**Abstract**

This paper points out certain flaws in the semantics for lexical rule specifications developed by Meurers (2001). Under certain circumstances, certain words may not be licit inputs to a rule according to this semantics while one would expect them to be from inspecting the specification of the rule, i.e. the rule violates the criterion of *Universality*. It is not universally applicable to all the words that its specification seems to characterise as licit inputs. The reasons for this are shown to be that whether properties of paths should be transferred from the input of a rule to its output is decided considering only the respective paths and their properties in isolation, ignoring the 'non-local' effects that transferring their properties can have. Furthermore, the semantics is insensitive to the possible shapes of inputs to the rule, which also makes it possible that inputs of certain shapes are unexpectedly not accepted. An alternative semantics is developed that does not suffer from these deficits.

## 1 Introduction

In HPSG theorising, *lexical rules* (henceforth *LRs*) play a prominent role. Such rules are employed to derive new words from existing words systematically. What precisely this means depends, of course, on what a *word* is taken to be. Pollard & Sag (1994) take *word* to mean *lexical entry*, envisaging LRs as a means of deriving new lexical entries from given ones (basic or themselves derived). This approach, which Meurers (2001) calls the *meta-level* approach, has however never been worked out formally in a satisfactory manner. An alternative approach is developed by Meurers (2001). Meurers suggests to view LRs as a means of deriving *lexical items* – the objects described by lexical entries – from given ones (basic or themselves derived). This approach can straightforwardly be formalised in model-theoretic HPSG by introducing a sort *lex_rule* with appropriate attributes IN and OUT, both of which take objects of sort *word* as values. The value of IN can then be viewed as the input and the value of OUT as the output of a LR. The content of any rule, i.e. the input-output relation it is supposed to encode, can be stated by an appropriate description that constrains the licenced *lex_rule* objects to those in which this relation holds between the values of IN and OUT. The actual content of a LR is thus given by an ordinary description of the same kind as a lexical entry. Meurers (2001) thus calls this a *description-level* approach.

In this paper, I exclusively assume this latter approach and will not concern myself with the idea of formalising LRs as relating lexical entries to lexical entries. Furthermore, I assume the model-theoretic foundations for HPSG developed by King (1989) and Richter (2004), where only the formalism developed by King (SRL) will be used here. As a consequence, I will not concern myself with approaches that have not been or cannot be satisfactorily formalised within these

[†]I thank two anonymous reviewers for helpful comments and Frank Richter, Manfred Sailer and Olivier Bonami for valuable discussion. The work reported here originates with Lahm (2012). The current paper is due to funding by the German Research Foundation (DFG) under grant GRK 2016/1.

frameworks, which leaves (Meurers, 2001) as the only serious attempt at specifying a semantics for LRs.

Usually, LRs are stated using abbreviations, which I also call Lexical Rule Specifications (LR descriptions). An LR descriptions is of the form $A \mapsto B$, where $A$ and $B$ are AVMs. It is supposed to express a rule that ($i$) accepts any input that is described by $A$, ($ii$) yields an output described by $B$ and ($iii$) effects not more than the smallest changes to the input necessary to guarantee ($i$) and ($ii$). So the output should still be like the input to the greatest possible extent compatible with its being described by $B$. The LRs given in the literature employ this abbreviatory notation, without any exceptions I would be aware of. The central aim of Meurers (2001) and this paper is to explicate what exactly the intuitive ideas ($i$)-($iii$) are supposed to mean.

The idea seems simple enough. If some path $\pi$ is not mentioned in the specification of a rule, its value can be assumed to be the same (token-identical) in the input and output of the rule and is thus transferred from the input to the output. Similarly, the sorts of path values in the input should become those of the values of the corresponding paths in the output if this is possible. Meurers's semantics of LRs operates according to these ideas, but the details turn out to be more involved than one might expect. In particular, it can be shown that Meurers's system does not fulfill ($i$) above, which will be stated more explicitly below as the criterion of **Universality**: it is possible to construct LRs $A \mapsto B$ which, according to Meurers's semantics, do not accept every word described by $A$ but impose further restrictions on their inputs in a manner that seems unexpected from their specification and therefore undesirable.

One reason for this will turn out to be that Meurers's semantics transfers properties of paths in a local manner, considering only one path at a time and disregarding the effects that such transfers can have on the possible values of other paths. If the properties of two paths cannot both be transferred at the same time, the semantics will thus not detect this, which leads to inconsistencies and thus violations of Universality. The alternative account offered here will instead allow for transferring the properties of all maximal sets of paths whose properties can consistently be transferred together. It thus takes into account the global effects of the transfers that are performed and acknowledges the fact that transferring one property may come at the cost of not being able to transfer another.

This alone however is not sufficient to guarantee Universality. In addition, the semantics needs to be made sensitive to the possible shapes of a rule's input according to its specification in order to guarantee that all of these are actually accepted by the rule. This problem will also be addressed in the approach presented here, which can be shown to no more violate Universality.

## 2 Preliminaries

### 2.1 The scope of this paper

The present paper is an investigation into lexical rules, an expressive mechanism often employed in HPSG grammars, and the way they are specified. It discusses the proposal by Meurers (2001) on how the meaning of LRs specifications can be made precise in the context of model-theoretic HPSG based on SRL as developed by King (1989). It points out certain problems with the approach advocated by Meurers. The problems identified are of a conceptual nature. It is shown that, in principle, the semantics that Meurers defines for LR specifications will fail to fulfill certain expectations that a semantics would be required to fulfill by intuitions which I expect everyone who employs LR specifications to share.[1]

The LRs as well as the sort hierarchy this paper uses are artificially constructed for the purpose of discussing the basic ideas behind the semantics outlined and the conceptual shortcomings of Meurers's approach. What the paper hence does not offer is any concrete example of the semantics developed by Meurers failing in the case of any actual LR that has been employed in the literature. It thus remains silent on whether the conceptual problems outlined also are practical problems that have adverse effects on any actual grammars that employ LRs. Even if this were not the case, it is worth keeping in mind that a merely conceptual problem could easily turn into such a practical problem with each new rule that is proposed.[2]

