Arbeitspapiere des SFB 340
Bericht Nr. 00, Xxxx 19??

RSRL FOR HPSG

FRANK RICHTER

ABSTRACT. This paper defines an AVM syntax for RSRL (Richter, Sailer, and Penn 1999), thus closing the gap between a common if informal linguistic notation and an all-comprehensive formalism for HPSG. With a well-defined and easily readable AVM syntax and RSRL's model theory for grammars, one obtains a direct and rigorous mathematical interpretation for specifications of natural language fragments of the kind presented in Pollard and Sag (1994).

1. INTRODUCTION

Linguists are familiar with a certain informal notation called AVM (attribute-value matrix) diagrams, which have evolved into a *lingua franca* for HPSG and related grammar frameworks because of their apparent graphical perspicuity and aptness for depicting complex principles in a highly compact fashion. But if one strips them of the surrounding text which explains their meaning, one quickly discovers that interpreting complex AVM diagrams is, in fact, often not as simple and straightforward as it may appear. Consider the following example, which is adapted from Kathol (1995, p. 291):[1]

$$(1)\quad sign \rightarrow \neg \left(\begin{bmatrix} \textsc{dom} & \boxed{2} \\ \textsc{ss loc context} & \boxed{3} \end{bmatrix} \right.$$

RSRL originated in joint work with Manfred Sailer. I am grateful to Tilman Höhle, Paul King, Detmar Meurers, Gerald Penn, Carl Pollard, Manfred Sailer, and Kiril Simov for extensive discussions on RSRL and on the formalization of HPSG. Thanks to Mike Calcagno and Adam Przepiórkowski for reading earlier versions of this paper. They suggested many more improvements than I managed to incorporate. Thanks are also due to Christian Richard for helping me with the English language.

[1]For transparency, I translate the functional notation for relations of Kathol (1995) into relational notation.

ISSN 0947-6954/??

$$\wedge\ \texttt{shuffle}\left(\left\langle \begin{bmatrix} \text{TOPO } \neg\, nf \\ \text{SS LOC CONTENT } \boxed{1} \end{bmatrix} \right\rangle, \boxed{4}, \boxed{2}\right)$$

$$\wedge\ \texttt{member}\left(\begin{bmatrix} anchor \\ \text{ARG1 } index \\ \text{ARG2 } \boxed{1}\ psoa \end{bmatrix}, \boxed{3}\right)\Bigg)\Bigg)$$

Among the questions which could be asked about the AVM diagram in (1) might be the following: What exactly is the meaning of the two relational expressions under the scope of negation? Assuming the tags in the diagram stand for variables, which in turn suggests that they are either bound by quantifiers or get their value from an assignment function, by which — if any — quantifiers are they bound, and what is the scope of those quantifiers? Can they be standard quantifiers of first order logic? Both relations contain one argument which is itself a complex AVM diagram. Is there also an implicit quantification on those arguments? Assuming there is such a quantification, how is the variable $\boxed{1}$ bound relative to that quantification, or does it not matter?

This paper gives answers to all of these questions. It does so by defining an AVM syntax for RSRL (Relational Speciate Re-entrant Language), a logical language which provides a comprehensive formalism for HPSG. The meaning of AVM diagrams like (1) will be reduced to the simple and well-understood semantics of RSRL's description language, and thus be made fully explicit. RSRL is an extension by restricted quantification and relations of SRL (Speciate Re-entrant Logic), a logic of descriptions that was specifically designed for HPSG (King 1989; King 1994). The suitability of RSRL to current HPSG grammar specifications is discussed at length in Richter, Sailer, and Penn (1999). The interested reader is referred there for the linguistic motivation and illustrations of examples from the literature. While I assume that RSRL provides an adequate formalism that is faithful to the requirements put forth in Pollard and Sag (1994) and other HPSG literature, I do not suppose that the reader is familiar with RSRL; and I will present and discuss all components of the syntax and semantics of the language of RSRL.[2]

In recent work in HPSG, RSRL has been expressly espoused as underlying formalism in order to state complex grammar principles (Calcagno, Hukari, and Levine 1998; Meurers 1998; Penn 1999; Przepiórkowski 1997;

[2]A preliminary version resulting from joint work with Manfred Sailer was published in Richter (1997). However, the treatment of *chains* there is not compatible with a non-algebraic characterization of exhaustive models as adopted in DEFINITION 2.22 below, since it does not distinguish lists and chains syntactically. Richter (1997) fails to see this.

Przepiórkowski 1998; Richter and Sailer 1998; Richter and Sailer 1999). Unfortunately, the explicit use of RSRL in linguistic work is partly hampered by the idiosyncratic syntax of its formal language. Since RSRL is a direct descendant of King's (1989) SRL, it does not have a syntax that lends itself to writing formulae which are readable for someone who is not a feature logic acolyte. Although the particular format of the syntax of a formal language is of course irrelevant for the purposes of mathematical precision, it is an important issue for linguistic applications. To be of maximal use, a successful formalism for HPSG should provide an exact definition and an interpretation of a language that comes as close to the actual practice in the literature as the pervasive informal conventions permit. A formalism whose syntax complies with this demand is not only intuitively readable for the linguist; at the same time, it is suitable for rigorous formal specifications of natural language fragments.

The objective of this paper is to clarify the meaning of AVM diagrams by establishing an AVM syntax for RSRL. In section 2, the syntax and semantics of the formal language of RSRL are defined. I say what a grammar is and what it means for a grammar to be true of a natural language. Section 3 presents the syntax of the AVM language. The meaning of its expressions is determined by a translation function from AVM formulae to traditional RSRL formulae, and it is shown that, for every grammar written in RSRL syntax, there is a grammar written in AVM syntax that is true of the same natural language. In other words, the AVM syntax is, in a linguistically relevant sense, equivalent to the original syntax of RSRL. Finally, in section 4, I discuss examples of grammatical principles in AVM notation and introduce some abbreviatory conventions which further simplify its syntax. With those conventions, the AVM diagrams of Pollard and Sag (1994) can essentially be interpreted in RSRL.

2. RSRL

Following SRL, descriptions of RSRL are true or false of objects in an interpretation. RSRL provides a class of formal languages consisting of a signature and a class of interpretations. The signature provides the sets of symbols which generate the formal language. Each interpretation of a signature provides a set of objects and assigns each symbol of the signature a meaning in terms of those objects. Descriptions are built from variables, the sets of symbols of the signature, and a small set of reserved symbols. A theory is a set of descriptions. Interpretations induce a description denotation function and a theory denotation function. The description denotation function assigns each description the set of objects of which it is true, and the theory denotation function assigns each theory the set of objects of which each description in the theory is true. A grammar is a signature together with a theory. The meaning of a grammar is defined as a class of interpretations which meet certain conditions. Informally, a grammar denotes interpretations of its signature such that its theory is true of every object in the interpretation, and the interpretation contains instances of all conceivable configurations of objects of which the descriptions in the theory are true.

The syntactic entities of each formal language are given by the set of variables and by the signature:

DEFINITION 2.1. Var *is a countably infinite set of symbols.*

I call each element of Var a variable.

DEFINITION 2.2. Σ *is a signature iff*

Σ *is a septuple* $\langle \mathcal{G}, \sqsubseteq, \mathcal{S}, \mathcal{A}, \mathcal{F}, \mathcal{R}, \mathcal{AR} \rangle$,
$\langle \mathcal{G}, \sqsubseteq \rangle$ *is a finite partial order,*
$$\mathcal{S} = \left\{ \sigma \in \mathcal{G} \;\middle|\; \begin{array}{l} \textit{for each } \sigma' \in \mathcal{G}, \\ \textit{if } \sigma' \sqsubseteq \sigma \textit{ then } \sigma = \sigma' \end{array} \right\},$$
$\mathcal{A}$ *is a set,*
$\mathcal{F}$ *is a partial function from the Cartesian product of* $\mathcal{G}$ *and* $\mathcal{A}$ *to* $\mathcal{G}$,
for each $\sigma_1 \in \mathcal{G}$, *for each* $\sigma_2 \in \mathcal{G}$ *and for each* $\alpha \in \mathcal{A}$,
if $\mathcal{F}\langle \sigma_1, \alpha \rangle$ *is defined and* $\sigma_2 \sqsubseteq \sigma_1$
then $\mathcal{F}\langle \sigma_2, \alpha \rangle$ *is defined and* $\mathcal{F}\langle \sigma_2, \alpha \rangle \sqsubseteq \mathcal{F}\langle \sigma_1, \alpha \rangle$,
$\mathcal{R}$ *is a finite set, and*
$\mathcal{AR}$ *is a total function from* $\mathcal{R}$ *to* $\mathbb{N}^+$.

I call each element of $\mathcal{G}$ a sort, $\langle \mathcal{G}, \sqsubseteq \rangle$ the sort hierarchy, each element of $\mathcal{S}$ a maximally specific sort or a species, each element of $\mathcal{A}$ an attribute, $\mathcal{F}$ the appropriateness function, each element of $\mathcal{R}$ a relation symbol, and $\mathcal{AR}$ the arity function. I say that sort σ_2 is at least as specific as sort σ_1 iff $\sigma_2 \sqsubseteq \sigma_1$; conversely, I call σ_1 a supersort of σ_2. The symbols $\forall$, $\exists$, $:$, $\sim$, $\approx$, $\neg$, [,], $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$, $\dagger$, $\triangleright$, *chain, echain, nechain* and *metatop* are reserved symbols, and I assume that none of them are a variable, a sort, an attribute or a relation symbol. The appropriateness function, $\mathcal{F}$, respects the usual postulates about attribute inheritance in sort hierarchies. If an attribute, α, is appropriate to a sort σ_1 then it is appropriate to any sort σ_2 which is more specific than σ_1, and $\mathcal{F}\langle \sigma_2, \alpha \rangle$ is at least as specific as $\mathcal{F}\langle \sigma_1, \alpha \rangle$. The arity function, $\mathcal{AR}$, specifies the arity of each relation symbol as a positive integer.

Suppose M is a set. M^* is the set of finite sequences or tuples of elements of M. Throughout this paper I write $\overline{M}$ as an abbreviation for $M \cup M^*$. Consequently, $\overline{M}^*$ will stand for $(M \cup M^*)^*$. RSRL signatures are interpreted as follows:

DEFINITION 2.3. *For each signature* $\Sigma = \langle \mathcal{G}, \sqsubseteq, \mathcal{S}, \mathcal{A}, \mathcal{F}, \mathcal{R}, \mathcal{AR} \rangle$, *$I$ is a Σ interpretation iff*

> *I is a quadruple $\langle U, S, A, R \rangle$,*
> *U is a set,*
> *S is a total function from U to $\mathcal{S}$,*
> *A is a total function from $\mathcal{A}$ to the set of partial functions from U to U,*
> *for each $\alpha \in \mathcal{A}$ and each $u \in U$,*
>   *if $A(\alpha)(u)$ is defined*
>   *then $\mathcal{F}\langle S(u), \alpha \rangle$ is defined, and $S(A(\alpha)(u)) \sqsubseteq \mathcal{F}\langle S(u), \alpha \rangle$, and*
> *for each $\alpha \in \mathcal{A}$ and each $u \in U$,*
>   *if $\mathcal{F}\langle S(u), \alpha \rangle$ is defined then $A(\alpha)(u)$ is defined,*
> *R is a total function from $\mathcal{R}$ to the power set of $\overline{U}^*$, and*
>   *for each $\rho \in \mathcal{R}$, $R(\rho) \subseteq \overline{U}^{\mathcal{AR}(\rho)}$.*

U is the set of objects in the universe. I call S the species assignment function, A the attribute interpretation function, and R the relation interpretation function.

S partitions the objects of the universe by assigning a species to every object. Informally, one might say that each species denotes a set of objects, and each object is in the denotation of exactly one species. Extending this terminology to sorts in general, I say that each sort, σ, denotes the union of the sets of objects denoted by the species of which σ is a supersort. The function A provides a denotation for attributes. Each attribute denotes a partial function from objects to objects. The denotation of attributes in a Σ interpretation is governed by the appropriateness function, $\mathcal{F}$. An attribute, α, is defined on all or no objects in the denotation of a species, σ, depending on whether $\mathcal{F}\langle \sigma, \alpha \rangle$ is defined or not, and the value of the partial function denoted by α on an object of species σ is in the denotation of $\mathcal{F}\langle \sigma, \alpha \rangle$. R interprets each relation symbol as a set of tuples of the form $\langle u_1, \ldots, u_n \rangle$, where: (a) each u_i is an object from U, or (b) each u_i is a sequence of members of the universe, or (c) some u_i are objects and others are sequences of objects of the universe. n is determined by the arity of the relation as given by the arity function. To refer to sequences of objects or sequences of sequences of objects, I will use two terms, tuple and chain, which will indicate the perspective from which I am looking at sequences of entities. When talking about the denotation of a relation, ρ, I will call a sequence of entities, $\langle u_1, \ldots, u_n \rangle$, in the denotation of the relation, $R(\rho)$, a tuple. When talking about sequences of objects as being in the denotation of any other expression of the formal language or as an element, u_i, of a tuple, $\langle u_1, \ldots, u_n \rangle$, in the denotation of a relation, I will refer to them as chains. The main purpose of chains is to treat certain uses of lists in the HPSG literature where the lists occur in the arguments of relations without being a component of the objects that are described. Only the elements of these "virtual lists" are components of the described objects, but not the apparent lists themselves.[3]

For reasoning with chains, I need to supplement the sort hierarchy, $\langle \mathcal{G}, \sqsubseteq \rangle$, the set of species, $\mathcal{S}$, and the set of attributes, $\mathcal{A}$. With the quasi-attributes $\dagger$ and $\triangleright$, chains can be navigated, and the symbols *chain, echain* and *nechain* serve as quasi-sorts for the description of chains, empty chains and nonempty chains, and are ordered in an extension to the sort hierarchy in the intuitive way:

DEFINITION 2.4. $\langle \mathsf{Chain}, \sqsubseteq^c \rangle$ *is the smallest partial order such that*
$\mathsf{Chain} = \{chain, echain, nechain\}$,
$echain \sqsubseteq^c chain$, $nechain \sqsubseteq^c chain$.

The symbols *chain, echain* and *nechain* are the elements of a finite partial order, where *echain* and *nechain* are below *chain* with respect to the chain ordering relation, $\sqsubseteq^c$. I call *echain* and *nechain* maximally specific with respect to the chain ordering relation. In DEFINITION 2.5, signatures are augmented by the partial order just defined and three more symbols, $\dagger, \triangleright$, and *metatop*, which is the top of the expanded sort hierarchy:

DEFINITION 2.5. *For each signature* $\Sigma = \langle \mathcal{G}, \sqsubseteq, \mathcal{S}, \mathcal{A}, \mathcal{F}, \mathcal{R}, \mathcal{AR} \rangle$,
$\widehat{\mathcal{G}} = \mathcal{G} \cup \mathsf{Chain} \cup \{metatop\}$,
$\widehat{\sqsubseteq} = \sqsubseteq \cup \sqsubseteq^c \cup \left\{ \langle \sigma, metatop \rangle \,\middle|\, \sigma \in \widehat{\mathcal{G}} \right\}$,[4]
$\widehat{\mathcal{S}} = \mathcal{S} \cup \{echain, nechain\}$, *and*
$\widehat{\mathcal{A}} = \mathcal{A} \cup \{\dagger, \triangleright\}$.

[3]For an illustration, see the discussion of the `shuffle` relation at the end of section 4 below.

[4]Note that $\left\langle \widehat{\mathcal{G}}, \widehat{\sqsubseteq} \right\rangle$ is a finite partial order.

