Chapter 3

Formal background

Frank Richter
Goethe Universität Frankfurt

This chapter provides a very condensed introduction to a formalism for Pollard & Sag (1994) and explains its fundamental concepts. It pays special attention to the model-theoretic meaning of HPSG grammars. In addition, it points out some links to other, related formalisms, such as feature logics of partial information, and to related terminology in the context of grammar implementation platforms.

1 Introduction

The two HPSG books by Pollard and Sag (Pollard & Sag 1987; 1994) do not present grammar formalisms with the intention to provide precise definitions. Instead they refer to various inspirations in the logics of typed feature structures or in predicate logic, informally characterize the intended formalisms, and explain them as they are used in concrete grammars of English. Pollard & Sag (1994) further clarify their intentions in an appendix which lists most (but not all) of the components of their grammar of English explicitly, and summarizes most of their core assumptions. With this strategy, both books leave room for interpretation.

There are a number of challenges with reviewing the formal background of HPSG. Some of them have to do with the long publication history of relevant papers and books, some with the considerable influence of grammar implementation platforms, which have their own formalisms and shape the way in which linguists think and talk about grammars with their platform-specific terminology and notational conventions. Salient examples include convenient notations for phrase structure rules, the treatment of lexical representations or the lexicon, mechanisms for lexical rules, and notations for default values, among many

Frank Richter. 2020. Formal background. In Stefan Müller, Anne Abeillé, Robert D. Borsley & Jean-Pierre Koenig (eds.), *Head-Driven Phrase Structure Grammar: The handbook*, 45–80. Berlin: Language Science Press. DOI:??

other devices. Many of these notations are well-known in the HPSG community; they are convenient, compact and arguably even necessary to write readable grammars. At the same time, they are a meta-notation in the sense that they do not (directly) belong to the syntax of the assumed feature logics. However, even if they are outside a declarative, logical formalism for HPSG, there is usually a way to interpret them in HPSG-compatible formalisms, but the necessary re-interpretation can deviate to a larger or lesser degree from what their users have in mind when they write their grammars. For example, a phrase structure rule in the sense of a context-free or context-sensitive rewrite system is not the same as an ID Schema written in a feature logic, which might matter in some cases but not in others. To name one difference, an ID Schema may easily leave the number of a phrase's daughters unspecified (and thus potentially infinite). The differences may be sometimes subtle and sometimes significant, but they entail that the meaning of the notations seen through the lens of logic is not what their users might assume either based on their meaning in other contexts or on what is gleaned from the behavior of a given implementation platform for parsing or generation which employs that kind of syntax. Similarly, terminology that belongs to the computational environment of implementations is often transferred to grammar theory, and again, when checking the technical specifics, a re-interpretation in terms of a feature-logical HPSG formalism can sometimes be trivial and sometimes nearly impossible, and different available re-interpretation choices lead to significantly different results.

Reviewing HPSG's formal background, it is not only the multi-purpose character and flexibility of the ubiquitous informal attribute-value matrix (AVM) notation and its practical notational enhancements (for lexical rules, decorated sort hierarchies, phrase structure trees, etc.) that one needs to be aware of, but also early changes in foundational assumptions and terminology. When first presented in a book in 1987, HPSG was conceived of as a *unification-based* grammar theory, a name, the authors explain, which "arises from the algebra that governs partial information structures" (Pollard & Sag 1987: 7). This algebra was populated by partial feature structures with unification as a fundamental algebraic operation. In the framework envisioned seven years later in Pollard & Sag (1994), that algebra did not exist anymore, feature structures were no longer partial but total objects in models of a logical theory, and unification was no longer defined in the new setting (as the relevant algebra was gone). However, most of the notation and considerable portions of the terminology of 1987 remain with us to this day, such as the *types* of feature structures (the term that replaces the *sorts* of 1994, when the term *type* was used for a different concept, to be discussed below), the

pieces of information (for 1987-style feature structures) or even the word *unification*, which took on a casual life of its own without the original algebra in which it had been defined. Occasionally these words still have a precise technical interpretation in the language of grammar implementation environments or in their run-time system, which may reinforce their use in the community despite their lack of meaning in the standard formalism of HPSG. Implementation platforms also often add their own technical and notational devices, thereby inviting linguists to import them as useful tools into their theoretical grammar writing.

This handbook article cannot disentangle the history of and relationships between the various formalisms leading to an explication of the 1994 version of HPSG, nor of those that existed and still exist in parallel. It sets out to clarify the terminology and structure of a formalism for Pollard & Sag (1994) and presents a canonical formalism of the final version of HPSG in Pollard & Sag (1994). Only occasionally will it point out some of the differences to its 1987 precursor where the older terminology is still present in current HPSG papers and may be confusing to an audience unaware of the different usages of terms. Similarly, it does not cover SBCG (Sign-Based Construction Grammar, Müller (2020), Chapter 34 of this volume).

The main sources of the present summary are the construction of a model theory for HPSG by King (1999) and Pollard (1999), and their synoptic reconstruction on the basis of a comprehensive logical language for HPSG, *Relational Speciate Re-entrant Language* (RSRL) by Richter (2004), including the critique and extensions sketched in Richter (2007). Section 2 gives a largely non-technical introductory overview which should provide sufficient background to follow all linguistic chapters of the present handbook. The subsequent sections (3–6) are for readers keen on obtaining a deeper understanding or looking for clarification of what might remain vague and imprecise in an initial broad overview. Those sections might be more challenging for the casual reader, but in return offer a fairly self-contained and comprehensive summary, omitting only the mathematical groundwork and definitions needed to spell out alternative model-theories, as this goes beyond what can reasonably be compressed to handbook format.

2 Essentials: An informal overview

This section presents an informal summary of the essentials of an HPSG formalism in the sense of Pollard & Sag (1994) as it emerged from their original outline and its subsequent elaboration. From here on, the term "HPSG formalism" always refers to this tradition, unless explicitly stated otherwise. All later sections

in this chapter will flesh out the basic ideas introduced here with a precise technical treatment of the relevant notions. Readers who are already familiar with feature logics and are specifically interested in technical details may want to skip ahead to Section 3.

At the heart of HPSG is a fundamental distinction between descriptions and described objects: a grammar avails itself of descriptions with the purpose of describing linguistic objects. Pollard & Sag (1994: 17–18, 396) commit to the ontological assumption that linguistic objects only exist as complete objects. Partial linguistic objects do not exist. Descriptions of linguistic objects, however, are typically *partial*, i.e. they do not mention many, or even most, properties of the objects in their denotation. They are *underspecified*. A word can be described as being nominal and plural, leaving all its other properties (gender, case, number and category of its arguments, etc.) unspecified. But any concrete word being so described will have all other properties that a plural noun can have, with none of them missing. A single underspecified description can therefore describe many distinct linguistic objects. Grammatical descriptions often describe an infinity of objects. Again considering plural nouns, English can be thought of as having a very large number or an infinity of them due to morphological processes such as compounding, depending on the choice of morphological analysis.

Descriptions are couched in a (language of a) feature logic rather than in English for precision. Linguistic objects as the subject of linguistic study are sharply distinguished from their logical descriptions and are entities in the denotation of the grammatical descriptions. The feature logic of HPSG can be seen as a particularly expressive variant of description logics. With this architecture, HPSG is a *model-theoretic* grammar framework as opposed to *generative-enumerative* grammar frameworks, which have rewrite systems that generate expressions from some start symbol(s) (Pullum & Scholz 2001).

A small digression might be in order to prevent confusion arising from the coexistence of different versions of feature logics. Varieties of HPSG more closely related to the tradition of Pollard & Sag (1987) do not make the same distinction between descriptions and described objects. Instead they employ a notion of *feature structures* as entities carrying *partial information*. These partial feature structures are, or correspond to, logical expressions in a certain normal form and are ordered in an algebra of partial information according to the amount of information they carry. In informal notation, they are written as AVMs just like the descriptions of the formalism we are presently concerned with, and this notational similarity contributes to obscuring substantial differences. When two partial feature structures carry compatible information, they are said to be unifiable.

Their unification returns a unique third feature structure in the given algebra that carries the more specific information that is obtained when combining the previous two pieces of information (supposing they were not the same to begin with). These ideas and the properties of algebras employed by feature logics of partial information are still essential for all current HPSG implementation platforms (see Bender & Emerson (2020), Chapter 26 of this volume), which is presumably one of the reasons why the terminology of unification and unification-based grammars is still popular in the HPSG community. Returning to Pollard & Sag (1994), in a certain informal and casual sense, combining two non-contradictory descriptions into one single bigger description by logical conjunction could be called – and often is called – their unification. However, since the logical descriptions of HPSG in the tradition of Pollard & Sag (1994) can no longer be arranged in an appropriate algebra, there is no technical interpretation of the term in this context.[1]

HPSG employs partial descriptions in all areas of grammar, comprising at least syntax, semantics (Koenig & Richter 2020, Chapter 23 of this volume), pragmatics (Lücking, Ginzburg & Cooper 2020, Chapter 27 of this volume) and morphology (Crysmann 2020, Chapter 22 of this volume). The descriptions are normally notated as AVMs and contain sort symbols (by convention in italics with lower case letters) and attribute symbols (in small caps). These are augmented by the standard logical connectives (conjunction, disjunction, negation and implication) and relation symbols. So-called tags, boxed numbers, function as variables. (1) shows a typical example in which *word*, *noun* and *plural* are sorts and SYNSEM, LOCAL, CATEGORY, etc. are attributes.[2] The AVM is a description of plural nouns.

(1)
$$\begin{bmatrix} \textit{word} \\ \textsc{synsem}|\textsc{local} \begin{bmatrix} \textsc{category} \begin{bmatrix} \textsc{head} & \textit{noun} \end{bmatrix} \\ \textsc{content}|\textsc{index}|\textsc{number} & \textit{plural} \end{bmatrix} \end{bmatrix}$$

A description such as (1) presupposes a declaration of the admissible nonlogical symbols: as in any formal logical theory, the vocabulary of the formal language in which the logical theory is written must be explicitly introduced as the *alphabet* of the language, together with a set of logical symbols. This means that the sorts, attributes and relation symbols must be listed. HPSG goes beyond

[1] This state of affairs is also responsible for the fact that implementation platforms often provide only a restricted syntax of descriptions and may also supply additional syntactic constructs which extend their logic of partial information toward the expressiveness of a feature logic with classical interpretation of negation and relational expressions.

[2] Tags, relations and logical connectives in descriptions will be illustrated later, in (3).

merely stating the nonlogical vocabulary as sets of symbols by imposing additional structure on the set of sorts and on the relationship between sorts and attributes. This additional structure is known as the *sort hierarchy* and the *feature* (*appropriateness*) *declarations*.

The sort hierarchy and the feature declarations essentially provide the space of possible structures of the linguistic universe that an HPSG grammar talks about with its grammar principles. Metaphorically speaking, they generate a space of possible structures which is then constrained to the actual, well-formed structures which a linguist deems the grammatical structures of a language. The interaction between sort hierarchy and feature declarations is regulated by assumptions about feature inheritance and feature value inheritance. This can best be explained with a small example, using the tiny (and slightly modified) fragment from the sort hierarchy and feature appropriateness of Pollard & Sag (1994) shown in Figure 1.