[1]I am not concerned here with the question whether all of Meurers's ideas about the correct meanings of LRs are tenable or not. LRs have been employed long before any precise semantics for them was available and, as a consequence, the intuitions of linguists can with good reason be expected to provide the standard of adequacy for any semantics to be proposed after the fact, not *vice versa*. Unfortunately however, nothing even guarantees that intuitions are stable enough among linguists to make a semantics that suits them all even possible. Hence it is hardly surprising if any concrete attempt at specifying such a semantics does not satisfy everyone. At the very least however, it provides a precise starting point from which a discussion about the benefits or shortcomings of certain decisions can be productively entered and opinions like that of one reviewer, who thinks that Meurers's system and mine alike are 'broken by design' (which, as I understand it, is supposed to be due at least in part to the choice of SRL) can sensibly be put forth, due to the presence of an actual design. Yet what I will be addressing here are not questions of semantic detail but rather something I think would be a conceptual problem for *any* semantics LRs might be given. This is not to say of course that semantic detail does not matter, but the system presented here is not supposed to carve the semantics in stone but is open to modification. For instance, the sceptical reviewer wondered why a certain rule would transfer to the output the species of paths that were equated in its input but not leave said structure sharing intact. It will be seen below that the present system is readily adapted so as to leave the structure sharing intact as well. Where intuitions run counter to the use of SRL however, there is no remedy in the present context. This does not mean that I think that arguments against its use could not sensibly be made or that the status of LRs might not figure prominently in such arguments. But unlike the arguments made here, such arguments cannot, it seems, be made on grounds that are purely formal apart from appealing to a single intuition about the meanings of LRs, namely that they should respect Universality, which seems to be very basic and has not yet been contested. Instead, they actually need to take into account how actual LRs behave in actual grammars when interpreted according to the system proposed here, and this issue is orthogonal to the one addressed here.

[2]The reviews of this paper remarked that offering an example of a problematic rule with genuine

## 2.2 The presupposed formalism

I shall, like Meurers (2001), assume the formalism in which HPSG theories are expressed to be Speciate Reentrant Logic (SRL) as developed in King (1989). Speciate Reentrant Logic was expressly designed for the formulation of HPSG grammars and incorporates assumptions on the linguistic ontology that are endorsed in Pollard & Sag (1994), total well-typedness and sort-resolvedness. The present section briefly discusses these assumptions, as they should be kept in mind in what follows.

Each linguistic object is assumed to have some sort. While sorts are organised in a partially ordered sort-hierarchy, so that it makes sense to speak of more general sorts subsuming more specific ones, each object needs to be assigned exactly one maximal sort that does not subsume any further sorts. Maximal sorts are also called species. Thus, if an object is said to be of a sort $\sigma$ that subsumes exactly the species $\{s_1, ..., s_n\}$, this means that the object is of one and only one of the species in this set.

*Attributes* may be defined on linguistic objects. If defined on some object, they will have a value on that object, which also is an object. The species of an object determines which attributes are defined on the object. Since the objects are *totally well-typed*, each attribute that can be defined on objects of some species also must be defined on each of them. The information which attributes are appropriate to which species is a part of the sort hierarchy, and so is the information which species the value of an attribute that is defined on some object may have, which depends on the species of this object.

A string of attributes is a *path*. Its value on an object, if it has any, is determined by determining the value of its leftmost attribute on this object and then evaluating the remainder of the path on the result, if any. If any value is undefined on the way, so is the path. Two distinct paths evaluated on an object may yield the same object as their value.

I shall not give a full characterisation of the model theory of SRL, whose details are not essential to understanding the paper, but content myself with a highly intuitive sketch. SRL provides for two types of atomic descriptions, *species assignments* and *path equations*. A species assignment $:\pi \sim s$ describes all objects on which the path $\pi$ has a value of species $s$[3] and a path equality $\pi =\pi'$ describes all

linguistic content would be helpful, and I agree. Unfortunately, I have not been able to construct such an example in the available time and it may be the case that no such examples will ever come up in linguistic practice. If they could be shown to be irrelevant, this might actually be seen as a welcome result, as the semantics of (Meurers, 2001) is of a considerably lower complexity than the one offered here while both coincide in cases where violations of Universality as discussed below do not actually occur. Hence, if one could be certain that the problems discussed will never attain relevance in actual grammar writing, staying with (Meurers, 2001) would definitely seem the right choice. Still, if it is agreed that the problems pointed out are genuine conceptual problems (and this no one has denied so far), it seems that the burden of proof regarding their harmlessness should be on those who prefer to leave things unchanged.

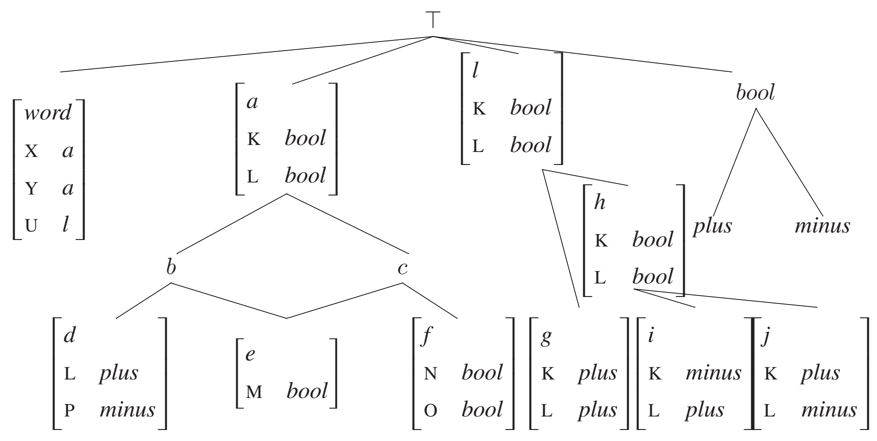
[3]It is convenient to assume that non-maximal sorts can also appear in such assignments, but these can straightforwardly be reduced to disjunctions of species assignments as long as no sort subsumes infinitely many species.

objects on which the value of $\pi$ is the same object as the value of $\pi'$. The set of objects described by a description is called the description's *denotation*. The syntax of SRL is like that of ordinary propositional logic, where e.g. $\wedge$ is interpreted as intersecting the denotations of the conjoined descriptions, but the syntax is hardly important in this paper. If a description cannot have a non-empty denotation, i.e. if no object at all can be described by it, the description is called *unsatisfiable*.

I trust that the reader will see how the familiar AVM notation relates to SRL. Precise specifications of this relation can be found in King (1989) and Richter (2004) (for RSRL, which contains SRL as a fragment). It is important to keep in mind that AVMs, just like SRL expressions, are only descriptions of linguistic objects that are conceived of as total in the sense of being totally well-typed and sort-resolved.

