Global Grammar: Building a cross-linguistic construction typology using HPSG

We demonstrate a methodology for representing cross-linguistic inventories of construction types in a grammar-driven, unitary formalism and in a unitary conceptual system. The grammatical basis resides in what we will call a *global level of grammatical analysis*. We first explain this notion.

Although languages, including their grammars, differ in countless ways, there is a set of semantic and syntactic parameters which we may call *global* parameters of analysis. They are, technically viewed, aspects of grammatical analysis which apply to a sentence as a whole rather than to its constituents, thus at a ‘global’ rather than ‘local’ level of sentence analysis. At the same time they reflect a repertoire of parameters for which probably all grammars are defined one way or another, and thus constitute a cross-linguistically ‘global’ dimension of analysis.\(^1\) Parameters of global specification include the following:

\(1\) ‘Global parameters’ (aspects of grammatical analysis which apply to a sentence as a whole rather than to its constituents, thus at a ‘global’ rather than ‘local’ level of sentence analysis):
- syntactic argument relations, described in terms such as ‘subject’, ‘object’, etc., called *grammatical functions*, or a related system with ‘A’, ‘S’, ‘P’ (see Witzlack 2011);
- semantic argument structure, that is, how many *participants* are present in the situation depicted, and which *roles* they play (such as ‘agent’, ‘patient’, etc.);
- linkage between syntactic and semantic argument structure, i.e., which grammatical functions express which roles;
- identity relations, part-whole relations, etc., between arguments;
- aspect and Aktionsart, that is, properties of the situation expressed by a sentence in terms of whether it is dynamic/stative, continuous/instantaneous, completed/ongoing, etc.;
- type of the situation expressed, in terms of some classificatory system of situation types
- derivational history of the sentence in terms of operations affecting the above properties.

A suitable mode of representation of these parameters is *attribute-value matrices (AVM)*, and the extension thereof, *Typed Feature Structures (TFS)*, both being standard formalisms in linguistics, and readily interpretable for digital purposes, and in particular used in complex systems like grammars, such as frameworks like *Lexical Functional Grammar (LFG)* (Bresnan 2001, Dalrymple 2001, Butt et al. 1999), and *Head-Driven Phrase Structure Grammar (HPSG)* (Pollard and Sag 1994, Sag et al. 2003). In this perspective, the parameters in (1) constitute a repertoire of specifications for which probably all grammars are defined, and so can serve as a ‘core’ in the creation of an aligned, unified, inventory of possible grammars, and as possible discriminants in a typology of construction types across languages.

For functional purposes related to such a typology, a formal ‘switch’ will be defined, mapping between TFS representations and a *string* based format suited for compact annotation and enumeration of construction and valence types. We thereby have the means for modeling entire grammars and lexicons, although their functionalities need not expose the HPSG system as such. The system of HPSG in combination with the string based format is also flexible enough to allow its organization to enhance the functionalities, through modularization based on types and unification. We demonstrate this in a design for a large scale construction typology, with representations of ‘global parameters’ as a key factor. In parallel we develop a methodology for inducing grammars on a cross-linguistic basis, taking the construction typology as basis.

\(^1\) We could have used ‘universal’ in this context, but this term has many fixed uses distinct from what we have presently in mind.
We see grammars as systems by which the modeling of global properties will be in principle always traceable, and thus in essence *compositional*. It therefore makes sense to define a ‘grammar core’ in terms of the global parameters, and radiating from this, grammars of various types, thereby making a step towards the creation of an aligned, unified, inventory of possible grammars.\(^2\) The construction type inventories, described in terms of global parameters, are stepping stones in this development.

These items are summarized in (2), and in this chapter we will briefly illustrate their content and how they can be interrelated.

\[ (2) \]
\[
\begin{array}{ccc}
\text{Global parameters} & / & \text{String representations} \\
\text{TFS representations} & \leftrightarrow & \text{Construction- and Valency enumerations} \\
\text{Grammar cores} & \leftrightarrow & \text{Valence ontologies} \\
\text{Grammar structures} & & \text{Valence corpora} \\
& & \text{IGT-cum-valency}
\end{array}
\]

1. **TFS for global parameters**

The essential ideas behind typed feature structures are manifest in daily life, in situations where we present inventories or plan actions. Generally speaking, there is an *issue* at hand, and it has a number of ‘respects’ or ‘concerns’ – ‘concerning this, the issue requires so-and-so’, and ‘concerning that, the issue requires so-and-so’. Each *concern* may have a *solution* nameable by a given thing or person, but it may also be that it introduces another *issue*, such as a complex of sub-actions. In an AVM, *attributes*, here written in capital letters, are used to encode *concerns*, and the *value* of an attribute, written in small letters, is the *solution* to the concern – it could be represented by the name of a person, or a label for a new issue. Schematically this gives the patterning possibilities shown in (3), where the items in italicized small letters – the values - are *types*. Types thus either serve as ultimate values, or as ‘issues’ introducing a new set of attributes:

\[ (3) \]
\[
\begin{array}{c}
\text{issue} \\
\text{CONCERN1 thing} \\
\text{CONCERN2 issue} \\
\text{CONCERN1 thing}
\end{array}
\]

The following exposition of such a design follows Copestake (2002), which is an introduction to the *Linguistic Knowledge Builder (LKB)* system, which underlies one of the computational platforms of HPSG, and Pollard and Sag (op. cit.). In this design, when a type occurs in a non-final (‘non-leaf’) position in a path, we say that it *declares* or *introduces* the attributes that occur immediately to its right. The following two principles govern the introduction of attributes:

---

\(^2\) Our strategy should be kept distinct from initiatives such as ‘Grammatical Framework’ (‘GF’; cf. Ranta 2011), the ‘HPSG Grammar Matrix’ (‘The Matrix’ - cf. Bender et al. 2010), and ‘The Core Grammar’ project (http://hpsg.fu-berlin.de/~stefan/Pub/coregram.html). The former two reside in providing constructive ‘kits’ from which one can start building grammar applications, the GF kits being computational structures and the Matrix kits residing in grammar structures; they do not orient themselves relative to a ‘parameter priority’ or aim at modeling a constructional typology, as the present initiative does. The latter points also apply to the third.
A given type introduces the same attribute(s) no matter in which environment it is used. A given attribute is declared by one type only (but occurs with all of its subtypes).

1a Attributes for Grammatical Functions and for Semantic Participants

GF is the attribute introducing grammatical functions, and its sub-attributes are conceived partly like the inventory used in LFG, ‘SUBJ’ (for ‘subject’), ‘OBJ’ (for ‘(direct) object’), OBJ2, COMP (for complement clauses not acting as objects), OBL, IOBJ (for ‘indirect object’), and SECPR (for ‘secondary predicate’, rather than ‘XCOMP’).