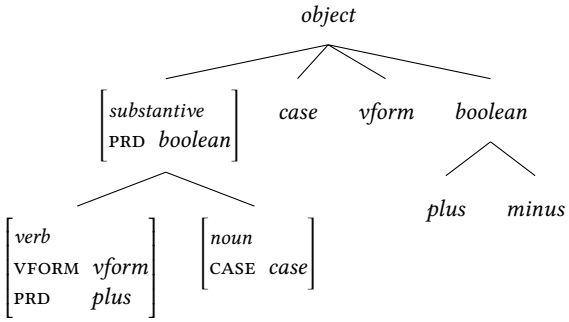

Figure 1: Example of sort hierarchy with feature declarations

According to Figure 1, a top sort *object* is the highest sort with immediate subsorts *substantive*, *case*, *vform* and *boolean*. The two sorts *substantive* and *boolean* have their own immediate subsorts: *verb* and *noun*, and *plus* and *minus*, respectively. All are subsorts of *object*. The six sorts *verb*, *noun*, *case*, *vform*, *plus* and *minus* are *maximally specific* in this hierarchy, because they do not have proper subsorts. Such sorts are called *species*. The four sorts *case*, *vform*, *plus* and *minus* are also called *atomic*, because they are species and they do not have attributes appropriate to them.

Figure 1 contains nontrivial feature declarations for the sorts *substantive*, *verb* and *noun*, and it also illustrates the idea behind feature inheritance. First of all, *verb* and *noun* have attributes which are only appropriate to them but to no other sort: VFORM is only appropriate to *verb*, and CASE is only appropriate to *noun*.

But there is one more attribute appropriate to both due to feature inheritance: the attribute PRD is declared appropriate to *substantive*, and appropriateness declarations are inherited by subsorts, so PRD is also appropriate to *verb* and *noun*. The sort *noun* inherits the declaration unchanged from *substantive*.

Finally, we have to consider attribute values and their inheritance mechanism. Whereas attributes are called appropriate *to* a sort, I call a sort appropriate *for* an attribute at a given sort when talking about attribute values. For example, the non-maximal sort *boolean* is declared appropriate for the attribute PRD at *substantive*. This value declaration is also inherited by the subsorts, with a slight twist to it: at any subsort, the value for an attribute can become *more specific* (but not less specific) than at its supersort(s), and this is what happens here at the subsort *verb* of *substantive*. At *verb* the value of PRD must be one particular subsort of *boolean*, namely *plus*.[3]

A further crucial aspect of the sort hierarchy and the feature declarations is their significance for the meaning of grammars. Structures in the denotation of a grammar must fulfill all their combined restrictions plus the constraints imposed by all grammar principles. Every denoted object must be of a maximally specific sort, i.e. Figure 1 allows only objects of the six species in the hierarchy. In addition, all attributes declared appropriate for a species (possibly by inheritance) must be present on objects of that species, with the values of course also obeying the feature declarations and being maximally specific. For example, an object of sort *noun* has CASE and PRD properties. The object that is the CASE value must be of sort *case* (because *case* is a species in the present example, unlike in real grammars where *case* has subsorts), and the sort of the PRD value must be either *plus* or *minus*, one of the two species which are maximally specific subsorts of *boolean*. With these restrictions, specifications like in Figure 1 determine the ontology of possible structures in the denotation of a grammar. The possible structures are further narrowed down by the grammar principles, leaving the well-formed structures as the predictions of a grammar.

This is a good opportunity to reconsider underspecified descriptions. With the sort hierarchy and feature declarations of Figure 1, there are a number of ways to underspecify the description of structures of sort *noun*. All following AVMs describe the same structures but differ in their degree of explicitness:

(2) a. $\begin{bmatrix} \textit{noun} & \\ \textsc{case} & \textit{case} \\ \textsc{prd} & \textit{plus} \vee \textit{minus} \end{bmatrix}$

[3] The *plus* value for PRD at verbs is introduced here to create a useful example; it is not usually found in grammars.

b. $\left[\textit{noun}\right]$

c. $\left[\begin{array}{ll}\textit{noun} & \\ \textsc{case} & \textit{object}\end{array}\right]$

d. $\left[\begin{array}{ll}\textit{object} & \\ \textsc{case} & \textit{object}\end{array}\right]$

e. $\left[\begin{array}{lll}\textit{noun} & & \\ \textsc{case} & \textit{case} & \\ \textsc{prd} & \textit{plus}\end{array}\right] \vee \left[\begin{array}{lll}\textit{noun} & & \\ \textsc{case} & \textit{case} & \\ \textsc{prd} & \textit{minus}\end{array}\right]$

All AVMs in (2) denote the same two configurations as the fully specific AVM description in (2a): two *noun* structures with the CASE property *case* and the PRD property *plus* or the PRD value *minus*. But a description of these structures can be underspecified in many different ways. For *noun* structures in general (2b), the two just described are the only two structural choices, as can be verified by inspecting Figure 1. The description could mention in addition to what (2b) says that the structures have a CASE property, leaving its value underspecified (2c), but that does not make a difference with respect to the shape of the structures satisfying the description. Moreover, the only *object*s with CASE (2d) are nouns, but since that leaves exactly the two possible PRD values *plus* and *minus*, (2d) is yet another way to underspecify the two structures which (2a) describes exhaustively. Omitting the sort symbol *object* in the upper left-hand corner of (2d) would in fact be one more way to describe all nouns to the exclusion of everything else, because saying that something is an *object* does not restrict the range of choices. Finally, the disjunction embedded in the AVM in (2a) can be lifted to the top level of the description, yielding (2e).

Grammar principles are descriptions which every structure is supposed to obey, together with all its substructures. The Head Feature Principle, shown in (3a), is a frequent example. Every phrase whose syntax is a headed phrase (*hd-ph*) is such that its HEAD value equals the HEAD value of its head daughter, indicated by the repeated occurrence of tag $\boxed{1}$ as the value of the two HEAD features. Every structure which is described by the AVM to the left of the implication symbol (in this case simply a sort, but see (3d)) must also fulfill the requirements in the AVM to its right. If something is not a *hd-ph*, it is not restricted by the Head Feature Principle because it is not described by the antecedent of the principle. At the same time, a structure which is not described by *hd-ph* still satisfies the Head Feature Principle as an implicational statement. For example, the SYNSEM value of each phrase is usually assumed to be an object of sort *synsem*, i.e. it is not a phrase of sort *hd-ph*. As a *synsem* object, it is not described by the antecedent of

(3a), thereby still fulfilling the principle. In classical logic, $A \rightarrow B$ is equivalent to $\neg A \vee B$, so something that is not an A satisfies $\neg A \vee B$. This is highly relevant for the ultimate idea that a structure is only licensed by an HPSG grammar when it is well-formed in all its components with respect to all the grammar principles: every component of each structure that is described by the antecedent of a grammar principle also obeys what the consequent of the principle requires, or a given component of the structure is licensed by not being described by the antecedent of the given principle.

The tag [1] signals the identity of the value found at the end of the two distinct attribute paths leading to its occurrences. This state of affairs is often referred to as *token identity*. In the Head Feature Principle, the tag notation could be an informal notation for a *path equation*, or it could mean that [1] plays the role of a variable. The description language of Sections 3–4 offers both options for rendering such occurrences of tags in the syntax of RSRL.

(3) a. *hd-ph* $\Rightarrow$ $\begin{bmatrix} \textsc{synsem|local|category|head} & \boxed{1} \\ \textsc{head-dtr|synsem|local|category|head} & \boxed{1} \end{bmatrix}$

b. *word* $\Rightarrow$ $(\mathrm{LE}_1 \vee \mathrm{LE}_2 \vee \ldots \vee \mathrm{LE}_n)$

c. *sign* $\Rightarrow$ $\left[\begin{array}{ll} \textsc{synsem|loc} & \left[\begin{array}{ll} \textsc{qstore} & \boxed{1} \\ \textsc{pool} & \boxed{2} \end{array}\right] \\ \textsc{retrieved} & \boxed{3} \end{array}\right]$
$\wedge$ `set-of-elements` $(\boxed{3},\boxed{4}) \wedge \boxed{4} \subseteq \boxed{2} \wedge \boxed{1} = \boxed{2} - \boxed{4}$

d. $\left[\textsc{retrieved}\ \textit{nelist}\right] \Rightarrow \left[\textsc{synsem|local|content}\ \textit{psoa}\right]$

The licensing of words by the grammar can also be understood as a consequence of a grammar principle with the shape of an implication. (3b) is known as the Word Principle (Höhle 2018: 500). LE_1 to LE_n in (3b) are the *lexical entries* of the grammar, descriptions of words. If an object is a word, it must be described by (at least) one of the disjuncts in the consequent of the Word Principle.

The semantic principle in (3c), taken from Pollard & Yoo (1998: 420),[4] illustrates one more syntactic construct of HPSG's description language, relations. The consequent of the principle consists of an AVM description conjoined with three relational expressions. Relations in HPSG often occur in connection with lists and sets, and so do the relations here: the binary relation `set-of-elements` relates the RETRIEVED value (a list) to a set [4] containing the elements on list [3] such that the set value [2] of POOL is a superset of [4] (using the subset relation),

[4] This principle is also discussed in the semantics chapter, Koenig & Richter (2020: example (10)), Chapter 23 of this volume.

and the set value [1] of QSTORE contains those elements of [2] which are not on the RETRIEVED list (using set difference). In other words, each element of POOL is either in QSTORE or a list element on RETRIEVED, and nothing else is in QSTORE or on the RETRIEVED list.[5]

The grammar principle (3d), which is also from Pollard & Yoo (1998: 421), is a case of a principle with a complex description in the antecedent, unlike (3a)–(3c), in which the antecedent consists of a sort symbol. Any kind of description may serve as antecedent of a grammar principle.

An HPSG grammar is a signature consisting of a sort hierarchy, feature appropriateness declarations and relation symbols, together with a set of grammar principles. The meaning of the grammar is given by a class of structures (linguistic objects) which obey the structural restrictions of the signature and are completely well-formed with respect to the grammar principles. The nature of the *linguistic objects* and how the relevant models of an HPSG grammar should be conceived of has been subject to intense discussion. Pollard & Sag (1994: 8–9) think of them as *types* and want to construct them as a set of totally well-typed and sort-resolved abstract feature structures. Each such type is supposed to correspond to the set of token occurrences of the same utterance. For example, in this view, the English utterance *Breakfast is ready*, which may occur as a concrete utterance token at different places and at different times, always belongs to the unique type *Breakfast is ready*, rendered as an abstract feature structure licensed by an HPSG grammar of English.

All HPSG model theories after Pollard & Sag (1994) give up the idea of postulating types as objects in the intended grammar model and do not construct models which are populated with feature structures.[6] King (1999) suggests *exhaustive models*, collections of possible language tokens. Whereas two types are always distinct, linguistic tokens in exhaustive models can be isomorphic when they are different token occurrences of the same utterance. Pollard (1999) rejects the idea that models contain possible tokens and essentially uses a variant of

[5] One additional interesting property of this principle concerns the set designated by tag [4]. Structures described by the consequent of the principle do not necessarily contain an attribute with the set value [4]. However, the list [3] and the sets [1] and [2] are all attribute values which are restricted in (3c) by reference to set [4]. Such constellations motivate the introduction of *chains* in the description language. Chains model lists (or sets) of objects that are not themselves attribute values, but whose members are (see Section 3 for the syntax and Section 4 for the semantics of chains). [4] is best described as a chain.

[6] Pollard (1999: 294) still uses the term *feature structure*, but it is applied to a special kind of *interpretation* in the sense of Definition 7. See the more detailed characterization of these structures in Section 6 below.

