A Unified Valence Resource

Lars Hellan

NTNU

LACompLing 2021, December 15-17, 2021

Verb Valence

The main component of a language is obviously its lexicon, with word meanings that represent a communal understanding of the world at large. Grammar is more like a membrane on top of the lexicon, administrating how words are put together. Word valence is how words accommodate their meaning(s) relative to the patterns made available by grammar; in particular, verb valence is how verbs accommodate their meanings relative to the patterns of what grammar offers as *sentences*. Verbs select among the patterns in highly different ways, and the array of possible patterns is rich; however, it is finite. For a given language, it is therefore possible to establish a total view of these selections-per-verb. The patterns are not meanings, but once we assume that a verb's meaning affects its choice of patterns, or 'frames', even an assembly of predominantly formal frames can be a contribution to a study of how verbs accommodate their meanings to grammar. We just need to study this for very many verbs.

A Unified Resource

With this as a general perspective, the present project of Norwegian verb valence establishes what is the totality of relevant patterns made available by the grammar of Norwegian, and in turn how the verbs avail themselves of the patterns, as their ‘valence frames’.

The unified resource consists of a Valence Catalogue (‘NorVal’), two computational grammars (‘NorSource’ and ‘TypeGram’), and channels for exposing and using their content, all with a common encoding. The Catalogue has a direct interface for linguistic specification.

We outline the main aspects of the resource as it has been developed for Norwegian, and sketch designs whereby it can serve in a multilingual setting

A Valence Catalogue for Norwegian

The catalogue organizes valence information for most verbs of the language; the information

- can be easily encoded in the relevant files
- is easily searchable
- holds in-depth information of grammar and structural meaning

A description is given in (Hellan 2022). Some main points follow:

Overall organization - *Lexval* entries

6,300 verb lemmas are represented. Given that a verb can have more than one valence frame, a valence resource needs two kinds of entries, viz. entries for *verb lemmas*, and entries for *a verb with a given frame*; the latter we refer to as a '*lexically instantiated valence frame'*, for short '*lexval*'. There are currently 15,700 lexvals.

Lexval entry illustrated

The format of a *lexval* entry is illustrated by one of the frame environments for the verb lemma *huske* ‘remember’ in (2), instantiating the general coding pattern in (1):

(1) Lemma – selectedItem (if any) __ FrameType

(2) huske-på__intrObl-oblDECL

(2) reads as an entry with the lemma *huske*, the selected item *på* (‘on’), and the frame type ‘intransitive with oblique’, where the oblique argument consists of the preposition *på* and a declarative clause, as in (3):

(3) *Han husket på at det var søndag* ‘he remembered that it was Sunday’

Overall organization - Lemma entries and *valpods*

The format of a multivalent *verb lemma* entry is illustrated in (4), where each constituting lexval is represented with ‘V’ as placeholder for the lemma (‘EqSuInf’ stands for ‘infinitive equi-controlled by subject’); such a structure we call a *valpod*:

(4)

huske:V__intr & V-på__intrObl-oblDECL & V-på__intrObl-oblEqSuInf & V-på__intrObl-oblINTERR & V-på__intrObl-oblN & V__tr & V__tr-obDECL & V__tr-obEqSuInf & V__tr-obINTERR

The structure of a valpod is essentially a *set*, although represented with ordering conventions among its members (e.g., intransitives before transitives).

The number of *multi-membered* valpods (like (4)) is about 3360.

Frame Types

There are currently *304 Frame Types*, defined in terms of:

- valence profiles like *intransitive, transitive, ditransitive, copular*, and more;
- argument parameters such as ‘direct’ vs. ‘oblique’;
- grammatical functions such as ‘subject’, ‘object’, etc.;
- for an argument to be noun-headed vs. being an *embedded clause* (*declarative, interrogative*, or *infinitival*, in canonical or ‘extraposed’ position);
- for an argument to relate syntactically and semantically to the same predicate or not;
- occurrence of particles;
- basic structures of ‘Logical Form’ as far as argument structures go.

Notation for Frame Types

The notation for frame types uses the system *Construction Labeling* ('CL') (cf. Hellan and Dakubu 2010, Dakubu and Hellan 2017, and Hellan 2019), which characterizes verb-headed constructions and verb valence frames through strings of symbols built up in the following way:

(5) Head-POS – Global Label – Argument Label1- ArgumentLabel2 - …

An *Argument Label* describes a *constituent* of the construction or frame.