The AVM format in principle lends itself as naturally to the representation of participants relative to the situation expressed by a sentence, as it does to the GFs constituting the syntactic structure of the sentence. For a sentence like John kicks Peter, we may at the outset consider (5) below as a sign representation expanded from (4). The participants in the situation type expressed by the sentence are introduced by the attribute ACTNTS (for ‘actants’, the notion used by Tesnière (1959) and Melchuk (2004)), being distinguished as ‘actant 1’ (ACT1), and ‘actant 2’ (ACT2) (see below for discussion of these notions and their relatedness to the notion ‘role’). The GF and ACT values are interlinked through the individuals serving as bearers of the actant functions, identified by a pointer entered as value of the ACT attribute, which can at the same time be seen as the referents – introduced by the attribute INDX – of the grammatical functions:

\[
\begin{align*}
\text{GF} & \begin{cases} 
\text{SUBJ} & \text{INDEX} [1] \\
\text{OBJ} & \text{INDEX} [2] \\
\text{ACTNTS} & \begin{cases} 
\text{ACT1} & [1] \\
\text{ACT2} & [2] 
\end{cases}
\end{cases}
\end{align*}
\]

Technically, the paths ‘SUBJ [INDEX [1]]’ and ‘ACT [1]’ both lead to the same individual, or ‘index’ – identified by the boxed number ‘[1]’. Such a use of identical boxed numbers is often referred to as reentrancy or ‘identity’.

The format in (5) allows one to model cases of ‘failed’ linking – a syntactic item lacking a semantic counterpart, or the opposite. The following cases may be considered, with short examples:

(6)

a. There is a boy sitting outside.
   b. The boy is eating.
   c. The apple was eaten.
   d. The apple eats easily.

For a case where a subject is an expletive pronoun and by assumption lacks a semantic participant, as in (6a), the constellation would be as in (7):

---

3 Except for the presence of the initial attribute GF – f-structures in LFG consist for the most part of GF attributes, hence no such ‘assembling’ attribute is needed, whereas in the present system, other aspects of the sign are contained in the same AVM as grammatical relations. Also different from an f-structure is the lack of a line ‘PRED ‘kick <SUBJ, OBJ>’”, as would be used in LFG for a case like this – we return to this below.

4 It may be noted that in HPSG, the use of such attributes is uncommon; however, to develop a global level representation of syntactic structure, such information is needed. A discussion of the issue from an HPSG-point of view is given in xxx.

5 Notationally, instead of boxes, one can use ‘#’, so that the above pair would come out as ‘SUBJ [INDEX #1]’ and ‘ACT1 #1’. As an alternative to embedding of brackets as in ‘SUBJ [INDEX #1]’, one can use a bar between the attributes, as in ‘SUBJ[INDEX #1]’, or a dot, as in ‘SUBJ.INDEX #1’.

---
(7) ‘Expletive subject’:
\[
\begin{array}{c}
\text{GF} \\
\text{OBJ} \\
\text{ACTNTS}
\end{array}
\begin{array}{c}
\text{SUBJ [HEAD pron]} \\
\text{INDX [ ]} \\
\text{ACT1 [ ]}
\end{array}
\]

For a case with an ‘implicit argument’ as object, as commonly assumed for a sentence like (6b), the constellation would be the converse, with the appearance of an ACT2 participant not linked to syntax:

(8) ‘Implicit object’:
\[
\begin{array}{c}
\text{GF} \\
\text{ACTNTS}
\end{array}
\begin{array}{c}
\text{SUBJ [INDX [ ]]} \\
\text{ACT1 [ ]} \\
\text{ACT2 [ ]}
\end{array}
\]

The value index when used as value as in (8), indicates that there is a referent, and this implicit actant could in principle be syntactically realized as argument of the lexical item carrying the participant role (here, as object). As is well known this is different from ‘agents’ of passive and middle verbs in English, as in (6c,d), which can only – if at all – be activated by prepositions, not by the verb, a situation often referred to as ‘blocking’. A way of expressing this difference within the format given is by distinguishing two subtypes of index, one being realizable index, and one being block(ed) index. If one lets INDX inside the path from SUBJ or OBJ always be real index, representations like (8) will stay as given, and blocked actants will come out as in (9), suitable for a middle form of a transitive verb, like in the book reads well:

(9) ‘Blocked subject role’:
\[
\begin{array}{c}
\text{GF} \\
\text{ACTNTS}
\end{array}
\begin{array}{c}
\text{SUBJ [INDX [ ]]} \\
\text{ACT1 [blockindex]} \\
\text{ACT2 [ ]}
\end{array}
\]

1b The ‘actants’ (‘arguments’) enumeration
The attributes ACT1, ACT2 etc. as used here are partly role labels, and partly enumeration markers: As enumeration markers, they list the participants present in the situation expressed (including implicit ones), starting with ACT1, using ACT2 only if there is an ACT1, and using ACT3 only if there is an ACT2. (This is analogous to the conventional listing of arguments of an operator in logical notation, where in expressions like ‘P(x,y)’ one introduces a comma only if there is more than one argument; and distinct from the conventions in PropBank.) As role markers, when there is more than one argument, they express something close to ‘macro’ or ‘proto’ roles, so that when there is an ACT1 and an ACT2, ACT1 is the role associated with emanation of force, and ACT2 is the ‘impacted’ part relative to the force; an ACT3 would then express a slightly less directly involved participant than the ACT2, such as the recipient or benefactive in a ditransitive sentence; in these contrasts, the ACTs have the same intuitive basis as Dowty’s (1991) proto-roles.\footnote{The Paninian roles kharta and karma are the earliest in this tradition (Parnini’ ref.). The conventions described contrast with the use of ARG0, ARG1, … in PropBank, which represent fixed roles, again at the level of calibration as in the proto-roles.}

When there is only one actant, it will be marked as ACT1, regardless of its role. (Again, this is analogous to conventional logical notation.)
Distinguishing between no more than three (or four) participant types, the \textit{ACTn} attributes by no means purport to fully differentiate between all types of roles that can be recognized. On the other hand they are not mere replica of the GFs of the sentence represented: apart from the circumstance that also implicit participants receive an \textit{ACTn}, the ordering among the \textit{ACTn} s does not necessarily reflect the actual GFs carried by the constituents expressing the participants in question. Such situations arise when – intuitively speaking – there has been an argument frame-changing operation whereby an ‘original’ correspondence ‘<subject - ACT1, direct object- ACT2, indirect object – ACT3>’ has been obliterated. For instance, although in a sentence like (6d), \textit{the apple} is a subject, it will correspond to the ACT2 participant, and this will reflect the circumstance that in the ‘underlying’ structure of this construction, it is the agent which is expressed as subject and carries the ACT1 status. For the moment, the exact status of this assumed ‘underlying’ representation format, and its adherence to a correspondence pattern of the type ‘<subject - ACT1, direct object- ACT2, indirect object – ACT3>’, has not been formally quite defined.\footnote{We assume that in a construction like English \textit{The ball was kicked by John}, the ACT1 of \textit{kicked} is blocked, and \textit{by} introduces its own ACT1 and ACT2, one being the event and the other coindexed with the ACT1 of \textit{kicked}.}

In line with common assumptions in semantics, one more attribute in the \textit{ACTn} family is the \textit{index of a situation}, often referred to as the \textit{event index}. As a locus of this index (following HPSG) we use an attribute name \textit{ACT0}, also introduced inside ACTNTS. This information is doubled with an attribute \textit{INDX} sitting at the outermost layer of the \textit{sign}; these attributes are both illustrated in (13) further below.