King's exhaustive models for constructing sets of unique mathematical idealizations of linguistic utterances: any well-formed utterance finds its structurally isomorphic unique counterpart in this model, called the *strong generative capacity* of the grammar. The relationship between the elements of the strong generative capacity and empirical linguistic events is much tighter than it is for Pollard and Sag's object types: for the former, it is a relationship of structural isomorphism, for the latter it is only a conventional notion of correspondence. Moreover, Pollard's models avoid an ontological commitment to the reality of types. Richter (2007) points out shortcomings with the postulated one-to-one correspondence between linguistic types (Pollard & Sag 1994) or mathematical idealizations (Pollard 1999) and the groups of linguistically indistinguishable utterances they are supposed to represent (e.g. the group of realizations of *Breakfast is ready*). The failure of achieving the intended one-to-one correspondence is due to technical properties of the structure of the respective models and to imprecisions of actual HPSG grammar specifications, and the two factors are partially independent. Richter (2007) suggests schematic amendments to grammars (by a small set of axioms and an extended signature), leading to *normal form grammars* whose *minimal exhaustive models* exhibit the intended one-to-one correspondence between structural configurations in the model and (groups of linguistically indistinguishable) empirically observable utterance events. Despite being a certain kind of exhaustive model, minimal exhaustive models are not token models and do not suffer from the problematic concept of *potential token* models which is characteristic of King's approach.

HPSG as a model-theoretic grammar framework provides linguists with an expressive class of logical description languages. Their semantics makes it possible to investigate closely the predictions of a given set of grammar principles and the internal and mutual consistency of different modules of grammar. At a more foundational level, HPSG is exceptional with its alternative characterizations of the meaning of grammars based on one and the same set of core definitions of the syntax and semantics of its descriptive devices. This common core in the service of philosophically different approaches to the scientific description of human languages makes their respective advantages and disadvantages comparable within one single framework, and it renders the discussion of very abstract concepts from the philosophy of science unusually concrete. Alternative approaches to grammatical meaning based on different views of the nature of scientific description of an empirical domain can be investigated and compared with a degree of detail that is hardly achieved elsewhere in linguistics.

The structure of the remainder of this chapter is as follows: Section 3 turns to

the syntax of RSRL, defines signatures with sort hierarchies and feature appropriateness for the non-logical vocabulary, and introduces terms and formulæ as expressions. A subclass of formulæ is called descriptions and corresponds to the informal AVMs augmented with logical connectives and relational expressions which we saw above in (1)–(3). Section 4 furnishes the syntactic expressions with a semantics similar to what is familiar from classical logic, except that formulæ and descriptions denote sets of objects rather than truth values. Section 5 turns to the meaning of grammars, taking King's exhaustive models as a concrete example of the four explications outlined above, since it is technically the easiest to define. The final section (Section 6) outlines how the other three approaches to the meaning of HPSG grammars differ from King's possible token models without fully defining all constructs they involve.

The function of Sections 3–6 is thus to spell out in more depth what the present section summarized in much broader strokes. Readers who do not wish to pursue HPSG's formal foundations further can stop here without missing anything fundamentally new.

3 Signatures and descriptions

As logical theories of entities in a domain of objects, HPSG grammars consist of two main components. First, a logical signature, which provides the symbols for describing the domain of interest, in this case a natural language. And second, an exact delineation of all and only the legitimate entities in the denotation of the grammar, written as a collection of statements about their configuration. These statements are descriptions within a logical language and are composed from logical constants, variables, quantifiers, brackets and the symbols provided by the signature. They are variously known to linguists as principles of grammar, constraints, or rules. In the following, I will use the term *principles* to designate these statements. Linguists often use abbreviatory conventions for conceptually distinguished groups of principles, such as grammar rules, lexical entries, or lexical rules. From a logical perspective, then, a grammar is a pair consisting of a signature and a collection of principles. The appendix of Pollard & Sag (1994) provides an early example in HPSG of this conception.

Signatures in HPSG go beyond supplying non-logical symbols for descriptions, they impose additional restrictions on the organization of the non-logical symbols. These restrictions ultimately have an effect on how the domain of described objects is structured. Let us first investigate the two most prominent sets of non-logical symbols: sorts and attributes. The set of *sort* symbols is arranged in a *sort*

hierarchy, and that sort hierarchy is in turn connected to the set of *attribute* symbols (also known as *features*). The sort hierarchy is a partial order,[7] and attributes are declared *appropriate to* sorts in the sort hierarchy. This appropriateness declaration must not be entirely random: if an attribute is declared appropriate to some sort, it must also be declared appropriate to all its subsorts. This requirement is known as *feature inheritance*.[8] Moreover, for each sort σ and attribute ϕ such that ϕ is appropriate to σ, some other sort σ' is *appropriate for* ϕ at σ. In other words, a certain attribute value (σ') is declared appropriate for ϕ at σ. These attribute values must not be completely random either: for any subsort of σ, the value of an appropriate feature ϕ of σ is of course also appropriate to that subsort (by feature inheritance), but in addition, the value of ϕ at that subsort must be at least as specific as it is at σ. This means the value is either σ' or a subsort thereof. It may not be less specific, or, to put it differently, it may not be a supersort of σ'.

Some sorts in the sort hierarchy enjoy a special status by being *maximally specific*. They are called *species*. Species are sorts without proper subsorts. Sorts that are maximally specific and lack any appropriate attribute receive a special name and are called *atomic* sorts or simply *atoms*.

In addition to sorts and attributes, a signature provides relation symbols. Well-known examples are a ternary `append` relation and a binary `member` relation, but grammars may also require relations such as (often ternary) `shuffle` and binary `o-command`. Each relation symbol comes with a positive natural number for the number of arguments, its *arity*.

Putting all of this together, we obtain a definition of signatures as a septuple with sort hierarchy $\langle S, \sqsubseteq \rangle$, species S_{max}, attributes A, and relation symbols R; the function F handles the feature appropriateness and function Ar is for the number of arguments of each relation.

Definition 1 Σ *is a* signature *iff*
Σ *is a septuple* $\langle S, \sqsubseteq, S_{max}, A, F, R, Ar \rangle$,
$\langle S, \sqsubseteq \rangle$ *is a partial order,*
$S_{max} = \{\sigma \in S \mid \text{for each } \sigma' \in S, \text{if } \sigma' \sqsubseteq \sigma \text{ then } \sigma = \sigma'\}$,
A *is a set,*
F *is a partial function from* $S \times A$ *to* S,

[7] A partial order is given by a set whose elements stand in a reflexive, antisymmetric and transitive ordering relation.

[8] See Figure 1 and its explanation in Section 2 for an example which also points out the subtle distinction between the use of the term *appropriate to* (feature to sort) vs. the term *appropriate for* (sort value for a feature at a given sort).

for each $\sigma_1 \in S$, *for each* $\sigma_2 \in S$, *for each* $\phi \in A$,
if $F(\sigma_1, \phi)$ *is defined and* $\sigma_2 \sqsubseteq \sigma_1$
then $F(\sigma_2, \phi)$ *is defined and* $F(\sigma_2, \phi) \sqsubseteq F(\sigma_1, \phi)$,
R *is a finite set, and*
Ar *is a total function from* R *to the positive integers.*

The partial order $\langle S, \sqsubseteq \rangle$ is the sort hierarchy, and the set of sorts S, just like the set of attributes A, can in principle be infinite. In actual grammars it is finite, and in HPSG grammars it is also assumed that S contains a top element, which is a sort that subsumes all other sorts in the sort hierarchy. S_{max} is the set of maximally specific sorts, which will play a prominent role in the semantics of descriptions. F is a function for fixing the appropriateness conditions on attributes and attribute values, and the conditions on that function reflect HPSG's restrictions on feature declarations. F is called the *(feature) appropriateness function*. The last two lines of the definition provide the set of relation symbols, R, with their arity, Ar. We assume that relations are at least unary.

Relations in HPSG often express relationships between lists (`append`, `shuffle`) or sets (`union`, `intersection`). Lists are usually encoded in HPSG with attributes FIRST and REST, and sorts *list*, *elist* (for empty list) and *nelist* (for non-empty list), but of course the exact naming does not matter. A fragment of the sort hierarchy which declares the sorts and attributes for regular lists is shown in Figure 2.

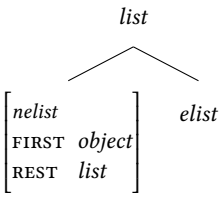

Figure 2: Fragment of a sort hierarchy for encoding lists

An AVM description of a list with two *synsem* objects can then be notated as in example (4a). Of course, grammar writers usually abbreviate list descriptions in AVMs by a syntax with angled brackets for superior readability, as shown in (4b), a more transparent rendering of (4a), but that is just a convention that presupposes the existence of a sort hierarchy fragment like in Figure 2.

(4) a.
$$\begin{bmatrix} \textit{nelist} & & \\ \textsc{first} & \textit{synsem} & \\ & \begin{bmatrix} \textit{nelist} & & \\ \textsc{first} & \textit{synsem} \\ \textsc{rest} & \textit{elist} \end{bmatrix} \\ \textsc{rest} \end{bmatrix}$$

b. $\left\langle \left[\textit{synsem}\right], \left[\textit{synsem}\right] \right\rangle$

In combination with relations, grammarians occasionally require a more generalized use of lists (and sets) than their basic encoding above supports. Starting already with Pollard & Sag (1994), we find structures in arguments of relations which behave like regular lists or sets, except that they do not occur as attribute values anywhere in the structures in which the relations are supposed to hold.[9] In order to account for these applications of lists and sets in arguments of relations, RSRL introduces *chains*. Chains are handled with dedicated sorts and attributes with a fixed interpretation that extend every signature. They can be thought of as a more flexible treatment of lists alongside their regular explicit encoding in HPSG.

RSRL adds chains to all signatures. Informally, the extra symbols act very much like sorts and attributes for lists: *chain* for *list*, *echain* and *nechain* for *elist* and *nelist*, respectively, and the reserved symbols † and ▹ for FIRST and REST. In order to integrate the reserved new sort symbols with any signature a linguist might specify, a distinguished sort *metatop* serves as unique top element of the extended sort hierarchy. The extensions are defined for any signature by adding reserved *pseudo-sorts* and *pseudo-attributes* and structuring the expanded sort hierarchy in the desired way:

Definition 2 *For each signature* $\Sigma = \langle S, \sqsubseteq, S_{max}, A, F, R, Ar \rangle$,
$\widehat{S} = S \cup \{chain, echain, nechain, metatop\}$,
$\widehat{\sqsubseteq} = \sqsubseteq \cup \{\langle echain, chain\rangle, \langle nechain, chain\rangle\} \cup \left\{\langle \sigma, \sigma\rangle \,|\sigma \in \widehat{S}\backslash S\right\}$
$\quad \cup \left\{\left\langle \sigma, metatop\right\rangle |\sigma \in \widehat{S}\right\}$,
$\widehat{S}_{max} = S_{max} \cup \{echain, nechain\}$, *and*
$\widehat{A} = A \cup \{\dagger, \triangleright\}$.

The extended sort hierarchy relation, $\widehat{\sqsubseteq}$, simply integrates the new pseudo-sorts into the given relation by ordering *echain* and *nechain* under *chain*, keeping the reflexive closure intact and ordering every sort and pseudo-sort under the

[9] See (3c) above for an example in the second argument of a binary relation `set-of-elements`.

new top element of the partial order, *metatop*. Corresponding to *elist* and *nelist* above, *echain* and *nechain* are treated as maximally specific by including them in the extension of S_{max}, designated as $\widehat{S}_{max}$.[10] An AVM describing a chain with two *synsem* objects corresponding to the description of a list with two *synsem* objects in (4a) now appears as follows:

(5)
$$\begin{bmatrix} \textit{nechain} \\ \dagger\ \textit{synsem} \\ \triangleright \begin{bmatrix} \textit{nechain} \\ \dagger\ \textit{synsem} \\ \triangleright\ \textit{echain} \end{bmatrix} \end{bmatrix}$$

Apart from the non-logical constants from (expanded) signatures and some logical symbols, we also need a countably infinite set of variables, which will be symbolized by V. We use lower-case letters from the Latin alphabet as variable symbols, typically x.