I call $\widehat{\mathcal{G}}$ the expanded sort set, $\left\langle \widehat{\mathcal{G}}, \widehat{\sqsubseteq} \right\rangle$ the expanded sort hierarchy, $\widehat{\sqsubseteq}$ the expanded sort hierarchy relation, $\widehat{\mathcal{S}}$ the expanded species set, and $\widehat{\mathcal{A}}$ the expanded attribute set. Besides the sorts of the signature, $\widehat{\mathcal{G}}$ contains the chain symbols and *metatop*, which, according to the definition of $\widehat{\sqsubseteq}$, is the top element of the expanded sort hierarchy. Moreover, the expanded sort hierarchy relation properly contains the chain ordering relation and the ordering relation of the sort hierarchy. The expanded species set, $\widehat{\mathcal{S}}$, includes the two maximally specific elements in the chain ordering relation, *echain* and *nechain*. For each σ_1 and for each σ_2 in the expanded sort hierarchy, I say that σ_1 subsumes σ_2 iff $\sigma_2 \mathrel{\widehat{\sqsubseteq}} \sigma_1$. Finally, the expanded attribute set, $\widehat{\mathcal{A}}$, contains two reserved symbols, † and ▷, besides the elements of the set of attributes, $\mathcal{A}$.

Parallel to the extension of the signature, the species assignment, S, and the attribute interpretation function, A, must be extended to give the new symbols *echain*, *nechain*, †, and ▷ a denotation in an interpretation:

Definition 2.6. *For each signature* $\Sigma = \langle \mathcal{G}, \sqsubseteq, \mathcal{S}, \mathcal{A}, \mathcal{F}, \mathcal{R}, \mathcal{AR} \rangle$, *for each* Σ *interpretation* $I = \langle U, S, A, R \rangle$,

$\widehat{S}$ *is the total function from* $\overline{U}$ *to* $\widehat{\mathcal{S}}$ *such that*

for each $u \in U$, $\widehat{S}(u) = S(u)$,

for each $u_1 \in U$, ..., *for each* $u_n \in U$,

$$\widehat{S}(\langle u_1, \ldots, u_n \rangle) = \begin{cases} \textit{echain} & \textit{if } n = 0, \\ \textit{nechain} & \textit{if } n > 0 \end{cases}, \textit{ and}$$

$\widehat{A}$ *is the partial function from* $\widehat{\mathcal{A}}$ *to the set of partial functions from* $\overline{U}$ *to* $\overline{U}$ *such that*

for each $\alpha \in \mathcal{A}$, $\widehat{A}(\alpha) = A(\alpha)$,

$\widehat{A}(\dagger)$ *is the total function from* U^+ *to* U *such that for each* $\langle u_0, \ldots, u_n \rangle \in U^+$,

$$\widehat{A}(\dagger)(\langle u_0, \ldots, u_n \rangle) = u_0, \textit{ and}$$

$\widehat{A}(\triangleright)$ *is the total function from* U^+ *to* U^* *such that for each* $\langle u_0, \ldots, u_n \rangle \in U^+$,

$$\widehat{A}(\triangleright)(\langle u_0, \ldots, u_n \rangle) = \langle u_1, \ldots, u_n \rangle.$$

I call $\widehat{S}$ the expanded species assignment function, and $\widehat{A}$ the expanded attribute interpretation function. For all elements of U in its domain, $\widehat{S}$ equals the species assignment function, S. In addition, the empty chain is assigned *echain*, and each nonempty finite chain is assigned *nechain*. Informally, I say that *echain* denotes the singleton set containing the empty chain, and *nechain* denotes the set of nonempty chains. Generalizing the terminology introduced for sorts to the expanded sort set, I say that each symbol, σ, in the expanded sort set denotes the union of the sets of objects or chains denoted by the maximally specific symbols which σ subsumes in the expanded sort hierarchy relation. $\widehat{A}$ equals the attribute interpretation function, A, for all attributes in its domain; but its domain also includes $\dagger$ and $\triangleright$, and its range includes partial functions whose domains are chains. Under $\widehat{A}$, $\dagger$ denotes the function from each nonempty chain to its first element, and $\triangleright$ denotes the function from each nonempty chain, c, to the chain which is exactly like c except that the first element of c is missing.

Since the formal language of RSRL contains variables, variable assignments will be needed to define denotations for its expressions.

DEFINITION 2.7. *For each signature Σ, for each Σ interpretation $I = \langle U, S, A, R\rangle$,*

$Ass_I = \overline{U}^{\mathsf{Var}}$ *is the set of variable assignments in I.*

I call each element of Ass_I a variable assignment in I. Elements of Ass_I are functions which assign objects or chains of objects of the universe to variables.

We are now ready to define the syntax and semantics of the formal languages of RSRL. First, terms and their denotations are defined. Second, the terms are used as basic syntactic expressions for constructing complex formulae and their denotation.

DEFINITION 2.8. *For each signature $\Sigma = \langle \mathcal{G}, \sqsubseteq, \mathcal{S}, \mathcal{A}, \mathcal{F}, \mathcal{R}, \mathcal{AR}\rangle$, $\mathcal{T}^{\Sigma}$ is the smallest set such that*

$: \; \in \mathcal{T}^{\Sigma}$,
for each $v \in \mathsf{Var}$, $v \in \mathcal{T}^{\Sigma}$, and
for each $\alpha \in \widehat{\mathcal{A}}$ and each $\tau \in \mathcal{T}^{\Sigma}$, $\tau\alpha \in \mathcal{T}^{\Sigma}$.

I call each element of $\mathcal{T}^{\Sigma}$ a Σ term. A Σ term consists of either the reserved symbol $:$ or a variable followed by a (possibly empty) string of symbols of the expanded attribute set. Σ terms are interpreted as follows:

DEFINITION 2.9. *For each signature $\Sigma = \langle \mathcal{G}, \sqsubseteq, \mathcal{S}, \mathcal{A}, \mathcal{F}, \mathcal{R}, \mathcal{AR}\rangle$, for each Σ interpretation $I = \langle U, S, A, R\rangle$, for each* $\mathrm{ass} \in Ass_I$*, T_I^{ass} is the total function from $\mathcal{T}^{\Sigma}$ to the set of partial functions from U to $\overline{U}$ such that for each $u \in U$,*

$T_I^{\mathrm{ass}}(:)(u)$ is defined and $T_I^{\mathrm{ass}}(:)(u) = u$,
for each $v \in \mathsf{Var}$, $T_I^{\mathrm{ass}}(v)(u)$ is defined and $T_I^{\mathrm{ass}}(v)(u) = \mathrm{ass}(v)$,

for each $\tau \in \mathcal{T}^{\Sigma}$, *for each* $\alpha \in \widehat{\mathcal{A}}$,
$T_I^{\text{ass}}(\tau\alpha)(u)$ *is defined*
iff $T_I^{\text{ass}}(\tau)(u)$ *is defined and* $\widehat{A}(\alpha)(T_I^{\text{ass}}(\tau)(u))$ *is defined, and*
if $T_I^{\text{ass}}(\tau\alpha)(u)$ *is defined*
then $T_I^{\text{ass}}(\tau\alpha)(u) = \widehat{A}(\alpha)(T_I^{\text{ass}}(\tau)(u))$.

I call T_I^{ass} the Σ term interpretation function with respect to I under a variable assignment in I, ass. Each Σ term is interpreted as a partial function from U to $\overline{U}$. The colon denotes the identity function in U. A variable denotes whatever object or chain of objects the variable assignment in I assigns to that variable. Finally, the meaning of complex Σ terms results from functional composition of the denotation of each of the symbols of the expanded attribute set succeeding the colon or variable in reverse order of their appearance in the Σ term, and the meaning of the colon or variable. Notice that, in case τ begins with a variable, the value of $T_I^{\text{ass}}(\tau)(u)$, and the question of whether it is defined, is independent of u.

Note that only Σ terms which begin with a variable which is followed by a (possibly empty) string of the reserved symbol $\triangleright$ may map elements of U to chains of objects of U. All other terms denote functions which, if they are defined on any object of the universe at all, may only map them to objects of U. I say that a Σ term, τ, is *defined on an object*, u, iff τ begins with the colon and $T_I^{\text{ass}}(\tau)(u)$ is defined.

The syntax of formulae is based on terms:

DEFINITION 2.10. *For each signature* $\Sigma = \langle \mathcal{G}, \sqsubseteq, \mathcal{S}, \mathcal{A}, \mathcal{F}, \mathcal{R}, \mathcal{AR} \rangle$, $\mathcal{D}^{\Sigma}$ *is the smallest set such that*
for each $\sigma \in \widehat{\mathcal{G}}$, *for each* $\tau \in \mathcal{T}^{\Sigma}$, $\tau \sim \sigma \in \mathcal{D}^{\Sigma}$,
for each $\tau_1 \in \mathcal{T}^{\Sigma}$, *for each* $\tau_2 \in \mathcal{T}^{\Sigma}$, $\tau_1 \approx \tau_2 \in \mathcal{D}^{\Sigma}$,
for each $\rho \in \mathcal{R}$, *for each* $x_1 \in \mathsf{Var}$, ... , *for each* $x_{\mathcal{AR}(\rho)} \in \mathsf{Var}$,
$\rho(x_1, \ldots, x_{\mathcal{AR}(\rho)}) \in \mathcal{D}^{\Sigma}$,
for each $x \in \mathsf{Var}$, *for each* $\delta \in \mathcal{D}^{\Sigma}$, $\exists x\, \delta \in \mathcal{D}^{\Sigma}$,
for each $x \in \mathsf{Var}$, *for each* $\delta \in \mathcal{D}^{\Sigma}$, $\forall x\, \delta \in \mathcal{D}^{\Sigma}$,
for each $\delta \in \mathcal{D}^{\Sigma}$, $\neg\delta \in \mathcal{D}^{\Sigma}$,
for each $\delta_1 \in \mathcal{D}^{\Sigma}$, *for each* $\delta_2 \in \mathcal{D}^{\Sigma}$, $[\delta_1 \wedge \delta_2] \in \mathcal{D}^{\Sigma}$,
for each $\delta_1 \in \mathcal{D}^{\Sigma}$, *for each* $\delta_2 \in \mathcal{D}^{\Sigma}$, $[\delta_1 \vee \delta_2] \in \mathcal{D}^{\Sigma}$,
for each $\delta_1 \in \mathcal{D}^{\Sigma}$, *for each* $\delta_2 \in \mathcal{D}^{\Sigma}$, $[\delta_1 \rightarrow \delta_2] \in \mathcal{D}^{\Sigma}$, *and*
for each $\delta_1 \in \mathcal{D}^{\Sigma}$, *for each* $\delta_2 \in \mathcal{D}^{\Sigma}$, $[\delta_1 \leftrightarrow \delta_2] \in \mathcal{D}^{\Sigma}$.

I call each element of $\mathcal{D}^{\Sigma}$ a Σ formula. Σ formulae of the form $\tau \sim \sigma$ are called sort assignments, Σ formulae of the form $\tau_1 \approx \tau_2$ are called path equations, and Σ formulae of the form $\rho(x_1, \ldots, x_{\mathcal{AR}(\rho)})$ are called relational Σ formulae. They constitute the atomic Σ formulae. Complex Σ formulae can be formed from atomic Σ formulae by means of existential and universal quantification and the standard logical connectives.

For each Σ term and formula, I define the set of free occurrences of variables in that Σ term or formula:

DEFINITION 2.11. *For each signature* $\Sigma = \langle \mathcal{G}, \sqsubseteq, \mathcal{S}, \mathcal{A}, \mathcal{F}, \mathcal{R}, \mathcal{AR} \rangle$,

$FV(:) = \emptyset$,

for each $v \in \mathsf{Var}$, $FV(v) = \{v\}$,

for each $\tau \in \mathcal{T}^{\Sigma}$, *for each* $\alpha \in \widehat{\mathcal{A}}$, $FV(\tau\alpha) = FV(\tau)$,

for each $\tau \in \mathcal{T}^{\Sigma}$, *for each* $\sigma \in \widehat{\mathcal{G}}$, $FV(\tau \sim \sigma) = FV(\tau)$,

for each $\tau_1, \tau_2 \in \mathcal{T}^{\Sigma}$, $FV(\tau_1 \approx \tau_2) = FV(\tau_1) \cup FV(\tau_2)$,

for each $\rho \in \mathcal{R}$, *for each* $x_1, \ldots, x_{\mathcal{AR}(\rho)} \in \mathsf{Var}$,
$FV(\rho(x_1, \ldots, x_{\mathcal{AR}(\rho)})) = \{x_1, \ldots, x_{\mathcal{AR}(\rho)}\}$,

for each $\delta \in \mathcal{D}^{\Sigma}$, *for each* $v \in \mathsf{Var}$, $FV(\exists v\, \delta) = FV(\delta)\backslash\{v\}$,

for each $\delta \in \mathcal{D}^{\Sigma}$, *for each* $v \in \mathsf{Var}$, $FV(\forall v\, \delta) = FV(\delta)\backslash\{v\}$,

for each $\delta \in \mathcal{D}^{\Sigma}$, $FV(\neg\delta) = FV(\delta)$,

for each $\delta_1 \in \mathcal{D}^{\Sigma}$, *for each* $\delta_2 \in \mathcal{D}^{\Sigma}$,
$FV([\delta_1 \wedge \delta_2]) = FV(\delta_1) \cup FV(\delta_2)$,

for each $\delta_1 \in \mathcal{D}^{\Sigma}$, *for each* $\delta_2 \in \mathcal{D}^{\Sigma}$,
$FV([\delta_1 \vee \delta_2]) = FV(\delta_1) \cup FV(\delta_2)$,

for each $\delta_1 \in \mathcal{D}^{\Sigma}$, *for each* $\delta_2 \in \mathcal{D}^{\Sigma}$,
$FV([\delta_1 \rightarrow \delta_2]) = FV(\delta_1) \cup FV(\delta_2)$, *and*

for each $\delta_1 \in \mathcal{D}^{\Sigma}$, *for each* $\delta_2 \in \mathcal{D}^{\Sigma}$,
$FV([\delta_1 \leftrightarrow \delta_2]) = FV(\delta_1) \cup FV(\delta_2)$.

Crucially, the definition of free occurrences of variables distinguishes between Σ terms that begin with the colon, which do not contain free variables, and Σ terms that begin with a variable, which is then free in that Σ term.

As motivated at length in Richter, Sailer, and Penn (1999), the semantics of existential and universal quantification in RSRL is restricted to quantification over components. Sets of components are defined relative to a given object in U in a Σ interpretation, I:

DEFINITION 2.12. *For each signature* $\Sigma = \langle \mathcal{G}, \sqsubseteq, \mathcal{S}, \mathcal{A}, \mathcal{F}, \mathcal{R}, \mathcal{AR} \rangle$, *for each* Σ *interpretation* $I = \langle U, S, A, R \rangle$, *and for each* $u \in U$,

$$Co^u_I = \left\{ u' \in U \;\middle|\; \begin{array}{l} \textit{for some } \mathrm{ass} \in Ass_I, \\ \textit{for some } \pi \in \mathcal{A}^*, \\ \quad T^{\mathrm{ass}}_I(:\pi)(u) \textit{ is defined, and} \\ \quad u' = T^{\mathrm{ass}}_I(:\pi)(u) \end{array} \right\}.$$

I call Co^u_I the set of components of u in I. Informally, DEFINITION 2.12 limits the set of components of an object u in I to those objects in U that are accessible from u by Σ terms defined on u. Note that the denotation of $:\pi$ is independent of the variable assignment in I, since there is no variable in Σ terms of that form.

As a final step towards the definition of the meaning of Σ formulae, I introduce a simple notational convention which I will need to fix the meaning of quantifiers:

DEFINITION 2.13. *For each signature Σ, for each Σ interpretation $I = \langle U, S, A, R\rangle$, for each* $\mathrm{ass} \in Ass_I$*, for each* $v \in \mathsf{Var}$*, for each* $w \in \mathsf{Var}$*, for each* $u \in \overline{U}$*,*

$$\mathrm{ass}\tfrac{u}{v}(w) = \begin{cases} u & \textit{if } v = w \\ \mathrm{ass}(w) & \textit{otherwise.} \end{cases}$$

For each variable assignment in I, ass, $\mathrm{ass}\frac{u}{v}$ is just like the function ass, except that it assigns object u to variable v.