### 2.3 The sort hierarchy

Throughout the paper, the following sort hierarchy will be presupposed, which is derived from the one in (Meurers, 2001, p. 188). The sorts in the bottom line are species. Lines indicate the subsumtion relation: a sort subsumes another if it is above it and connected to it by a line.



## 3 Lexical rules and Universality

Lexical Rules are specified by Lexical Rule Specifications, typically given in the shape $A \mapsto B$. These are understood to mean that for every word described by $A$ there is one described by $B$ that has as much in common with $A$ as possible without violating $B$. The purpose of the semantics discussed in this paper is to make this idea explicit; in order for the output to have 'as much in common with $A$ as possible, properties of the input must be transferred to the output, and how this is supposed to happen needs to be made precise.

The semantics of Meurers (2001) as well as the one proposed here proceed by starting out from the class of objects described by $A \mapsto B$, the LR descriptions as it is before any transfers of properties are specified. $A \mapsto B$ is understood as an abbreviation for the AVM in (1).

(1) $\begin{bmatrix} \textit{lex\_rule} & \\ \text{IN} & A \\ \text{OUT} & B \end{bmatrix}$

The objects described by an LR descriptions as such are also called *proto-instances*. From the class of proto-instances, those are singled out in which properties are transferred in the way desired. These form the class of the rule's *instances*. The semantics will be spelt out indirectly by specifying a translation function that assigns to each LR descriptions a Lexical Rule Description (LR description) that only describes its licit instances. Section 6 briefly discusses this translation. The discussion of the intended meaning of LR descriptions will however present the contents of the proposals in a more direct manner, talking about objects (proto-instances and instances) directly instead of descriptions of them.

Since the class of instances of a rule is arrived at by preventing an appropriate subclass of the proto-instances from becoming instances, it follows that every instance of a rule, i.e. every actual pairing of an input with an output, is also a proto-instance of that rule. This seems right, as both input and output should at least conform to what is stated by the LR descriptions. Additional requirements, i.e. property transfers, are imposed monotonically without contradicting the specification provided by the LR descriptions.

Furthermore, one would expect an LR descriptions that is not in itself unsatisfiable, i.e. one that has proto-instances, to have instances, too. So merely trying to transfer certain properties should not make it impossible to pair an input with an output. This requirement can be called the criterion of **Preservation**.

**Criterion (Preservation).** *If a rule has a proto-instance, it also has an instance.*

Preservation is a consequence of the stronger criterion of **Universality**.

**Criterion (Universality).** *For every PI of an LR, there is an instance of this LR such that the values of the* IN *attribute on the PI and on the instance are congruent.*

Two objects $u_1$ and $u_2$ are *congruent* iff there is a bijection $f$ from the components of $u_1$ to those of $u_2$ such that for each component $v$ of $u_1$, $v$ and $f(v)$ are of the same species and for every attribute $\alpha$ and component $v$ of $u_1$, the value of $\alpha$ on $f(v)$ (if defined) is identical to $f(\text{the value of } \alpha \text{ on } v)$. A *component* of an object $u$ is any object that is the value of some path evaluated on $u$. In other words, congruent objects are look-alikes and any description that describes one of them also describes the other. Perhaps a bit more intuitively, the criterion can hence be thought of as demanding that an object that is the IN-value of a proto-instance must also be the IN-value of some instance.[4] Universality clearly implies that any rule that has proto-instances has instances, hence Preservation. But it goes beyond that in demanding, in effect, that the rule also must have something to say about every word that can be the IN-value of a proto-instance. This means that inspecting an LR descriptions should be sufficient to determine whether the specified rule will relate a given word to an output. Universality in effect limits the scope of property transfers, requiring that such transfers may not be carried out at the cost of reducing the applicability of the rule to any given input that conforms to its specification. So if, for instance, the input specification of a rule is simply *word* (as will be the case in some of the examples below), the rule should be expected to be applicable to any word whatsoever rather than to nothing at all or only to words that satisfy certain path equations.

In the following section, I shall discuss the semantics proposed by Meurers (2001) and show that it does not fulfill the criterion of Universality and not even that of Preservation. In section 5, an alternative semantics will be developed that can be shown to fulfill Universality (and thus Preservation).

## 4 Meurers's Semantics of Lexical Rule Specifications

The following exposition of the semantics for LR descriptions given in Meurers (2001) is in part my interpretation of what is intended in this paper. The formal definition of the semantics in the paper comes in the form of an algorithm that is supposed to translate LR descriptions into SRL descriptions that capture their intended semantics. Unfortunately however, the algorithm is flawed in that the rule responsible for transferring species can never apply, which renders it useless and leaves all path species untransferred. This is definitely not what was intended and so remaining faithful to (Meurers, 2001) in this respect would render about half of Meurers's proposal entirely uninteresting. What I present here is thus my understanding of what the algorithm was in fact supposed to achieve, based on the informal discussion in (Meurers, 2001).

It should be noted that the differences between my reading and the actual statement do not affect the way in which values are transferred. In this respect, the discussion below can be regarded as entirely faithful to (Meurers, 2001).

[4]Note that the formulation of Universality differs from demanding that, given an LR descriptions $A \mapsto B$, there must be an instance of the rule for every object described by $A$ such that the instance's IN-value is congruent to that object. This requirement would be to strong since an LR descriptions may specify certain components of the input and output to be token-identical. But if a token-identity between an input path $\pi$ and an output-path $\pi'$ is specified, it may be the case, for instance, that $\pi$ can have values of sorts that are not possible as sorts of values of $\pi'$. So there could be objects described by $A$ for which it would simply not be possible to apply the rule in a way that completely conforms to the specification, and in such a case, it seems, the rule should thus not pair the object with any output at all. But for such objects there also will not be any proto-instances with congruent IN-values to begin with. So under the formulation of Universality chosen here, these objects will not be considered at all.

## 4.1 Sort Transfer

The purpose of having a semantics for LR descriptions is to guarantee the transfer of properties of words which the LR descriptions does not mention from the rule's input to its output. There are two kinds of properties which need to be considered: the sorts of path values and path values themselves. The latter, which will be dealt with in the next section, is perhaps the more salient kind of transfer: (at least) if a path $\pi$ is not mentioned in an LR descriptions at all, the token identity $\text{IN}\,\pi = \text{OUT}\,\pi$ should hold if this is possible (in a sense to be made more precise below).