1c How to express a richer array of participant roles I

A semantic representation should expose what sets the meaning of the sign represented apart from the meaning of other signs; and at the same time expose what the meaning of the sign in question has in common with the meaning of other signs. From the perspective of participant roles, such a demand for expressiveness calls for approaches beyond what has so far been considered, and one approach is to introduce \textit{ROLE} as an attribute inside of \textit{ACTn}, with role notions as value:

\begin{equation}
\text{(10) Representing roles: } \left[ \begin{array}{c}
\text{ACTNTS} \\
\text{ACT1 index \text{ROLE agent}} \\
\text{ACT2 index \text{ROLE theme}} \\
\end{array} \right]
\end{equation}

An alternative would be to go directly to role labels as attributes, and not via the \textit{ACTn}:

\begin{equation}
\text{(11) Representing roles, alternative: } \left[ \begin{array}{c}
\text{ACTNTS} \\
\text{AGENT index} \\
\text{THEME index} \\
\end{array} \right]
\end{equation}

There is in many cases a need for leaving a role status underspecified. This can be easily done in the format in (10) by using as value of \textit{ROLE} simply a super-type of all the candidate role names, i.e., \textit{role}, whereas in the format in (11), an actant cannot be indicated without a specific role.

The frameworks LFG and HPSG both feature an attribute \textit{PRED}, used in f-structures in LFG and semantic representations in HPSG. The value of this attribute is simply a letter-string identical to the spelling of the word acting as head of the construction being analyzed, and is not intended as exposing what sets the meaning of the sign in question apart from the meaning of other signs (apart from, trivially, suggesting that it \textit{is} different), or what the meaning of the sign in question has in
common with the meaning of other signs: the PRED-value at most can be seen as a placeholder for whichever formal representation would be offered for the meaning in question. We for the present will use a PRED-attribute notation just as mentioned, which will be helpful in exposing to a reader what ‘meaning’ is being discussed, but it is not a contribution to the question of representing situational meaning; it is illustrated in the structure (12) below.

1d Types in the feature formalism – a first illustration

In a general sense, types - with subtypes and super-types - constitute hierarchies of the sort one will want for organizing situation types, Aktionsarten, semantic roles, parts-of-speech, and more. Types also can be introduced as constitutive to the modelling of grammatical information, in such a way that every attribute occurrence has a type as value, thus, also ‘inner’ and ‘initial’ attributes in an AVM path will have types as values. As mentioned, in the adopted design following Copestake (2002), the following two principles govern the introduction of attributes, repeated:

(4) [A] A given type introduces the same attribute(s) no matter in which environment it is used.
[B] A given attribute is declared by one type only (but occurs with all of its subtypes).

The structure shown in (12) illustrates some type declarations. This is a typed AVM for a ditransitive construction, and exemplifies the ‘inner’ types, and amongst others also the use of the attributes ACT3 and ACT0 mentioned above:

\[
\begin{array}{c}
\text{HEAD} \\
\text{GF grmfct} \\
\text{INDX} \\
\text{ACTNTS sit} \\
\text{PRED give _rel} \\
\text{ACT0 0} \\
\text{ACT1 1 ROLE agent} \\
\text{ACT2 2 ROLE theme} \\
\text{ACT3 3 ROLE beneficiar} \\
\text{AKTRT aktionsart} \\
\end{array}
\]

Here, the outermost occurrence of the type \textit{sign} declares the attributes HEAD, GF, INDX and ACTNTS; the type \textit{grmfct} declares SUBJ, OBJ and IOBJ; and the type \textit{sit} declares ACT0, ACT1, ACT2, ACT3, and AKTRT. The attributes SUBJ, OBJ and IOBJ all take \textit{sign} as value in turn. In return, as prescribed by principle (4)[B], these features HEAD, GF, INDX and ACTNTS are introduced only by the type \textit{sign}.

In contrast to the latter, a given type can be the value of more than one attribute: for instance, in (12), the type \textit{index} is a value of INDX as well as of ACT0, ACT1, ACT2, and ACT3.

Under the regulations (4), it is in practical exposition defensible to leave out many of the types mentioned in (12): for instance, since INDX, HEAD, GF etc are all declared by \textit{sign}, the mentions in (12) of \textit{sign} can in practical exposition be left out. Moreover, if there is little to say about an attribute in a given exposition, there is no need to represent it – by the general attribute declaration of the type concerned, one knows which attributes will in principle occur there.

\[9\] These ‘inner’ occurrences of \textit{sign} all are shown only as introducing the attribute INDX, but in principle they also declare HEAD, GF, and ACTNTS, as prescribed by principle [A].
Illustrating ‘non-isomorphy’ between the structures of GF and ACTNTS

Why not have just (13) instead of (12), indicating roles as declared by index as in (10), but without mention of the ACT attributes?

(13)

One type of consideration is as follows.
Ditransitive constructions may in many languages be formed through a causation marker on the stem of a transitive verb, typically yielding a linking between syntactic and semantic argument structure schematically looking as in (14) (here using ‘OBJ2’ as GF rather than ‘IOBJ’):

This structure exposes the subject as the ‘causer’, and the ‘caused situation’ as the ACT2, the ACT1 of which situation (‘the ‘caussee’’) is realized as (‘surface’) object and whose ACT2 (‘the underlying object’) is realized as (‘surface’) indirect object. An example is given in (15):

(15) (example of the structure in (14), from Citumbuka (Jean Chavula, p.c.))

Mary wa-ka-mu-phik-isk-a Tumbikani nchunga
Mary 1SM-pst-1OM-cook-Caus-fV Tumbikani beans
‘Mary made Tumbikani cook beans’

Clearly, in such a structure there is no longer an ‘isomorphy’ between GFs and participants in a way that would justify a representation like (13).
Clauses with ‘derived’ structures as illustrated in (7)-(9) constitute another type of ‘non-isomorphic’ constellation. We thus in principle need the full ACTNTS structure alongside the GF structure as envisaged (but can occasionally use the format in (13) for abbreviation).