For expository reasons, the syntax of descriptions, to be introduced next, does not employ AVMs, the common lingua franca of constraint-based grammar formalisms. The reasons are twofold: most importantly, although AVMs provide an extremely readable and flexible notation, they are quite cumbersome to define as a rigorous logical language which meets all the expressive needs of HPSG. Some of this awkwardness in explicit definitions derives from the very flexibility and redundancy in notation that makes AVMs perfect for everyday linguistic practice. Second, the original syntax of RSRL is, by contrast, easy to define, and, as long as it is not used for descriptions as complex as they occur in real grammars, its expressions are still transparent for everyone who is familiar with AVMs. Readers who want to explore how our description syntax relates to a formal syntax of AVMs are referred to Richter (2004) for details and a correspondence proof.

The definition of the syntax of descriptions proceeds in two steps, quite similar to first-order predicate logic. We will first introduce terms and then build formulæ and descriptions from terms. Terms are essentially what is known to linguists as *paths*, sequences of attributes:

Definition 3 *For each signature* $\Sigma = \langle S, \sqsubseteq, S_{max}, A, F, R, Ar\rangle$, T^Σ *is the smallest set such that*
$: \in T^\Sigma$,
for each $x \in V$, $x \in T^\Sigma$,
for each $\phi \in \widehat{A}$ *and each* $\tau \in T^\Sigma$, $\tau\phi \in T^\Sigma$.

[10] Extending the appropriateness function, F, is unnecessary since the relevant effects follow immediately from the semantics of the new reserved symbols in Definition 8.

Simply put, sequences of attributes (including the two pseudo-attributes † and ▹) starting either with the colon or a single variable are Σ terms. Equipped with terms, we can immediately proceed to formulæ, the penultimate step on the way to descriptions. There are three kinds of simple formulæ: formulæ that assign a sort to the value of a path, formulæ which state that two paths have the same value (*structure sharing*, in linguistic terminology), and relational formulæ. Complex formulæ can be built from these by existential and universal quantification, negation, and the classical binary logical connectives.

Definition 4 *For each signature* $\Sigma = \langle S, \sqsubseteq, S_{max}, A, F, R, Ar\rangle$, D^Σ *is the smallest set such that*
for each $\sigma \in \widehat{S}$, *for each* $\tau \in T^\Sigma$, $\tau \sim \sigma \in D^\Sigma$,
for each $\tau_1, \tau_2 \in T^\Sigma$, $\tau_1 \approx \tau_2 \in D^\Sigma$,
for each $\rho \in R$, *for each* $x_1, \ldots, x_{Ar(\rho)} \in V$, $\rho(x_1, \ldots, x_{Ar(\rho)}) \in D^\Sigma$,
for each $x \in V$, *for each* $\delta \in D^\Sigma$, $\exists x\delta \in D^\Sigma$, *(analogous for* $\forall$*)*
for each $\delta \in D^\Sigma$, $\neg\delta \in D^\Sigma$,
for each $\delta_1, \delta_2 \in D^\Sigma$, *and* $(\delta_1 \wedge \delta_2) \in D^\Sigma$. *(analogous for* $\vee, \rightarrow, \leftrightarrow$*)*

In this syntax, the Head Feature Principle of (3a) can be rendered as in (6a) or, equivalently, as in (6b).[11]

(6) a. $(: \sim$ *hd-ph*$) \rightarrow$
$(:$ SYNSEM LOCAL CATEGORY HEAD $\approx$
: HEAD-DTR SYNSEM LOCAL CATEGORY HEAD$)$

b. $(: \sim$ *hd-ph*$) \rightarrow$
$\exists x\,(:$ SYNSEM LOCAL CATEGORY HEAD $\approx x \wedge$
: HEAD-DTR SYNSEM LOCAL CATEGORY HEAD $\approx x)$

Finally, *FV* is a function that determines for every Σ term and Σ formula the set of variables that occur free in them.

Definition 5 *For each signature* $\Sigma = \langle S, \sqsubseteq, S_{max}, A, F, R, Ar\rangle$,
$FV(:) = \{\}$,
for each $x \in V$, $FV(x) = \{x\}$,
for each $\tau \in T^\Sigma$, *for each* $\phi \in \widehat{A}$, $FV(\tau\phi) = FV(\tau)$,
for each $\tau \in T^\Sigma$, *for each* $\sigma \in \widehat{S}$, $FV(\tau \sim \sigma) = FV(\tau)$,
for each $\tau_1, \tau_2 \in T^\Sigma$, $FV(\tau_1 \approx \tau_2) = FV(\tau_1) \cup FV(\tau_2)$,
for each $\rho \in R$, *for each* $x_1, \ldots, x_{Ar(\rho)} \in V$, $FV(\rho(x_1, \ldots, x_{Ar(\rho)})) = \{x_1, \ldots, x_{Ar(\rho)}\}$,

[11] The brackets in the antecedent are for readability.

for each $\delta \in D^\Sigma$, *for each* $x \in V$, $FV(\exists x\delta) = FV(\delta)\backslash\{x\}$, *(analogous for* $\forall$*)*
for each $\delta \in D^\Sigma$, $FV(\neg\delta) = FV(\delta)$,
for each $\delta_1, \delta_2 \in D^\Sigma$, $FV((\delta_1 \wedge \delta_2)) = FV(\delta_1) \cup FV(\delta_2)$. *(analogous for* $\vee, \rightarrow, \leftrightarrow$*)*

Informally, an occurrence of a variable is free in a Σ term or a Σ formula if it is not bound by a quantifier. We single out Σ formulæ without free occurrences of variables as a kind of formula of special interest and reserve the term Σ *description* for them:

Definition 6 *For each signature* Σ, $D_0^\Sigma = \left\{\delta \in D^\Sigma | FV(\delta) = \{\}\right\}$.

D_0^Σ is the set of Σ descriptions. When a signature is fixed by the context, or when the exact signature is irrelevant in the discussion, we can simply speak of *descriptions* instead of Σ descriptions. Descriptions are the syntactic units that linguists use in grammar writing. (6a) and (6b) are descriptions. Grammars, as we will see in Section 5, are written by declaring a signature and stating a set of descriptions. But before we can investigate grammars and what they mean, we have to explain the meaning of signatures and of descriptions.

4 Meaning of signatures and descriptions

Descriptions of RSRL are interpreted similarly to expressions of classical logics such as first order logic, except that they are not evaluated as true or false in a given structure; instead, they denote collections of structures.

Defining the meaning of descriptions begins with delineating the structures which interpret signatures. In particular, species and attributes must receive a meaning, which should be tied to the HPSG-specific intentions behind sort hierarchies and feature declarations; and so must relation symbols, whose interpretation should heed their arity. Due to some extra restrictions which we will ultimately want to put on the interpretation of relation symbols (to meet intuitions of grammarians) and whose formulation presupposes a notion of term interpretation, we start with *initial interpretations*. They will be refined in a second step to full interpretations (Definition 13).

Some additional notation is convenient in the upcoming definition of initial interpretations. If S is a set, S^* is the set of all finite sequences (or n-tuples) of elements of S. S^+ is the same set without the empty sequence. $\overline{S}$ is short for the set $S \cup S^*$. Initial interpretations employ a set U of entities which form the domain of grammars. The functions S, A and R interpret sort symbols, attribute symbols and relation symbols in that domain, respecting certain general restrictions

which come with HPSG's ontological assumptions about languages. In particular, the behavior of attribute interpretation is tied to the feature appropriateness conditions, i.e. feature inheritance in the sort hierarchy.

Definition 7 *For each signature* $\Sigma = \langle S, \sqsubseteq, S_{max}, A, F, R, Ar \rangle$, I *is an* initial Σ interpretation *iff*
$\mathsf{I} = \langle \mathsf{U}, \mathsf{S}, \mathsf{A}, \mathsf{R} \rangle$,
U *is a set,*
S *is a total function from* U *to* S_{max},
A *is a total function from* A *to the set of partial functions from* U *to* U,
for each $\phi \in A$ *and each* $u \in \mathsf{U}$
  if $\mathsf{A}(\phi)(u)$ *is defined*
  then $F(\mathsf{S}(u), \phi)$ *is defined, and* $\mathsf{S}(\mathsf{A}(\phi)(u)) \sqsubseteq F(\mathsf{S}(u), \phi)$, *and*
for each $\phi \in A$ *and each* $u \in \mathsf{U}$,
  if $F(\mathsf{S}(u), \phi)$ *is defined then* $\mathsf{A}(\phi)(u)$ *is defined,*
R *is a total function from* R *to the power set of* $\bigcup_{n \in \mathbb{N}} \overline{\mathsf{U}}^n$, *and*
  for each $\rho \in R$, $\mathsf{R}(\rho) \subseteq \overline{\mathsf{U}}^{Ar(\rho)}$.

Initial Σ interpretations are quadruples consisting of four components. The first three of them will remain unchanged in full Σ interpretations (Definition 13). The elements of U are entities which populate the universe of structures. Their ontological status has been debated fiercely in HPSG, and will be discussed in Sections 5 and 6. For the moment, assume that they are either linguistic objects or appropriate abstractions thereof. S assigns each object in the universe a species, which is another way of saying that each object is of exactly one maximally specific sort. This is what is known as the property of being *sort-resolved*. The attribute interpretation function A interprets each attribute symbol as a (partial) function that assigns an object of the universe to an object of the universe, and as such it obeys the restrictions of the feature declarations of the signature, embodied in the function F: attributes are defined on all and only those objects u_1 which have a species to which the attributes are appropriate according to F; and the object which u_1 is mapped to by the attribute must in turn be of a species which is appropriate for the attribute (at the species of u_1). This is what is known as the property of interpreting structures of being *totally well-typed*. Originally both of these properties of interpreting structures were formulated with respect to so-called *feature structures*, but, as we will see below, this conception of interpreting structures for grammars was soon given up for philosophical

reasons.[12] The relation interpretation function R finally interprets n-ary relation symbols as sets of n-tuples of objects. However, there is an additional option, which makes the definition look more complex: an object in an n-tuple may in fact not be an atomic object; it can alternatively be a tuple of objects itself. These tuples in argument positions of relations will be described as *chains* with the extra symbols pseudo-sort and pseudo-attributes, which were added to signatures in Definition 2 above. As pointed out there, chains are a construct which gives grammarians the flexibility to use (finite) lists in all the ways in which they are put in relations in actual HPSG grammars (see (3c) for an example).

Since chains are provided by an extension of the set of sort symbols and attributes (Definition 2), the interpretation of the additional symbols must be defined separately. This is very simple, since these symbols behave essentially analogously to the conventional sort and attribute symbols of HPSG's list encoding.

