(\_,\boxed{1}\textit{canon}, \left\langle \boxed{1} \right\rangle)$

$$\textit{map-arg}(\_, {}_{\textit{gap}}\begin{bmatrix} \text{LOC} & \boxed{1} \\ \text{SLASH} & \left\{\boxed{1}\right\} \end{bmatrix}, \langle\ \rangle)$$
$$\textit{map-arg}(\textit{non-cl-phon},\ \boxed{1}\,{}_{\textit{aff}}\begin{bmatrix} \text{SLASH} & \emptyset \end{bmatrix}, \left\langle \boxed{1} \right\rangle)$$
$$\textit{map-arg}(\textit{cl-phon},\ {}_{\textit{aff}}\begin{bmatrix} \text{SLASH} & \emptyset \end{bmatrix}, \langle\ \rangle)$$

6.2 Clitic Climbing

Miller and Sag argue that their analysis of clitics as pronominal affixes, which attach to a host-verb in the lexicon, is compatible with *clitic climbing*:

(48) a. Marie l' a vu
Marie him has seen
Marie has seen him

b. Le livre lui a été donné
The book him has been given
The book has been given to him

The key to their solution is the lexical specification for auxiliaries. Given the assumption that such verbs inherit the complements of their verbal argument on ARG-ST, as well as COMPS, clitic climbing can be accounted for:

(49)
$$\begin{bmatrix} \text{PHON} & \textit{avoir/être} \\ \text{SUBJ} & \left\langle \boxed{1} \right\rangle \\ \text{COMPS} & \left\langle \boxed{2} \begin{bmatrix} \text{HEAD} & \textit{v[psp]} \\ \text{SUBJ} & \left\langle \boxed{1} \right\rangle \\ \text{COMPS} & \boxed{3} \end{bmatrix} \right\rangle \oplus \boxed{3} \\ \text{ARG-ST} & \left\langle \boxed{1}, \boxed{2} \right\rangle \oplus \boxed{3} \end{bmatrix}$$

This analysis is compatible with the approach adopted here. The sign in (49) corresponds with the following (infected) verbal lexeme:[13]

(50) $$\textit{verbal-lexeme}\left(a, \left\langle \boxed{1}\text{NP}, \begin{bmatrix} \text{HEAD} & v[psp] \\ \text{SUBJ} & \left\langle \boxed{1} \right\rangle \\ \text{COMPS} & \boxed{2} \end{bmatrix} \right\rangle \oplus \boxed{2},\ \textit{avoir-sem}\right)$$

which in its turn gives rise to the following instantiation of *verb*:

(51) $$\textit{verb}\left(\begin{bmatrix} \text{PHON} & \boxed{1} \\ \text{HEAD} & v \\ \text{ARG-ST} & \boxed{2} \left\langle \boxed{3}\text{NP}, \begin{bmatrix} \text{HEAD} & v[psp] \\ \text{SUBJ} & \left\langle \boxed{3} \right\rangle \\ \text{COMPS} & \boxed{4} \end{bmatrix} \right\rangle \oplus \boxed{4} \\ \text{CONT} & \text{avoir-sem} \end{bmatrix}\right) \leftarrow \textit{praf}(a, \boxed{2}, \boxed{1})$$

The example in (48a) is now accounted for as follows. The auxiliary combines with a transitive verb, which implies that its ARG-ST is instantiated as follows:

(52) $$\textit{verb}\left(\begin{bmatrix} \text{PHON} & \boxed{1} \\ \text{HEAD} & v \\ \text{ARG-ST} & \boxed{2} \left\langle \boxed{3}\text{NP}, \begin{bmatrix} \text{HEAD} & v[psp] \\ \text{SUBJ} & \left\langle \boxed{3} \right\rangle \\ \text{COMPS} & \left\langle \boxed{4} \right\rangle \end{bmatrix}, \boxed{4}\text{NP} \right\rangle \\ \text{CONT} & \text{avoir-sem} \end{bmatrix}\right) \leftarrow \textit{praf}(a, \boxed{2}, \boxed{1})$$