If The type index
First, as a formal point, it should be noted that in (14), the partial specification

\[
\begin{bmatrix}
\text{HEAD verb} \\
\text{GF grnmfc}\text{ sign} \\
\text{OBJ sign} \\
\text{IOBJ sign} \\
\text{INDX 0} \\
\text{ACTNTS sit} \\
\text{ACT0 0} \\
\text{AKTRT aktionsart}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\text{SUBJ sign} \\
\text{OBJ sign} \\
\text{OBJ2 sign} \\
\text{PRED cause-rel} \\
\text{ACTNTS ACT1 1} \\
\text{ACT1 2} \\
\text{ACT2 3}
\end{bmatrix}
\]
is strictly speaking not admitted by the regulations (4): the value of ACT2 is generally the type index, and index has no declaration allowing it to introduce (the inner) ACT1, ACT2 as attributes. This situation can be resolved by letting the type index in general declare an attribute KIND, abbreviated K, whose value will be kind with subtypes sit and indiv(idual):

\[
\begin{array}{c}
\text{kind} \\
/ \\
\text{indiv} \quad \text{sit}
\end{array}
\]

The type sit is already the value of ACTNTS, as exemplified in (13), but can also occur as value of other attributes, as here K, and will thus be able to introduce ACT1 and ACT2 etc. as needed, as in (16):

\[(16)\quad \text{ACT2 index} \quad \begin{bmatrix} \text{K sit} \\
\text{ACT1} \begin{bmatrix} 2 \\ \text{index} \end{bmatrix} \\
\text{ACT2} \begin{bmatrix} 3 \\ \text{index} \end{bmatrix} \end{bmatrix}\]

An amended version of (14) will thereby be:

\[(17)\quad \begin{bmatrix}
\text{HEAD verb} \\
\text{SUBJ sign} \begin{bmatrix} \text{INDX} \begin{bmatrix} 1 \\ \text{index} \end{bmatrix} \end{bmatrix} \\
\text{OBJ sign} \begin{bmatrix} \text{INDX} \begin{bmatrix} 2 \\ \text{index} \end{bmatrix} \end{bmatrix} \\
\text{OBJ2 sign} \begin{bmatrix} \text{INDX} \begin{bmatrix} 3 \\ \text{index} \end{bmatrix} \end{bmatrix} \\
\text{PRED cause_rel} \\
\text{ACT0} \begin{bmatrix} 4 \end{bmatrix} \\
\text{ACT1} \begin{bmatrix} 1 \\ \text{index} \end{bmatrix} \\
\text{ACT2 index} \begin{bmatrix} \text{K sit} \\
\text{ACT1} \begin{bmatrix} 2 \\ \text{index} \end{bmatrix} \\
\text{ACT2} \begin{bmatrix} 3 \\ \text{index} \end{bmatrix} \end{bmatrix} \end{bmatrix}\]

It may be useful at this point to summarize what has been said about the type index. It occurs as value of either the attribute INDX, or one of the ACT0-ACT3. Itself, index is so far declared for the attributes ROLE and K(IND). ROLE takes as value a so far unspecified range of role labels, instantiated by agent and theme in examples above. This is illustrated in (18):

\[(18)\quad \begin{bmatrix}
\text{ACTNTS sit} \\
\text{PRED read_rel} \\
\text{ACT0 index} \begin{bmatrix} \text{K sit} \end{bmatrix} \\
\text{ACT1 index} \begin{bmatrix} \text{ROLE agent} \\
\text{K indiv} \end{bmatrix} \\
\text{ACT2 index} \begin{bmatrix} \text{ROLE theme} \\
\text{K indiv} \end{bmatrix} \end{bmatrix}\]

An issue now concerns ‘coreference’. A referent’s role might well vary between its instantiation as subject or object, as putatively in a sentence like John admires himself, a ‘first hunch’
representation of which might have (19) as its semantic part, with identical value for ACT1 and ACT2:

(19) \[
\begin{bmatrix}
\text{PRED admire rel} \\
\text{ACTNTS} \\
\text{ACT1 index} \\
\text{ACT2 index}
\end{bmatrix}
\]

A re-entrancy symbol however covers everything contained in the feature structure to its right, so that in an expanded form of index as suggested, (19) would correspond to the illicit structure (20), where the re-entered feature structures are not identical:

(20) Illicit use of re-entrancy symbol:

\[
\begin{bmatrix}
\text{PRED admire rel} \\
\text{ACT0 index [K sit]} \\
\text{ACT1 index [ROLE agent K indiv]} \\
\text{ACT2 index [ROLE theme K indiv]}
\end{bmatrix}
\]

Given this, we may either want to represent reflexive binding in a more convoluted way than (19), for instance by stating in a specific semantic relation that two distinct indices are co-referring, or enrich the index definition with an attribute with somehow ‘spearheads’ the referential identity as such, like an attribute ‘HEACCITAS’\(^{10}\), or HEAC, with ROLE as a different specification path, whereby the intention behind (19) would come out in the licit feature structure (21):

(21) ‘HAEC(itas)’ - licit use of re-entrancy symbol:

\[
\begin{bmatrix}
\text{PRED admire rel} \\
\text{ACT0 index [K sit]} \\
\text{ACT1 index [HAEC role K indiv]} \\
\text{ACT2 index [HAEC role K indiv]}
\end{bmatrix}
\]

We may assume this as the more detailed structure for stating coreference, but will nevertheless employ the simpler format of (19) as a shorthand notation when role is not explicitly in the discussion. In principle, however, the type index now declares three attributes: HAEC, ROLE and K(IND).

\(^{10}\) The medieval philosophy term for ‘thisness’; cf. David Kaplan’s term ‘haeccity’.

\(^{11}\) link
types at various levels, the system will presumably be a multiple inheritance hierarchy, with the verb-counterparts constituting the ‘leaf’ nodes. If at all constructible, this will obviously be a huge system, with the non-leaf nodes added to the leaf ones. Situation types as such are presumably ‘universal’ in the sense of not being by definition part of a specific language system, however lexicalization is language particular, hence if one is aiming at an ‘all-languages-included’ situation system, this will be even larger than the situation type system of a particular language.

When a system takes such a degree of complexity, a typed feature system as we use here may well use the possibility of not just ordering types in hierarchies, but articulating the hierarchical relations by means of attributes allowing for specification of what sets subtypes of a given super-type apart from each other, and in what respects these subtypes are more specific than the super-types. Figure 1 illustrates this idea with a highly delimited hierarchy for a set of verb-correlated situation types in English: here the higher nodes represent types of a high degree of generality, and the attributes introduce role specifications typical of these types. These attributes are all inherited down the tree, and certain of the lower types in turn introduce new attributes; mention of inherited attributes is made only when their values are identical; this is all in observance of the principles (4).

![Figure 1](image_url)

**Figure 1** Excerpt of possible situation-type hierarchy

This design is in principle not unlike that used in FrameNet, except that in FrameNet, the interplay between frames (corresponding to situation types) and the role attributes they correlate with, is not strictly governed by the principles (4), or any rigorously maintained counterpart of these. As a result, the FrameNet system is quite open to contributions being made, but less formally tractable than a system observing such constraints would be.

Relating to the issue of roles, by having role names as attributes, the design is in principle of the type illustrated in (11) above, so that adopting this course might seem to formally contradict the strategy we have chosen, the latter illustrated in (10). Also, having two formats for representing roles might seem redundant. To motivate such a design, let us first consider how a formal ‘cohabitation’ of the two approaches can be designed.

As attribute hosting situation type, we use AKTRT, as mentioned. The type *throw* in Figure 1 will come out as under AKTRT in the following representation, showing its connection to the ‘established’ role representation from (10) at the same time:
As is seen, the general value of the role attributes is here *haeccitas*, introduced above, whereby one avoids introducing role specifications twice along the same feature path.