But even if a path $\pi$ is mentioned in the LR descriptions and the LR descriptions prohibits transferring its value because the rule effects some change on it, one might still expect $\text{IN}\,\pi$ and $\text{OUT}\,\pi$ to be of the same species whenever this is possible. So even if the value of a path cannot be transferred (which would of course imply the transfer of its species), the species possibly can and, in view of the goal of transferring as much as possible, it should.

The intuition behind the transfer of path value sorts is best conveyed by an example. Consider the simple rule in (2), taken from (Meurers, 2001, p. 188).

(2) $word \mapsto \begin{bmatrix} \text{X} & c \end{bmatrix}$

This rule takes any word as its input and its output should be a word that is like the input except for having an X-value of sort $c$, which subsumes the species $e$ and $f$. On *word*s, X is allowed to have values of sort $a$, which subsumes $d$, $e$ and $f$. The rule should then allow for the following configurations:

(3) a. $\begin{bmatrix} \text{IN X} & d \\ \text{OUT X} & e \end{bmatrix}$

b. $\begin{bmatrix} \text{IN X} & d \\ \text{OUT X} & f \end{bmatrix}$

c. $\begin{bmatrix} \text{IN X} & e \\ \text{OUT X} & e \end{bmatrix}$

d. $\begin{bmatrix} \text{IN X} & f \\ \text{OUT X} & f \end{bmatrix}$

In (3c) and (3d), the species of X in the input is subsumed by $c$ and thus compatible with what the output demands. In such a case, the species is transferred: if X is of species, say, $e$ in the input, it also needs to be in the output, ruling out $f$ as a possible output species in such cases. But if X in the input is of species $d$, which is not subsumed by $c$, the species of X in the output must differ from that in the input if the LR descriptions is to be obeyed. In this case, where no transfer is possible, any species subsumed by $c$ is allowed.

$$\begin{bmatrix} \textsc{in} & \begin{bmatrix} \textsc{u} & \begin{bmatrix} g & \\ \textsc{k} & \textit{plus} \\ \textsc{l} & \textit{plus} \end{bmatrix} \end{bmatrix} \\ \textsc{out} & \begin{bmatrix} \textsc{u} & \begin{bmatrix} i & \\ \textsc{k} & \textit{minus} \\ \textsc{l} & \textit{plus} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

(a) U-value of species $i$

$$\begin{bmatrix} \textsc{in} & \begin{bmatrix} \textsc{u} & \begin{bmatrix} g & \\ \textsc{k} & \textit{plus} \\ \textsc{l} & \textit{plus} \end{bmatrix} \end{bmatrix} \\ \textsc{out} & \begin{bmatrix} \textsc{u} & \begin{bmatrix} j & \\ \textsc{k} & \textit{plus} \\ \textsc{l} & \textit{minus} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

(b) U-value of species $j$

Figure 1: Proto-instances cancel each other out.

Meurers (2001) implements this intuition about sort transfer as follows: the basic idea is to allow no proto-instance $x$ of a rule to become an instance if another proto-instance $y$ can be found such that, for some path $\pi$, IN $\pi$ and OUT $\pi$ have values of the same species $s$ on $y$, IN $\pi$ also has a value of species $s$ on $x$ but OUT $\pi$ has a value of a species distinct from $s$ on this proto-instance. In such a case, $x$ is said to be *cancelled out by* $y$. In the case of rule (2), proto-instances configured as in (4) are cancelled out due to the existence of those configured as in (3c).

(4) $\begin{bmatrix} \textsc{in x} & e \\ \textsc{out x} & f \end{bmatrix}$

## 4.2 Problems

Meurers's way of transferring sorts leads to a violation of the criterion of Universality. This can be seen by inspecting rule (5).

(5)
$$\begin{bmatrix} \textsc{u} & g \end{bmatrix} \mapsto \begin{bmatrix} \textsc{u} & \begin{bmatrix} h & \\ \textsc{k} & \textit{bool} \\ \textsc{l} & \textit{bool} \end{bmatrix} \end{bmatrix}$$

The input is required to have a U-value of species $g$. Objects of species $g$ only allow for objects of species *plus* as values of the attributes K and L. In the output, the value of U is supposed to be of sort $h$, i.e. of species $i$ or $j$. Objects of species $i$ only allow K-values of species *minus* and L-values of species *plus*. The same in reverse holds for $j$. This is illustrated in Fig. 1.

It is easily seen that using Meurers's way of transferring sorts will result in the rule having no instances at all, i.e. a violation of the criteria of both Preservation and, *a fortiori*, Universality. Proto-instances configured as in Fig. 1(a) cancel out those configured as in Fig. 1(b): the U L-value of the proto-instances described by Fig. 1(a) is of species *plus* in both the input and output. Regarding the proto-instances described by Fig. 1(b), it is of species *plus* in the input, too, but of species *minus* in the output. According to Meurers's approach to sort transfers, the proto-instances described by Fig. 1(b) are thus cancelled out and may not become instances. The same clearly holds in reverse if the path U K is considered instead of U L.

So while the rule should accept any word as input with a U-value of species $g$, it in fact will not accept any input at all, thus violating both Preservation and Universality.

## 4.3 Value Transfer

Meurers suggests to transfer path values from a rule's input to its output along the following lines: $(i)$ only the values of paths which are not mentioned in the rule's output specification are transferred. $(ii)$ the value of each such path has to be transferred whenever possible.

$(i)$ is meant to keep the semantics from becoming overly complex. Because enforcing token-identity between input and output for paths that also get further specified in the output may of course lead to certain conflicts if the input and the specification of the output are incompatible, Meurers suggests that complex strategies will be needed to prevent inconsistencies. In this paper, I follow Meurers in adopting $(i)$, but without strongly endorsing it.

$(ii)$ can be made precise as follows: Let $\mathsf{Approp}(s, \alpha)$ denote the set of species the value of $\alpha$ can have on an object of species $s$. Now let $\pi$ be a path such that OUT $\pi$ is mentioned in the LR descriptions in question and let $\alpha$ be an attribute such that OUT $\pi\alpha$ is not mentioned in the LR descriptions, in accord with $(i)$. Then IN $\pi\alpha$ = OUT $\pi\alpha$ must hold of every instance of the rule on which IN $\pi$ has a species $s$ and OUT $\pi$ has a species $s'$ such that $\mathsf{Approp}(s, \alpha) \cap \mathsf{Approp}(s', \alpha) \neq \emptyset$. This means that if it is possible for objects of the species of IN $\pi$ and OUT $\pi$ to have identical $\alpha$-values, then they must have identical $\alpha$-values.