Everything is now in place to define the denotation function for Σ formulae, D^{ass}_I:

DEFINITION 2.14. *For each signature $\Sigma = \langle \mathcal{G}, \sqsubseteq, \mathcal{S}, \mathcal{A}, \mathcal{F}, \mathcal{R}, \mathcal{AR}\rangle$, for each Σ interpretation $I = \langle U, S, A, R\rangle$, for each* $\mathrm{ass} \in Ass_I$*, D^{ass}_I is the total function from $\mathcal{D}^\Sigma$ to the power set of U such that*

for each $\tau \in \mathcal{T}^\Sigma$, for each $\sigma \in \widehat{\mathcal{G}}$,

$$D^{\mathrm{ass}}_I(\tau \sim \sigma) = \left\{ u \in U \;\middle|\; \begin{array}{l} T^{\mathrm{ass}}_I(\tau)(u) \textit{ is defined, and} \\ \widehat{S}(T^{\mathrm{ass}}_I(\tau)(u)) \mathrel{\widehat{\sqsubseteq}} \sigma \end{array} \right\},$$

for each $\tau_1 \in \mathcal{T}^\Sigma$, for each $\tau_2 \in \mathcal{T}^\Sigma$,

$$D^{\mathrm{ass}}_I(\tau_1 \approx \tau_2) = \left\{ u \in U \;\middle|\; \begin{array}{l} T^{\mathrm{ass}}_I(\tau_1)(u) \textit{ is defined,} \\ T^{\mathrm{ass}}_I(\tau_2)(u) \textit{ is defined, and} \\ T^{\mathrm{ass}}_I(\tau_1)(u) = T^{\mathrm{ass}}_I(\tau_2)(u) \end{array} \right\},$$

for each $\rho \in \mathcal{R}$, for each $x_1 \in \mathsf{Var}$, ..., for each $x_{\mathcal{AR}(\rho)} \in \mathsf{Var}$,

$$D^{\mathrm{ass}}_I(\rho(x_1, \ldots, x_{\mathcal{AR}(\rho)})) = \left\{ u \in U \;\middle|\; \left\langle \mathrm{ass}(x_1), \ldots, \mathrm{ass}(x_{\mathcal{AR}(\rho)})\right\rangle \in R(\rho) \right\},$$

for each $v \in \mathsf{Var}$, for each $\delta \in \mathcal{D}^\Sigma$,

$$D^{\mathrm{ass}}_I(\exists v\, \delta) = \left\{ u \in U \;\middle|\; \begin{array}{l} \textit{for some } u' \in \overline{Co^u_I}, \\ u \in D^{\mathrm{ass}\frac{u'}{v}}_I(\delta) \end{array} \right\},$$

for each $v \in \mathsf{Var}$, *for each* $\delta \in \mathcal{D}^{\Sigma}$,

$$D_I^{\mathrm{ass}}(\forall v\, \delta) = \left\{ u \in U \;\middle|\; \begin{array}{l} \textit{for each } u' \in \overline{Co_I^u}, \\ u \in D_I^{\mathrm{ass}\frac{u'}{v}}(\delta) \end{array} \right\},$$

for each $\delta \in \mathcal{D}^{\Sigma}$,

$$D_I^{\mathrm{ass}}(\neg\delta) = U \backslash D_I^{\mathrm{ass}}(\delta),$$

for each $\delta_1 \in \mathcal{D}^{\Sigma}$, *for each* $\delta_2 \in \mathcal{D}^{\Sigma}$,

$$D_I^{\mathrm{ass}}([\delta_1 \wedge \delta_2]) = D_I^{\mathrm{ass}}(\delta_1) \cap D_I^{\mathrm{ass}}(\delta_2),$$

for each $\delta_1 \in \mathcal{D}^{\Sigma}$, *for each* $\delta_2 \in \mathcal{D}^{\Sigma}$,

$$D_I^{\mathrm{ass}}([\delta_1 \vee \delta_2]) = D_I^{\mathrm{ass}}(\delta_1) \cup D_I^{\mathrm{ass}}(\delta_2),$$

for each $\delta_1 \in \mathcal{D}^{\Sigma}$, *for each* $\delta_2 \in \mathcal{D}^{\Sigma}$,

$$D_I^{\mathrm{ass}}([\delta_1 \rightarrow \delta_2]) = (U \backslash D_I^{\mathrm{ass}}(\delta)) \cup D_I^{\mathrm{ass}}(\delta_2), \textit{ and}$$

for each $\delta_1 \in \mathcal{D}^{\Sigma}$, *for each* $\delta_2 \in \mathcal{D}^{\Sigma}$,

$$\begin{aligned} &D_I^{\mathrm{ass}}([\delta_1 \leftrightarrow \delta_2]) \\ &= ((U \backslash D_I^{\mathrm{ass}}(\delta_1)) \cap (U \backslash D_I^{\mathrm{ass}}(\delta_2))) \cup (D_I^{\mathrm{ass}}(\delta_1) \cap D_I^{\mathrm{ass}}(\delta_2)). \end{aligned}$$

I call D_I^{ass} the Σ formula interpretation function with respect to I under a variable assignment in I, ass. When looking at the denotation of Σ formulae, it is important to remember that in RSRL, just like in SRL, expressions denote a subset of the objects of the universe rather than truth values. This formalizes the intuition underlying both formalisms that their expressions describe objects and are, thus, true *of objects in an interpretation.*

To understand the denotation of sort assignments and path equations, it is intuitively useful to distinguish cases when their Σ terms start with the colon from cases when they start with variables. Sort assignments, $\tau \sim \sigma$, in which τ starts with the colon denote the set of objects on which the Σ term τ is defined and leads to an object which is in the denotation of σ. Path equations, $\tau_1 \approx \tau_2$, in which τ_1 and τ_2 start with the colon denote the set of objects on which τ_1 and τ_2 are defined and lead to the same object.

The denotation of sort assignments and path equations whose Σ terms start with a variable crucially depends on the chosen variable assignment in I. They either denote the entire universe or the empty set. Because the value of $T_I^{\mathrm{ass}}(\tau)(u)$, and the question of whether it is defined, is independent of u if τ starts with a variable, $\tau \sim \sigma$ denotes the entire universe if the function denoted by the string of symbols of the expanded attribute set following the variable is defined on the object or chain denoted by the variable and leads to an object or chain in the denotation of the symbol

σ; and the empty set otherwise. $\tau_1 \approx \tau_2$ denotes the entire universe if the functions denoted by the string of symbols of the expanded attribute set following the two variables are defined on the object or chain the respective variables denote and lead to the same object or chain; and the empty set otherwise.

An n-ary relational Σ formula, $\rho(x_1, \dots, x_n)$, is interpreted as the set of objects such that the n-tuple of variables describing its arguments denotes an n-tuple of objects and chains which is in the relation interpretation of ρ. Similar to sort assignments and path equations with Σ terms beginning with variables, a relational Σ formula either denotes the entire universe or the empty set, depending on the variable assignment in I, ass. It denotes U if the tuple of variables describing its arguments denotes an n-ary tuple of objects or chains (or an n-ary tuple of objects and chains) which is in the relation interpretation; and the empty set otherwise.

Below we will restrict ourselves to Σ formulae without free variables. In that case, the denotation of relational Σ formulae, sort assignments and path equations involving variables becomes more interesting. Before I turn to the properties of the denotation of Σ formulae without free variables, notice that conjunction, disjunction, implication and equivalence are interpreted classically; and negation, of course, corresponds to set complement. Notice also the crucial appeal to the set comprising components and chains of components of u in I, $\overline{Co_I^u}$, in the interpretation of existential and universal quantification.

In a complex Σ formula without free variables, an n-ary relational Σ formula denotes the set of objects whose n-tuples of components or chains of components, as determined by the quantification, are in the denotation of the relation. If all variables in a relational Σ formula are existentially bound, it denotes the set of objects, u, which have an n-tuple of components or chains of components in I that are in the denotation of the relation; if they are all universally bound, it denotes the set of objects, u, whose components and chains of components in I are all in the denotation of the relation. Similarly, if the variable in the Σ term, τ, of a sort assignment, $\tau \sim \sigma$ is existentially bound, it denotes the set of objects which have at least one component or chain of components, c, such that the function denoted by the string of symbols from the expanded attribute set following the variable in τ is defined on c and yields an object or chain in the denotation of σ. If the variables in the Σ terms τ_1 and τ_2 of a path equation, $\tau_1 \approx \tau_2$, are existentially bound, that path equation denotes the set of objects such that there is at least one component or chain of components, c_1, on which the function denoted by the string of symbols from the expanded attribute set following the variable in τ_1 is defined, there is at least one component or chain of components, c_2, on which the function denoted by the string of symbols from the expanded attribute set following the variable in τ_2 is defined, and the two functions have the same object or chain as value on c_1 and c_2, respectively. Corresponding observations hold for universal quantification.

PROPOSITION 2.15, which can be verified by a simple inductive proof, states that a Σ formula, δ, denotes the same set of objects under two different variable assignments in I if the two variable assignments agree on the free variables in δ. This observation will ultimately allow us to abstract away from variable assignments when I move from arbitrary Σ formulae to Σ formulae without free variables.

PROPOSITION 2.15. *For each signature* Σ, *for each* Σ *interpretation* I, *for each* $\mathrm{ass}_1 \in Ass_I$, *for each* $\mathrm{ass}_2 \in Ass_I$,

for each $\tau \in \mathcal{T}^\Sigma$,

if for each $v \in FV(\tau)$, $\mathrm{ass}_1(v) = \mathrm{ass}_2(v)$
then $T_I^{\mathrm{ass}_1}(\tau) = T_I^{\mathrm{ass}_2}(\tau)$, *and*

for each $\delta \in \mathcal{D}^\Sigma$,

if for each $v \in FV(\delta)$, $\mathrm{ass}_1(v) = \mathrm{ass}_2(v)$
then $D_I^{\mathrm{ass}_1}(\delta) = D_I^{\mathrm{ass}_2}(\delta)$.

If two variable assignments in I agree on the variables that occur free in a given Σ term, τ, then τ denotes the same partial function under the Σ term interpretation function with respect to I under both variable assignments in I. Similarly, if two variable assignments in I agree on the variables that occur free in a given Σ formula, δ, then δ denotes the same set of objects of the universe under the Σ formula interpretation function with respect to I under both variable assignments in I.

I distinguish the set of Σ formulae without free variables by a special symbol, $\mathcal{D}_0^\Sigma$:

DEFINITION 2.16. *For each signature* Σ,

$$\mathcal{D}_0^\Sigma = \{\delta \in \mathcal{D}^\Sigma \mid FV(\delta) = \emptyset\}.$$

I call each element of $\mathcal{D}_0^\Sigma$ a Σ description, and each subset of $\mathcal{D}_0^\Sigma$ a Σ theory. When working with a fixed signature, I will henceforth refer to terms, formulae and descriptions. As a corollary of PROPOSITION 2.15, we get the following result for the meaning of Σ descriptions:

COROLLARY 2.17. *For each signature* Σ, *for each* $\delta \in \mathcal{D}_0^\Sigma$, *for each* Σ *interpretation* I, *for each* $\mathrm{ass}_1 \in Ass_I$, *for each* $\mathrm{ass}_2 \in Ass_I$,

$D_I^{\mathrm{ass}_1}(\delta) = D_I^{\mathrm{ass}_2}(\delta)$.

According to COROLLARY 2.17 the meaning of Σ descriptions is independent from variable assignments in I.

DEFINITION 2.18. *For each signature Σ, for each Σ interpretation $I = \langle U, S, A, R\rangle$, D_I is the total function from $\mathcal{D}_0^\Sigma$ to the power set of U such that for each $\delta \in \mathcal{D}_0^\Sigma$,*

$$D_I(\delta) = \left\{ u \in U \,\middle|\, \begin{array}{l} \textit{for each } \mathrm{ass} \in Ass_I, \\ u \in D_I^{\mathrm{ass}}(\delta) \end{array} \right\}.$$

I call D_I the Σ description denotation function with respect to I. D_I abstracts away from particular variable assignments in I. With D_I in hand it is now straightforward to define the denotation of Σ theories:

DEFINITION 2.19. *For each signature Σ, for each Σ interpretation $I = \langle U, S, A, R\rangle$, Θ_I is the total function from the power set of $\mathcal{D}_0^\Sigma$ to the power set of U such that for each $\theta \subseteq \mathcal{D}_0^\Sigma$,*

$$\Theta_I(\theta) = \left\{ u \in U \,\middle|\, \begin{array}{l} \textit{for each } \delta \in \theta, \\ u \in D_I(\delta) \end{array} \right\}.$$

I call Θ_I the theory denotation function with respect to I. Under Θ_I a Σ theory, θ, denotes the set of objects of the universe which are in the denotation of all Σ descriptions in θ.

DEFINITION 2.20. *Γ is a grammar iff*

Γ is a pair $\langle \Sigma, \theta \rangle$,
Σ is a signature, and
$\theta \subseteq \mathcal{D}_0^\Sigma$.

A grammar is a signature, Σ, together with a set of Σ descriptions, θ. Clearly, not every Σ interpretation is an interesting interpretation of a grammar, $\langle \Sigma, \theta \rangle$. For example, the ungrammatical sentence *Sandy walks Kim Fido* could very well be in an interpretation of the grammar of Pollard and Sag (1994), but that interpretation does certainly not reflect the predictions the authors make about English. The notion of a model of a grammar is a first approximation to the interpretations in which linguists are interested when they write grammars:

DEFINITION 2.21. *For each grammar $\Gamma = \langle \Sigma, \theta \rangle$, for each Σ interpretation $I = \langle U, S, A, R\rangle$,*

I is a Γ model iff $\Theta_I(\theta) = U$.

In a Γ model, each description in the theory is true of each object in the universe of the interpretation. Informally, the grammar denotes the whole universe of the interpretation. The notion of a model formalizes the idea that with a grammar linguists want to characterize natural language as objects which, in all their components, obey every principle of the grammar. However, as King (1995) was the first to observe, the concept of a model is not precise enough to characterize natural languages adequately. In a sense, an arbitrary model of a grammar might not be big enough, since it might not contain an instance of an object which could, in principle, be licensed by the grammar. In other words, there are other models of the same grammar which contain objects of that kind. A trivial example for an uninteresting model is the model where U is the empty set. A less trivial example of a model that is too small is a model of Pollard and Sag's (1994) grammar which contains exactly the sentence *Kim likes bagels* and nothing else.

The notion of an exhaustive model delimits the kind of model which is of the right form:

DEFINITION 2.22. *For each grammar* $\Gamma = \langle \Sigma, \theta \rangle$, *for each* Σ *interpretation* I,

I is an exhaustive Γ model iff
I is a Γ model, and
for each $\theta' \subseteq \mathcal{D}_0^{\Sigma}$, for each Σ interpretation I',
if I' is a Γ model and $\Theta_{I'}(\theta') \neq \emptyset$ then $\Theta_I(\theta') \neq \emptyset$.

An exhaustive Γ model, I, is a Γ model such that any theory which has a nonempty denotation in any other Γ model also has a nonempty denotation in I. Informally, an exhaustive Γ model contains instances of all objects that can be licensed by Γ. Among those objects are, of course, all sentences that are licensed by Γ. In linguistic terms this means that a grammar is true of a natural language if the natural language is an exhaustive model of the grammar. For a more thorough discussion of exhaustive models and alternative mathematical characterizations in SRL, see King (1999) and Pollard (1999).[5]

A very long and technical proof by Richter and King (1997) shows the following central theorem about exhaustive models of RSRL grammars:

THEOREM 2.23. *For each signature Σ, for each Σ theory θ, there exists a Σ interpretation I such that I is an exhaustive $\langle \Sigma, \theta \rangle$ model.*

As a corollary of THEOREM 2.23, we get immediately that if a grammar has a nonempty model, it also has a nonempty exhaustive model. In other

[5]The results obtained for SRL carry over to RSRL.

words, a non-vacuous grammar has a linguistically sensible meaning in the class of its exhaustive models.