Our intuition about the usefulness of such a double set of specifications is as follows: At one level we want an inventory of role labels closely associated with syntactically realized arguments, covering also the cases where such arguments are implicit, as discussed above. At another level we want a somewhat freer navigation space where we can perform the types of analysis expected within lexical semantics; this is a distinction Melchuk recognizes in the contrast xxx – yyy. The role *LAUNCHER* in (22) will be a case in point, representing, e.g., an *arm*, which is essential in throwing, but not represented in the standard argument structure associated with the verb *throw* in English.

To summarize, each situation type and subtype can introduce certain attributes to more closely characterize a situation. These attributes may be seen as role attributes, whereby we have by now considered three formats for role specification – the ‘proto-role’ format mixed with enumeration represented by the ACTn attributes, the specification inside the ROLE attribute, and ‘full-fledged’ role attributes like LAUNCHER as illustrated in Figure 1. The first format is robust and easy to use, the second format depends on a choice of role type hierarchy which we have not yet provided, and the third format has very much the status of a project; they address different depths of specification, and could thereby all be present in the overall design, but should be stepwise developed according to the needs of research.

1h Situation types and Aktionsarten

In (22) above, the attribute *AKTRT* is serving as ‘host’ of the entire situational description. Arguably, Aktionsart is part of a full situational description, but should as a minimum introduce the most common Aktionsarten, as in the following hierarchy of types and accompanying features, following in essence Vendler (1967) and Smith (1991, 1997):
The idea will be to integrate systems like those in Figure 1 and 2 in one and the same type hierarchy, such that each type in Figure 1 will inherit from one type in Figure 2, and situation types will thus be characterized both by role features and Aktionsart features, which will seem reasonable.

**II Illustrating the deployment of global specifications**

We have now established some of the main structures of global sentence specification, i.e., for representing properties of a construction per se, without regard to the compositional structure of the sentence. It is conceivable that many regularities (or necessities) relating to factors at the global level can be stated with reference to this level exclusively, thus making it possible to ‘levitate’ some aspects of grammatical analysis from the morpho-syntactic composition of the sentence. This can be illustrated by the long-recognized dependency between the Aktionsart accomplishment and features such as definiteness and count-specificity of an incrementally affected object, exemplified by sentences such as (23):

(23) German:
   a. *Johan isst* [Activity]
      Johan eats
   b. *Johan isst von dem Apfel* [Activity]
      Johan eats of the apple
   c. *Johan isst den Apfel* [Accomplishment]
      Johan eats the apple
   d. *Johan isst drei Äpfel* [Accomplishment]
      Johan eats three apples
   e. *Johan isst Äpfel* [Activity]
      Johan eats apples

The feature ‘**BOUNDDED** +’ represents that an NP is either definite, or specific, or with a quantifier or numeral determiner. The commonality between the two sentences (23c,d) with accomplishment as Aktionsart is represented by means of the partial AVM (24):
A schema such as (26) will state these possibilities (accomplishment being characterized by ‘COMPL +’, cf. Figure 2):

We are not thereby saying exactly how this schema can act inside of a grammar, our focus at this point being on the expressive potential of the Typed Feature Structure system outlined till now.

Ij Situating the deployment of Typed Feature Structures

We have now outlined and illustrated the essential content of the TFS system to be used, a system corresponding to the upper colored items in the overview figure (2), repeated:

We move on to presenting the way in which this TFS can sustain a valency and construction type typology. This will involve the presentation of a string representation format, and how it is interlinked with TFS.
2. A typology of valence- and construction types based on global parameters

2a Valunits and enumeration

Here we outline a procedure by which one can build an enumeration, and in turn an ontology, of valence and construction types. It partly resides in a string-based system for summarizing construction properties, described in detail in Hellan and Dakubu 2010, which has been used in establishing fairly large-scale construction inventories for a few languages from Germanic, Niger-Congo and Ethio-Semitic. Construction types, as well as valence types, in this system are represented by strings of labels and hyphens, where each minimal label – which we may call a valunit, reflecting a minimal unit of valence information - represents a property of the construction. CL’s potential resides in a comprehensive stock of valunits from a range of language types, in transparency of the manner in which these units are combined into construction representations, called construction templates, and in the flexibility of these combinations. Below is an example of how valunits combine into construction templates:

(27) a. Examples of valunits: each unit specifies a property of a construction X:

<table>
<thead>
<tr>
<th>Valunit</th>
<th>Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>v</td>
<td>X is headed by a verb</td>
</tr>
<tr>
<td>ditr</td>
<td>X is ditransitive</td>
</tr>
<tr>
<td>obPostp</td>
<td>X has as object a postpositional phrase (or an NP with locative head)</td>
</tr>
<tr>
<td>suAg</td>
<td>X has a subject carrying the role of Agent</td>
</tr>
<tr>
<td>obEndpt</td>
<td>X has an object carrying the role as Endpoint</td>
</tr>
<tr>
<td>ob2Mover</td>
<td>X has a second object carrying the role as Mover</td>
</tr>
<tr>
<td>PLACEMENT</td>
<td>X expresses a situation of type Placement</td>
</tr>
</tbody>
</table>

b. Combination of valunits into a template, where the construction X is represented as having all of the properties represented by the individual units:

v-ditr-obPostp-suAg-obEndpt-ob2Mover-PLACEMENT

An enumeration of construction types using this notation will be called a v(alence)-profile or c(onstruction)-profile. (28) illustrates part of such a v-profile displaying specifications for some construction types (all ditransitive), out of a full set of 200 specifications, for Ga; for convenience, illustrative examples are given for each type:

(28) v-ditr-suAg_obAff-ob2Instr-CUTTING
Nuu le baŋ le klænte
man DEF AOR slash 3S cutlass
‘The man slashed him with a cutlass.’

v-ditr-suAg-ob2Instr-PENETRATION
E-gbu le kækla
3S.AOR-pierce 3S knife
‘He stabbed him with a knife.’

v-ditr-suAg_obLoc-ob2Res-CUTTING
Nuu le baŋ mi-hi e gbɛ
man DEF AOR slash 3S.POSS-face scar
‘The man cut marks on my face.’

v-ditr-suAg_obTrgt-ob2Endpt-COMMUNICATION
Mii-da bo shi
1S.PROG-thank 2S down
‘I thank you.’

v-ditr-suAg_iobTrgt-obThmover-COMMUNICATION
E-fɔ mi ninε
3S.AOR-throw 1S hand
‘She waved to me; invited me.’