If the *praf* constraint is satisfied nontrivially (i.e. there is an actual affixation process at this point), (52) leads to:

(53) $$\textit{verb}\left(\begin{bmatrix} \text{PHON} & \boxed{1} \\ \text{HEAD} & v \\ \text{ARG-ST} & \boxed{2} \left\langle \boxed{3}\text{NP}, \begin{bmatrix} \text{HEAD} & v[psp] \\ \text{SUBJ} & \left\langle \boxed{3} \right\rangle \\ \text{COMPS} & \left\langle \boxed{4} \right\rangle \end{bmatrix}, \boxed{4}\text{NP} \right\rangle \\ \text{CONT} & \text{avoir-sem} \end{bmatrix}\right) \leftarrow \textit{affixes}(\boxed{2}, \boxed{5}) \wedge \textit{affixation}(a, \boxed{5}, \boxed{1})$$

The ARG-ST list $\boxed{2}$ consists of a subject NP, a verb, and an (inherited) object NP. The *affixes* constraint can be satisfied by instantiating the object NP as an affix. This gives rise to:

[13] Note that this lexeme is itself the result of combining a corresponding listeme and subcategorization frame. No account of inflection has been given in this paper. However, a constraint-based approach to inflection, which determines the actual phonological value of a verb on using a relational constraint which takes as arguments a *stem* phonological value provided by a listeme, morphosyntactic features such as VFORM, AGR, and TENSE, and PHON, can easily be integrated.

$$(54)\quad verb\left(\begin{bmatrix}\textsc{phon} & \boxed{1}\\ \textsc{head} & v\\ \textsc{arg-st} & \left\langle \boxed{3}\textsc{np}[non\text{-}aff], {}_{non\text{-}aff}\begin{bmatrix}\textsc{head} & v[psp]\\ \textsc{subj} & \left\langle\boxed{3}\right\rangle\\ \textsc{comps} & \left\langle\boxed{4}\right\rangle\end{bmatrix}, \boxed{4}\textsc{np}[aff, acc]\right\rangle\\ \textsc{cont} & \text{avoir-sem}\end{bmatrix}\right) \leftarrow affixation(a, \left\langle\boxed{4}\right\rangle, \boxed{1})$$

which will give the result:

$$(55)\quad verb\left(\begin{bmatrix}\textsc{phon} & \text{cl-phon}^{\text{l'a}}\\ \textsc{head} & v\\ \textsc{arg-st} & \left\langle \boxed{3}\textsc{np}[non\text{-}aff], {}_{non\text{-}aff}\begin{bmatrix}\textsc{head} & v[psp]\\ \textsc{subj} & \left\langle\boxed{3}\right\rangle\\ \textsc{comps} & \left\langle\boxed{4}\right\rangle\end{bmatrix}, \boxed{4}\textsc{np}[aff, acc]\right\rangle\\ \textsc{cont} & \text{avoir-sem}\end{bmatrix}\right)$$

which in turn gives rise to:

$$(56)\quad pred\left(\begin{bmatrix}\textsc{phon} & \text{cl-phon}^{\text{l'a}}\\ \textsc{head} & v\\ \textsc{arg-st} & \left\langle\boxed{1},\boxed{2},\boxed{3}\textsc{np}[aff, acc]\right\rangle\\ \textsc{subj} & \left\langle\boxed{1}\ \textsc{np}[canon]\right\rangle\\ \textsc{comps} & \left\langle\boxed{2}\ {}_{canon}\begin{bmatrix}\textsc{head} & v[psp]\\ \textsc{subj} & \left\langle\boxed{1}\right\rangle\\ \textsc{comps} & \left\langle\boxed{3}\right\rangle\end{bmatrix}\right\rangle\\ \textsc{cont} & \text{avoir-sem}\end{bmatrix}\right)$$

Note that the *map-args* constraint, which 'maps' argument structure onto the valence features, will prevent the affix from occurring on COMPS. Furthermore, elements which are realized on COMPS (or SUBJ) are assigned the type *canon*.