In terms of cancelling out of proto-instances, this approach amounts to cancelling out any proto-instance for which $\mathsf{Approp}(s, \alpha) \cap \mathsf{Approp}(s', \alpha)$ is not empty but IN $\pi\alpha$ = OUT $\pi\alpha$ does not hold.

This approach to transferring values also leads to violations of the requirement of Universality. For one thing, this is due to the fact that the transfer of an attribute $\pi\alpha$ is required whenever $\mathsf{Approp}(s, \alpha) \cap \mathsf{Approp}(s', \alpha) \neq \emptyset$ for $s$, $s'$ the species of IN $\pi$ and OUT $\pi$, respectively. Universality is then not generally respected since, if e.g. $\mathsf{Approp}(s, \alpha) = \{t, q\}$ and $\mathsf{Approp}(s', \alpha) = \{q\}$, the condition is fulfilled and the corresponding paths will be identified, but this clearly restricts the possible species for the IN-path to $q$, ruling out $t$ even if there are proto-instances on which the IN-path has this species.

Additionally, the following rules all lead to violations of Universality.

(6) a. $$\begin{bmatrix}\text{X K} & \textit{plus}\end{bmatrix} \mapsto \begin{bmatrix}\text{X} & \boxed{1} \\ \text{Y} & \boxed{1}\begin{bmatrix}\text{K} & \textit{minus}\end{bmatrix}\end{bmatrix}$$

b. $\begin{bmatrix} \text{X K} & \textit{plus} \\ \text{Y K} & \textit{minus} \end{bmatrix} \mapsto \begin{bmatrix} \text{X} & \boxed{1} \\ \text{Y} & \boxed{1} \end{bmatrix}$

c. $word \mapsto \begin{bmatrix} \text{X} & \boxed{1} \\ \text{Y} & \boxed{1} \end{bmatrix}$

Regarding (6a), the value of X K will be required to be transferred: according to the signature, all species allowed as values of X ($d$, $e$ and $f$) allow for values of K of species *plus* and *minus* alike. So the set of commonly accepted species is non-empty and X K will hence be required to be transferred and consequently of species *plus*. But this contradicts the output specification which requires OUT X K = OUT Y K and that OUT Y K be of species *minus*, which of course implies that OUT X K also is of this species. The description that Meurers's approach derives from the LR descriptions in (6a) is thus contradictory and cannot licence any $lex\_rule$ objects at all. The rule thus has no effect.

(6b) leads to the same kind of contradiction, but in this case it is not due to the output specification but to the simultaneously required transfer of the values of X K and Y K, which are required to be of different species in the input but need to be identical in the output. So this rule, again, has no effect at all.

Unlike rules (6a) and (6b), rule (6c) is not contradictory and hence respects Preservation, but it still violates Universality. To see this, consider again the attribute K. Since rule (6c) should accept any word, it should accept words with an X K-value of species *plus* and a Y K-value of species *minus* in particular. But in fact it does not: according to Meurers's approach, the values of both X K and Y K will be transferred from the input to the output, i.e. OUT X K = IN X K and OUT Y K = IN Y K in any instance. Likewise, OUT X and OUT Y need to be the same object in any instance according to the LR descriptions itself. This implies that OUT X K = OUT Y K also holds and hence IN X K = IN Y K. Contrary to expectations, thus, rule (6c) will only accept words as its inputs on which IN X K and IN Y K are the same object. Since there are proto-instances of the rule for which this does not hold, this is again a violation of Universality.

## 5 The Alternative

To see which issues precisely an alternative proposal needs to address, first note that at the heart of the violations of Universality in the system of (Meurers, 2001) observed in the preceding sections are the following two properties of this semantics:

- Whether a proto-instance is cancelled is decided by inspecting paths in isolation.
- Comparison of proto-instances is not sensitive to whether they do have congruent IN-values or not.

The first point is illustrated by all the problematic rules presented so far. Regarding example (5), a proto-instance can be cancelled because it does not transfer the species of U K while another does. This happens without regard to the fact that such a proto-instance will transfer the species of U L while the one it is cancelled by will not. Similarly rule (6a), for instance, will always 'try' to transfer the value of X K. Since this is impossible without violating the output specification, the rule cannot apply to any word at all. This merely local consideration of paths is misguided and a more global approach called for. Such an approach is developed in this section. It will not rely on the inspection of paths in isolation in order to effect cancellation of proto-instances but instead take into consideration the set of paths whose properties are transferred by a proto-instance. A proto-instance will be cancelled if the set of paths that it transfers the properties of is a proper subset of the corresponding set for some other proto-instance, i.e. if the latter transfers the properties of more paths than the former.

This alone would however not suffice to guarantee Universality, due to the second of the two problems named above. It would still be possible for all proto-instances with an input object of a certain shape to be cancelled, leaving no instance with an input object congruent to those of the cancelled instances. This problem is solved here by allowing cancellation only inside of classes of proto-instances with congruent input objects. Thus, since congruence clearly is an equivalence relation, the class of proto-instances is partitioned into 'input congruence classes' and only inside of these classes proto-instances are cancelled that do not transfer the properties of a maximal set of paths.

## 5.1 Spelling out the Alternative

As stated above, the alternative approach to transferring path properties proposed here rests on the idea of maximising the set of paths whose properties are transferred. For each proto-instance, call its *SFrame* the set of paths with species transferred and its *VFrame* the set of paths with values transferred, defined as in (7). For any LR descriptions $\lambda$, $\mathit{Men}(\lambda)$ is the set of paths $\pi$ such that $\textsc{out}\,\pi$ is mentioned in $\lambda$. *Edge* is the set of all paths with values that might be transferred by a given LRs, i.e. the set of mentioned paths that extend an unmentioned one by one attribute: $\mathit{Edge}(\lambda) = \{\pi\alpha \mid \pi \in Men(\lambda) \,\&\, \pi\alpha \notin Men(\lambda)\}$.