A *Global Label* categorizes the *construction/frame as a whole*.

Argument Labels are composed of a prefix indicating the grammatical function (GF) of the constituent, followed by one or more parts indicating inherent properties of the constituent. For instance, `obDECL` is an Argument Label with `ob` as GF-indicating prefix and `DECL` indicating that the constituent is a declarative clause.

Global Labels consist minimally of a symbol for overall valence, such as `tr` for 'transitive', in many cases with additional symbols indicating further structure. Thus, the frame representation `intrObl-oblDECL` in (2) has `intrObl` as Global Label.

Notion of GF	Representation as prefix in Argument Label
subject	su
direct object	ob
indirect object	iob
complement	comp
oblique	obl
oblique2	obl2
presented	pres
secondary predicate	sc
extraposed	expn
extralinked	exlnk
identifier	id
adverb	adv
particle	prtcl

ArgumentLabel	Carrier of the GF	Embedded in argment-PP	Semantic role or Function	Target of dep	Sem-arg status
suExpl	Expl				
obExpl	Expl				
expnDECL	DECL				
exlnkDECL	DECL				
prtcl					
oblN		N			
suDECL	DECL				
obDECL,	DECL				
oblDECL		DECL			
iobRefl	Refl				
obRefl	Refl				
oblRefl		Refl			
scPPrefl		Refl			
suDir			Dir		
obDir.					
oblLoc.	Loc				
oblPRTOFob			PRTFob		
scSuNrg				Su	Nrg
scObNrg,				Ob	Nrg
scObNrgCsd			Csd	Ob	Nrg
obEqSuInf	Inf		Eq	Su	
expnEqIobInf	Inf		Eq	Iob	

Global Label	GFs declared	Semantic arity	Subject expletive	Predication target ('Nrg' ='not sem.arg of verb')	Correlate of extrapos-clause
intr	su	1			
tr	su, ob	2			
ditr	su, iob, ob	3			
impers	su	0	X		
intrObl	su, obl	2			
trObl	su, ob, obl	3			
impersObl	su, obl	1	X		
intrScpr	su, sc	1		subj Nrg	
intrScpr	su, sc	2		subj Arg	
trScpr	su, ob, sc	2		obj Nrg	
trScpr	su, ob, sc	3		obj Arg	
intrPresnt	su, pres	1	X		
trPresnt	su, ob, pres	2	X		
intrExpn	su, expn	1	X		'logical subject'
trExpnSu	su, ob, expn	2	X		'logical subject'
trExpnOb	su, ob, expn	2	X		'logical object'
copAdj	su, sc	1			
copIdN	su, id	2			

Clausal arguments

Among the totality of the 15,700 lexvals in the resource, 1140 lexvals contain an argument specified with `DECL` as a defining label, 849 lexvals contain an argument specified with `INTERR` as a defining label, 1066 lexvals contain an argument specified as a *controlled infinitive*, and 267 lexvals contain an argument specified as an absolute infinitive.

This means that more than 3000 lexvals, or about 20% of all the lexvals, contain a clausal argument.

Distributions are shown in the following tables.

Univalent verbs – no clausal arguments

Number	Frame type
2,145	transitive
656	intransitive
88	transitive with light reflexive object
65	intransitive with oblique
42	transitive with a particle
36	intransitive with a directional subject
28	ditransitive
22	impersonal
16	transitive with directional object,
15	transitive plus oblique,
14	ditransitive with light reflexive as indirect object ,
12	transitive with light reflexive object plus oblique ,
12	transitive with light reflexive directional object,
11	transitive with a particle,
10	transitive with a particle and light reflexive object, …
…	….
2	subject-controlled infinitive as unique frame: *plikte å*, *unnlate å*
0	declarative or interrogative argument, extraposed clause, or absolute infinitive

Number of lexvals with clausal arguments of type ‘declarative’

ArgumentLabel	Instances
suDECL	87
obDECL	460
oblDECL	485
expnDECL	89
oblExlnkDECL	5
DECL	**1142**

Number of lexvals with clausal arguments of type ‘interrogative’