While RSRL is apt to formalize current HPSG grammars rigorously[6], it leaves its potential users with a syntax of descriptions that is rather arcane. All that linguists are really interested in for writing grammars are descriptions and, perhaps, formulae. A further distinction between terms and formulae is superfluous for their purposes, and potentially confusing. In addition, the syntax of descriptions above is entirely linear in contrast to the conventional, two-dimensional AVM diagrams. A linear notation necessitates multiple repetitions of attributes where a two-dimensional matrix notation only needs to mention them once, and the linear notation is, thus, much harder to read. But for the lack of a definition of AVM diagrams as a formal language with an interpretation in RSRL, AVM diagrams can so far only be understood as sketches of the theory of the intended grammar, and it takes a logician and a discussion with the linguist about the intended meaning of his sketches to translate them into meaningful expressions of RSRL. The goal of section 3 is to close the mathematical gap between AVM diagrams and their empirical predictions.

3. An AVM syntax for RSRL

In this section, I am going to define an AVM language for RSRL grammars. Then I show that each theory of a grammar that can be written with descriptions can also be written with AVM descriptions. In linguistic terms, each natural language that can be characterized with descriptions can be characterized with AVM descriptions. The AVM language is simply a different notation for writing grammars in RSRL.

3.1. **Definition of the AVM syntax.** The basic building blocks of AVM diagrams are matrices, enclosed between square brackets, which allow a two-dimensional arrangement of attribute names and sort symbols. This way it is possible to avoid redundant repetitions of attribute names in formulae. Matrices can optionally be preceded by so-called *tags*, which are conventionally depicted as boxed integers, and are usually used to indicate that two (or more) paths share their value. Again, this notation avoids repetitions of attribute names, and it can combine describing path values with stating that they have identical values. For lists, there is a special graphical notation that enumerates the descriptions of the elements

[6]See Richter, Sailer, and Penn (1999) for discussion.

of a list as AVM diagrams between angled brackets. Since this notation is built into the graphics of AVM diagrams, we need to presuppose that the underlying signature provides at least the usual sorts, hierarchical relations, attributes and appropriateness conditions for lists: *nelist* and *elist* are species, they are the only immediate subsorts of sort *list*, no attributes are appropriate to *list* and *elist*, exactly FIRST and REST are appropriate to *nelist*, and $\mathcal{F}\langle nelist, \text{REST}\rangle = list$. I call a signature which comprises these definitions a list signature. From now on, whenever signatures are mentioned, list signatures are meant. Finally, AVM diagrams occasionally contain the negation symbol or the disjunction symbol with AVM diagrams as their argument(s). DEFINITION 3.1 below generalizes this practice to the standard logical connectives. Since RSRL includes chains, it is necessary to add an appropriate syntax of chains, which I derive from the syntax of lists, to the AVM syntax.

The AVM syntax will be defined in two steps. First, I define simple AVM diagrams, which I call boxes. Then I take certain boxes as basic units for potentially complex formulae which, among other constructs, might involve quantification and relations. For tags, I use the set of variables, Var. I write $\mathsf{Var}^{:}$ as an abbreviation for the set $\mathsf{Var} \cup \{:\}$.

DEFINITION 3.1. *For each signature* $\Sigma = \langle \mathcal{G}, \sqsubseteq, \mathcal{S}, \mathcal{A}, \mathcal{F}, \mathcal{R}, \mathcal{AR}\rangle$, BOX^Σ, UBOX^Σ *and* TBOX^Σ *are the smallest sets such that*

$\mathsf{TBOX}^\Sigma \cup \mathsf{UBOX}^\Sigma \subseteq \mathsf{BOX}^\Sigma$,

for each $\sigma \in \widehat{\mathcal{G}}$, *for each* $\alpha_1 \in \widehat{\mathcal{A}}$, ... , *for each* $\alpha_n \in \widehat{\mathcal{A}}$, *for each* $\beta_1 \in \mathsf{BOX}^\Sigma$, ... , *for each* $\beta_n \in \mathsf{BOX}^\Sigma$,

$$\begin{bmatrix} \sigma & \\ \alpha_1 & \beta_1 \\ \vdots & \\ \alpha_n & \beta_n \end{bmatrix} \in \mathsf{UBOX}^\Sigma,$$

for each $\beta_1 \in \mathsf{BOX}^\Sigma$, ... , *for each* $\beta_n \in \mathsf{BOX}^\Sigma$,

$\langle \beta_1, \ldots, \beta_n\rangle \in \mathsf{UBOX}^\Sigma$,

for each $\beta \in \mathsf{BOX}^\Sigma$, $\neg\beta \in \mathsf{UBOX}^\Sigma$,

for each $\beta_1 \in \mathsf{BOX}^\Sigma$, *for each* $\beta_2 \in \mathsf{BOX}^\Sigma$, $[\beta_1 \wedge \beta_2] \in \mathsf{UBOX}^\Sigma$,

for each $\beta_1 \in \mathsf{BOX}^\Sigma$, *for each* $\beta_2 \in \mathsf{BOX}^\Sigma$, $[\beta_1 \vee \beta_2] \in \mathsf{UBOX}^\Sigma$,

for each $\beta_1 \in \mathsf{BOX}^\Sigma$, *for each* $\beta_2 \in \mathsf{BOX}^\Sigma$, $[\beta_1 \rightarrow \beta_2] \in \mathsf{UBOX}^\Sigma$,

for each $\beta_1 \in \mathsf{BOX}^\Sigma$, *for each* $\beta_2 \in \mathsf{BOX}^\Sigma$, $[\beta_1 \leftrightarrow \beta_2] \in \mathsf{UBOX}^\Sigma$,

for each $v \in \mathsf{Var}$, *for each* $\beta_1 \in \mathsf{BOX}^\Sigma$, ... , *for each* $\beta_n \in \mathsf{BOX}^\Sigma$,

$v \, \|\beta_1, \ldots, \beta_n\| \in \mathsf{TBOX}^\Sigma$, *for each* $v \in \mathsf{Var}^{:}$, *for each* $\beta \in \mathsf{UBOX}^{\Sigma}$, $v\ \beta \in \mathsf{TBOX}^{\Sigma}$.

I call each element of TBOX^{Σ} a tagged Σ box, each element of UBOX^{Σ} an untagged Σ box, and each element of BOX^{Σ} a Σ box. Notice that $\mathsf{TBOX}^{\Sigma} \cap \mathsf{UBOX}^{\Sigma} = \emptyset$ and $\mathsf{TBOX}^{\Sigma} \cup \mathsf{UBOX}^{\Sigma} = \mathsf{BOX}^{\Sigma}$. Untagged Σ boxes of the form $[\ \sigma\ ]$ and tagged Σ boxes of the form $v[\ \sigma\ ]$, where v is a variable, I call atomic Σ matrices. A Σ matrix need not be atomic. There can be any finite number of symbols of the expanded attribute set in vertical order under the symbol, σ, of the expanded sort set, each of them followed by a Σ box. I call a Σ box of that form a(n) (untagged) Σ matrix. A Σ box of the form $\langle\beta_1, \ldots, \beta_n\rangle$, where a finite number of arbitrary Σ boxes is enumerated between angled brackets, I call a(n) (untagged) Σ list description. Finally, Σ boxes can be combined with the standard logical connectives in the usual way.

Every untagged Σ box can be turned into a tagged Σ box by writing a variable or the colon in front of it. The graphical notation for chains is similar to the notation for Σ list descriptions, except that in this case Σ boxes are enumerated between two vertical bars instead of angled brackets on each side. I call a complex tagged Σ box of the form $v\,\|\beta_1, \ldots, \beta_n\|$ a (tagged) Σ chain description. Note that Σ chain descriptions only occur as tagged Σ boxes in which the tag must be a variable, whereas all other types of tagged Σ boxes have untagged counterparts and may have the colon as a tag. This is a syntactic reflection of the fact that only formulae whose terms start with a variable can possibly denote chains. A chain notation with untagged Σ boxes would, thus, only supply expressions which would always denote the empty set.

DEFINITION 3.1 provides a syntax for single AVM diagrams. Crucially, it does not include quantification and relations. Linguistic notation usually does not make quantification explicit, and there is no standard notation of relations. The following definition provides a pragmatic solution that fits both linguistic practice and the need for mathematical brevity.

DEFINITION 3.2. *For each signature* $\Sigma = \langle \mathcal{G}, \sqsubseteq, \mathcal{S}, \mathcal{A}, \mathcal{F}, \mathcal{R}, \mathcal{AR}\rangle$, AVM^{Σ} *is the smallest set such that*

for each $\beta \in \mathsf{TBOX}^{\Sigma}$, $\beta \in \mathsf{AVM}^{\Sigma}$,
for each $\rho \in \mathcal{R}$, *for each* $v_1 \in \mathsf{Var}$, $\ldots$, *for each* $v_{\mathcal{AR}(\rho)} \in \mathsf{Var}$,
$\rho(v_1, \ldots, v_{\mathcal{AR}(\rho)}) \in \mathsf{AVM}^{\Sigma}$,
for each $v_1 \in \mathsf{Var}^{:}$, *for each* $v_2 \in \mathsf{Var}^{:}$, $v_1 = v_2 \in \mathsf{AVM}^{\Sigma}$,
for each $v \in \mathsf{Var}$, *for each* $\kappa \in \mathsf{AVM}^{\Sigma}$, $\exists v\ \kappa \in \mathsf{AVM}^{\Sigma}$,
for each $v \in \mathsf{Var}$, *for each* $\kappa \in \mathsf{AVM}^{\Sigma}$, $\forall v\ \kappa \in \mathsf{AVM}^{\Sigma}$,

for each $\kappa \in \mathsf{AVM}^{\Sigma}$, $\neg\kappa \in \mathsf{AVM}^{\Sigma}$,
for each $\kappa_1 \in \mathsf{AVM}^{\Sigma}$, *for each* $\kappa_2 \in \mathsf{AVM}^{\Sigma}$, $(\kappa_1 \wedge \kappa_2) \in \mathsf{AVM}^{\Sigma}$,
for each $\kappa_1 \in \mathsf{AVM}^{\Sigma}$, *for each* $\kappa_2 \in \mathsf{AVM}^{\Sigma}$, $(\kappa_1 \vee \kappa_2) \in \mathsf{AVM}^{\Sigma}$,
for each $\kappa_1 \in \mathsf{AVM}^{\Sigma}$, *for each* $\kappa_2 \in \mathsf{AVM}^{\Sigma}$, $(\kappa_1 \rightarrow \kappa_2) \in \mathsf{AVM}^{\Sigma}$,
for each $\kappa_1 \in \mathsf{AVM}^{\Sigma}$, *for each* $\kappa_2 \in \mathsf{AVM}^{\Sigma}$, $(\kappa_1 \leftrightarrow \kappa_2) \in \mathsf{AVM}^{\Sigma}$.

I call each element of AVM^{Σ} a Σ AVM formula. Every tagged Σ box is also an atomic Σ AVM formula. I call each Σ formula of the form $\rho(v_1, \ldots, v_{\mathcal{AR}(\rho)})$ a relational Σ AVM formula, and each equation between elements of $\mathsf{Var}^{:}$ a Σ AVM equation. Both relational Σ AVM formulae and Σ AVM equations are atomic Σ AVM formulae. Complex Σ AVM formulae can be formed with quantification and the standard logical connectives in the usual way. When working with a fixed signature, I refer to Σ boxes and Σ AVM formulae as boxes and AVM formulae, respectively.

With Σ AVM formulae in hand, it is possible to write HPSG principles in a format that is much more familiar to linguists than the RSRL description syntax of section 2. As an example, consider the SUBCATEGORIZATION PRINCIPLE of Pollard and Sag (1994) written as an AVM formula:

$$(2)\quad {}^{:}\begin{bmatrix} \textit{phrase} & \\ \textsc{dtrs} & \begin{bmatrix}\textit{headed-struc}\end{bmatrix} \end{bmatrix} \rightarrow$$

$$\exists\boxed{1}\ \exists\boxed{2}\ \exists\boxed{3}\ \exists\boxed{4}$$
$$\left(\begin{array}{l}
{}^{:}\begin{bmatrix} \textit{phrase} & \\
\textsc{synsem} & \begin{bmatrix} \textit{synsem} & \\ \textsc{loc} & \begin{bmatrix} \textit{loc} & \\ \textsc{cat} & \begin{bmatrix} \textit{cat} & \\ \textsc{subcat} & \boxed{1}\begin{bmatrix}\textit{list}\end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix} \\
\textsc{dtrs} & \begin{bmatrix} \textit{headed-struc} & \\ \textsc{head-dtr} & \begin{bmatrix} \textit{sign} & \\ \textsc{synsem} & \begin{bmatrix} \textit{synsem} & \\ \textsc{loc} & \begin{bmatrix} \textit{loc} & \\ \textsc{cat} & \begin{bmatrix} \textit{cat} & \\ \textsc{subcat} & \boxed{2}\begin{bmatrix}\textit{list}\end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix} \\ \textsc{comp-dtrs} & \boxed{3}\begin{bmatrix}\textit{list}\end{bmatrix} \end{bmatrix}
\end{bmatrix} \\
\wedge\ \texttt{sign-to-synsem}(\boxed{3}, \boxed{4}) \\
\wedge\ \texttt{append}(\boxed{1}, \boxed{4}, \boxed{2})
\end{array}\right)$$

Following linguistic conventions, the variables in (2) are written as boxed integers. The antecedent of the implication is a matrix which is tagged with the colon. I write the colon at the upper left-hand corner of matrices. All variables in the consequent — which is a complex AVM formula — are bound by existential quantifiers, and the outermost matrix of the complex matrix in the consequent is again preceded by the colon. The outermost matrices of (potentially nested) matrices in an AVM formula I call top matrices. The two matrices tagged with the colon are the top matrices of (2). In section 4, I will introduce a few abbreviatory conventions which further simplify the AVM notation.

What remains to be done in section 3.2 is to give Σ AVM formulae a meaning in RSRL and to show that Σ AVM formulae are indeed a complete notation for grammars.