Definition 8 *For each signature* $\Sigma = \langle S, \sqsubseteq, S_{max}, A, F, R, Ar\rangle$*, for each initial* Σ *interpretation* $\mathsf{I} = \langle \mathsf{U}, \mathsf{S}, \mathsf{A}, \mathsf{R}\rangle$*,*
$\widehat{\mathsf{S}}$ *is the total function from* $\overline{\mathsf{U}}$ *to* $\widehat{S}$ *such that*
for each $u \in \mathsf{U}$, $\widehat{\mathsf{S}}(u) = \mathsf{S}(u)$,
for each $u_1, \ldots, u_n \in \mathsf{U}$, $\widehat{\mathsf{S}}\left(\langle u_1, \ldots, u_n\rangle\right) = \begin{cases} \mathit{echain} & \mathit{if}\ n = 0, \\ \mathit{nechain} & \mathit{if}\ n > 0 \end{cases}$*, and*
$\widehat{\mathsf{A}}$ *is the total function from* $\widehat{A}$ *to the set of partial functions from* $\overline{\mathsf{U}}$ *to* $\overline{\mathsf{U}}$ *such that*
for each $\phi \in A$, $\widehat{\mathsf{A}}\left(\phi\right) = \mathsf{A}\left(\phi\right)$,
$\widehat{\mathsf{A}}(\dagger)$ *is the total function from* U^+ *to* U *such that for each* $\langle u_0, \ldots, u_n\rangle \in \mathsf{U}^+$,
$\widehat{\mathsf{A}}(\dagger)\left(\langle u_0, \ldots, u_n\rangle\right) = u_0$*, and*
$\widehat{\mathsf{A}}(\triangleright)$ *is the total function from* U^+ *to* U^* *such that for each* $\langle u_0, \ldots, u_n\rangle \in \mathsf{U}^+$,
$\widehat{\mathsf{A}}(\triangleright)\left(\langle u_0, \ldots, u_n\rangle\right) = \langle u_1, \ldots, u_n\rangle$.

$\widehat{\mathsf{S}}$ is the *expanded species assignment function*, and $\widehat{\mathsf{A}}$ is the *expanded attribute interpretation function*. The pseudo-species symbols *echain* and *nechain* label empty chains and non-empty chains, respectively. Given a non-empty chain, the pseudo-attribute † picks out its first member, corresponding to the function of the FIRST attribute on non-empty lists. Conversely, $\triangleright$ cuts off the first element of a non-empty chain and returns the remainder of the chain, as does the standard attribute REST for lists.

In addition to attributes, terms may also contain variables (Definition 3). Term interpretation thus requires a notion of *variable assignments* in (initial) interpre-

[12] Of course, the informal term *feature structure* is still alive among linguists, and in a technical sense, feature structures are essential constructs for implementation platforms.

tations.

Definition 9 *For each signature* Σ*, for each initial* Σ *interpretation* $\mathsf{I} = \langle \mathsf{U}, \mathsf{S}, \mathsf{A}, \mathsf{R} \rangle$*,* $\mathsf{G}_\mathsf{I} = \overline{\mathsf{U}}^V$ *is the set of variable assignments in* I*.*

An element of G_I (the set of total functions from the set of variables to the set of objects and chains of objects of U) will be notated as g, following a convention frequently observed in predicate logic. With variable assignments in (initial) interpretations, variables denote objects in the universe U and chains of objects of the universe.

Terms map objects of the universe to objects (or chains of objects) of the universe as determined by a term interpretation function T_I^g:

Definition 10 *For each signature* $\Sigma = \langle S, \sqsubseteq, S_{max}, A, F, R, Ar \rangle$*, for each initial* Σ *interpretation* $\mathsf{I} = \langle \mathsf{U}, \mathsf{S}, \mathsf{A}, \mathsf{R} \rangle$*, for each* $g \in \mathsf{G}_\mathsf{I}$*,* T_I^g *is the total function from* T^Σ *to the smallest set of partial functions from* U *to* $\overline{\mathsf{U}}$ *such that for each* $u \in \mathsf{U}$*,*

$\mathsf{T}_\mathsf{I}^g(:)(u)$ *is defined and* $\mathsf{T}_\mathsf{I}^g(:)(u) = u$*,*

for each $x \in V$*,* $\mathsf{T}_\mathsf{I}^g(x)(u)$ *is defined and* $\mathsf{T}_\mathsf{I}^g(x)(u) = g(x)$*,*

for each $\tau \in T^\Sigma$*, for each* $\phi \in \widehat{A}$*,*

$\mathsf{T}_\mathsf{I}^g(\tau\phi)(u)$ *is defined iff* $\mathsf{T}_\mathsf{I}^g(\tau)(u)$ *is defined and* $\widehat{\mathsf{A}}(\phi)(\mathsf{T}_\mathsf{I}^g(\tau)(u))$ *is defined, and if* $\mathsf{T}_\mathsf{I}^g(\tau\phi)(u)$ *is defined then* $\mathsf{T}_\mathsf{I}^g(\tau\phi)(u) = \widehat{\mathsf{A}}(\phi)(\mathsf{T}_\mathsf{I}^g(\tau)(u))$*.*

T_I^g is called the *term interpretation function under* I *under* g*.* Σ terms either start with a variable or with the special symbol colon (':'). The colon denotes the identity function. Interpreted on any object, it returns that object. If a term τ starts with the colon, its term interpretation starts, so to speak, at the object u to which it is applied ($\mathsf{T}_\mathsf{I}^g(\tau)(u)$) and, if each subsequent attribute in τ is defined on the object to which the interpretation of the earlier attribute(s) took us, the term interpretation will yield the object reached by the last attribute. When a Σ term starts with a variable x, the given variable assignment g will determine the starting point of interpreting the sequence of attributes ($g(x)$). Of course, variables may be assigned chains of objects, in which case the symbols of the expanded attribute set can be used to navigate the elements of the chain.

The set of objects which are reachable from a single given object in an interpretation by following sequences of attribute interpretations is important for the way in which quantification is conceived of by grammarians. It also plays a role in thinking about which objects can in principle stand in a relation, and it is crucial for explicating different notions of the meaning of grammars. Definition 11 captures this notion, which we call the set of components of an object

in an (initial) interpretation. Note that all terms in Definition 11 start with the colon.

Definition 11 *For each signature* $\Sigma = \langle S, \sqsubseteq, S_{max}, A, F, R, Ar\rangle$*, for each initial* Σ *interpretation* $\mathsf{I} = \langle \mathsf{U}, \mathsf{S}, \mathsf{A}, \mathsf{R}\rangle$*, for each* $u \in \mathsf{U}$*,*

$$\mathsf{C}_\mathsf{I}^u = \left\{ u' \in \mathsf{U} \;\middle|\; \begin{array}{l} \text{for some } g \in \mathsf{G}_\mathsf{I}, \\ \text{for some } \pi \in A^*, \\ \quad \mathsf{T}_\mathsf{I}^g(:\pi)(u) \text{ is defined, and} \\ \quad u' = \mathsf{T}_\mathsf{I}^g(:\pi)(u) \end{array} \right\}.$$

C_I^u is the set of components of u in I. The purpose of C_I^u is to capture the set of all objects that are reachable from some object u in the universe by following a path of interpreted attributes. Thinking of these configurations as directed graphs, the set of components of u in I is the set of nodes that can be reached by following any sequence of vertices (in the direction of attribute interpretation) starting from u. This corresponds to how linguists normally conceive of the substructures of some structured object.[13] The set of components of objects is used in two ways in the definitions of full interpretations and description denotations: it restricts the set of objects that are permitted in relations, and it provides the domain of quantification in quantificational expressions of the logical language.

According to Definition 7 of initial interpretations, relation symbols are simply interpreted as tuples of objects (and chains of objects) in the universe of interpretation. However, HPSGians have a slightly more restricted notion of relations: for them, relations hold between objects that occur within a sign (or a similar kind of larger linguistic structure); they are not relations between objects that occur in separate (unconnected) signs. The following notion of *possible relation tuples in an interpretation* captures this intuition.

Definition 12 *For each signature* $\Sigma = \langle S, \sqsubseteq, S_{max}, A, F, R, Ar\rangle$*, for each initial* Σ *interpretation* $\mathsf{I} = \langle \mathsf{U}, \mathsf{S}, \mathsf{A}, \mathsf{R}\rangle$*,*

$$\mathsf{RT}_\mathsf{I} = \bigcup_{n \in \mathbb{N}} \left\{ \langle u_1, \ldots, u_n\rangle \in \overline{\mathsf{U}}^n \;\middle|\; \begin{array}{l} \text{for some } u \in \mathsf{U}, \\ \text{for each } i \in \mathbb{N}, 1 \leq i \leq n \\ \quad u_i \in \overline{\mathsf{C}_\mathsf{I}^u} \end{array} \right\}.$$

[13] Phrasing this more carefully, the object itself is not structured, but there is a structure generated by the object by following the vertices, or more technically, by the composition of functions which interpret attribute symbols.

RT_I is the set of possible relation tuples in I. Possible relation tuples in an initial interpretation are characterized by the existence of some object in the interpretation from which each object in a relation tuple can be reached by a sequence of attribute interpretations. In case an argument in a tuple is a chain, then the objects on the chain are thus restricted.

The notion of *full interpretations* integrates the restriction on possible relations, keeping everything else unchanged from initial interpretations:

Definition 13 *For each signature* $\Sigma = \langle S, \sqsubseteq, S_{max}, A, F, R, Ar\rangle$*, for each initial* Σ *interpretation* $\mathsf{I}' = \langle \mathsf{U}', \mathsf{S}', \mathsf{A}', \mathsf{R}'\rangle$*, for the set of possible relation tuples in* I'*,* $\mathsf{RT}_{\mathsf{I}'}$*,* $\mathsf{I} = \langle \mathsf{U}, \mathsf{S}, \mathsf{A}, \mathsf{R}\rangle$ *is a full* Σ *interpretation iff*
$\mathsf{U} = \mathsf{U}'$*,* $\mathsf{S} = \mathsf{S}'$*,* $\mathsf{A} = \mathsf{A}'$*, and* R *is a total function from* R *to the power set of* $\mathsf{RT}_{\mathsf{I}'}$*, and for each* $\rho \in R$*,* $\mathsf{R}(\rho) \subseteq \left(\mathsf{RT}_{\mathsf{I}'} \cap \overline{\mathsf{U}}^{Ar(\rho)}\right)$.

It can be checked that variable assignments in initial interpretations and sets of components of objects in initial interpretations are the same as in corresponding full interpretations with the same universe, species interpretation and attribute interpretation functions, since variable assignments and sets of components of objects do not depend on the interpretation of relations. From now on, all of the above will be used with respect to full interpretations, and full interpretations will simply be called interpretations.

We are now ready to define the meaning of formulæ in interpretations as sets of objects in an interpretation. A sort assignment formula constructed from a term, a reserved assignment symbol and a sort symbol such as :CASE∼*nominative* denotes the set of objects in the interpretation on which the CASE attribute is defined and, when interpreted on them, leads to an object of sort *nominative*; and the path equation :SYNSEM LOCAL CATEGORY HEAD ≈ :HEAD-DTR SYNSEM LOCAL CATEGORY HEAD denotes those objects on which the two given paths are defined and lead to the same object. Relational formulæ, the third kind of atomic formula, also denote sets of objects and will be discussed in more detail below. Existential quantification and universal quantification are restricted to components of objects; and the logical connectives are treated with the familiar operations of set union (disjunction), set intersection (conjunction) and set complement (negation), or with combinations thereof (implication, bi-implication). The definition of Σ formula denotation for quantificational expressions needs a notation for modifying variable assignments with respect to the value of designated variables. For any variable assignment $g \in \mathsf{G}_\mathsf{I}$, for $g' = g[x \mapsto u]$, g' is just like g except that g' maps variable x to object u (possibly a tuple).