(7) a. $\mathit{SFrame}(x) =$
$\{\pi \in \mathit{Men}(\lambda) \mid \textsc{in}\,\pi \text{ and } \textsc{out}\,\pi \text{ have the same species on } x\}$

b. $\mathit{VFrame}(x) =$
$\{\pi \in \mathit{Edge}(\lambda) \mid \textsc{in}\,\pi, \textsc{out}\,\pi \text{ are defined and } \textsc{out}\,\pi = \textsc{in}\,\pi \text{ on } x\}$

As a first approach, one might let a proto-instance $x$ cancel another proto-instance $y$ if $\mathit{SFrame}(x) \supset \mathit{SFrame}(y)$ or $\mathit{VFrame}(x) \supset \mathit{VFrame}(y)$, i.e. if $x$ transfers the sorts or values of all paths of which $y$ transfers them and also of some additional paths for which $y$ does not. But, as has already been mentioned above, it could very well happen then that some proto-instance cancels another whose IN-value is not congruent to its own. E.g., in the case of rule (8), only proto-instances with a IN U-value of species $g$ allow for a *VFrame* containing both U K and U L.

(8) $\mathit{word} \mapsto \begin{bmatrix} \text{U} & g \end{bmatrix}$

The existence of such proto-instances would effect the cancellation of all those with IN U-values of species $i$ or $j$, leading to a violation of Universality again. The solution to this problem, likewise mentioned above, is to allow cancellation only within the equivalence classes of proto-instances according to the equivalence relation of having congruent IN-values.

Letting the meaning of $\mathit{IVC}(x, y)$ be that the *lex_rule* objects $x$ and $y$ have congruent IN-values, the intended way of performing sort and value transfer can now be expressed as in definitions 1 and 2.

**Definition 1** (Species Transfers)**.**
$\mathit{STrans}(PI) =$
$\{x \in PI \mid \mathit{For\ no\ } y \in PI : \mathit{IVC}(x, y) \mathit{\ and\ SFrame}(y) \supset \mathit{SFrame}(x)\}$

**Definition 2** (Value Transfers)**.**
$\mathit{VTrans}(PI) =$
$\{x \in PI \mid \mathit{For\ no\ } y \in PI : \mathit{IVC}(x, y) \mathit{\ and\ VFrame}(y) \supset \mathit{VFrame}(x)\}$

Let $PI(\lambda)$ denote the set of proto-instances of a LR specification $\lambda$.[5] Denote by $\mathit{STrans}(PI(\lambda))$ the set of all proto-instances $x$ of $\lambda$ such that no proto-instance with an IN-value congruent with that of $x$ exists that transfers the species of a proper superset of the paths whose species are transferred by $x$. $\mathit{VTrans}(PI(\lambda))$ is the analogous notion for value transfers. Since there will clearly exist maximal elements wrt $\supset$ (note that both *SFrame* and *VFrame* are finite), these sets are guaranteed to be non-empty and to contain, for every proto-instance of $\lambda$, some element with a congruent IN-value. So Universality is clearly respected by *STrans* and *VTrans*.

The set of instances of the rule is given by

(9) $\mathit{Transfers}(PI(\lambda)) = \mathit{VTrans}(PI(\lambda)) \cap \mathit{STrans}(PI(\lambda))$

Does *Transfers* still respect Universality? One can show that

$$\begin{array}{rl} & \mathit{STrans}(\mathit{VTrans}(PI(\lambda))) \\ = & \mathit{VTrans}(\mathit{STrans}(PI(\lambda))) \\ = & \mathit{VTrans}(PI(\lambda)) \cap \mathit{STrans}(PI(\lambda)) \end{array}$$

So the order in which the transfers are performed (values before sorts, sorts before values or in parallel) is immaterial and, since each of the transfer operations respects Universality, so does *Transfers* itself.

[5]Strictly speaking, this 'set' is of course a proper class. It is possible however to find sets which contain, for any PI, a congruent object. Since the actual formalization of the ideas laid out here proceeds indirectly, as sketched in section 6, thus operating on descriptions instead of the objects themselves, there is no reason to worry here.

## 5.2 Application

Let us consider the problematic rules given above to see the results of the proposed semantics. Since *Transfers* is known to respect Universality, what remains to be seen is only which Transfers the proposed semantics will actually licence in these cases.

Regarding rule (5) and the two interesting classes of proto-instances which this rule has, shown in Figures 1(a) and 1(b), the former of these have *SFrame*s that contain U L but not U K while those of the latter contain U K but not U L. Consequently, neither *SFrame*$(x) \subset$ *SFrame*$(y)$ nor *SFrame*$(y) \subset$ *SFrame*$(x)$ can ever hold if $x$ is a proto-instance described by Figure 1(a) and $y$ is one described by Figure 1(b). Of course, inside of the classes of proto-instances described by each of 1(a) and 1(b), cancellation will take place: e.g. some proto-instances will have *SFrame*s that contain X while others will not, and the same is true regarding the *VFrames*. Thus those proto-instances from whose frames X is missing will be cancelled due to those in whose frames it is contained. But 'across' 1(a) and 1(b), no cancellation can happen and thus proto-instances described by each of these will be among the instances. Transfer of U K and U L thus happens, but only on one of the paths U K and U L at a time, as it is impossible for both together.

Consider next rule (6a). No proto-instance of this rule can fulfill OUT X $=$ IN X, i.e. have a *VFrame* that contains X, which would require X K to have the species *plus* and *minus* at once. But under the present account, this does not have the effect of cancelling all proto-instances of the rule. In Meurers's semantics, this is the result because it uncompromisingly demands that X be transferred in the case of rule (6a) and thus cancels each proto-instance that does not transfer it. Under the approach presented here, there just is no proto-instance whose *VFrame* contains X, but this does of course not mean that there are none with maximal *VFrame*s and *SFrame*s. These will in fact exist for every satisfiable LR descriptions, and these will be instances of the rule.

Regarding (6b), the situation is slightly different and reminiscent of rule 5: there are *VFrame*s that contain X, and these cannot contain Y, which would again require X K to have as its species both *plus* and *minus*. Conversely and analogously, there are also *VFrame*s that contain Y but not X. A proto-instance whose *VFrame* contains X can thus never cancel one whose *VFrame* contains Y and *vice versa*. As a result, analogously to the case of rule (5), one of X and Y will be transferred in each instance but never both.

Basically the same is true regarding rule (6c), but in this case it is important that cancellation can only take place within a congruence class. While proto-instances of this rule can be found whose *VFrame*s contain both X and Y, none of these can have an input with X K and Y K-values of distinct species. But proto-instances with such IN-values exist, and inside of the congruence classes of such proto-instances, transferring the values of both X K and Y K is as impossible as it is in general in the case of rule (6b). For inputs of this shape, thus, it also holds that one and only one of these paths will have its value transferred.