vHab-ditr-suNrg-ob2DECLcmp-obSens-ob2Thsit-COGNITION
E-fe-ɔ mi akr noko brɛ mli
3S-do-HAB 1S COMP something is not inside
‘It seems to me that it isn’t true’
By a construction ontology we mean a subsumption hierarchy of construction types. Conceivable items in such a hierarchy could be typed feature structures, for instance clause-representing AVMs of an HPSG grammar. These are complex objects, but one could define sub-AVMs to represent specific properties of the construction, and thus more abstract construction types; for instance, relative to a clausal AVM as depicted in (29), representing a clause in German with *beissen* ‘bite’ as main verb,

$$\text{(29)}$$

$$\begin{align*}
\text{HEAD} & \text{verb} \\
\text{SUBJ} & \text{[HEAD[CASE nom]} \\
\text{INDEX} & \text{[ROLE agent]}] \\
\text{OBJ} & \text{[HEAD[CASE acc]} \\
\text{INDEX} & \text{[ROLE patient]}] \\
\text{PRED} & \text{beissen-rel} \\
\text{ACT1} & \text{1} \\
\text{ACT2} & \text{2} \\
\text{AKTRT} & \text{achievement}
\end{align*}$$

a ‘sub-AVM’ as now alluded to could be (30),

$$\text{(30)}$$

$$\text{GF [SUBJ [HEAD [CASE nom]]]}$$

representing the clausal property of having a subject with nominative case. This AVM, as will be noted, is a subpart of (29), and can be seen as a super-type of it. So can also (31), representing the clausal (multi-)property having a subject with nominative case and an object with accusative case:

$$\text{(31)}$$

$$\text{GF [SUBJ [HEAD [CASE nom]] [OBJ [HEAD [CASE acc]]]}$$

The AVM (30) will in turn be a supertype of the AVM (31). Thus, once we focus on restricted parts of clausal AVMs, subsumption relations may be possible to establish.

Formulas like (30) and (31) are still a bit cumbersome for being used as node labels in an ontology tree – in this function the labels in (32) would be more convenient, based on valunits:

$$\text{(32)}$$

a. suNom (for (30))

b. suNom-obAcc (for (31))

c. suNom-obAcc-suAg-obPat-ACHIEVEMENT (for (29))

Using those, the hierarchy in question can be expressed as in (33):

$$\text{(33)}$$

$$\text{suNom}$$

$$\text{suNom-obAcc$$}

$$\text{suNom-obAcc-suAg-obPat-ACHIEVEMENT}$$

Given such a composition of node labels, subsumption relations can be automatically computed, reflecting the circumstance that the valunit set \{suNom\} is a subset of the set \{suNom, obAcc\}, and \{suNom, obAcc\} is a subset of \{suNom, obAcc, suAg, obPat, ACHIEVEMENT\}. 

15
Link between valunits and Typed Feature Structures (TFS)

What we have seen above, thus, is a potential design where, from a valence profile expressed in terms of valunits, one can automatically or semi-automatically generate a construction ontology. This design at the same time involves a systematic link to TFS, whereby the correspondences stated in (32) are not just stipulations. This link between the CL system and the TFS representations resides in the circumstance that the valunits of the CL expressions systematically match top level types and attributes in the TFS. Examples of such matches are indicated in the correspondences in (34) below reflecting simple analytic statement such as ‘head is a verb’, ‘construction is transitive’, ‘subject is an Agent’, ‘object is Incrementally affected’, and ‘Aktionsart is Accomplishment’, a set of statements which together describe a sentence like The boy eats the apple (serving as the sentence X – cf. (27a) above):

\[
\begin{align*}
\text{(34)} & \quad \text{v} \quad \text{tr} \quad \text{suAg} \quad \text{obAffincrem} \quad \text{ACCOMPLISHMENT} \\
\end{align*}
\]

In the valunit string (template) (35),

\[
\begin{align*}
\text{(35)} & \quad \text{v-tr-suAg_obAffincrem-ACCOMPLISHMENT} \\
\end{align*}
\]

the hyphenation and underline notation is formally construed as unification, whereby the AVMs in the right column of (34) are ‘assembled’ to the structure (36), the TFS (36) thus counting as inter-convertible with the string (35):

\[
\begin{align*}
\text{(36)} & \quad \text{[HEAD verb]} \\
\text{GF} \quad \text{SUBJ [INDX \[\] [ROLE agent]]} \\
\text{OBJ [INDX \[\] [ROLE aff-increm]]} \\
\text{ACTNTS ACT1 \[\] \[\]} \\
\text{ACT2 \[\] \[\]} \\
\text{PRED 'eat'} \\
\text{AKTRT accomplishment} \\
\end{align*}
\]

Structure of templates

Inside of a template, the area occupied by each type of valunit is referred to as a slot. Slot 1 consists of a label for Part of Speech of the head of the entire construction, including the category of possible formatives marked on the head. Slot 2 consists of a label for argument structure/valency specification - like intr (intransitive), tr (transitive), ditr (ditransitive), and varieties thereof (see below). Slot 3 consists of one or more labels for specification of syntactic constituents, identified by their grammatical function (subject, object, etc.). Slot 4 consists of one or more labels for specification of participant roles: agent, theme, instrument etc. Slot 5 consists of a label for aspect and Aktionsart, written in CAPS. Slot 6 consists of a label for the situation type or general
semantics of the construction, also written in CAPS. Thus, all of the aspects of global specification discussed above are addressed in a template. Slots 1 and 2 are obligatorily filled, the others not.

The valunits defined for the various slots are distinct, hence no valunit specification is ambiguous with regard to which type of information it concerns. Likewise, no valunits are formally distinguished only in terms of capitalization vs. not. The valunits are interconnected exclusively by ‘-’ (hyphen) or ‘_’ (underline), in such a way that cooccurring valunits inside a slot are interconnected by ‘_’, and valunits across two slots by ‘-’.

Derivational history regarding argument structure is reflected in a template, so that an example like (15) above will have the (partial) template

\[ v- dbobCs- obCuAg \]

meaning that it is a double object construction derived by means of causativization, and that the object represents an Agent, but with a derivational history where it stems from being a subject. These and further capacities of the valunit system are explained and illustrated in later chapters, and implemented in the grammatical demo accompanying this exposition.

In terms of the diagram (2), we have now connected the blue-colored items as described in section 1, to the red-colored items described in the present section.

(2)

```
Global parameters
          / \
TFS representations   String representations
|    |        |
Grammar cores   Construction- and Valency enumerations
|    |        |
Grammar structures Valence ontologies
|    |        |
Valence corpora IGT-cum-valency
```

We subsequently show how the TFS system can be used in building grammar cores.

3. Deriving a grammar core and in turn grammar structures from the global parameters

3a Adding grammatical functions to an HPSG combination mechanism

To derive grammars, the specification format so far surveyed must be embedded in a grammar formalism. If we let that be the standard HPSG design, this already has a TFS format, and most of the specifications given above can be accommodated in the HPSG formalism. Only one aspect of the present global parameter design does not have a direct counterpart in the HPSG formalism, namely the grammatical functions. We outline how to accommodate those, and thus merge the global parameter model with an HPSG type grammar formalism.