Definition 14 *For each signature* $\Sigma = \langle S, \sqsubseteq, S_{max}, A, F, R, Ar\rangle$*, for each (full)* Σ *interpretation* $\mathsf{I} = \langle \mathsf{U}, \mathsf{S}, \mathsf{A}, \mathsf{R}\rangle$*, for each* $g \in \mathsf{G_I}$*,* D_I^g *is the total function from* D^Σ *to the power set of* U *such that*

$$\textit{for each } \tau \in T^\Sigma\textit{, for each } \sigma \in \widehat{S},\ \mathsf{D}_\mathsf{I}^g(\tau \sim \sigma) = \left\{ u \in \mathsf{U} \;\middle|\; \begin{array}{l} \mathsf{T}_\mathsf{I}^g(\tau)(u) \textit{ is defined, and} \\ \widehat{\mathsf{S}}\left(\mathsf{T}_\mathsf{I}^g(\tau)(u)\right) \mathrel{\widehat{\sqsubseteq}} \sigma \end{array} \right\},$$

$$\textit{for each } \tau_1, \tau_2 \in T^\Sigma,\ \mathsf{D}_\mathsf{I}^g(\tau_1 \approx \tau_2) = \left\{ u \in \mathsf{U} \;\middle|\; \begin{array}{l} \mathsf{T}_\mathsf{I}^g(\tau_1)(u) \textit{ is defined,} \\ \mathsf{T}_\mathsf{I}^g(\tau_2)(u) \textit{ is defined, and} \\ \mathsf{T}_\mathsf{I}^g(\tau_1)(u) = \mathsf{T}_\mathsf{I}^g(\tau_2)(u) \end{array} \right\},$$

$$\textit{for each } \rho \in R\textit{, for each } x_1, \ldots, x_{Ar(\rho)} \in V,\ \mathsf{D}_\mathsf{I}^g\left(\rho(x_1, \ldots, x_{Ar(\rho)})\right) = \left\{ u \in \mathsf{U} \;\middle|\; \left\langle g(x_1), \ldots, g(x_{Ar(\rho)})\right\rangle \in \mathsf{R}(\rho) \right\},$$

$$\textit{for each } x \in V\textit{, for each } \delta \in D^\Sigma,\ \mathsf{D}_\mathsf{I}^g\,(\exists x\delta) = \left\{ u \in \mathsf{U} \;\middle|\; \begin{array}{l} \textit{for some } u' \in \overline{\mathsf{C}_\mathsf{I}^u} \\ u \in \mathsf{D}_\mathsf{I}^{g[x\mapsto u']}(\delta) \end{array} \right\},$$

$$\textit{for each } x \in V\textit{, for each } \delta \in D^\Sigma,\ \mathsf{D}_\mathsf{I}^g\,(\forall x\delta) = \left\{ u \in \mathsf{U} \;\middle|\; \begin{array}{l} \textit{for each } u' \in \overline{\mathsf{C}_\mathsf{I}^u} \\ u \in \mathsf{D}_\mathsf{I}^{g[x\mapsto u']}(\delta) \end{array} \right\},$$

$$\textit{for each } \delta \in D^\Sigma,\ \mathsf{D}_\mathsf{I}^g(\neg\delta) = \mathsf{U}\backslash\mathsf{D}_\mathsf{I}^g(\delta),$$

$$\textit{for each } \delta_1, \delta_2 \in D^\Sigma,\ \mathsf{D}_\mathsf{I}^g\left((\delta_1 \wedge \delta_2)\right) = \mathsf{D}_\mathsf{I}^g(\delta_1) \cap \mathsf{D}_\mathsf{I}^g(\delta_2)$$

$$\textit{for each } \delta_1, \delta_2 \in D^\Sigma,\ \mathsf{D}_\mathsf{I}^g\left((\delta_1 \vee \delta_2)\right) = \mathsf{D}_\mathsf{I}^g(\delta_1) \cup \mathsf{D}_\mathsf{I}^g(\delta_2)$$

$$\textit{for each } \delta_1, \delta_2 \in D^\Sigma,\ \mathsf{D}_\mathsf{I}^g\left((\delta_1 \rightarrow \delta_2)\right) = \left(\mathsf{U}\backslash\mathsf{D}_\mathsf{I}^g(\delta_1)\right) \cup \mathsf{D}_\mathsf{I}^g(\delta_2)\textit{, and}$$

$$\textit{for each } \delta_1, \delta_2 \in D^\Sigma,\ \mathsf{D}_\mathsf{I}^g\left((\delta_1 \leftrightarrow \delta_2)\right) = ((\mathsf{U}\backslash\mathsf{D}_\mathsf{I}^g(\delta_1)) \cap (\mathsf{U}\backslash\mathsf{D}_\mathsf{I}^g(\delta_2))) \cup (\mathsf{D}_\mathsf{I}^g(\delta_1) \cap \mathsf{D}_\mathsf{I}^g(\delta_2)).$$

D_I^g is the Σ *formula interpretation function with respect to* I *under a variable assignment,* g*, in* I. Sort assignment formulæ, $\tau \sim \sigma$, denote sets of objects on which the attribute path τ is defined and leads to an object u' of sort σ. If σ is not a species, the object u' must be of a maximally specific subsort of σ. Path equations of the form $\tau_1 \approx \tau_2$ hold of an object u when path τ_1 and path τ_2 lead to the same object u'. And an n-ary relational formula $\rho(x_1, \ldots, x_n)$ denotes a set of objects such that the n-tuples of objects (or chains of objects) assigned to the variables x_1 to x_n are in the denotation of the relation ρ. This means that a relational formula either denotes the entire universe U or the empty set, depending on the variable assignment g in I. For example, according to Definition 14, the formula $\mathsf{append}(x_1, x_2, x_3)$ denotes the universe of objects if the triple $\left\langle g(x_1), g(x_2), g(x_3)\right\rangle$ is in $\mathsf{R}(\mathsf{append})$, or else the empty set. We will return to the meaning of relational formulæ after defining the meaning of grammars to confirm that this is a useful way to determine their denotation.

Negation is interpreted as set complement of the denotation of a formula, conjunction and disjunction of formulæ as set intersection and set union of the denotations of two formulæ, respectively. The meaning of implication and biimplication follows the pattern of classical logic and could alternatively be defined on the basis of negation and disjunction (or conjunction) alone. Quantificational expressions are special in that they implement the idea of restricted quantification by referring to the set of components of objects in $\mathfrak{I}$. An existentially quantified formula, $\exists x\delta$, denotes the set of objects u such that there is at least one component (or chain of components) u' of u, and interpreting x as u' leads to δ describing u. With universal quantification, the corresponding condition must hold for *all* components (or chains of components) of the objects u in the denotation of the quantified formula. Again turning to the application of these definitions of formula denotations in grammar writing, the intuition is that linguists quantify over the components of grammatical structures (sentences, phrases), and not over a universe of objects that may include unrelated sentences and grammatical structures, or components thereof: a certain kind of object exists within a given structure, or all objects in a certain structure fulfill certain conditions.

A standard proof shows that the denotation of Σ formulæ without free occurrences of variables, i.e. the denotation of Σ descriptions, is independent of the initial choice of variable assignment. For Σ descriptions, we can thus define a simpler Σ *description denotation function with respect to an interpretation* $\mathfrak{I}$, $\mathsf{D}_\mathfrak{I}$:

Definition 15 *For each signature* $\Sigma = \langle S, \sqsubseteq, S_{max}, A, F, R, Ar\rangle$, *for each (full)* Σ *interpretation* $\mathfrak{I} = \langle \mathsf{U}, \mathsf{S}, \mathsf{A}, \mathsf{R}\rangle$, $\mathsf{D}_\mathfrak{I}$ *is the total function from* D_0^Σ *to the power set of* U *such that* $\mathsf{D}_\mathfrak{I}(\delta) = \left\{ u \in \mathsf{U} \,\middle|\, \textit{for each } g \in \mathsf{G}_\mathfrak{I},\ u \in \mathsf{D}_\mathfrak{I}^g \right\}$.

For each description δ, $\mathsf{D}_\mathfrak{I}$ returns the set of objects in the universe of $\mathfrak{I}$ that are described by δ. With Σ descriptions and their denotation as sets of objects, we have everything in place to symbolize all grammar principles of a grammar such as the one presented by Pollard & Sag (1994) in logical notation, and the grammar principles receive an interpretation along the lines informally characterized by Pollard and Sag. A comprehensive logical rendering of their grammar of English can be found in Appendix C of Richter (2004). It includes the treatments of (finite) sets and of parametric sorts (such as *list(synsem)*), which are not specifically addressed – but implicitly covered – in the preceding presentation. Moreover, as shown there, all syntactic constructs of the logical languages above are necessary to achieve that goal without reformulating the grammar.

5 Meaning of grammars

Grammars comprise sets of descriptions, the principles of grammar. These sets of principles are often called *theories* in the context of logical languages for HPSG, although this terminology can occasionally be confusing.[14] Theories, i.e. sets of descriptions, are symbolized with θ. A grammar is simply a theory together with a signature:

Definition 16 Γ *is a* grammar *iff*
Γ *is a pair* $\langle \Sigma, \theta \rangle$, *where* Σ *is a signature, and* $\theta \subseteq D_0^\Sigma$.

Essentially, the denotation of a theory can be thought of as the denotation of the conjunction of the descriptions in the theory. The difference is that theories can, in principle (and contrary to deliberate linguistic convention), be infinite in the sense of containing infinitely many descriptions. Conjunctions of descriptions are finite, since conjunctive formulæ are finite.

Definition 17 *For each signature* Σ, *for each* Σ *interpretation* $\mathsf{I} = \langle \mathsf{U}, \mathsf{S}, \mathsf{A}, \mathsf{R} \rangle$, Θ_I *is the total function from the power set of* D_0^Σ *to the power set of* U *such that for each* $\theta \subseteq D_0^\Sigma$,
$\Theta_\mathsf{I}(\theta) = \left\{ u \in \mathsf{U} \;\middle|\; \text{for each } \delta \in \theta, u \in \mathsf{D}_\mathsf{I}(\delta) \right\}$.

Θ_I is the *theory denotation function with respect to* I. A theory consisting of a set of descriptions holds of every object u in the universe exactly if every description in the theory holds of u. In short, a theory denotes the set of objects that are described by everything in the theory. These objects do not violate any restriction that the theory expresses in one of its descriptions.

A first approximation to the meaning of grammars is provided by the notion of a Γ model, a model of a grammar Γ:

Definition 18 *For each grammar* $\Gamma = \langle \Sigma, \theta \rangle$, *for each* Σ *interpretation* $\mathsf{I} = \langle \mathsf{U}, \mathsf{S}, \mathsf{A}, \mathsf{R} \rangle$, I *is a* Γ model *iff* $\Theta_\mathsf{I}(\theta) = \mathsf{U}$.

A Γ model is an interpretation $\mathsf{I} = \langle \mathsf{U}, \mathsf{S}, \mathsf{A}, \mathsf{R} \rangle$ in which every description in the theory of grammar Γ describes every object in the interpretation's universe U. In other words, each object in the interpretation fulfills all conditions which

[14] The problem with this term is that it can be argued that theories, defined this way, do not constitute what would traditionally be called a *theory of a language*, since many central aspects of a theory in the latter sense are not embodied in that kind of formalized theory.

are imposed by the grammar principles. There is no object in a Γ model that violates any principle.