There is one subtle difference at this point with cases where affixation does not apply. This is relevant if we consider the entry for *vu* that should combine with *l'a*. This is an ordinary transitive verb, whose object is to be realized on COMPS. Thus, we have:

$$(57)\quad verb\left(\begin{bmatrix}\textsc{phon} & \boxed{1}\\ \textsc{head} & v[psp]\\ \textsc{arg-st} & \boxed{2}\left\langle\textsc{np},\textsc{np}\right\rangle\\ \textsc{cont} & \text{voir-sem}\end{bmatrix}\right) \leftarrow praf(vu, \boxed{2}, \boxed{1})$$

Since the *praf* constraint is satisfied trivially in this case, this leads to:

$$(58)\quad verb\left(\begin{bmatrix}\textsc{phon} & \text{non-cl-phon}\ vu\\ \textsc{head} & v[psp]\\ \textsc{arg-st} & \left\langle\textsc{np},\textsc{np}\right\rangle\\ \textsc{cont} & \text{voir-sem}\end{bmatrix}\right)$$

Note that by satisfying the *praf* constraint trivially, the phonology of the verb is typed *non-cl-phon* and the list of ARG-ST elements is not typed as being either *non-aff* or *aff*. Thus, the *map-args* constraint will not be allowed to 'skip' affixes on ARG-ST. Instead, all (non-gap, non-subject) elements on ARG-ST must go to COMPS:

(59) $$verb(\begin{bmatrix} \text{PHON} & \text{non-cl-phon}\,vu \\ \text{HEAD} & v[psp] \\ \text{ARG-ST} & \left\langle \boxed{1},\boxed{2} \right\rangle \\ \text{SUBJ} & \left\langle \boxed{1}\text{NP} \right\rangle \\ \text{COMPS} & \left\langle \boxed{2}\text{NP} \right\rangle \\ \text{CONT} & \text{voir-sem} \end{bmatrix})$$

The element on COMPS is still underspecified with respect to the distinction between *aff* and *non-aff*, and thus it can actually serve as complement of the complex verb *l'a*.

6.3 En Affixation

The *En* AFFIXATION lexical rule of Miller and Sag (1996) allows on element on SLASH of one of the arguments to be added to ARG-ST. The element containing the SLASH-value which is 'bound' by the verb to which the *en*-AFFIXATION rule must be an unmarked NP.[14] The element on SLASH must be an NP with *de*-marking (i.e. an NP precedes by the preposition (marker) *de*). The rule is repeated below for convenience.

(60) **En-Affixation LR (EnLR):**

$$\begin{bmatrix} \text{HEAD} & verb \\ \text{COMPS} & \boxed{1} \\ \text{ARG-S} & \boxed{2} \oplus \left\langle \text{NP}_{unm}\begin{bmatrix} \text{LOC} & \boxed{3} \\ \text{SLASH} & \{\} \end{bmatrix} \right\rangle \oplus \boxed{4} \end{bmatrix}_{lexeme} \mapsto$$

$$\begin{bmatrix} \text{COMPS} & \boxed{1} \oplus \boxed{7} \\ \text{ARG-S} & \boxed{2} \oplus \left\langle {}_{canon}\begin{bmatrix} \text{LOC} & \boxed{3} \\ \text{SLASH} & \left\{\boxed{6}\right\} \end{bmatrix} \right\rangle \oplus \boxed{4} \oplus \boxed{7}\left\langle \text{NP}_{aff,de}\begin{bmatrix} \text{LOC} & \boxed{6} \end{bmatrix} \right\rangle \end{bmatrix}$$

The rule has several aspects that require attention. First of all, this rule only makes sense in combination with Miller and Sag's PRONOMINAL AFFIXATION lexical rule. The latter rule removes a list of *affix synsem*'s from COMPS, and realizes these as pronominal affixes. The *En* AFFIXATION lexical rule adds an affix to (ARG-S and) COMPS. The only reason to do this is that in this way it can be included in the list of affixes to be realized by the pronominal affixation rule. This somewhat cumbersome detour, which adds an element to COMPS by means of a lexical rule, only to remove it in the next lexical rule, is thoroughly nonmonotonic.