According to the valence specification in the standard HPSG formalism, the information associated with a transitive verb will be as in (37), applying the semantic categories and features mentioned above. Here, the semantic indices of the object NP and the subject NP are copied into the ACTNTS specification, which is also the ACTNTS specification of the dominating node, in accordance with the combination schema (38) for verb-object combination, and a similar schema for the combination between subject and VP:
The empty valence lists in the S node provided through these mechanism give no information about grammatical functions at the ‘global’ level, hence to accommodate the full global specification in this framework, we need to align the specification types in (37)/(38) with the format for GF representation outlined above. This can be achieved through the addition of the GF feature complex in (37)/(38), together with a requirement that the member on the COMPS list in (37) is identical to the value of the path GF.OBJ in the added features, and that the member on the SPR list is identical to the value of GF.SUBJ. This is displayed in the verb specification schema (39):

A combination for a transitive clause starting with (39) as main verb will thereby end up with (40), which includes the global representations we have been assuming, but reflects the combinatorial algorithm by which an HPSG grammar functions as a parser, here through the empty SPR and COMPS lists licensing the TFS in (40) as a formally valid parsing output for a sentence:

Having thereby enriched the composition mechanism of standard HPSG with the addition of grammatical functions, we have, ‘in return’, situated the global specification format within an HPSG type grammar formalism. We have thus partly constructed a grammar core based on the
global parameters, and prepared the ground for constructing partial grammars corresponding to valence- and construction profiles.

This core will be described in chapter xxx. It by now accommodates a large set of construction types from various language types, such as about 30 types of Serial Verb Constructions, 30 verb-extension constructions like those found in Bantu, 20 secondary predicate constructions as found in Germanic, defined through a balance between lexical types, lexical rule types and syntactic combination types, and sustained by a central inventory of type definitions. With its coverage of features from many languages in one system, the Core may be called a ‘Pan’-grammar.

3b Inducing grammar structures from valence- and construction profiles

From any template in the CL formalism, one can induce a partial grammar covering the information encoded in that template. That this should be in principle feasible is suggested by the correspondence table in (34) above, although by itself this table of course does not constitute a partial grammar. What we need is a way of construing the template itself – e.g. the template ‘v-ditr-obPostp-suAg_obEndpt_ob2Mover-PLACEMENT’ from (27b) – as a sign level type in the grammar, whereby that type, and the types serving as values of the various attributes inside that type, sum up to partial declarations of a grammar. We now show how this can be achieved.

The following will be among the type definitions of the grammar in question (where ‘:=’ means ‘is a subtype of’ and ‘&’ is the operation of unification, as defined in the tdl code – see Copestake op. cit.):

\[
\text{v-ditr-obPostp-suAg_obEndpt_ob2Th-PLACEMENT} := \\
v \& \text{ditr} \& \text{obPostp} \& \text{suAg} \& \text{obEndpt} \& \text{ob2Th} \& \text{PLACEMENT}.
\]

Such definitions are given for all the templates in a valence-profile, so that a convenient name for this kind of type file for a grammar may be template-types, ranging from 100 to 300 definitions for a language.

These definitions are – technically speaking - based exclusively on unification of types which correspond to valunits in the CL design, in the grammar to be referred to as valunit-types. Examples of definitions of these are given in (42), using the tdl code again:

\[
\begin{align*}
\text{v} & := \text{sign} \& \{\text{HEAD headverb}\}. \\
\text{ditr} & := \text{ditr-lex}. \\
\text{obPostp} & := \text{sign} \& \{\text{GF.OBJ poss-sign} \& \{\text{ACTNTS.PRED spatial-coord_rel}\}\}. \\
\text{suAg} & := \text{sign} \& \{\text{GF.SUBJ.INDX.ROLE agent}\}. \\
\text{obEndpt} & := \text{sign} \& \{\text{GF.OBJ.INDX #1} \& \{\text{ROLE endpnt}, \text{ACTNTS.DIR.ACT2 #1}\}\}. \\
\text{ob2Th} & := \text{sign} \& \{\text{GF.OBJ.INDX.ROLE theme-locative}\}. \\
\text{PLACEMENT} & := \text{sign} \& \{\text{SIT-TYPE placement_sit}\}.
\end{align*}
\]

Items that here occur on the right side of ‘:=’ are all defined in the terms of the global parameters, with types like sign, ditr-lex, poss-sign having further definitions inside of this system in turn.

Seen from the perspective of inducing partial grammars, for the creation of any specific grammar, the overall flow of valence-information exchange is as in (43):

\[
\begin{align*}
\text{v-profile-of-L} & \quad \rightarrow \quad \text{valunit strings, templates of L} \\
\text{v-profile-of-L} & \quad \rightarrow \quad \text{inter-defininitions between Core and valunit types} \\
\text{Grammar structure-of-L} & \quad \rightarrow \quad \text{Argument syntax-of-L} \\
\text{Lexicon-types-of-L} & \quad \rightarrow \quad \text{‘Core’ (‘Pan’-Grammar)}
\end{align*}
\]
The file with definitions like (42) will, compared to *template-types*, have a more language independent status, since valunit types are expected to occur across template types and across languages. Obviously, also the definition of template types is in principle language independent, since the same template types can occur across languages. Still, the point where a specific language is represented is through a set of template-types relative to that language, formally a sub-set of the total template-types inventory.

This is now where typologies of valence- and construction types can be investigated: relative to the all-inclusive *template-types* inventory, how much larger is this than the inventory for a single language; how much overlap is there between the single-language inventories, and which profiles of overlap do we find as we go from language type to language type, and from language family to language family? A study of language typology, based on valence- and construction profiles, here meets with a strategy of grammar induction, based on a general design of TFS for global parameters.

3c Further aspects of grammar induction

We have shown how all of the colored items can be interconnected, giving us grammar structures based on global parameters and valence profiles:

\[\text{(2) Global parameters}\]

\[
\text{TFS representations} \quad \langle - - - \rangle \quad \text{String representations}
\]

\[
\text{Grammar core} \quad \langle - - - \rangle \quad \text{Construction- and Valency enumerations}
\]

\[
\text{Grammar structures} \quad \text{Valence ontologies}
\]

\[
\text{Valence corpora}
\]

\[
\text{IGT-cum-valency}
\]

Filling the grammar structures with lexical instantiations of the lexical types, and morphologyntactic instantiations of the grammar structures, can of course happen in many ways. One line, projecting from what has been described, is building valence corpora with the CL code as annotation code, and combining these corpora with IGT, and do grammar induction from there (Hellahan and Beermann 2011, to appear).

\[\text{(44) Lexically and morpho-syntactically instantiated Grammar-of-L}\]

\[
\text{Lexicon-items-of-L} \quad \text{Morphology-of-L}
\]

\[
\text{IGT-of-L}
\]

\[
\text{inter-defininitions between Core and gloss-units}
\]

\[
\text{‘Core’ (‘Pan’-Grammar)}
\]

To illustrate, the perfective verb form *ettee* in Ga with an annotation as indicated in the TC annotation snippet in (45) is associated with a lexicon entry and an inflection rule as summarized in the lower part of (45); attributes such as ‘ORTH’, ‘AKTRT’ etc., and value categories such as *v*-*lxm*, *perf* and *word*, are all defined in the ‘pan’-grammar, and inherited by the actual grammar being derived.
In the following, we will keep this construction type as a case where, arguably, the argument structure of the construction comprises more than the argument structure of the main verb, and where the difference between the argument structures follows the pattern mentioned. The discussion will address how a compositional grammar can construe such a situation. Empirically we will stay with Germanic languages.