ArgumentLabel	Instances
suINTERR	22
obINTERR	235
compINTERR	77
expnINTERR	48
oblExlnkINTERR	1
oblINTERR	432
INTERR	**849**

Number of lexvals with a ‘controlled infinitive’ as argument

ArgumentLabel	Instances
`suEqObInf` (‘subject is an infinitive controlled by object’)	21
`obEqSuInf` (‘object is an infinitive controlled by subject’)	135
`obEqIobInf` (‘object is an infinitive controlled by indirect object’)	51
`oblEqSuInf` (‘oblique is an infinitive controlled by subject’)	291
`oblEqObInf` (‘oblique is an infinitive controlled by object’)	476
`expnEqObInf` (‘extraposed is an infinitive controlled by object’)	31
Controlled Infinitive with infinitival marker	**1066**

Clausal arguments of types ‘absolute infinitive’ and ‘bare infinitive’

ArgumentLabel	Instances
suAbsinf	17
obAbsinf	35
oblAbsinf	160
expnAbsinf	28
Absinf total	**267**

ArgumenLabel	Instances
obEqIobBareinf (‘object is a bare infinitive controlled by indirect object’)	2
scBareinf	18
obEqBareinf (‘object is a bare infinitive controlled by subject’)	2
Bare controlled infinitives total	**22**

Oblique clausal arguments summarized

For the arguments specified with `INTERR` as a defining label, about half of them appear in an oblique PP, the same holds for arguments with *infinitive* as a defining label, and it holds for almost half of the arguments specified with `DECL` as a defining label. The exact numbers of each are summarized below:

ArgumentLabel	Instances
`oblDECL`	485
`oblINTERR`	432
`oblEqSuInf`	291
`oblEqObInf`	476
`oblAbsinf`	160
Oblique clausal arguments	**1844**

Semantics behind?

Given the verbs, with the prepositions, that fall into each group, one can investigate common properties of the verbs and the prepositions.

It appears that 217 verbs can take all three kinds of clausal arguments; can one find factors that distinguish them from other verbs, thus different from those verbs that take only declaratives or only interrogatives, etc.?

Results in these respects may well extend across languages.

.

Querying in *lexval* and *valpod* files

Valpods and lexvals are defined on the format of one-line sequences of strings, and the main mode of search in the files is by string identity, with query specifications ranging from full valpods down to parts of CL-labels, in accordance with the ‘morphological’ partitions exemplified above. The statistics shown earlier is based on results obtained by query by string.

However, it may be desirable to have recourse to further modes.

For instance, suppose that we want to know whether the lexval types ‘`V__intr`’ and ‘`V-opp__trPrtcl`’ are instantiated in any valpod. Suppose that among the defined valpods, one valpod has the form:

(6) `:{V__intr & V__tr & V-opp__trPrtcl}`

The two lexvals in question are the first and last in this triple, but if one searches for their joint occurrence via the search string (7),