Models of grammars are an appropriate starting point for revisiting the denotation of relational formulæ. Assume we want to define a unary relation `synsem-rel` which contains all objects of sort *synsem* of a typical HPSG grammar. To achieve this, we declare the relation symbol `synsem-rel` in the signature and we add the description in (7) to the theory of the grammar Γ:

(7) $\forall x \left(\texttt{synsem-rel}(x) \leftrightarrow x \sim \mathit{synsem}\right)$

Consider a non-empty Γ model I containing words and phrases. Since by assumption (7) is in the theory of Γ, and we consider a *model*, (7) describes every object in the universe of I. By the bi-implication, every component object of every object u in I which is of sort *synsem* is in $\mathsf{R}(\texttt{synsem-rel})$ (right to left), and every element of $\mathsf{R}(\texttt{synsem-rel})$ is a *synsem* object (left to right). But if the bi-implication in (7) holds in both directions in I, it follows that the expression $\exists x\, \texttt{synsem-rel}(x)$ describes every object u in I which has a component that is in the `synsem-rel` relation. The expression $\forall x\, \texttt{synsem-rel}(x)$ describes every object in I all of whose components are in the `synsem-rel` relation.[15]

Now assume we have a description much like (7) in our grammar theory, but instead of defining the meaning of `synsem-rel`, it defines the meaning of `append`: the new description says that for every object in a grammar model which contains three lists as components, the lists are in the ternary `append` relation as triple $\langle l_1, l_2, l_3\rangle$ iff l_3 is the concatenation of l_1 and l_2 (in that order). Then we can use this `append` relation in yet another grammar principle as follows:[16]

(8) $\mathit{head\text{-}fill\text{-}ph} \Rightarrow \exists \boxed{1}\, \exists \boxed{2}\, \exists \boxed{3} \left(\begin{bmatrix} \textsc{phon} & \boxed{3} \\ \textsc{non-hd-dtrs first phon} & \boxed{1} \\ \textsc{head-dtr phon} & \boxed{2} \end{bmatrix} \wedge \texttt{append}(\boxed{1}, \boxed{2}, \boxed{3}) \right)$

The filler daughter is the only non-head daughter in a head-filler phrase. In English, the phonology of the filler daughter precedes the phonology of the head daughter. According to (8), a head-filler phrase has three components, $\boxed{1}$, $\boxed{2}$, and $\boxed{3}$ such that they are the list values of the PHON attributes of the non-head daughter,

[15] Of which there are none, given the usual structure of signs where *synsem* objects always have components of other sorts.

[16] In RSRL syntax, (8) can be written as
: $\sim \mathit{head\text{-}fill\text{-}ph} \rightarrow$
$\exists x_1\, \exists x_2\, \exists x_3$
$\left(\textsc{:phon} \approx x_3 \wedge \textsc{:head-dtr phon} \approx x_2 \wedge \textsc{:non-hd-dtrs first phon} \approx x_1 \wedge \texttt{append}(x_1, x_2, x_3)\right)$

the head daughter and the phrase as a whole, and they are in the append relation (in the given order). But being in the append relation means that list 3 is the concatenation of list 1 and list 2. Obviously, the denotation of relational formulæ works as intended in grammar models.

Linguists use grammars to make predictions about the grammatical structures of languages. In classical generative terminology, a grammar undergenerates if there are grammatical structures it does not capture. It overgenerates if it permits structures that are deemed ungrammatical. It is uncontroversial that an appropriate notion of the meaning of a grammar should support linguists in making such predictions with their grammars. However, the notion of Γ models in Definition 18 is not strong enough for this purpose. To see this, suppose there is a signature Σ which is fit to describe the entire English language, and there is a theory θ which expresses correctly all and only what there is to say about English. Interestingly, a $\langle \Sigma, \theta \rangle$ model $\mathfrak{I}$ of this perfect grammar of English can be arbitrarily small, as long as every object in the Σ interpretation $\mathfrak{I}$ is described by every grammar principle in θ, as this is a condition on models of a grammar. Therefore a $\langle \Sigma, \theta \rangle$ model of our perfect grammar may consist of nothing but a structure of the single sentence *Elon is going to Mars*. This follows from the definition of Γ models, because any appropriate grammar of English must describe all objects that together make up this well-formed sentence. But this one-sentence-model of the grammar of English is obviously too small to count as a good candidate for the English language, because English contains much more than this single sentence. We conclude that in arbitrarily chosen models, it cannot be detected if a grammar undergenerates or overgenerates.

King's (1999) *exhaustive models* are a possibility to define the meaning of grammars in such a way that the models reflect the basic expectations of generative linguists. The underlying intuition is to choose a maximal model which contains a congruent copy of any configuration of objects which can be found in some model of the grammar. This way, the model chosen for the meaning of a grammar is in a relevant sense big enough so that all the consequences of the grammar can be observed in it. If the grammar overgenerates, the model will contain ill-formed structures. If the grammar undergenerates, expected well-formed structures will be absent.

The simplest way to spell this out is by considering each and every alternative model $\mathfrak{I}'$ of a grammar and observing that whenever you can describe something in an alternative model $\mathfrak{I}'$ with an arbitrary set of descriptions, that set of descriptions also picks out something in the targeted, sufficiently large model $\mathfrak{I}$:

Definition 19 *For each grammar* $\Gamma = \langle \Sigma, \theta \rangle$*, for each* Σ *interpretation* $\mathfrak{I}$*,*
$\mathfrak{I}$ *is an* exhaustive Γ model *iff*
$\mathfrak{I}$ *is a* Γ *model, and*
for each $\theta' \subseteq D_0^\Sigma$*, for each* Σ *interpretation* $\mathfrak{I}'$*,*
if $\mathfrak{I}'$ *is a* Γ *model and* $\Theta_{\mathfrak{I}'}(\theta') \neq \emptyset$ *then* $\Theta_{\mathfrak{I}}(\theta') \neq \emptyset$*.*

Any grammar with a non-empty model also has a non-empty exhaustive model. In addition to being a model of a given grammar $\Gamma = \langle \Sigma, \theta \rangle$, an exhaustive Γ model $\mathfrak{I}$ has the property that each arbitrarily chosen set of descriptions θ' which denotes anything at all in any Γ model also denotes something in $\mathfrak{I}$. An alternative algebraic way to characterize this requirement is to say that any configuration of objects in any Γ model has a congruent counterpart in an exhaustive Γ model. At the same time, since an exhaustive model is from a special class of *models*, if a description in θ does not describe some object in a Γ interpretation $\mathfrak{I}'$, then this object in $\mathfrak{I}'$ cannot have a counterpart in an exhaustive Γ model.

This is sufficient to capture relevant grammar-theoretic notions of linguistics: a grammar Γ of a language $\mathcal{L}$ overgenerates iff an exhaustive Γ model contains configurations that are not (congruent to) grammatical expressions in $\mathcal{L}$; it undergenerates iff an exhaustive Γ model does not contain configurations which are (congruent to) grammatical expressions in $\mathcal{L}$.

6 Alternative conceptions of the meaning of grammars

Section 2 gave an informal overview of four different ways to conceive of models which explain the meaning of HPSG grammars: Theory T1 of Pollard & Sag (1994) views the adequate model as a collection of the object types of the expressions of the language $\mathcal{L}$ that a given grammar describes. T2 by King (1999) takes the intended model to be one from a class of models which contains all possible linguistic tokens of $\mathcal{L}$. T3 (Pollard 1999) constructs the model Γ of language $\mathcal{L}$ as a collection of mathematical idealizations such that each grammatical structure of $\mathcal{L}$ should find a structurally isomorphic counterpart in the model. This model is called the *strong generative capacity* of grammar Γ. And T4 by Richter (2007) defines a schematic extension to grammars called their normal form which guarantees the existence of a model (a minimal exhaustive model) in which all and only the grammatical utterances of $\mathcal{L}$ find exactly one structurally matching configuration each, without commiting to the ontological status of the configurations in the model.

All four share the common core of aiming at capturing the predictions of a

grammar in the sense of directly reflecting possible overgeneration or undergeneration (Section 5): all and only the grammatical structures of $\mathcal{L}$ are supposed to be in the intended model or to find a corresponding counterpart in it. The significant differences between T1, T2, T3 and T4 reside in their assumptions about the nature of the model. The decision of what kind of entities populate the model determines the ontological and structural properties of the entities in the model, which in turn leads to substantial technical differences in the construction of the models. The four theories T1–T4 are numbered chronologically in the order in which they were developed.

Deviating from chronological order, we begin with T2, the theory of exhaustive models (Definition 19). T2 has the distinguished property of insisting on a *token* model of the language $\mathcal{L}$ of a given grammar, $\langle \Sigma, \theta \rangle$. According to T2, actual well-formed linguistic tokens are the immediate object of grammatical description. They are the objects u in the intended exhaustive model $\mathfrak{I} = \langle \mathsf{U}, \mathsf{S}, \mathsf{A}, \mathsf{R} \rangle$. For any occurrence of an utterance of $\mathcal{L}$ in the real world, the intended exhaustive model contains the actual utterance itself. Since linguists cannot know how often an utterance of a concrete token in $\mathcal{L}$ did occur and will occur in the world, exhaustive models are a class of models. For T2 it does not matter how often the token utterance *Elon is going to Mars* is encountered at a concrete place and time in the world, because among the class of exhaustive models of English there is one with the correct number of occurrences for this utterance and all other actual utterances, and that exhaustive model is the intended one. However, there is a crucial complication: it is clear that most conceivable well-formed expressions of any given human language were never produced and never will be. Since, by construction, an exhaustive model must contain all potential well-formed expressions of a language which obey the principles of grammar, in addition to actual utterance tokens, the theory of exhaustive models must admit *potential tokens* in the intended exhaustive model for those utterances which never occur in the real world. If token models are already suspicious (or unacceptable) to many linguists, models comprising non-actual tokens are even more contentious.

T2 is designed in deliberate opposition to the chronologically preceding theory T1 of Pollard & Sag (1994), the only one which employs feature structures. T1 proposes that a grammar $\Gamma = \langle \Sigma, \theta \rangle$ denotes a set of mathematical representations of *types* of linguistic events. The main idea is that the object types abstract away from individual circumstances of token occurrences, because for T1 a grammar of a language is assumed not to be concerned with individual linguistic events or tokens. The object types capture individual linguistic token events in the sense that an object type conventionally corresponds to "those imaginable linguistic

objects that are actually predicted to be possible ones" (Pollard & Sag 1994: 7) in the language $\mathcal{L}$ that $\langle \Sigma, \theta \rangle$ describes. The postulated intuitive correspondence is not explicated further, but it is expected that a trained linguist will recognize which object type a linguistic token encountered in the real world corresponds to. When observing a token expression of English in the world, for example in a situation in which someone exclaims *Elon is going to Mars!*, the linguist recognizes the corresponding object type. The informality of the relationship between the denotation of a grammar (mathematical objects serving as object types) and the domain of empirically measurable events (utterances of grammatical expressions of a language) is one of the reasons to reject T1. In addition to the weak connection between the object types and the domain of empirically accessible data, object types have been criticized for being ontologically dubious and in any case superfluous and thus falling victim to Occam's razor. A theory of meaning without such an additional ontological postulate is deemed to be stronger.