\[12\] ‘By themselves’, annotated valence corpora and valence lexicons – both being aided by the availability of construction type inventories and valence type inventories – are probably the more useful creations for the linguistics community, compared with computational grammars (in the sense of ‘deep’ grammars as here considered). In this perspective, such grammars are tests – tests for consistency, and for whether one has managed to find the right abstractions, notions and classifications on a multi-language basis. This is also a reason why there is a point in creating construction type inventories and valence type inventories ‘with’ HPSG.

\[13\] Resp.: Kjøleskapet ble spist tomt; Pasientene ble sunget friske.
Before entering the discussion, we mention a construction type where the argument structure of the construction does comprise more than the argument structure of the main verb, but where there are arguable more than one main verb – this is the situation often found in multi-verb constructions such as Serial Verb Constructions (SVCs). An area where these are typically found is in West African languages, where they appear as a sequencing of any number of VPs, with pervasive uniformity between the verbs, both in their morphology and regarding their arguments. Interpretations range from temporal sequences of events reflecting the sequencing of VPs, to pair-wise more special combinations. (47) is an example of the latter, from Ga (Dakubu 2010):

(47) Á-gbele  gbɛ  á-ha  bo  
   3.PRF-open  road  3.PRF-give  2S  
   V  N  V  Pron  
   ‘You have been granted permission.’

This SVC has two verbs, both with an expressed object; their subjects are identical, and likewise their aspects. Questions are whether the separate verb meanings (informally using the English glosses) ‘add up’ to constituting a situation type that could be named ‘permission’, as the English free translation would suggest; also not indicated is whether V1 and V2 are syntactically related in the fashion of head plus complement, head plus adjunct, or as coordinated conjuncts, or something else - the question whether these are at all adequate categories in the analysis of this construction type is a central issue in the analysis of SVCs. Both points suggest that the concepts of verb valency on the one hand, and the concept of constructional/global argument structure on the other, be kept in principle even more separate than what we concluded in the above discussion of Germanic structures.

4b  Constructions vs. ‘Constructions’

The framework now described takes ‘compositionality’ as a given, in the sense that the structure assigned to a sentential construction is a function of the structure of the main verb and the structures of the items it combines with. In our sketch so far this assumption has been without exception, but it is a view being partly challenged in Construction Grammar (also in an HPSG version thereof, under the name ‘Sign based Construction grammar (SBCG)’; and even on ‘HPSG immanent’ grounds there are construction types that may feel rather ‘non-compositional’; we therefore address some of these.

First, expressions like (48) presumably mean the same:

(48) a. “You are wrong” (English) 
   b. “Du tar feil” (Norwegian) (literally: ‘you take wrong’) 
   c. “Tu te trompes” (French) (literally ‘you wrong yourself’) 

These constructions establish their content by means of word combinations that in some sense of ‘literal meaning’ compose the content in quite different ways across the languages, and for none of these ways can one say that one is more or less ‘literal’ or ‘figurative’ than any of the others. They are as ‘direct’, and in a Saussurean sense ‘arbitrary’, as word level entities normally are, and yet they are composed of more than one word, through recognized rules of composition. Hence they are instances of what are commonly called ‘Multi-Word Expressions’ (MWEs).

To discuss the case more closely, what may prompt the perception of a combination like (48b) as being non-compositional, is that the verb ta has standard uses implying gaining possession of the referent of the object, which is not the case in (48b). It thus looks as if the meaning of ta in this case is suppressed or overruled, and now under the control of the ‘construction’.

This case contrasts with commonly quoted locutions like ‘kick the bucket’, which, although ‘kick’ and ‘bucket’ in no way compose to convey the content ‘die’, may be seen as a fully compositional expression of a situational image which, by convention of ‘preserved metaphor’, is used as label of a specific situation type. (There also happens to be a word “die” in this same language, naming this same meaning and counting

14 Goldberg 1996, 2006 
as the official way of expressing it.) This is not the case for (48b), and we take the latter as a more representative case of putative ‘non-compositionality’ than ‘kick the bucket’.

A strategy for formally rendering such a combination as *compositional* could be by distinguishing multiple variants of the verb *ta*, and let one of them carry a special meaning that in combination with *feil* would induce the meaning of (48b). This would be a ‘meaning’ never attested independently of the occurrence of *feil* as object, but formally there is nothing preventing such a move.

Technically the strategy could be implemented by marking words with ‘sense indices’ consistently 1-to-1-related to their meanings, so that in *du tar feil*, the verb *ta* would carry a different sense index than it does in *jeg tar mat* (“I take food”). The standard way of assigning such marking in the style presently used is by defining PRED-values distinguished by integers, such as in the possible value expressions *ta_1_rel*, *ta_2_rel*, *ta_3_rel*, *ta_4_rel*, ... The PRED-value of *ta in du tar feil* could then for instance have number 16 in such an inventory, and one would know that none of the semantic expectations going along with the other “*ta*-variants would carry over to this case, thus, e.g., excluding inferences which imply taking possession or control over something. The relevant lexical entry would include a COMPS list consisting of an NP required to be headed by “*feil*” (perhaps also to be a ‘bare’ singular).

It is obvious that a plain numbering of verb senses inside of a monolingual grammar provides little basis for obtaining an interesting representation of shared meaning, as one might want for an example set like (48) – the numbering even in its own enumeration is arbitrary, and as far as multilingual use of the formalism is concerned, since verbs are not shared between languages, there are not even sequences of numberings to compare. To instead establish a representation of the common meaning of cases like (48), one would have to construct a point in a semantic space representing this exact meaning. This would have to be in an ontology of predicates, or situation types, as in the fragment shown in Figure 1 in chapter 1, and this ‘point’ would be included in the lexical entry in question such as in (49), for convenience here representing the situation type as *se-tromper*; the representation skirts the issue of whether the object in this case would correspond to an ACT2:

(49) 

Against this rather formalistic stance, it could be maintained that many of the MWE type expressions are half-frozen metaphors, all linked to a ‘core’ through different aspects of the meaning of this core, or even to a small set of cores, with similarities in the design of family resemblance, and that this should be revealed in the lexical representations. Then, simply stating ‘metaphor’ in a meaning description is of course not enough, since there are many aspects to a verb meaning from which a metaphorical extension could take place. Clearly, in this respect a schema like (49) is just a skeleton onto which further meaning description could be built, but nevertheless a necessary step to enable such.

4c The pan-grammar approach – other instances: the HPSG Grammar Matrix

Where the present system will use a TFS design as exemplified in (50), the Matrix uses the structure design exemplified in (51) (for an impressionistic view):

(50)
Orthogonal to this difference in formal design (but both obeying the LKB constraints) are the immediate purposes of the approaches, to which we return at a later point.

References
Dakubu and Hellan, to appear. A format for multi-lingual valence classification. In Hellan et al. (ed.).