## 6 Indirect formulation

In this section, I shall briefly discuss how the semantics described above is actually implemented under Meurers's and my approach. As remarked above, the semantics is given in an indirect manner. Thus LR descriptions are not given an interpretation themselves under which they denote their instances but are translated into SRL descriptions (LR descriptions), and these in turn denote the instances according to the semantics of SRL. That the semantics can be realised in this way shows that the expressive means used do not actually go beyond SRL. Specifying LRs is thus just a more convenient way of writing SRL descriptions.

Meurers's translation function builds on the notion of what I call the *Kepser Normal Form* (KNF) of SRL descriptions. This kind of normal form was introduced by Kepser (1994) as a tool for deciding satisfiability of SRL descriptions. A description in KNF is a description in Disjunctive Normal Form (DNF) that has certain closure properties.

For the sake of convenience, a description in DNF can be represented by a set of sets of literals, where a literal is an atomic description or a negated atomic description. Adopting the terminology used by Meurers (2001), I call such a set a *matrix* and its elements *clauses*. The meaning of a DNF $\delta$ will then be that of the SRL description

$$\bigvee_{\alpha\in\delta}\bigwedge_{\beta\in\alpha}\beta$$

i.e. the disjunction of the conjunctions of the literals contained in each of its clauses.

A matrix is in KNF iff each of its clauses is in KNF. Discussing the exact definition of a clause in KNF is beyond the scope of this paper, but what matters here is that such clauses are highly explicit about the objects they describe as far as the paths are regarded that are mentioned in them: for each path $\pi$ that occurs at all in the literals of a clause in KNF and is defined on any of the objects it describes, there is some species $s$ for which the sort assignment $:\pi \sim s$ is contained in the clause. The same holds for the equality $:\pi =:\pi$ (which is not completely trivial but expresses the definedness of $\pi$.) Furthermore, each clause in a matrix in KNF is satisfiable.[6]

These properties of normal clauses make them well suited to represent the proto-instances of LRs in the translation process. Hence, as the first step of the translation, the LR descriptions is normalised. This results in a matrix in KNF. Sort transfers can then be formulated as above, but instead of cancelling proto-instances directly, the clauses that denote them are dropped from the matrix. In Meurers's system, if the matrix $\mathcal{M}$ contains clauses $\mathcal{C}$ and $\mathcal{C}'$ such that $:\text{IN}\,\pi\ \sim$

[6]Hence normalisation, for which Kepser (1994) provides an algorithm, also provides the decision procedure that he is after: a description is satisfiable iff its KNF is not empty.

$s, {:}\text{OUT}\,\pi \sim s \in \mathcal{C}$ and ${:}\text{IN}\,\pi \sim s, {:}\text{OUT}\,\pi \sim s' \in \mathcal{C}'$ for some path $\pi$ and (distinct) species $s$ and $s'$, then $\mathcal{C}'$ is dropped from the matrix. This means to drop all proto-instances described by $\mathcal{C}'$, which is what is wanted.

Meurers (2001) realises value transfers by adding ${:}\text{IN}\,\pi\alpha = {:}\text{OUT}\,\pi\alpha$ to every clause $\mathcal{C}$ that fulfills the following:

- ${:}\text{OUT}\,\pi = {:}\text{OUT}\,\pi \in \mathcal{C}$
- ${:}\text{OUT}\,\pi\alpha = {:}\text{OUT}\,\pi\alpha \notin \mathcal{C}$
- ${:}\text{IN}\,\pi \sim s, {:}\text{OUT}\,\pi \sim s' \in \mathcal{C}$
- $\mathsf{Approp}(s, \alpha) \cap \mathsf{Approp}(s', \alpha) \neq \varnothing$

Since $\mathcal{C}$ is in KNF, the first condition ensures that $\text{OUT}\,\pi$ is mentioned in the rule and defined on each of the proto-instances licenced by $\mathcal{C}$. The second condition ensures that the path resulting from this one by appending $\alpha$ is not defined and mentioned in the rule. The last two conditions ensure that $\pi\alpha$ can in principle have the same value on the input and output of the proto-instances the clause describes. Adding ${:}\text{IN}\,\pi\alpha = {:}\text{OUT}\,\pi\alpha$ then requires these values to actually be the same, i.e. it cancels all proto-instances described by the original clause on which they are not, just as required by the semantics stated above.

This highlights an asymmetry between the way in which sorts and values are transferred under Meurers's approach: while sorts are transferred by dropping certain clauses from the matrix, values are transferred by adding literals to clauses. In contrast, the present approach treats sort and value transfers in an exactly parallel fashion, but this requires an additional step. The reason is that normal clauses are not yet explicit enough: while a normal clause assigns a species to the value of every path that occurs in it and is defined on the objects it describes, the clauses are not in general explicit about which path equalities the objects they describe satisfy. So while the notion of an *SFrame* is straightforwardly adapted to the indirect semantic account as in (10), the notion of a *VFrame* is not.

(10) $\textit{SFrame}(\mathcal{C}) = \{\pi \mid \text{For some species } s, {:}\text{IN}\,\pi \sim s, {:}\text{OUT}\,\pi \sim s \in \mathcal{C}\}$

While every path that is defined on the objects a normal clause describes is also explicitly assigned a species by the clause, the absence of a literal ${:}\text{OUT}\,\pi = {:}\text{IN}\,\pi$ from a clause does not mean that this equality may not describe some proto-instances the clause licences; it just does not, in general, need to describe them. It is thus not possible to compare the clauses with regard to the value transfers they enforce because proto-instances in which some such transfer is performed and those in which it is not performed are not in general described by distinct clauses.

The missing explicitness about transferred values can be supplied in the following way. Let the set of value transfer specifications be the set of all possible equations between corresponding paths such that the $\text{OUT}$ path is mentioned in $\mathcal{C}$: $VTr(\mathcal{C}) := \{{:}\,\text{IN}\,\pi = {:}\,\text{OUT}\,\pi \mid {:}\,\text{OUT}\,\pi \in \mathcal{C}\}$. Now consider the KNF of an LR descriptions $\lambda$. Under reasonable assumptions about the form of LR descriptions,[7] this will be a matrix $\mathcal{M}$ such that for each $\mathcal{C} \in \mathcal{M}$ and $\mathcal{C}' \in \mathcal{M}$, $VTr(\mathcal{C}) = VTr(\mathcal{C}')$, which can thus be referred to as $VTr(\mathcal{M})$. Now a new matrix can be defined as in (11). This matrix consists of all clauses that are the union of some clause in $\mathcal{M}$ with some subset of $VTr(\mathcal{M})$. So for every set of mentioned paths, there is a clause in this matrix that specifies all elements of this set as transferred.