The use of constraints suggests a somewhat different solution. Note that the combined effect of the PRONOMINAL and *en* affixation rules is that certain elements are realized as affixes, and not as complements or members SLASH. The elements in question can all be found on ARG-ST: either they are an element of ARG-ST, or they are the single member of SLASH of an element of ARG-ST. This suggests the following reformulation of the *affixes* constraint:

(61) a. $affixes(\langle\rangle,\langle\rangle)$

[14] Note that Miller and Sag treat subcategorized-for PP's as NP's with a marking corresponding to the preposition which precedes the NP.

b. $affixes(\langle \boxed{1} aff | \boxed{2} \rangle, \langle \boxed{1} | \boxed{3} \rangle) \leftarrow affixes(\boxed{2},\boxed{3})$

c. $affixes(\left\langle \begin{bmatrix} non\text{-}aff & \\ \text{HEAD} & np[unm] \\ \text{SLASH} & \left\{\boxed{1}\right\} \end{bmatrix} | \boxed{2} \right\rangle, \left\langle \begin{bmatrix} aff & \\ \text{LOC} & \boxed{1} \begin{bmatrix}\text{HEAD} & \text{np[de]}\end{bmatrix} \end{bmatrix} | \boxed{3} \right\rangle) \leftarrow affixes(\boxed{2},\boxed{3})$

d. $affixes(\langle non\text{-}aff | \boxed{2} \rangle, \boxed{3}) \leftarrow affixes(\boxed{2},\boxed{3})$

The third clause accounts for *en* clitization. If an NP contains a *de*-marked NP on SLASH, it may be added to the list of affixes.

A remarkable aspect of Miller and Sag's *En* AFFIXATION lexical rule is that its input contains an NP with an empty slash value, whereas on the result there is an NP whose SLASH value is non-empty. They explain:[15]

> Note that the NP's SLASH value ($\left\{\boxed{6}\right\}$) is not amalgamated into the SLASH-value of the rule output, because the input's SLASH value (which has amalgamated the unmarked NP's SLASH value, i.e. the empty set) is carried over unchanged to the rule output (although the canonical element introduced after the list $\boxed{2}$ is distinct from the unmarked NP present in the input's ARG-ST, the amalagamation of SLASH values is not 'recomputed' on the output of the LR). This entails that there is no inheritance of $\left\{\boxed{6}\right\}$ up from an EnLR output - the EnLR output itself serves as a bindee of that SLASH dependency, whether it is the object or the subject whose SLASH value is bound off.

Such complications are avoided to a large extent in the monotonic approach. As there is no rule input or output, the question whether a certain value needs to be 'recomputed' or not does not arise. The issue of 'binding' the SLASH value does require attention, however. The *slash-amalgamation* constraint defines the value of SLASH in terms of the SLASH-values of ARG-S members. This constraint now needs to be modified in a way that is very similar to the modifications that have been made to the *valence* constraint. That is, instead of simply collecting all SLASH values of all ARG-S members, and assigning SLASH a value which is the union of these, a more refined definition is needed which ignores SLASH values containing an *affix-loc* value.[16]

As was the case for *valence*, *slash-amalgamation* must be made sensitive to the type of the phonology. The reason is that affix local value's must be included in SLASH if no affixation has taken place, whereas they must be ignored if such affixation has taken place. This implies that *slash-amalgamation* is now a relation between a phonological type, ARG-ST, and SLASH:

(62) $canonical(\boxed{0} \begin{bmatrix} \text{PHON} & \boxed{1} \\ \text{ARG-S} & \boxed{2} \\ \text{SLASH} & \boxed{3} \end{bmatrix}) \leftarrow (\ major(\boxed{0}) \lor minor(\boxed{0})\ ) \land sl\text{-}amalgamation(\boxed{1},\boxed{2},\boxed{3})$

Slash amalgamation is now redefined as follows:

(63) a. $sl\text{-}amalgamation(\_,\langle p \rangle,\emptyset)$

[15] It should probably be noted that the rule as given by Miller and Sag is not entirely correct. The intended effect is clearly that the output of the rule selects for an NP with a non-empty SLASH-value, without including this value in the SLASH-set of the verb. Note, however, that the NP which as assigned an empty SLASH-value is present on COMPS as well (i.e. it is a member of the list denoted by $\boxed{1}$), which is shared by the output. This implies, however, that the ouput selects for a complement with an empty SLASH -value, instead of for an NP with a *de*-NP on SLASH. This problem is also solved in the constraint-based approach.

[16] The type *affix-loc* is a subtype of *local*, which is appropriate for *affix synsem*'s. The subtype *non-affix-loc* is assigned to LOC values of *non-affix synsem*'s.

b. $\textit{sl-amalgamation}(\textit{cl-phon}, \left\langle \left[\text{SLASH } \boxed{1} \uplus \textit{list}(\textit{aff-loc})\right] | \boxed{3} \right\rangle, \boxed{1}\ \textit{list}(\textit{non-aff-loc}) \uplus \boxed{4}) \leftarrow$
$\textit{sl-amalgamation}(\textit{cl-phon}, \boxed{3}, \boxed{4})$

c. $\textit{sl-amalgamation}(\textit{canon-phon}, \left\langle \left[\text{SLASH } \boxed{1}\right] | \boxed{2} \right\rangle, \boxed{1} \uplus \boxed{3}) \leftarrow$
$\textit{sl-amalgamation}(\textit{canon-phon}, \boxed{2}, \boxed{3})$

As before, slash amalgamation is a constraint which steps through ARG-ST recursively. The recursion stops if ARG-ST is the empty list. In case affixation has applied, and the phonology is of type *cl-phon*, slash amalgamation includes into SLASH only the *non-affix local* values on SLASH of the argument (63b). If no affixation has taken place, slash amalgamation works as before (63c).

6.4 Remaining Issues

6.4.1 Argument Inheritance and Extraction

In Hinrichs and Nakazawa (1996) it is observed that the interaction of argument inheritance and a lexical rule for complement extraction may lead spurious ambiguity. If complements are extracted by means of a lexical rule, there are two derivations for examples such as:

The CELR could apply either to the embedded verb, or to the auxiliary.

The solution proposed in Hinrichs and Nakazawa (1996) is to let lexical rules apply only to lexical entries which are subsumed by the input specifications of the lexical rule. Since argument inheritance verbs have an underspecified COMPS value, the CELR can only extract those elements for which the auxiliary actually 'subcategorizes'. Crucially, inherited complements cannot be extracted from an auxiliary since their precence is not specified in the lexical entry of the auxiliary.

An appeal to subsumption to control the application of lexical rules has a number of disadvantages: Sag's recent CELR won't work, delayed evaluation is difficult, order-sensitivity ??, undecidability.

The current approach avoids this problem naturally. Instead of a (nonmonotonic) rule which deletes elements from COMPS, the map-args constraint decides for each element whether it must be present on COMPS or not. An advantage of this approach is that *gaps* are never on COMPS to begin with. That is, if an element is a member of COMPS, it must be of type *non-gap* (*canon* if affixes are realized phonologically). This implies that an *inherited complement*, which is inherited from a COMPS-list of another element on ARG-ST, must be of type *non-gap*, and thus cannot be realized as *gap* anymore. That is, only elements on ARG-ST that are not inherited can be extracted. Therefore, the spurious ambiguity problem disappears.