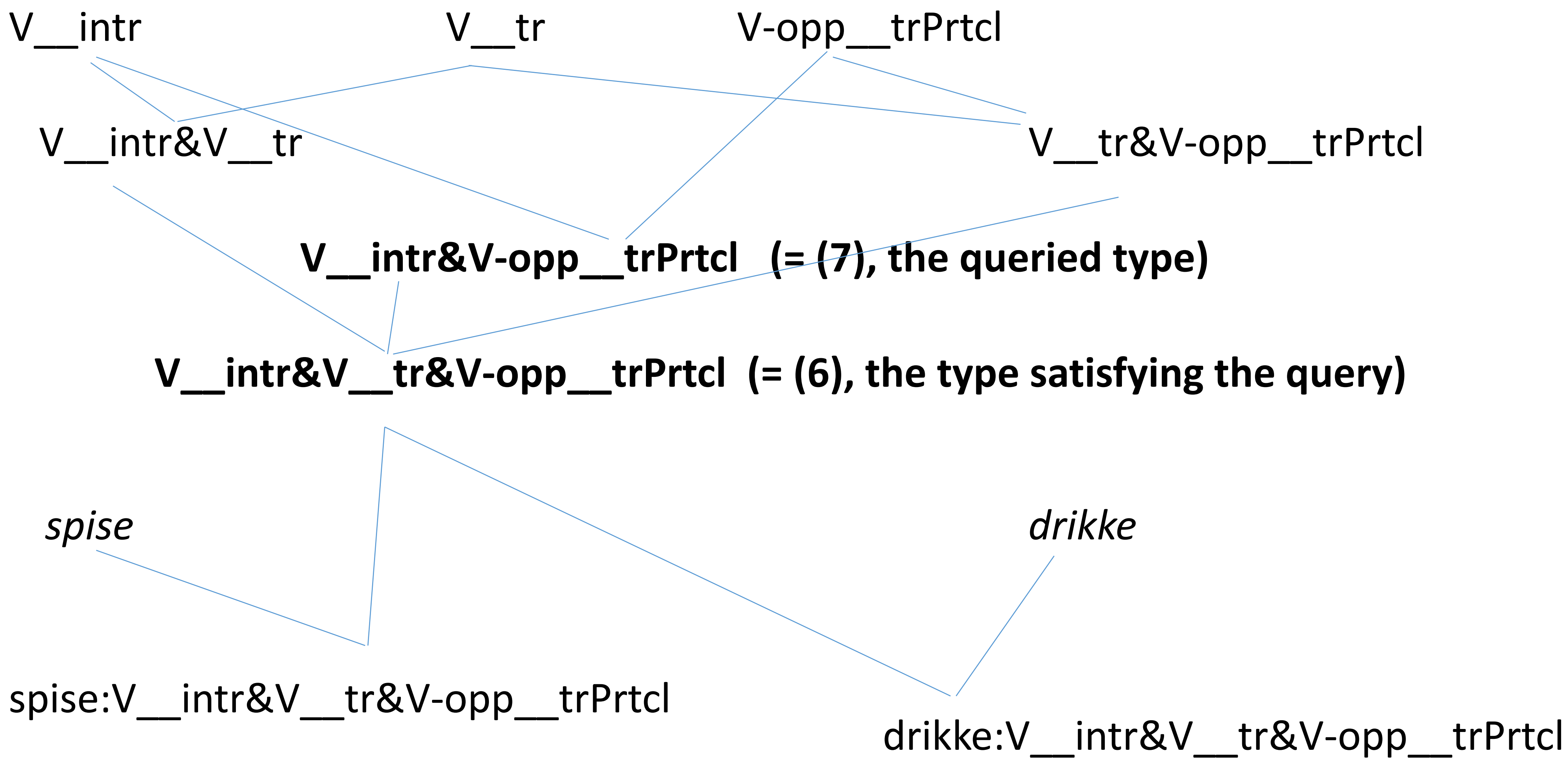
(7) `V__intr & V-opp__trPrtcl`,

this search will not get a hit in (6), since here the two lexvals ‘`V__intr`’ and ‘`V-opp__trPrtcl`’ are string-*dis*continuous

The most obvious solution is to make it possible to construe (7) as *a subset* of (6), so that in general, a ‘queried set’ has a ‘hit’ when the ‘hit’ valpod is a super-set of the ‘queried set’.

An alternative is to construe (6) and (7) as *types*, in a hierarchy constituted by *valpod types*, whereby (6) is a subtype of (7), and a ‘queried’ type generally has a ‘hit’ when the ‘hit’ valpod is a subtype of the ‘queried’ type.

The figure below illustrates the idea with a partial hierarchy relative to the types in question, with lemmas *spise* ‘eat’ and *drikke* ‘drink’ added so that one can infer, from the figure, that the lexvals ‘`V__intr`’ and ‘`V-opp__trPrtcl`’ are realized in the valpods of both *spise* and *drikke* (which both have much larger valpods, but including the frames in question).


V__intr
V__tr
V-opp__trPrtcl
V__intr&V__tr
V__tr&V-opp__trPrtcl
V__intr&V-opp__trPrtcl (= (7), the queried type)
V__intr&V__tr&V-opp__trPrtcl (= (6), the type satisfying the query)
spise
drikke
spise:V__intr&V__tr&V-opp__trPrtcl
drikke:V__intr&V__tr&V-opp__trPrtcl

Exposing content of the catalogue and the grammars

Exposing Frame Types

A special list is compiled to present the full inventory of the 304 Frame Types by their Argument Structure representations, illustrated below for a small snippet.