3.2. **Equivalence to RSRL syntax.** The meaning of Σ AVM formulae is determined by their translation to corresponding Σ formulae. First, I define a translation from pairs of Σ terms and Σ boxes, where the term is thought of as leading to the box, to Σ formulae. Then I use that translation to define the translation of Σ AVM formulae to Σ formulae.[7]

DEFINITION 3.3. *For each signature* Σ*,* btrans *is the total function from the Cartesian product of* $\mathcal{T}^\Sigma$ *and* BOX^Σ *to* $\mathcal{D}^\Sigma$*,* utrans *is the total function from the Cartesian product of* $\mathcal{T}^\Sigma$ *and* UBOX^Σ *to* $\mathcal{D}^\Sigma$*, and* ttrans *is the total function from the Cartesian product of* $\mathcal{T}^\Sigma$ *and* TBOX^Σ *to* $\mathcal{D}^\Sigma$ *such that*

for each $\tau \in \mathcal{T}^\Sigma$*, for each* $\beta \in \mathsf{BOX}^\Sigma$*,*

$$\mathsf{btrans}(\tau, \beta) = \begin{cases} \mathsf{utrans}(\tau,\beta) & \textit{if } \beta \in \mathsf{UBOX}^\Sigma, \\ \mathsf{ttrans}(\tau,\beta) & \textit{if } \beta \in \mathsf{TBOX}^\Sigma \end{cases},$$

for each $\tau \in \mathcal{T}^\Sigma$*, for each* $\sigma \in \widehat{\mathcal{G}}$*, for each* $\alpha_1 \in \widehat{\mathcal{A}}$*, ..., for each* $\alpha_n \in \widehat{\mathcal{A}}$*, for each* $\beta_1 \in \mathsf{BOX}^\Sigma$*, ..., for each* $\beta_n \in \mathsf{BOX}^\Sigma$*,*

$$\mathsf{utrans}\left(\tau, \begin{bmatrix} \sigma & \\ \alpha_1 & \beta_1 \\ \vdots & \\ \alpha_n & \beta_n \end{bmatrix}\right) =$$

$$\left[\tau \sim \sigma \;\wedge\; \bigwedge \left\{ \mathsf{btrans}(\tau\alpha_i, \beta_i) \;\middle|\; i \in \{1, \dots, n\} \right\}\right],$$

for each $\tau \in \mathcal{T}^\Sigma$*, for each* $\beta_1 \in \mathsf{BOX}^\Sigma$*, ..., for each* $\beta_n \in \mathsf{BOX}^\Sigma$*,*

$$\mathsf{utrans}\left(\tau, \langle \beta_1, \dots, \beta_n \rangle\right) =$$

$$[\tau \underbrace{\text{REST} \dots \text{REST}}_{n \textit{ times}} \sim \textit{elist} \wedge$$

[7]For convenience, I adopt the following meta-formula notation in DEFINITION 3.3: For each $\delta \in \mathcal{D}^\Sigma$, for each $\delta_0 \in \mathcal{D}^\Sigma$, ..., for each $\delta_{n+1} \in \mathcal{D}^\Sigma$, $\bigwedge\{\delta\} \stackrel{\text{def}}{=} \delta$, and $\bigwedge\{\delta_0, \dots, \delta_n, \delta_{n+1}\} \stackrel{\text{def}}{=} [\bigwedge\{\delta_0, \dots, \delta_n\} \wedge \delta_{n+1}]$.

$$\bigwedge\left\{\mathsf{btrans}(\tau \underbrace{\text{REST} \ \ldots \ \text{REST}}_{i-1 \ \textit{times}} \ \text{FIRST}, \beta_i) \,\middle|\, i \in \{1, \ldots, n\}\right\}\Bigg],$$

for each $\tau \in \mathcal{T}^{\Sigma}$, *for each* $\beta \in \mathsf{BOX}^{\Sigma}$,

$\mathsf{utrans}(\tau, \neg\beta) = [\tau \approx \tau \wedge \neg \ \mathsf{btrans}(\tau, \beta)]$,

for each $\tau \in \mathcal{T}^{\Sigma}$, *for each* $\beta_1 \in \mathsf{BOX}^{\Sigma}$, *for each* $\beta_2 \in \mathsf{BOX}^{\Sigma}$,

$\mathsf{utrans}\,(\tau, [\beta_1 \wedge \beta_2]) = [\mathsf{btrans}(\tau, \beta_1) \wedge \mathsf{btrans}(\tau, \beta_2)]$,

for each $\tau \in \mathcal{T}^{\Sigma}$, *for each* $\beta_1 \in \mathsf{BOX}^{\Sigma}$, *for each* $\beta_2 \in \mathsf{BOX}^{\Sigma}$,

$\mathsf{utrans}\,(\tau, [\beta_1 \vee \beta_2]) = [\mathsf{btrans}(\tau, \beta_1) \vee \mathsf{btrans}(\tau, \beta_2)]$,

for each $\tau \in \mathcal{T}^{\Sigma}$, *for each* $\beta_1 \in \mathsf{BOX}^{\Sigma}$, *for each* $\beta_2 \in \mathsf{BOX}^{\Sigma}$,

$\mathsf{utrans}\,(\tau, [\beta_1 \rightarrow \beta_2]) = [\mathsf{btrans}(\tau, \beta_1) \rightarrow \mathsf{btrans}(\tau, \beta_2)]$,

for each $\tau \in \mathcal{T}^{\Sigma}$, *for each* $\beta_1 \in \mathsf{BOX}^{\Sigma}$, *for each* $\beta_2 \in \mathsf{BOX}^{\Sigma}$,

$\mathsf{utrans}\,(\tau, [\beta_1 \leftrightarrow \beta_2]) = [\mathsf{btrans}(\tau, \beta_1) \leftrightarrow \mathsf{btrans}(\tau, \beta_2)]$, *and*

for each $\tau \in \mathcal{T}^{\Sigma}$, *for each* $v \in \mathsf{Var}$, *for each* $\beta_1 \in \mathsf{BOX}^{\Sigma}$, ..., *for each* $\beta_n \in \mathsf{BOX}^{\Sigma}$,

$$\mathsf{ttrans}\,(\tau, v \,\|\beta_1, \ldots, \beta_n\|) =$$

$$[\tau \approx v \ \wedge \ [\tau \underbrace{\triangleright \ldots \triangleright}_{n \ \textit{times}} \sim \textit{echain} \ \wedge$$

$$\bigwedge\left\{\mathsf{btrans}(\tau \ \underbrace{\triangleright \ldots \triangleright}_{i-1 \ \textit{times}} \ \dagger, \beta_i) \,\middle|\, i \in \{1, \ldots, n\}\right\}\Bigg]\Bigg],$$

for each $\tau \in \mathcal{T}^{\Sigma}$, *for each* $v \in \mathsf{Var}^{:}$, *for each* $\beta \in \mathsf{UBOX}^{\Sigma}$,

$\mathsf{ttrans}(\tau, v \ \beta) = [\tau \approx v \wedge \mathsf{utrans}(\tau, \beta)]$.

I call btrans the Σ box translation function. The basic intuition behind the definition of btrans is that the Σ term, τ, leads to the Σ box, β, and we prefix all paths with τ when we translate β. btrans distinguishes between the case when β is an untagged Σ box, which is treated by utrans, and the case when β is a tagged Σ box, which is treated by ttrans.

A Σ term, τ, and an untagged Σ matrix with the symbol, σ, from the expanded sort set, the symbols α_1 to α_n from the expanded attribute set followed by the Σ boxes β_1 to β_n, respectively, are translated to the Σ formula consisting of the sort assignment $\tau \sim \sigma$ and n conjuncts in which, for each line i of the Σ matrix, btrans is applied to the Σ term $\tau\alpha_i$ and the Σ box β_i. A Σ term, τ, and an untagged Σ list description, $\langle\beta_1, \ldots, \beta_n\rangle$, are translated to the Σ formula consisting of the sort assignment $\tau' \sim \textit{elist}$, where τ' is the Σ term which consists of τ followed by a string of n REST attributes, and n conjuncts in which, for i from 1 to n, btrans is applied to τ followed by $i - 1$ REST attributes and the ith Σ box in the

Σ list description. Note that, according to the Σ formula interpretation function, our definition of btrans makes sure that the Σ formula resulting from the translation of any pair, $\langle \tau, \beta \rangle$, can only be true of an object, u, if $T_I^{\mathrm{ass}}(\tau)(u)$ is defined. This property of the Σ box translation function also holds for the translation of Σ terms, τ, and a negated untagged Σ box, $\neg\beta$, where the only purpose of the first conjunct of the translation, $\tau \approx \tau$, is to enforce it. The reason I define btrans this way is, of course, that I think that it correctly reflects how linguists usually interpret their informal AVM diagrams. The translation of Σ terms and untagged Σ boxes combined by the standard logical connectives is straightforward.

For the translation of Σ terms, τ, and tagged Σ boxes, $v\ \beta$, the function ttrans simply uses utrans on τ and the untagged Σ box β, which we have already defined, and conjoins the resulting Σ formula with the path equation $\tau \approx v$. The only tagged Σ boxes this definition cannot apply to are Σ chain descriptions, since they do not have untagged counterparts. Therefore, there is a special definition for Σ chain descriptions which mirrors the case of untagged Σ list descriptions but adds the path equation $\tau \approx v$.

We are now ready to translate AVM formulae:

Definition 3.4. *For each signature Σ,* trans *is the total function from* AVM^Σ *to* $\mathcal{D}^\Sigma$ *such that*

for each $v \in \mathsf{Var}$, *for each* $\beta_1 \in \mathsf{BOX}^\Sigma$, $\ldots$, *for each* $\beta_n \in \mathsf{BOX}^\Sigma$,

$\mathsf{trans}\,(v\,\|\beta_1, \ldots, \beta_n\|) = \mathsf{ttrans}\,(v, v\,\|\beta_1, \ldots, \beta_n\|)$,

for each $v \in \mathsf{Var}^{:}$, *for each* $\beta \in \mathsf{UBOX}^\Sigma$,

$\mathsf{trans}(v\ \beta) = \mathsf{ttrans}(v, v\ \beta)$,

for each $\rho \in \mathcal{R}$, *for each* $v_1 \in \mathsf{Var}$, $\ldots$, *for each* $v_{\mathcal{AR}(\rho)} \in \mathsf{Var}$,

$\mathsf{trans}\,\big(\rho(v_1, \ldots, v_{\mathcal{AR}(\rho)})\big) = \rho(v_1, \ldots, v_{\mathcal{AR}(\rho)})$,

for each $v_1 \in \mathsf{Var}^{:}$, *for each* $v_2 \in \mathsf{Var}^{:}$,

$\mathsf{trans}\,(v_1 = v_2) = v_1 \approx v_2$,

for each $v \in \mathsf{Var}$, *for each* $\kappa \in \mathsf{AVM}^\Sigma$,

$\mathsf{trans}\,(\exists v\ \kappa) = \exists v\, \mathsf{trans}(\kappa)$,

for each $v \in \mathsf{Var}$, *for each* $\kappa \in \mathsf{AVM}^\Sigma$,

$\mathsf{trans}\,(\forall v\ \kappa) = \forall v\, \mathsf{trans}(\kappa)$,

for each $\kappa \in \mathsf{AVM}^\Sigma$,

$\mathsf{trans}\,(\neg\kappa) = \neg\mathsf{trans}(\kappa)$,

for each $\kappa_1 \in \mathsf{AVM}^\Sigma$, *for each* $\kappa_2 \in \mathsf{AVM}^\Sigma$,

$\mathsf{trans}\,((\kappa_1 \wedge \kappa_2)) = [\mathsf{trans}(\kappa_1) \wedge \mathsf{trans}(\kappa_2)]$,

for each $\kappa_1 \in \mathsf{AVM}^\Sigma$, *for each* $\kappa_2 \in \mathsf{AVM}^\Sigma$,
$\quad \mathsf{trans}\left((\kappa_1 \vee \kappa_2)\right) = [\mathsf{trans}(\kappa_1) \vee \mathsf{trans}(\kappa_2)]$,
for each $\kappa_1 \in \mathsf{AVM}^\Sigma$, *for each* $\kappa_2 \in \mathsf{AVM}^\Sigma$,
$\quad \mathsf{trans}\left((\kappa_1 \rightarrow \kappa_2)\right) = [\mathsf{trans}(\kappa_1) \rightarrow \mathsf{trans}(\kappa_2)]$,
for each $\kappa_1 \in \mathsf{AVM}^\Sigma$, *for each* $\kappa_2 \in \mathsf{AVM}^\Sigma$,
$\quad \mathsf{trans}\left((\kappa_1 \leftrightarrow \kappa_2)\right) = [\mathsf{trans}(\kappa_1) \leftrightarrow \mathsf{trans}(\kappa_2)]$.

I call trans the Σ AVM translation function. On the basis of the Σ box translation function, the definition of trans is fairly trivial. The function trans applied to a Σ AVM formula which is a tagged Σ box is simply the ttrans value of the tag, v, of that formula and the formula itself. Note that the path equation $v \approx v$, which ttrans introduces into the translation of v and a Σ box, is always true of all objects in any interpretation. Σ chain descriptions must, again, be translated separately, since they do not have untagged twins. Relational Σ AVM formulae and Σ AVM equations are trivially translated to their counterparts, relational Σ formulae and path equations. For quantification and complex Σ AVM formulae involving the logical connectives, the translation is likewise straightforward. Henceforth, when I talk about the denotation of a Σ AVM formula, κ, I mean the denotation of the translation of κ, $\mathsf{trans}(\kappa)$.

For each Σ AVM formula, κ, I define the set of free occurrences of variables in κ. I will need this definition below to show that free variables are preserved under the translation from AVM syntax to RSRL syntax.

Definition 3.5. *For each signature* $\Sigma = \langle \mathcal{G}, \sqsubseteq, \mathcal{S}, \mathcal{A}, \mathcal{F}, \mathcal{R}, \mathcal{AR} \rangle$,
for each $\sigma \in \widehat{\mathcal{G}}$, *for each* $\alpha_1 \in \widehat{\mathcal{A}}$, ..., *for each* $\alpha_n \in \widehat{\mathcal{A}}$, *for each* $\beta_1 \in \mathsf{BOX}^\Sigma$, ..., *for each* $\beta_n \in \mathsf{BOX}^\Sigma$,

$$FV^{AVM}\left(\begin{bmatrix} \sigma & \\ \alpha_1 & \beta_1 \\ \vdots & \\ \alpha_n & \beta_n \end{bmatrix}\right) = \left\{ v \in \mathsf{Var} \,\middle|\, \begin{array}{c} \textit{for some } i \in \{1, \ldots, n\}, \\ v \in FV^{AVM}(\beta_i) \end{array} \right\},$$

for each $\beta_1 \in \mathsf{BOX}^\Sigma$, ..., *for each* $\beta_n \in \mathsf{BOX}^\Sigma$,

$$FV^{AVM}\left(\langle \beta_1, \ldots, \beta_n \rangle\right) = \left\{ v \in \mathsf{Var} \,\middle|\, \begin{array}{c} \textit{for some } i \in \{1, \ldots, n\}, \\ v \in FV^{AVM}(\beta_i) \end{array} \right\},$$

for each $\beta \in \mathsf{BOX}^\Sigma$, $FV^{AVM}(\neg\beta) = FV^{AVM}(\beta)$,
for each $\beta_1 \in \mathsf{BOX}^\Sigma$, *for each* $\beta_2 \in \mathsf{BOX}^\Sigma$,
$\quad FV^{AVM}([\beta_1 \wedge \beta_2]) = FV^{AVM}(\beta_1) \cup FV^{AVM}(\beta_2)$,
for each $\beta_1 \in \mathsf{BOX}^\Sigma$, *for each* $\beta_2 \in \mathsf{BOX}^\Sigma$,

$$
\begin{aligned}
&\quad FV^{AVM}([\beta_1 \vee \beta_2]) = FV^{AVM}(\beta_1) \cup FV^{AVM}(\beta_2), \\
&\textit{for each } \beta_1 \in \mathsf{BOX}^\Sigma, \textit{ for each } \beta_2 \in \mathsf{BOX}^\Sigma, \\
&\quad FV^{AVM}([\beta_1 \rightarrow \beta_2]) = FV^{AVM}(\beta_1) \cup FV^{AVM}(\beta_2), \\
&\textit{for each } \beta_1 \in \mathsf{BOX}^\Sigma, \textit{ for each } \beta_2 \in \mathsf{BOX}^\Sigma, \\
&\quad FV^{AVM}([\beta_1 \leftrightarrow \beta_2]) = FV^{AVM}(\beta_1) \cup FV^{AVM}(\beta_2), \\
&\textit{for each } v \in \mathsf{Var}, \textit{ for each } \beta_1 \in \mathsf{BOX}^\Sigma, \ldots, \textit{ for each } \beta_n \in \mathsf{BOX}^\Sigma, \\
&\quad FV^{AVM}(v \, \| \beta_1, \ldots, \beta_n \|) \\
&\quad = \{v\} \cup \left\{ v' \in \mathsf{Var} \;\middle|\; \begin{array}{c} \textit{for some } i \in \{1, \ldots, n\}, \\ v' \in FV^{AVM}(\beta_i) \end{array} \right\}, \\
&\textit{for each } v \in \mathsf{Var}^{:}, \textit{ for each } \beta \in \mathsf{UBOX}^\Sigma, \\
&\quad FV^{AVM}(v \ \beta) = FV(v) \cup FV^{AVM}(\beta), \\
&\textit{for each } \rho \in \mathcal{R}, \textit{ for each } v_1 \in \mathsf{Var}, \ldots, \textit{ for each } v_{\mathcal{AR}(\rho)} \in \mathsf{Var}, \\
&\quad FV^{AVM}\big(\rho(v_1, \ldots, v_{\mathcal{AR}(\rho)})\big) = \{v_1, \ldots, v_{\mathcal{AR}(\rho)}\}, \\
&\textit{for each } v_1 \in \mathsf{Var}^{:}, \textit{ for each } v_2 \in \mathsf{Var}^{:}, \\
&\quad FV^{AVM}(v_1 = v_2) = FV(v_1) \cup FV(v_2), \\
&\textit{for each } v \in \mathsf{Var}, \textit{ for each } \kappa \in \mathsf{AVM}^\Sigma, \\
&\quad FV^{AVM}(\exists v \ \kappa) = FV^{AVM}(\kappa) \backslash \{v\}, \\
&\textit{for each } v \in \mathsf{Var}, \textit{ for each } \kappa \in \mathsf{AVM}^\Sigma, \\
&\quad FV^{AVM}(\forall v \ \kappa) = FV^{AVM}(\kappa) \backslash \{v\}, \\
&\textit{for each } \kappa \in \mathsf{AVM}^\Sigma, \ FV^{AVM}(\neg \kappa) = FV^{AVM}(\kappa), \\
&\textit{for each } \kappa_1 \in \mathsf{AVM}^\Sigma, \textit{ for each } \kappa_2 \in \mathsf{AVM}^\Sigma, \\
&\quad FV^{AVM}((\kappa_1 \wedge \kappa_2)) = FV^{AVM}(\kappa_1) \cup FV^{AVM}(\kappa_2), \\
&\textit{for each } \kappa_1 \in \mathsf{AVM}^\Sigma, \textit{ for each } \kappa_2 \in \mathsf{AVM}^\Sigma, \\
&\quad FV^{AVM}((\kappa_1 \vee \kappa_2)) = FV^{AVM}(\kappa_1) \cup FV^{AVM}(\kappa_2), \\
&\textit{for each } \kappa_1 \in \mathsf{AVM}^\Sigma, \textit{ for each } \kappa_2 \in \mathsf{AVM}^\Sigma, \\
&\quad FV^{AVM}((\kappa_1 \rightarrow \kappa_2)) = FV^{AVM}(\kappa_1) \cup FV^{AVM}(\kappa_2), \\
&\textit{for each } \kappa_1 \in \mathsf{AVM}^\Sigma, \textit{ for each } \kappa_2 \in \mathsf{AVM}^\Sigma, \\
&\quad FV^{AVM}((\kappa_1 \leftrightarrow \kappa_2)) = FV^{AVM}(\kappa_1) \cup FV^{AVM}(\kappa_2).
\end{aligned}
$$

Let AVM_0^Σ be the set of Σ AVM formulae without free variables. I call each element of AVM_0^Σ a Σ AVM description. When working with a fixed signature, I simply refer to them as AVM descriptions.