(11) $\{\mathcal{C} \cup E \,|\, \langle \mathcal{C}, E \rangle \in \mathcal{M} \times \mathcal{P}(VTr(\mathcal{M}))\}$

Obviously, not all of these clauses can be expected to be satifsiable, i.e. to describe anything at all. But normalising the new matrix again will remove all the inconsistent clauses.

The matrix that results from this second normalisation step thus contains, for each set of paths whose values can jointly be transferred, a clause that explicitly states the values to be transferred on the objects it describes. On the basis of this matrix it is now possible to define the notion of a *VFrame* appropriately, which is done in (12)

(12) $\textit{VFrame}(\mathcal{C}) = \{\pi \in Edge(\mathcal{C}) \,|\, \text{:IN}\,\pi = \text{:OUT}\,\pi \in \mathcal{C}\}$

$Edge(\mathcal{C})$ corresponds to $Edge(\lambda)$ as used in section 5 and can be defined as in definition 3.

**Definition 3.** *Edge of a clause*
$Edge(\mathcal{C}) =$
$\{\pi\alpha \,|\, \text{:OUT}\,\pi = \text{:OUT}\,\pi \in \mathcal{C} \;\&\; \text{:OUT}\,\pi\alpha = \text{:OUT}\,\pi\alpha \notin \mathcal{C}, \textit{for } \alpha \textit{ an attribute.}\}$.

The last missing ingredient is a way to ensure that cancellation of clauses only takes place within what I called a "congruence class" above. The details of how this can be done are rather involved, but the basic idea is straightforward: from any clause $\mathcal{C}$, a description $\mathcal{C}^{:\text{IN}}$ can be derived that describes all and only objects which are congruent with the :IN-value of some object described by $\mathcal{C}$.[8] Call $\mathcal{C}^{:\text{IN}}$ the IN-value description for $\mathcal{C}$. The notion of proto-instances having congruent IN-values can then be replaced by the notion of clauses having equivalent IN-value descriptions, which I notate as $EQV(\mathcal{C}, \mathcal{C}')$.[9]

The indirect semantics can now be stated in a fashion parallel to that used for the direct semantics above, making use of the following definitions.

[7]To be precise, it is assumed that an LR descriptions is stated by giving the input AVM, the output AVM and a (possibly empty) set of path inequalities. About the AVMs it is assumed that they do not contain any logical symbols. This means that each AVM is equivalent to a conjunction of SRL literals, but where it is allowed to use sort assignments that assign non-maximal sorts. Rule specifications found in the literature typically obey these constraints and a rule whose specification does not can equivalently be expressed by a set of rules that do.

[8]A detailed account of how to do this is given in Lahm (2012).

[9]$EQV(\mathcal{C}, \mathcal{C}')$ is decidable, e.g. by using the fact that $\delta$ and $\delta'$ are equivalent iff bot $\delta \wedge \neg\delta'$ and $\neg\delta \wedge \delta'$ are unsatisfiable.

**Definition 4** (Species Transfers)**.**
$STrans(\mathcal{M}) =$
$\{\mathcal{C} \in \mathcal{M} \,|\, \textit{For no } \mathcal{C}' \in \mathcal{M} : EQV(\mathcal{C}, \mathcal{C}') \textit{ and } SFrame(\mathcal{C}') \supset SFrame(\mathcal{C})\}$

**Definition 5** (Value Transfers)**.**
$VTrans(\mathcal{M}) =$
$\{\mathcal{C} \in \mathcal{M} \,|\, \textit{For no } \mathcal{C}' \in \mathcal{M} : EQV(\mathcal{C}, \mathcal{C}') \textit{ and } VFrame(\mathcal{C}') \supset VFrame(\mathcal{C})\}$

Each of the clauses in each of these two matrices has as its denotation a subclass of the proto-instances that the semantics specified in section 5 admits into the respective *STrans* and *VTrans* sets as defined in that section. Since the matrices are interpreted disjunctively, their denotation is the union of all these classes and thus the same as that of the transfer functions that were specified directly.

## 7 Conclusion

In this paper, I have shown that the semantics of LRs specified in Meurers (2001) disrespects what I have called the criterion of Universality. Words that one would expect to be licit inputs to a rule may not in fact be that under the semantics suggested by Meurers, i.e. there will be no output for them. In extreme cases, rules may become unsatisfiable although their specifications are not. I have shown the reason for that to be that in Meurers (2001) whether properties of a path are transferred from the input to the output is decided by inspection of that path in isolation, without regard for the non-local effects that transferring the properties may have, and that the system developed performs transfers without regard to the possible shapes of input objects to the rule. I further introduced an alternative semantics that solves both of these problems.

An interesting question that I must leave open is whether the problems pointed out actually affect realistic grammar writing. To argue that they do not would require showing that sort hierarchies and LRs of the kind that lead to the problems are by their very nature pathological and can be expected not to occur in actual grammar writing. This would actually be a welcome result since the complexity of the criticised account by Meurers (2001) seems to be considerably lower than that of mine.[10] I do not consider it unlikely that this might be the case, but determining whether it is is not the goal of this paper, but rather to show that the question matters.

## References

Kepser, Stephan. 1994. A satisfiability algorithm for a logic for typed feature structures. Arbeitspapiere des SFB 340 60 Universität Tübingen, Germany.

[10]Note that the size of the set specified in (11) grows exponentially with the number of $Edge$ paths.

King, Paul. 1989. *A logical formalism for head-driven phrase structure grammar*: University of Manchester dissertation.

Lahm, David. 2012. Revisiting the description-level approach to lexical rules in HPSG. Unpublished master's thesis, University of Tübingen.

Meurers, Walt Detmar. 2001. On Expressing Lexical Generalizations in HPSG. *Nordic Journal of Linguistics* 24(2). 161–217.

Pollard, Carl J. & Ivan A. Sag. 1994. *Head-driven phrase structure grammar*. Chicago: University of Chicago Press.

Richter, Frank. 2004. *A mathematical formalism for linguistic theories with an application in Head-Driven Phrase Structure Grammar*: Eberhard-Karls-Universität Tübingen Phil. dissertation (2000).