What if there is a gap INSIDE

6.4.2 Binding and Complement Inheritance

6.4.3 Binding and En-affixation

The constraint-based account of *en*-affixation is based on ARG-ST, but differs from the lexical rule given by Miller and Sag in that it never actually adds the NP corresponding to *en* to ARG-ST. In the presentation of Miller and Sag it is not obvious why an element is added to ARG-ST, and not just to COMPS. However, Daniele Godard (p.c.) points at the following contrast:

(64) a. no en

b. en

In the first example, a coreference between the subject and the embedded pronominal is allowed. However, in the second example, no coreference is possible between subject and *en*. As binding theory is defined on ARG-ST, these examples seem to provide evidence for the fact that *en*-clitization must have an effect on argument structure. At the moment, I cannot account for this.

6.4.4 No clitization on participles

6.4.5 Tous

References

Abeillé, Anne and Daniele Godard. 1994. The complementation of french auxiliaries. ms.

Gazdar, Gerald. 1981. Unbounded dependencies and coordinate structure. *Linguistic Inquiry*, 12:155–84.

Gazdar, Gerald. 1982. Phrase structure grammar. In Pauline Jacobson and Geoffrey K. Pullum, editors, *The Nature of Syntactic Representation*. Reidel, Dordrecht, pages 131–186.

Hinrichs, Erhard and Tsuneko Nakazawa. 1994. Linearizing AUXs in German verbal complexes. In John Nerbonne, Klaus Netter, and Carl Pollard, editors, *German in Head-driven Phrase Structure Grammar*, Lecture Note Series. CSLI, Stanford, pages 11–38.

Hinrichs, Erhard and Tsuneko Nakazawa. 1996. Applying lexical rules under subsumption. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING)*, pages 543–549, Copenhagen.

Hukari, Thomas E. and Robert D. Levine. 1994. Adjunct extraction. ms. Univerity of Victoria and Ohio State Univerity.

Iida, Masayo, Christopher Manning, Patrick O'Neill, and Ivan Sag. 1994. The lexical integrity of Japanese causatives. Paper presented at the LSA 1994 Annual Meeting.

Kasper, Robert. 1994. Adjuncts in the Mittelfeld. In John Nerbonne, Klaus Netter, and Carl Pollard, editors, *German in Head-driven Phrase Structure Grammar*, Lecture Note Series 46. CSLI, Stanford, pages 39–70.

Manning, Christopher and Ivan Sag. 1995. Dissociations between argument structure and grammatical relations. draft.

Manning, Christopher, Ivan Sag, and Masayo Iida. 1996. The lexical integrity of Japanese causatives. In Takao Gunji, editor, *Studies on the Universality of Constraint-based Phrase Structure Grammars*. Osaka University.

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

Miller, Philip and Ivan Sag. 1996. French clitic movement without clitics or movement. Ms., Stanford University.

Pollard, Carl and Ivan Sag. 1987. *Information Based Syntax and Semantics, Volume 1*. Center for the Study of Language and Information Stanford.

Pollard, Carl and Ivan Sag. 1994. *Head-driven Phrase Structure Grammar*. Center for the Study of Language and Information Stanford.

Pollard, Carl and Eun Jung Yoo. 1996. A unified theory of scope for quantifiers and wh-phrases.

Sag, Ivan. 1995a. Constraint-based Extraction (Without a Trace). Draft, Stanford University, November, 1995.

Sag, Ivan. 1995b. English relative clause constructions. Draft, Stanford University, October, 1995.

Sag, Ivan A. and Janet Dean Fodor. 1994. Extraction without traces. In *West Coast Conference on Formal Linguistics 13*, Stanford. CSLI Pulbications.

van Noord, Gertjan and Gosse Bouma. 1994. Adjuncts and the processing of lexical rules. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING)*, pages 250–256, Kyoto.