We can now show the correspondence between the set of variables that occur free in expressions of our AVM language and in the translations of those expressions. First, we must show the correspondence for btrans:

LEMMA 3.6. *For each signature* Σ*, for each* $\tau \in \mathcal{T}^\Sigma$*, for each* $\beta \in \mathsf{BOX}^\Sigma$*,*
$FV\left(\mathsf{btrans}(\tau, \beta)\right) = FV(\tau) \cup FV^{AVM}(\beta)$,

PROOF 3.7. *By induction on the complexity of* β. ■

LEMMA 3.6 says that the set of variables that occur free in the Σ formula obtained from applying the Σ box translation function, btrans, to a Σ term, τ, and a Σ box, β, equals the union of the set of free variables in τ and the set of free variables in β.

With LEMMA 3.6 in hand it is easy to show that the set of free variables in a Σ AVM formula, κ, equals the set of free variables in the Σ formula obtained when we apply the Σ AVM translation function, trans, to κ:

PROPOSITION 3.8. *For each signature* Σ*, for each* $\kappa \in \mathsf{AVM}^\Sigma$*,*
$FV^{AVM}(\kappa) = FV(\mathsf{trans}(\kappa))$.

PROOF 3.9. *By induction on the complexity of* κ *and* LEMMA *3.6.* ■

An important special case of this result is that our translation of a Σ AVM description is always a Σ description. With COROLLARY 2.17 we thus know that, as for Σ descriptions, the meaning of Σ AVM descriptions is independent of the variable assignments in I.

The next proposition is essential for the main result of this section, THEOREM 3.12. It says that, for each signature Σ and each Σ formula δ, we can find a Σ AVM formula κ such that the same variables occur free in δ and κ and, for certain variable assignments, δ and κ denote the same set of objects in each interpretation.

PROPOSITION 3.10. *For each signature* Σ*, for each* $\delta \in \mathcal{D}^\Sigma$*, there exists a* $\kappa \in \mathsf{AVM}^\Sigma$ *such that, for each interpretation* $I = \langle U, S, A, R \rangle$*, for each* $u \in U$*, for each* $\mathrm{ass} \in Ass_I$*,*
$FV(\delta) = FV^{AVM}(\kappa)$*, and*
if for each $v \in FV(\delta)$*,* $\mathrm{ass}(v) \in \overline{Co_I^u}$*,*
then $u \in D_I^{\mathrm{ass}}(\delta)$ *iff* $u \in D_I^{\mathrm{ass}}(\mathsf{trans}(\kappa))$.

PROOF 3.11. *By induction on the complexity of* δ*. Sketch: For each* $\delta \in \mathcal{D}^\Sigma$ *I have to provide a* $\kappa \in \mathsf{AVM}^\Sigma$ *such that the proposition holds. Base cases:*

For each $n \in \mathbb{N}$*, for* $\delta = v\alpha_1 \ldots \alpha_n \sim \sigma$*, choose* $v\begin{bmatrix} metatop \\ \alpha_1 \ \ldots \begin{bmatrix} metatop \\ \alpha_n \ [\sigma] \end{bmatrix} \ldots \end{bmatrix}$.

For each $n \in \mathbb{N}$*, for each* $m \in \mathbb{N}$*, for each* $\delta = v\alpha_1 \ldots \alpha_n \approx v'\alpha'_1 \ldots \alpha'_m$*, if* $n = 0$ *and* $m = 0$*, choose* $v = v'$*,* *if* $n \geq 1$ *and* $m = 0$, *choose* $v\begin{bmatrix} \textit{metatop} \\ \alpha_1 \ldots \begin{bmatrix} \textit{metatop} \\ \alpha_n \;\; v' \begin{bmatrix}\textit{metatop}\end{bmatrix}\end{bmatrix} \ldots \end{bmatrix}$,

if $n = 0$ *and* $m \geq 1$, *analogous to the previous case,*
if $n \geq 1$ *and* $m \geq 1$, *choose*

$$\exists v'' \left(v\begin{bmatrix} \textit{metatop} \\ \alpha_1 \ldots \begin{bmatrix} \textit{metatop} \\ \alpha_n \;\; v'' \begin{bmatrix}\textit{metatop}\end{bmatrix}\end{bmatrix} \ldots \end{bmatrix} \wedge \; v'\begin{bmatrix} \textit{metatop} \\ \alpha'_1 \ldots \begin{bmatrix} \textit{metatop} \\ \alpha'_m \;\; v'' \begin{bmatrix}\textit{metatop}\end{bmatrix}\end{bmatrix} \ldots \end{bmatrix} \right).$$

For each relational Σ *formula, choose the identical relational* Σ *AVM formula.*
The proposition follows by induction. ■

Everything is now in place for the main theorem. Informally, THEOREM 3.12 says that for each Σ description, δ, there exists a Σ AVM description, κ, such that δ and κ denote the same set of objects in each interpretation.

THEOREM 3.12. *For each signature* Σ, *for each* $\delta \in \mathcal{D}_0^\Sigma$, *there exists a* $\kappa \in \mathsf{AVM}_0^\Sigma$ *such that, for each interpretation* $I = \langle U, S, A, R\rangle$, *for each* $u \in U$, *for each* $\text{ass} \in Ass_I$,

$u \in D_I(\delta)$ *iff* $u \in D_I(\mathsf{trans}(\kappa))$.

PROOF 3.13. *Follows immediately from* PROPOSITION *3.10.*

THEOREM 3.12 has significant consequences. Since the theory, θ, of a grammar, $\Gamma = \langle \Sigma, \theta\rangle$, is a set of descriptions and, for each description, δ, we can find an AVM description which is true of an object in any interpretation of Γ exactly if δ is true of it, we can write θ as a set of AVM descriptions, θ^{AVM}, rather than as a set of descriptions. Because each element of θ has a corrensponding element with the same denotation in θ^{AVM}, the grammar $\langle \Sigma, \theta\rangle$ and the AVM grammar $\langle \Sigma, \theta^{AVM}\rangle$ have the same models and, thus, the same exhaustive models. But in linguistic terms that means that for each grammar, Γ, there is an AVM grammar, Γ^{AVM}, which is true of a natural language exactly if Γ is true of it. In other words, it does not matter whether one chooses RSRL's original syntax or the AVM syntax to write a grammar. For linguistic purposes, they are equivalent.

4. ABBREVIATORY CONVENTIONS

With AVM descriptions, we have a convenient syntax for specifying theories of HPSG grammars. However, a few abbreviatory conventions tailored to the specific purpose of our formal language can improve the usefulness of the new notation further. The principal application of the

AVM syntax is to write grammatical principles, or AVM descriptions, which lack free occurrences of variables. I will, therefore, introduce a number of notational conventions which take this into account and bring AVM descriptions even closer to actual linguistic practice. Throughout this section, if nothing else is indicated, I presuppose the signature of the appendix of Pollard and Sag (1994) and assume that it is augmented by the necessary relations, which the authors leave implicit.

As I have already done in example (2) above, I adopt the usual bracketing conventions of first order predicate logic and leave out the round brackets of AVM formulae when the omission cannot lead to confusion. With those conventions, the HEAD FEATURE PRINCIPLE of Pollard and Sag (1994), written as an AVM description, looks as follows:

$$
(3)\quad {:}\left[\begin{array}{ll} \textit{phrase} & \\ \textsc{dtrs} & \left[\textit{headed-struc}\right] \end{array}\right] \rightarrow
$$

$$
\exists\boxed{1}\; {:}\left[\begin{array}{ll}
\textit{phrase} & \\
\textsc{synsem} & \left[\begin{array}{ll} \textit{synsem} & \\ \textsc{loc} & \left[\begin{array}{ll} \textit{loc} & \\ \textsc{cat} & \left[\begin{array}{ll} \textit{cat} & \\ \textsc{head} & \boxed{1}\left[\textit{head}\right] \end{array}\right] \end{array}\right] \end{array}\right] \\
\textsc{dtrs} & \left[\begin{array}{ll} \textit{headed-struc} & \\ \textsc{head-dtr} & \left[\begin{array}{ll} \textit{sign} & \\ \textsc{synsem} & \left[\begin{array}{ll} \textit{synsem} & \\ \textsc{loc} & \left[\begin{array}{ll} \textit{loc} & \\ \textsc{cat} & \left[\begin{array}{ll} \textit{cat} & \\ \textsc{head} & \boxed{1}\left[\textit{head}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right] \end{array}\right]
\end{array}\right]
$$

There are a few notational properties that distinguish (3) from how linguists would write the HEAD FEATURE PRINCIPLE. First, the colon, which functions as the tag of the top matrix in the antecedent and in the consequent, is a particular feature of our AVM descriptions. Since by definition every top matrix in an AVM description must be tagged, I can introduce a convention that determines which unique tag I mean when I leave top matrices untagged:

CONVENTION 4.1. *The colon as the tag of a top matrix may be left out in AVM descriptions.*

Second, in every matrix there must be a symbol from the expanded sort set at the top. Clearly, this is not necessary for practical purposes, since most sort symbols in the matrices of (3) do not restrict the meaning of the description. On the other hand, they make the matrices look much more complex than they really are. It is, thus, convenient to leave those symbols out. Because *metatop* denotes all objects and chains of the universe, it can always be assumed that *metatop* should be inserted where no other symbol stands.

CONVENTION 4.2. *At the top of matrices, the symbol from the expanded sort set may be left out. If it is missing, then I regard metatop as the missing symbol.*

Third, once superfluous sort symbols are left out, (3) contains many angled brackets of matrices that do not serve any real graphical purpose, because there is only one attribute in the matrix. If the brackets do anything at all, they make the description harder to read. This is inappropriate for a practical notation, and I eliminate this notational overhead:

CONVENTION 4.3. *If a matrix which is not a top matrix contains only one attribute and no symbol from the expanded sort set or no attribute at all, then the brackets of that matrix may be left out.*

Finally, the single variable in (3) is existentially bound. As an investigation of the use of tags in the HPSG literature in the light of the meaning of AVM descriptions shows, they are almost always to be understood as being existentially bound variables, and, as in the HEAD FEATURE PRINCIPLE, it can be assumed that the existential quantifier which binds the variables takes wide scope. It is, therefore, practical to leave wide scope existential quantification implicit in grammatical principles, and only notate quantifiers which are exceptions to this default rule.

CONVENTION 4.4. *If a tag is not explicitly bound by a quantifier, then I assume that it is captured by an implicit existential closure with wide scope over the entire formula.*

Applying the notational conventions 4.1, 4.2, 4.3, and 4.4 to the HEAD FEATURE PRINCIPLE leads to the following notation:

$$(4)\quad \begin{bmatrix} \textit{phrase} \\ \textsc{dtrs}\ \ \textit{headed-struc} \end{bmatrix} \quad \rightarrow \quad \begin{bmatrix} \textsc{synsem loc cat head}\ \ \boxed{1} \\ \textsc{dtrs head-dtr synsem loc cat head}\ \ \boxed{1} \end{bmatrix}$$

Notice that, in accordance with the conventions 4.2 and 4.3, the tag $\boxed{1}$ in the consequent of example (4) abbreviates the tagged matrix $\boxed{1}[\textit{metatop}]$.

Finally, it proves to be useful for the description of lists and chains to borrow the standard notation of the programming language Prolog for referring to the head and the tail of a list or chain:

CONVENTION 4.5. *For each signature Σ, for each Σ AVM formula ϕ_1, and for each Σ AVM formula ϕ_2:*

$$\langle\phi_1|\phi_2\rangle = \left[\begin{array}{ll} \textit{list} & \\ \text{FIRST} & \phi_1 \\ \text{REST} & \phi_2 \end{array}\right], \textit{ and } \ \|\phi_1|\phi_2\| = \left[\begin{array}{ll} \textit{chain} & \\ \dagger & \phi_1 \\ \triangleright & \phi_2 \end{array}\right].$$

ϕ_1 and ϕ_2 may of course employ any of the abbreviatory conventions which are defined in this section.

HPSG grammars make increasingly intense use of relations. In theories, two types of appearances of relational AVM formulae can be distinguished. First, the theory of a grammar contains descriptions which serve to define the intended meaning of a relation, e.g., the meaning of `append` or `member`. Those descriptions are formally on a par with other principles of grammar such as the SUBCATEGORIZATION PRINCIPLE or the HEAD FEATURE PRINCIPLE, and, to emphasize this similarity, one could call them the `append` principle, the `member` principle, etc. Second, relational AVM formulae occur in other grammatical principles such us the SUBCATEGORIZATION PRINCIPLE, where they serve to ensure certain properties of linguistic objects. In this case, the occurrence of the relation makes use of the meaning of the relation which is defined in the relation principles.

Before I take a closer look at AVM descriptions which define the meaning of relations, I introduce a notational convention that is common linguistic practice. The lexical entry shown in (5) is adapted from Hinrichs and Nakazawa (1994, p.25). It is a typical example of a so-called argument raising predicate.[8]

(5)
$$\left[\begin{array}{ll} \textit{word} & \\ \text{PHON} & \langle \textit{haben} \rangle \\ \text{SYNSEM LOC CAT} & \left[\begin{array}{ll} \text{HEAD} & \left[\begin{array}{ll} \text{MAJ} & \textit{verb} \\ \text{AUX} & + \\ \text{FLIP} & \boxed{2} \end{array}\right] \\ \text{SUBCAT} & \boxed{3} \\ \text{NPCOMP} & - \end{array}\right] \end{array}\right]$$
$$\wedge\ \boxed{4} \left\langle \left[\begin{array}{ll} \text{LOC CAT} & \left[\begin{array}{ll} \text{HEAD} & \left[\begin{array}{ll} \text{MAJ} & \textit{verb} \\ \text{VFORM} & \textit{psp} \\ \text{FLIP} & \boxed{2} \end{array}\right] \\ \text{SUBCAT} & \boxed{1} \\ \text{NPCOMP} & - \end{array}\right] \end{array}\right] \right\rangle$$

[8]For this example, I assume that the signature is slightly changed by adding the *head* attributes MAJ and FLIP, and the *cat* attribute NPCOMP, with their respective appropriateness values.