T1 is implemented by constructing linguistic object types as abstract feature structures. In first approximation – to be refined presently – these can be thought of as rooted directed graphs, or, in terms of our previous grammar models, as configurations of objects under a root node. Definition 11 introduced $\mathsf{C}_{\mathfrak{I}}^{u}$ as the set of components of an object u in an interpretation $\mathfrak{I}$. The root node of the directed graph corresponds to the distinguished object u in a set $\mathsf{C}_{\mathfrak{I}}^{u}$. The abstract feature structures used as mathematical representations of object types, however, are not graph-like objects, as two distinct graphs could be isomorphic, in violation of the core idea of proposing unique object types for classes of linguistic events. Abstract feature structures are therefore defined as (tuples of) sets, representing each node v in the graph as an equivalence class of paths that lead to v from the root node. A labeling function assigns sorts to these abstract nodes in accordance with the feature appropriateness function of the signature, and relations are basically tuples of abstract nodes. A satisfaction function determines what it means for a feature structure to satisfy a description, which is then elaborated in the notion of grammars admitting sets of abstract feature structures. In terms of the exhaustive models of T2, the abstract feature structures admitted by a grammar Γ can be imagined as a normal form representation with the abstract feature structures (the linguistic types) serving as the objects u in a canonical exhaustive model $\mathfrak{I}$ of Γ.[17] The earlier ontological criticism of T1 amounts to rejecting the insinuation that linguists consider (abstract) feature structures the subject

[17] This characterization is slightly simplistic; see Richter (2004: Appendix A, Definition 80) for details. Abstract feature structures are in fact extended to *canonical entities* to obtain canonical interpretations/models/exhaustive models.

of their grammars and affirming that their real interest lies in the description of languages. Assuming the existence of abstract feature structures is then a superfluous detour in the linguistic enterprise.

Meaning theory T3 is positioned against the theory T1 of object types for classes of theoretically indistinguishable linguistic tokens, and against the theory T2 of perceiving the meaning of a grammar in an intended exhaustive model populated with actual and non-actual linguistic tokens. With T3, Pollard (1999) is firmly opposed to token models and sees mathematical idealizations as fundamental to grammatical meaning. The concept of non-actual tokens is deemed unacceptable and self-contradictory. However, Pollard (1999) also rejects T1's ontological commitment to object types and wants to strengthen the relationship between the structures in the denotation of a grammar and empirically observable token expressions. According to T3, no two structures in the *strong generative capacity*, the collection denoted by a grammar $\langle \Sigma, \theta \rangle$ of language $\mathcal{L}$, are structurally isomorphic, and each utterance token of language $\mathcal{L}$ which is judged grammatical finds a structurally isomorphic counterpart in the grammar's strong generative capacity. An occurrence of the question *Is Elon really going to Mars?*, just like the occurrence of any other grammatical token of English, must find a unique structurally isomorphic mathematical idealization in the strong generative capacity of an adequate grammar of English. With this requirement, T3 tightens the connection between observables and the mathematical model, cutting out the types and establishing a much stricter link between the predictions of a grammar and the domain of empirical phenomena than the abstract feature structure models of Pollard & Sag (1994) offer with their appeal to conventional correspondence.

T3 is spelled out on the basis of models (Definition 18),[18] offering three alternative ways of characterizing the strong generative capacity of a grammar. The structures in Pollard's models can be understood as pairs of interpretations $\mathfrak{I} = \langle \mathsf{U}, \mathsf{S}, \mathsf{A}, \mathsf{R} \rangle$ and a root node u whose set of components ($\mathsf{C}_{\mathfrak{I}}^{u}$) constitute $\mathfrak{I}$'s universe U. The objects in $\mathsf{C}_{\mathfrak{I}}^{u}$ are all defined as canonical representations by a construction employing equivalence classes of attribute paths originating at the root node: given a grammar Γ, its strong generative capacity is the set of all such canonical representations whose interpretations are Γ models. By construction, they are all pairwise non-isomorphic, and with their internal (set-theoretic) structure, they can be assumed to be structurally isomorphic to grammatical utterance tokens of a language, in contrast to the abstract feature structures of Pollard &

[18] Pollard (1999) is in fact based on Speciate Re-entrant Logic (SRL), King's precursor of RSRL, but a straightforward extension to full RSRL is provided in Richter (2004).

Sag (1994). The canonical representations in the strong generative capacity can be abstracted from each exhaustive model.

A central tenet of theories T1 and T3 of the meaning of grammars as sets of abstract feature structures and as mathematical idealizations in the strong generative capacity is the one-to-one correspondence either of object types or of mathematical idealizations to (linguistically indistinguishable groups of) grammatical utterances in a language. Richter (2007) investigates the models of existing HPSG grammars, such as the fragment of English developed in Pollard & Sag (1994), and notes that T1 and T3 necessarily trigger an unintended one-to-many relationship between grammatical utterances and structures in the denotation of typical HPSG grammars: one token utterance leads to more than one structure in the grammar denotation. The main reason is that, in both theories, each structure which corresponds to a grammatical utterance entails the presence of a large number of further structures. For the strong generative capacity, the additional structures come from the substructural nodes in the mathematical idealization of an utterance which, by design, must in turn function as root nodes of admissible structures. But these additional structures are not mathematical idealizations of empirically observable grammatical utterances. In fact, many of the structures present in the strong generative capacity do not correspond to structures which can occur in grammatical utterances at all.[19] While the abstract feature structures of T1 do not have substructures, the abstract feature structure admission relation relies on a mechanism with exactly the same effect: admitting the unique type of *Elon must be on his way to Mars* entails the existence of many other types, so-called reducts of the intended type, and these reducts do not have empirical counterparts in linguistic utterance tokens.

In response to these problems, T4 proposes *normal form grammars*, schematic signature and theory extensions applicable to any HPSG grammar. The core idea behind the canonical grammar extension is to partition the denotation of grammars into utterances and to guarantee by construction that every *connected configuration of objects* in a grammar's denotation is isomorphic to an utterance token in a language. For T1 and T3, this extension is insufficient to establish the intended one-to-one correspondence between observable utterances and object types or mathematical idealizations, because the structures predicted by T1 and T3 still generate additional linguistic types or mathematical idealizations corresponding to each feature structure reduct or substructure, respectively. However, normal form grammars allow the definition of *minimal exhaustive models*, because normal form grammars can be shown to have exhaustive models

[19] See Richter (2007: Section 4) for extensive discussion and examples.

which contain non-isomorphic connected configurations of objects with the special property that each of these configurations corresponds to a grammatical utterance. According to T4, *Elon must be on his way to Mars* corresponds to exactly one connected configuration in the minimal exhaustive model of a perfect grammar of English, and so does any other well-formed English utterance. Proposal T4 is not forced to make any assumptions about the ontological status of the inhabitants of minimal exhaustive models of normal form grammars, since they do not have to be defined as a particular kind of mathematical structure (nor is this option excluded if it is desired).[20] T4 shares with T3 the commitment to providing an isomorphic structure to each grammatical utterance of a given language rather than just a corresponding linguistic type. With King's theory T2, it shares the avoidance of mathematical entities representing linguistic facts.

HPSG is among a small group of grammar formalisms with a very precise outline of their formal foundations. This high degree of precision extends up to different but closely related ways of characterizing the meaning of grammars. The differences are in part of a very technical nature, but under the technical surface, they are due to different opinions of what grammars ought to describe. It is an advantage of HPSG as a grammar framework that all these approaches are built on the same explicit logical foundations. As a consequence, their relationships can be studied with the rigorous tools of mathematical logic. The philosophical debate regarding the adequacy of each interpretation of the nature and purpose of grammars is thus grounded in concrete mathematical structures. Finally, independent of philosophical arguments and preferences, proposal T1, enlisting typed feature structures as canonical structures in models, provides a bridge to the literature on feature logics, connecting linguistic theory to an interesting set of efficient computational methods, pursued in other chapters of the present handbook (Bender & Emerson 2020, Chapter 26 of this volume). This connection to computation and the rich literature on feature structures is untouched by whether these models are deemed adequate for linguistic theory.

Acknowledgements

I would like to thank Adam Przepiórkowski and the reviewers Jean-Pierre Koenig, Stefan Müller and Manfred Sailer for their helpful suggestions which helped improve this chapter considerably.

[20] The techniques enlisted in the construction of mathematical idealizations in T3 can easily be adapted to this end.

References

Bender, Emily M. & Guy Emerson. 2020. Computational linguistics and grammar engineering. In Stefan Müller, Anne Abeillé, Robert D. Borsley & Jean-Pierre Koenig (eds.), *Head-Driven Phrase Structure Grammar: The handbook*, 989–1033. Berlin: Language Science Press. DOI:??

Crysmann, Berthold. 2020. Morphology. In Stefan Müller, Anne Abeillé, Robert D. Borsley & Jean-Pierre Koenig (eds.), *Head-Driven Phrase Structure Grammar: The handbook*, 839–888. Berlin: Language Science Press. DOI:??

Höhle, Tilman N. 2018. Spurenlose Extraktion. In Stefan Müller, Marga Reis & Frank Richter (eds.), *Beiträge zur Grammatik des Deutschen: Gesammelte Schriften von Tilman N. Höhle* (Classics in Linguistics 5), 499–537. Berlin: Language Science Press. DOI:10.5281/zenodo.1169689

King, Paul. 1999. Towards truth in Head-Driven Phrase Structure Grammar. In Valia Kordoni (ed.), *Tübingen studies in Head-Driven Phrase Structure Grammar* (Arbeitsberichte des SFB 340 No. 132), 301–352. Tübingen: Universität Tübingen. http://www.sfs.uni-tuebingen.de/sfb/reports/berichte/132/132abs.html, accessed 2018-2-25.

Koenig, Jean-Pierre & Frank Richter. 2020. Semantics. In Stefan Müller, Anne Abeillé, Robert D. Borsley & Jean-Pierre Koenig (eds.), *Head-Driven Phrase Structure Grammar: The handbook*, 889–928. Berlin: Language Science Press. DOI:??

Lücking, Andy, Jonathan Ginzburg & Robin Cooper. 2020. Grammar in dialogue. In Stefan Müller, Anne Abeillé, Robert D. Borsley & Jean-Pierre Koenig (eds.), *Head-Driven Phrase Structure Grammar: The handbook*, 1035–1080. Berlin: Language Science Press. DOI:??

Müller, Stefan. 2020. HPSG and Construction Grammar. In Stefan Müller, Anne Abeillé, Robert D. Borsley & Jean-Pierre Koenig (eds.), *Head-Driven Phrase Structure Grammar: The handbook*, 1355–1407. Berlin: Language Science Press. DOI:??

Pollard, Carl J. 1999. Strong generative capacity in HPSG. In Gert Webelhuth, Jean-Pierre Koenig & Andreas Kathol (eds.), *Lexical and Constructional aspects of linguistic explanation* (Studies in Constraint-Based Lexicalism 1), 281–298. Stanford, CA: CSLI Publications.

Pollard, Carl J. & Ivan A. Sag. 1987. *Information-based syntax and semantics* (CSLI Lecture Notes 13). Stanford, CA: CSLI Publications.

Pollard, Carl J. & Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar* (Studies in Contemporary Linguistics). Chicago: The University of Chicago Press.

Pollard, Carl J. & Eun Jung Yoo. 1998. A unified theory of scope for quantifiers and *wh*-phrases. *Journal of Linguistics* 34. 415–445. DOI:10.1017/S0022226798007099

Pullum, Geoffrey K. & Barbara C. Scholz. 2001. On the distinction between generative-enumerative and model-theoretic syntactic frameworks. In Philippe de Groote, Glyn Morrill & Christian Retor (eds.), *Logical Aspects of Computational Linguistics: 4th International Conference* (Lecture Notes in Computer Science 2099), 17–43. Berlin: Springer Verlag.

Richter, Frank. 2004. *A mathematical formalism for linguistic theories with an application in Head-Driven Phrase Structure Grammar*. Universität Tübingen. (Phil. Dissertation (2000)). https://publikationen.uni-tuebingen.de/xmlui/handle/10900/46230, accessed 2018-2-25.

Richter, Frank. 2007. Closer to the truth: A new model theory for HPSG. In James Rogers & Stephan Kepser (eds.), *Model-theoretic syntax at 10 – Proceedings of the ESSLLI 2007 MTS@10 Workshop, August 13–17*, 101–110. Dublin: Trinity College Dublin.