$\wedge$ `append`($\boxed{1}$, $\boxed{4}$, $\boxed{3}$)

The lexical entry of *haben* above makes use of all conventions discussed so far, except for the list convention 4.5. The colon at the first top matrix is left implicit; most matrices do not contain sort symbols; most matrix brackets are omitted; and all tags are bound by implicit existential closure. However, contrary to the common notation in the HPSG literature, the description of the second argument of the `append` relation is written as a separate conjunct of the overall AVM description instead of being part of the relational AVM formula. Since the latter option seems to be an elegant way of representing the same description in a more compact fashion, I permit this notation:

CONVENTION 4.6. *A tagged box, ϕ, may be written into an argument slot of a relational AVM formula. This notation is interpreted as abbreviating a conjunction of ϕ with the relational AVM formula which contains only the tag of ϕ in the respective argument slot. If the tag of ϕ is the colon, the notation is interpreted as described, except that the relevant argument slot of the relational AVM formula now contains the alphabetically first variable, v, which does not occur yet in the overall description, and the AVM equation $:= v$ is conjoined with the relational AVM formula and ϕ.*

According to convention 4.6, the lexical entry of *haben* can be rewritten as follows:

(6)
$$\begin{bmatrix} \textit{word} & \\ \text{PHON} & \langle \textit{haben} \rangle \\ \text{SYNSEM LOC CAT} & \begin{bmatrix} \text{HEAD} & \begin{bmatrix} \text{MAJ} & \textit{verb} \\ \text{AUX} & + \\ \text{FLIP} & \boxed{2} \end{bmatrix} \\ \text{SUBCAT} & \boxed{3} \\ \text{NPCOMP} & - \end{bmatrix} \end{bmatrix}$$

$$\wedge\ \texttt{append}\left(\boxed{1}, \boxed{4}\left\langle \begin{bmatrix} \text{LOC CAT} & \begin{bmatrix} \text{HEAD} & \begin{bmatrix} \text{MAJ} & \textit{verb} \\ \text{VFORM} & \textit{psp} \\ \text{FLIP} & \boxed{2} \end{bmatrix} \\ \text{SUBCAT} & \boxed{1} \\ \text{NPCOMP} & - \end{bmatrix} \end{bmatrix} \right\rangle, \boxed{3}\right)$$

The `append` principle which must be a member of the theory of the grammar in order to give the relational AVM formula in the lexical entry in (6) the intended meaning looks as follows:

(7)
$$\begin{aligned} \forall x \forall y \forall z\ (\texttt{append}(x,y,z) \leftrightarrow & \\ & (x \langle\rangle \wedge\ y = z) \\ & \vee\ \exists\boxed{1}\ \exists\boxed{2}\ \exists\boxed{3}\ \ (x \langle \boxed{1}|\boxed{2} \rangle \wedge z \langle \boxed{1}|\boxed{3} \rangle \wedge \texttt{append}(\boxed{2}, y, \boxed{3}))) \end{aligned}$$

Note that in (7) the existential quantification over the variables which are written as boxed integers cannot be left implicit. The existential closure convention, 4.4, cannot be evoked to leave out the existential quantifiers, since existential closure would wrongly lead to wide scope of the existentials over the universals. If the AVM description (7) is in the theory of a grammar, Γ, it determines, for each object, u, in the universe of an exhaustive model of Γ, which components, x, y, and z, of u must be in the `append` relation. It is exactly those triples of components of u that are either described by the first disjunct (second line of (7)) or the second disjunct (third line of (7)) after the equivalence symbol.

In some cases, it proves to be advantageous to employ a slightly different notation for relation principles. That notation, again, owes its syntax to Prolog, and appeals to common notational habits of linguists working in HPSG. It breaks up the AVM description of relation principles like (7) into multiple clauses. The number of clauses determining the meaning of a relation depends on the number of disjuncts to the right of the equivalence symbol in the corresponding AVM formula notation, and each clause is an independent piece of syntax. Because of its design, the clause notation is very convenient when there are many disjuncts that define the meaning of a relation. If a relation principle is written as clauses, it is split up in a modular fashion and its parts can be presented, motivated and discussed separately in the course of linguistic or formal reasoning. This is an important characteristic if a relation has linguistic significance as in a relational formalization of word order rules; whereas, for merely technical relations such as `append`, it is usually not necessary. As we will presently see, a clause notation will also make it easy to leave all the quantifiers of the `append` principle, (7), implicit.

To define conventions for a clause notation, I first need an adaptation to AVM formulae of the standard notion of the simultaneous substitution of variables. In DEFINITION 4.7, curly brackets are used for disambiguation when necessary and have no other significance.

DEFINITION 4.7. *For each signature* $\Sigma = \langle \mathcal{G}, \sqsubseteq, \mathcal{S}, \mathcal{A}, \mathcal{F}, \mathcal{R}, \mathcal{AR} \rangle$, *for all pairwise distinct variables* $x_1, \ldots x_n, v_1, \ldots, v_n$,

for each $v \in \mathsf{Var}^{:}$, $v\frac{v_1 \ldots v_n}{x_1 \ldots x_n} = \begin{cases} v, & \text{if } v \neq x_1, \ldots, v \neq x_n, \\ v_i, & \text{if } v = x_i, \end{cases}$

for each $\sigma \in \widehat{\mathcal{G}}$, *for each* $\alpha_1 \in \widehat{\mathcal{A}}$, ..., *for each* $\alpha_n \in \widehat{\mathcal{A}}$, *for each* $\beta_1 \in \mathsf{BOX}^{\Sigma}$, ..., *for each* $\beta_n \in \mathsf{BOX}^{\Sigma}$,

$$\begin{bmatrix} \sigma \\ \alpha_1 & \beta_1 \\ \vdots \\ \alpha_n & \beta_n \end{bmatrix} \frac{v_1 \ldots v_n}{x_1 \ldots x_n} = \begin{bmatrix} \sigma \\ \alpha_1 & \beta_1 \frac{v_1 \ldots v_n}{x_1 \ldots x_n} \\ \vdots \\ \alpha_n & \beta_n \frac{v_1 \ldots v_n}{x_1 \ldots x_n} \end{bmatrix},$$

for each $\beta_1 \in \mathsf{BOX}^\Sigma$, ..., *for each* $\beta_n \in \mathsf{BOX}^\Sigma$,

$$\langle \beta_1, \ldots, \beta_n \rangle \frac{v_1 \ldots v_n}{x_1 \ldots x_n} = \left\langle \beta_1 \frac{v_1 \ldots v_n}{x_1 \ldots x_n}, \ldots, \beta_n \frac{v_1 \ldots v_n}{x_1 \ldots x_n} \right\rangle,$$

for each $\beta \in \mathsf{BOX}^\Sigma$, $\{\neg\beta\} \frac{v_1 \ldots v_n}{x_1 \ldots x_n} = \neg \left\{ \beta \frac{v_1 \ldots v_n}{x_1 \ldots x_n} \right\}$,

for each $\beta_1 \in \mathsf{BOX}^\Sigma$, *for each* $\beta_2 \in \mathsf{BOX}^\Sigma$,

$$[\beta_1 \wedge \beta_2] \frac{v_1 \ldots v_n}{x_1 \ldots x_n} = [\beta_1 \frac{v_1 \ldots v_n}{x_1 \ldots x_n} \wedge \beta_2 \frac{v_1 \ldots v_n}{x_1 \ldots x_n}],$$

for each $\beta_1 \in \mathsf{BOX}^\Sigma$, *for each* $\beta_2 \in \mathsf{BOX}^\Sigma$,

$$[\beta_1 \vee \beta_2] \frac{v_1 \ldots v_n}{x_1 \ldots x_n} = [\beta_1 \frac{v_1 \ldots v_n}{x_1 \ldots x_n} \vee \beta_2 \frac{v_1 \ldots v_n}{x_1 \ldots x_n}],$$

for each $\beta_1 \in \mathsf{BOX}^\Sigma$, *for each* $\beta_2 \in \mathsf{BOX}^\Sigma$,

$$[\beta_1 \rightarrow \beta_2] \frac{v_1 \ldots v_n}{x_1 \ldots x_n} = [\beta_1 \frac{v_1 \ldots v_n}{x_1 \ldots x_n} \rightarrow \beta_2 \frac{v_1 \ldots v_n}{x_1 \ldots x_n}],$$

for each $\beta_1 \in \mathsf{BOX}^\Sigma$, *for each* $\beta_2 \in \mathsf{BOX}^\Sigma$,

$$[\beta_1 \leftrightarrow \beta_2] \frac{v_1 \ldots v_n}{x_1 \ldots x_n} = [\beta_1 \frac{v_1 \ldots v_n}{x_1 \ldots x_n} \leftrightarrow \beta_2 \frac{v_1 \ldots v_n}{x_1 \ldots x_n}],$$

for each $v \in \mathsf{Var}$, *for each* $\beta_1 \in \mathsf{BOX}^\Sigma$, ..., *for each* $\beta_n \in \mathsf{BOX}^\Sigma$,

$$\{v \, \|\beta_1, \ldots, \beta_n\|\} \frac{v_1 \ldots v_n}{x_1 \ldots x_n} = v \frac{v_1 \ldots v_n}{x_1 \ldots x_n} \left\| \beta_1 \frac{v_1 \ldots v_n}{x_1 \ldots x_n}, \ldots, \beta_n \frac{v_1 \ldots v_n}{x_1 \ldots x_n} \right\|,$$

for each $v \in \mathsf{Var}^{:}$, *for each* $\beta \in \mathsf{UBOX}^\Sigma$,

$$\{v \;\beta\} \frac{v_1 \ldots v_n}{x_1 \ldots x_n} = v \frac{v_1 \ldots v_n}{x_1 \ldots x_n} \; \beta \frac{v_1 \ldots v_n}{x_1 \ldots x_n},$$

for each $\rho \in \mathcal{R}$, *for each* $z_1 \in \mathsf{Var}$, ..., *for each* $z_{\mathcal{AR}(\rho)} \in \mathsf{Var}$,

$$\rho(z_1, \ldots, z_{\mathcal{AR}(\rho)}) \frac{v_1 \ldots v_n}{x_1 \ldots x_n} = \rho(z_1 \frac{v_1 \ldots v_n}{x_1 \ldots x_n}, \ldots, z_{\mathcal{AR}(\rho)} \frac{v_1 \ldots v_n}{x_1 \ldots x_n}),$$

for each $z_1 \in \mathsf{Var}^{:}$, *for each* $z_2 \in \mathsf{Var}^{:}$,

$$\{z_1 = z_2\} \frac{v_1 \ldots v_n}{x_1 \ldots x_n} = z_1 \frac{v_1 \ldots v_n}{x_1 \ldots x_n} = z_2 \frac{v_1 \ldots v_n}{x_1 \ldots x_n},$$

for each $v \in \mathsf{Var}$, *for each* $\kappa \in \mathsf{AVM}^\Sigma$, *for the alphabetically first variable,* z, *which does not occur in* κ *and is distinct from* $v_1, \ldots, v_n$, *for the variables* $x_{i_1}, \ldots x_{i_m}$ *such that* $\{x_{i_1}, \ldots x_{i_m}\} \subseteq \{x_1, \ldots, x_n\}$, $i_1 < \ldots < i_m$, *for each* $x \in \{x_{i_1}, \ldots x_{i_m}\}$, $x \in FV^{AVM}(\exists v\, \kappa)$, *and for no* $x \in \{x_1, \ldots x_n\} \backslash \{x_{i_1}, \ldots x_{i_m}\}$, $x \in FV^{AVM}(\exists v\, \kappa)$,

$$\{\exists v\, \kappa\} \frac{v_1 \ldots v_n}{x_1 \ldots x_n} = \begin{cases} \exists v \left\{ \kappa \frac{v_{i_1} \ldots v_{i_m}}{x_{i_1} \ldots x_{i_m}} \right\} & \textit{if } v \neq v_{i_1}, \ldots, v \neq v_{i_m}, \\ \exists z \left\{ \kappa \frac{v_{i_1} \ldots v_{i_m}, z}{x_{i_1} \ldots x_{i_m}, v} \right\} & \textit{if for some } j \in \{1, \ldots, n\}, \\ & \quad v = v_{i_j}, \end{cases}$$

for each $v \in \mathsf{Var}$, *for each* $\kappa \in \mathsf{AVM}^{\Sigma}$, *for the alphabetically first variable,* z, *which does not occur in* κ *and is distinct from* $v_1, \ldots, v_n$, *for the variables* $x_{i_1}, \ldots x_{i_m}$ *such that* $\{x_{i_1}, \ldots x_{i_m}\} \subseteq \{x_1, \ldots, x_n\}$, $i_1 < \ldots < i_m$, *for each* $x \in \{x_{i_1}, \ldots x_{i_m}\}$, $x \in FV^{AVM}(\exists v\ \kappa)$, *and for no* $x \in \{x_1, \ldots x_n\} \backslash \{x_{i_1}, \ldots x_{i_m}\}$, $x \in FV^{AVM}(\exists v\ \kappa)$,

$$\{\forall v\ \kappa\} \tfrac{v_1 \ldots v_n}{x_1 \ldots x_n} = \begin{cases} \forall v \left\{\kappa \frac{v_{i_1} \ldots v_{i_m}}{x_{i_1} \ldots x_{i_m}}\right\} & \textit{if } v \neq v_{i_1}, \ldots, v \neq v_{i_m}, \\ \forall z \left\{\kappa \frac{v_{i_1} \ldots v_{i_m}, z}{x_{i_1} \ldots x_{i_m}, v}\right\} & \textit{if for some } j \in \{1, \ldots, n\}, \\ & \quad v = v_{i_j}, \end{cases}$$

for each $\kappa \in \mathsf{AVM}^{\Sigma}$, $\{\neg\kappa\} \frac{v_1 \ldots v_n}{x_1 \ldots x_n} = \neg \left\{\kappa \frac{v_1 \ldots v_n}{x_1 \ldots x_n}\right\}$,

for each $\kappa_1 \in \mathsf{AVM}^{\Sigma}$, *for each* $\kappa_2 \in \mathsf{AVM}^{\Sigma}$,
$(\kappa_1 \wedge \kappa_2) \frac{v_1 \ldots v_n}{x_1 \ldots x_n} = (\kappa_1 \frac{v_1 \ldots v_n}{x_1 \ldots x_n} \wedge \kappa_2 \frac{v_1 \ldots v_n}{x_1 \ldots x_n})$,

for each $\kappa_1 \in \mathsf{AVM}^{\Sigma}$, *for each* $\kappa_2 \in \mathsf{AVM}^{\Sigma}$,
$(\kappa_1 \vee \kappa_2) \frac{v_1 \ldots v_n}{x_1 \ldots x_n} = (\kappa_1 \frac{v_1 \ldots v_n}{x_1 \ldots x_n} \vee \kappa_2 \frac{v_1 \ldots v_n}{x_1 \ldots x_n})$,

for each $\kappa_1 \in \mathsf{AVM}^{\Sigma}$, *for each* $\kappa_2 \in \mathsf{AVM}^{\Sigma}$,
$(\kappa_1 \rightarrow \kappa_2) \frac{v_1 \ldots v_n}{x_1 \ldots x_n} = (\kappa_1 \frac{v_1 \ldots v_n}{x_1 \ldots x_n} \rightarrow \kappa_2 \frac{v_1 \ldots v_n}{x_1 \ldots x_n})$,

for each $\kappa_1 \in \mathsf{AVM}^{\Sigma}$, *for each* $\kappa_2 \in \mathsf{AVM}^{\Sigma}$,
$(\kappa_1 \leftrightarrow \kappa_2) \frac{v_1 \ldots v_n}{x_1 \ldots x_n} = (\kappa_1 \frac{v_1 \ldots v_n}{x_1 \ldots x_n} \leftrightarrow \kappa_2 \frac{v_1 \ldots v_n}{x_1 \ldots x_n})$.

The only interesting case in Definition 4.7 is quantification, where it must be ensured that no bound variable is replaced and no new variable gets bound by quantification. For that reason, only those x_i are replaced which occur free in κ, and if a variable which is substituted for some x_i equals the variable that is bound by the quantification, the name of the variable that is quantified over is changed.

After these technical preliminaries, I can come back to the definition of the clause notation:

Definition 4.8. *For each signature* $\Sigma = \langle \mathcal{G}, \sqsubseteq, \mathcal{S}, \mathcal{A}, \mathcal{F}, \mathcal{R}, \mathcal{AR} \rangle$, $\rho\left(x_1, \ldots, x_{\mathcal{AR}(\rho)}\right) \stackrel{\forall}{\Longleftarrow} \phi$ *is a clause iff*

$\rho \in \mathcal{R}$, $x_1 \in \mathsf{Var}$, $\ldots$, $x_{\mathcal{AR}(\rho)} \in \mathsf{Var}$, $x_1, \ldots, x_{\mathcal{AR}(\rho)}$ *are pairwise distinct,* ϕ *is a* Σ *AVM formula, and* $FV^{AVM}(\phi) \subseteq \{x_1, \ldots, x_{\mathcal{AR}(\rho)}\}$.

Clauses employ a new reserved symbol, $\stackrel{\forall}{\Longleftarrow}$. I call it the clause symbol and assume that it is not a variable, a sort, an attribute nor a relation symbol. Notice that the variables in the relational Σ AVM formula to the left of the clause symbol must be pairwise distinct. The universal quantifier which is part of the clause symbol stands for the universal quantification over the variables in the relational formula to its left, which is left implicit. The quantification becomes explicit when we say what a set of clauses defining the meaning of a relation, ρ, abbreviates:

CONVENTION 4.9. *Let* $C = \{\rho(x_{1_1}, \ldots, x_{1_{\mathcal{AR}(\rho)}}) \overset{\forall}{\Longleftarrow} \phi_1, \ldots, \rho(x_{n_1}, \ldots, x_{n_{\mathcal{AR}(\rho)}}) \overset{\forall}{\Longleftarrow} \phi_n\}$ *be a finite set of clauses. Suppose that* $v_1, \ldots, v_{\mathcal{AR}(\rho)}$ *are the alphabetically first pairwise distinct variables which do not occur in any clause in* C*. Then* C *is equivalent to the AVM description*

$$\forall v_1 \ldots \forall v_{\mathcal{AR}(\rho)} \left(\rho(v_1, \ldots, v_{\mathcal{AR}(\rho)}) \leftrightarrow \left(\phi_1 \frac{v_1 \ldots v_{\mathcal{AR}(\rho)}}{x_{1_1} \ldots x_{1_{\mathcal{AR}(\rho)}}} \vee \ldots \vee \phi_n \frac{v_1 \ldots v_{\mathcal{AR}(\rho)}}{x_{n_1} \ldots x_{n_{\mathcal{AR}(\rho)}}}\right)\right).$$

Using clauses and the clause notation convention, 4.9, the `append` principle, (7), can be reformulated as two clauses:

(8)
$$\texttt{append}(x, y, z) \overset{\forall}{\Longleftarrow} x\,\langle\rangle \wedge\ y = z$$
$$\texttt{append}(x, y, z) \overset{\forall}{\Longleftarrow} \exists\boxed{1}\,\exists\boxed{2}\,\exists\boxed{3}\ (x\,\langle\boxed{1}|\boxed{2}\rangle \wedge z\,\langle\boxed{1}|\boxed{3}\rangle \wedge \texttt{append}(\boxed{2}, y, \boxed{3}))$$

The clause notation lends itself easily to further abbreviatory conventions. Two expedient moves are to allow multiple occurrences of the same variable name in the relational AVM formula to the left of the clause symbol and to apply a variant of the existential closure convention, 4.4, to the AVM formula to its right. For this purpose, I loosen the syntactic restrictions on the notation of clauses accordingly and introduce pseudo-clauses. A pseudo-clause looks just like a clause, i.e., it is of the form $\rho\left(x_1, \ldots, x_{\mathcal{AR}(\rho)}\right) \overset{\forall}{\Longleftarrow} \phi$, but either not all the variables $x_1, \ldots, x_{\mathcal{AR}(\rho)}$ are pairwise distinct, or the set of free variables in the AVM formula ϕ is not a subset of $\{x_1, \ldots, x_{\mathcal{AR}(\rho)}\}$, or both. For pseudo-clauses, I state which actual clauses are meant when variable names occur several times in the relational AVM formula, ϕ_ρ, on the left of the clause symbol, and when the AVM formula on the right contains free variables distinct from the ones that occur in ϕ_ρ:

CONVENTION 4.10. *Suppose* $\phi_1 \overset{\forall}{\Longleftarrow} \phi_2$ *is a pseudo-clause. First, all free variables in* ϕ_2*, except for the ones occurring in* ϕ_1*, are regarded as being existentially bound, with the quantifiers scoping over* ϕ_2*.*

Second, if not all variables in ϕ_1 *are pairwise distinct, then proceed as follows: For each variable,* v*, in* ϕ_1*, go through* ϕ_1 *from left to right and check if* v *occurs again. If a second occurrence is found, replace* *it by the alphabetically first variable, v', which is not yet in the clause. Furthermore, conjoin ϕ_2 with the AVM equation $v = v'$. Repeat this procedure until all variables in ϕ_1 are pairwise distinct.*

With the pseudo-clause convention, 4.10, stating the `append` principle, (7), reduces to (9):

(9)
$$\texttt{append}(x, y, y) \overset{\forall}{\Longleftarrow} \\ x\,\langle\rangle$$
$$\texttt{append}(x, y, z) \overset{\forall}{\Longleftarrow} \\ x\,\langle\boxed{1}|\boxed{2}\rangle \wedge z\,\langle\boxed{1}|\boxed{3}\rangle \wedge \texttt{append}(\boxed{2}, y, \boxed{3})$$

The formal language of RSRL lets its users define relations as relations over components of objects. While Richter, Sailer, and Penn (1999) shows that this definition agrees with the actual use of relations and quantification in the HPSG literature, it also recognizes that there are some relations which are apparently at odds with that approach. Relations with list arguments are sometimes used in a way which implies that one or more of their arguments cannot be an actual list that is a component of the described objects. Rather, it is a "virtual list" or sequence of components of the described objects. The ubiquitous `shuffle` relation of so-called linearization grammars is a typical instance of a relation whose use presupposes such a generalized definition, but even `append` is occasionally evoked with "virtual lists" as arguments. To handle those uses of lists in relations over lists, the concept of chains was introduced into RSRL.

In practice, the linguist does not want to think about whether a particular instance of using a relation might involve chains in one or more arguments of that relation, and which argument(s) would be of which type. To make that kind of underspecification easily possible, it is convenient to have an abbreviatory notation which covers all conceivable distributions of lists or chains over a relation's arguments. CONVENTION 4.11 defines that notation.

CONVENTION 4.11. *For each signature Σ, for each $v \in \mathsf{Var}$, for each $\beta_1 \in \mathsf{BOX}^{\Sigma}$, ... , for each $\beta_n \in \mathsf{BOX}^{\Sigma}$,*

$$v \ll \beta_1, \dots, \beta_n \gg \; = v\,[\langle \beta_1, \dots, \beta_n \rangle \vee \| \beta_1, \dots, \beta_n \|].$$

I call $v \ll \beta_1, \dots, \beta_n \gg$ a (Σ) tape description. Tape descriptions may involve all other abbreviatory conventions, especially the head and tail convention, CONVENTION 4.5, for lists and chains. With tape descriptions, relation principles which involve listlike entities can easily be written in such a way that they enforce the intended relationship between both lists and chains. As an illustration, I define the meaning of `shuffle` with tape descriptions:

(10) $$\texttt{shuffle}(x,y,z) \stackrel{\forall}{\Longleftarrow} x \ll\gg \land\ y \ll\gg \land\ z \ll\gg$$

$$\texttt{shuffle}(x,y,z) \stackrel{\forall}{\Longleftarrow} x \ll \boxed{0} | \boxed{1} \gg \land\ z \ll \boxed{0} | \boxed{2} \gg \land\ \texttt{shuffle}(\boxed{1}, y, \boxed{2})$$

$$\texttt{shuffle}(x,y,z) \stackrel{\forall}{\Longleftarrow} y \ll \boxed{0} | \boxed{2} \gg \land\ z \ll \boxed{0} | \boxed{3} \gg \land\ \texttt{shuffle}(x, \boxed{2}, \boxed{3})$$

As the reader may verify, the `shuffle` relation holds between each appropriate triple of lists and chains in an exhaustive model of a grammar containing the relation principle (10).

In the introduction, I quoted a grammatical principle from Kathol (1995), which was formulated as an AVM diagram, and promised that I would provide a similarly-looking formal syntax with an interpretation in RSRL that answered the open questions about the exact meaning of such complex diagrams. In fact, I can now minimally modify that AVM diagram and turn it into an AVM description (under Kathol's signature) which, under certain assumptions about the empirical significance of the involved attributes and sorts, means what the author intended to say.[9] The formalization of the original diagram in terms of RSRL shows both how much was left implicit in the original AVM and how little the syntactic notation must be modified in order to turn the original diagram into a rigorously formalized statement in RSRL.

(11) $$\textit{sign} \rightarrow \neg\ \exists\boxed{1}\ \exists\boxed{2}\ \exists\boxed{3}\ \exists\boxed{4}\ \exists\boxed{5}\ \exists\boxed{6}$$
$$\left(\begin{bmatrix} \text{DOM} & \boxed{2} \\ \text{SS LOC CONTEXT} & \boxed{3} \end{bmatrix} \right.$$
$$\land\ \texttt{shuffle}\left(\boxed{5} \ll \begin{bmatrix} \text{TOPO} & \neg\ \textit{nf} \\ \text{SS LOC CONTENT} & \boxed{1} \end{bmatrix} \gg, \boxed{4}, \boxed{2} \right)$$
$$\left. \land\ \texttt{member}\left(\boxed{6} \begin{bmatrix} \textit{anchor} & \\ \text{ARG1} & \textit{index} \\ \text{ARG2} & \boxed{1}\ \textit{psoa} \end{bmatrix}, \boxed{3} \right) \right)$$

Notice that the existential quantifiers in (11) cannot be left implicit, since for the intended statement, they must be in the scope of the negation symbol in the consequent. Moreover, tags were added in front of the

[9] "What this constraint in effect says is that any domain containing a constituent whose content value is linked via the *anchor* relation to the index of some other entity is required to occur extraposed." (Kathol 1995, p. 291)

formulae in the first argument slot of the relational formulae. The final change compared with (1) concerns the 'list' in the first argument of `shuffle`, which must be described as a tape. The intention of the use of `shuffle` in that instance is that the one element of the tape, if it exists, is a member of the DOM list, $\boxed{2}$. But the entity that is described as the first argument of `shuffle` may only be a real list in the described sign if it is the last element of the DOM list of the sign. Otherwise a list of the described kind might not exist as a component of the sign, and the formula can only be interpreted as the description of a chain of length one whose element is somewhere on the DOM list of the sign.

An issue I have not discussed is the treatment of sets. As already mentioned in Richter, Sailer, and Penn (1999), finite sets can be encoded in RSRL. The basic idea is to employ chains to define the necessary access relations such as membership, union, and intersection for them, and enforce the relevant properties of sets. For AVM formulae, a notational convention would define the standard notation with curly brackets as an abbreviation for the set access relation. For reasons of space, I cannot spell out those relations and the convention just sketched, and I must refer the interested reader to (Richter in prep.) for further details.

References

Calcagno, M., T. Hukari, and R. Levine (1998). Parasitic Gaps in English: Some Overlooked Cases and their Theoretical Implications. In P. Culicover and P. Postal (Eds.), *Parasitic Gaps*. Cambridge, MA: MIT Press. To appear.

Hinrichs, E. and T. Nakazawa (1994). Linearizing AUXs in German Verbal Complexes. In J. Nerbonne, K. Netter, and C. Pollard (Eds.), *German in Head-Driven Phrase Structure Grammar*, Volume 46, pp. 11–37. CSLI.

Kathol, A. (1995). *Linearization-Based German Syntax*. Ph. D. thesis, Ohio State University.

King, P. J. (1989). *A logical Formalism for Head-Driven Phrase Structure Grammar*. Ph. D. thesis, University of Manchester.

King, P. J. (1994). An expanded logical formalism for Head-Driven Phrase Structure Grammar. Arbeitspapiere des SFB 340 59, Universität Tübingen.

King, P. J. (1995). From unification to constraint. Unpublished lecture notes, Universität Tübingen.

King, P. J. (1999). Towards Truth in Head-driven Phrase Structure Grammar. This volume.

Meurers, W. D. (1998). Raising spirits (and assigning them case). Draft of a paper presented at the *ESSLLI-98* Workshop on Current Topics in Constraint-Based Theories of Germanic Syntax. Available via http://www.sfs.nphil.uni-tuebingen.de/~dm/papers/raising-spirits.html.

Penn, G. (1999). Linearization and *wh*-extraction in HPSG: Evidence from Serbo-Croatian. In R. D. Borsley and A. Przepiórkowski (Eds.), *Slavic in HPSG*, pp. 149–182. CSLI Publications. In press.

Pollard, C. and I. A. Sag (1994). *Head-Driven Phrase Structure Grammar*. University of Chicago Press.

Pollard, C. J. (1999). Strong generative capacity in HPSG. In G. Webelhuth, J.-P. Koenig, and A. Kathol (Eds.), *Lexical and Constructional Aspects of Linguistic Explanation*, Chapter 18, pp. 281–297. CSLI Publications.

Przepiórkowski, A. (1997). Quantifiers, adjuncts as complements, and scope ambiguities. To appear in *Journal of Linguistics*. Version of December 2, 1997.

Przepiórkowski, A. (1998). 'A Unified Theory of Scope' revisited: Quantifier retrieval without spurious ambiguities. In G. Bouma, G.-J. M. Kruijff, and R. T. Oehrle (Eds.), *Proceedings of the FHCG-98, 14–16 August 1998, Saarbrücken*, pp. 185–195.

Richter, F. (1997). Die Satzstruktur des Deutschen und die Behandlung langer Abhängigkeiten in einer Linearisierungsgrammatik. Formale Grundlagen und Implementierung in einem HPSG-Fragment. In E. Hinrichs, W. D. Meurers, F. Richter, M. Sailer, and H. Winhart (Eds.), *Ein HPSG-Fragment des Deutschen. Teil 1: Theorie*, Volume 95 of *Arbeitspapiere des SFB 340*, pp. 13–188.

Richter, F. (in prep.). *A Mathematical Formalism for Linguistic Theories with an Application in Head-Driven Phrase Structure Grammar and a Fragment of German*. Ph. D. thesis, Universität Tübingen. Working Title. Version: November 5th, 1998.

Richter, F. and P. J. King (1997). On the Existence of Exhaustive Models in a Relational Feature Logic for Head-driven Phrase Structure Grammar. Manuscript. Available via http://www.sfs.nphil.uni-tuebingen.de/~fr/current/.

Richter, F. and M. Sailer (1998). Complementizers and finite verbs in German sentence structure. In T. Kiss and D. Meurers (Eds.), *Proceedings of the ESSLLI-98 Workshop on Current Topics in Constraint-Based Theories of Germanic Syntax*, pp. 133–148.

Richter, F. and M. Sailer (1999). LF Conditions on Expressions of Ty2: An HPSG Analysis of Negative Concord in Polish. In R. D. Borsley and A. Przepiórkowski (Eds.), *Slavic in HPSG*, pp. 247–282. CSLI Publications. In press.

Richter, F., M. Sailer, and G. Penn (1999). A formal interpretation of relations and quantification in HPSG. This volume.

Universität Tübingen, Seminar für Sprachwissenschaft, Abteilung Computerlinguistik
E-mail address: `fr@sfs.nphil.uni-tuebingen.de`