(Cf. https://typecraft.org/tc2wiki/NorVal_resources.)

The list displays for each Type its *CL string notation*, an interpretation of this notation in terms of a *feature structure* (AVM), and an exemplifying sentence. The AVM shows grammatical functions (GF) linked with what may be called a 'Logical Form' ('ACTNT').

The AVMs constitute analyses generated by the computational grammar *TypeGram*, a multilingual HPSG grammar based on Typed Feature Structures (cf. Hellan 2019).

CL-expression	*AVM format showing GF and Logical Form interlinked*	Illustrating sentence
tr-obDir	$\begin{bmatrix} \text{GF} & \begin{bmatrix} \text{SUBJ} & \begin{bmatrix} \text{INDX } \boxed{1} \end{bmatrix} \\ \text{OBJ} & \begin{bmatrix} \text{INDX } \boxed{2} \begin{bmatrix} \text{ROLE oriented-object} \end{bmatrix} \end{bmatrix} \end{bmatrix} \\ \text{ACTNT} & \begin{bmatrix} \text{ACT1 } \boxed{1} \\ \text{ACT2 } \boxed{2} \end{bmatrix} \end{bmatrix}$	*vi tømmer innholdet ut i elven* 'we empty the content out into the river'
tr-obEqBareinf	$\begin{bmatrix} \text{GF} & \begin{bmatrix} \text{SUBJ} & \begin{bmatrix} \text{INDX } \boxed{1} \end{bmatrix} \\ \text{OBJ} & \begin{bmatrix} \text{HEAD verb-BAREINF} \\ \text{INDX } \boxed{2} \\ \text{XACT } \boxed{1} \end{bmatrix} \end{bmatrix} \\ \text{ACTNT} & \begin{bmatrix} \text{ACT1 } \boxed{1} \\ \text{ACT2 } \boxed{2} \end{bmatrix} \end{bmatrix}$	*hun tør komme* 'she dares come'
tr-obEqSuInf	$\begin{bmatrix} \text{GF} & \begin{bmatrix} \text{SUBJ} & \begin{bmatrix} \text{INDX } \boxed{1} \end{bmatrix} \\ \text{OBJ} & \begin{bmatrix} \text{HEAD inf} \\ \text{INDX } \boxed{2} \\ \text{XACT } \boxed{1} \end{bmatrix} \end{bmatrix} \\ \text{ACTNT} & \begin{bmatrix} \text{ACT1 } \boxed{1} \\ \text{ACT2 } \boxed{2} \end{bmatrix} \end{bmatrix}$	*hun unnlater å komme* 'she fails to come'

Every label in CL has a TFS definition encoded in TypeGram. Thus, the label *tr* is defined as being the type of the following feature structure:

$$
tr\begin{bmatrix}
\text{GF} & \begin{bmatrix} \text{SUBJ } sign\begin{bmatrix}\text{INDX } \boxed{1}\end{bmatrix} \\ \text{OBJ } sign\begin{bmatrix}\text{INDX } \boxed{2}\end{bmatrix} \end{bmatrix} \\
\text{ACTNT} & \begin{bmatrix} \text{ACT1 } \boxed{1} \\ \text{ACT2 } \boxed{2} \end{bmatrix}
\end{bmatrix}
$$

Notion of GF	Representation in AVM
subject	SUBJ
direct object	OBJ
indirect object	IOBJ
complement	COMP
oblique	OBL
oblique2	OBL2
presented	PRESENTED
secondary predicate	SECPRD
extraposed	EXPN
extralinked	
identifier	IDNT
adverbial	ADV
particle	PRTCL

‘Logical Form’ attributes

ACT0	‘event index’ for a sentence and ‘thing index’ for an NP
ACT1	index of subject of verb used in active voice
ACT2	index of direct object of verb used in active voice
ACT3	index of indirect object of verb used in active voice
ACTobl	index of the governee of a PP functioning as oblique
XACT	index of subject or ‘external argument’ of a predicate (‘<e,t>’)
LOC	index of valence-bound locative specification
DIR	index of valence-bound directional specification

The computational HPSG grammar *NorSource* produces ‘Logical Form’-like representations in the MRS-part of its sentence parses. Below is the parse produced for the third sentence represented above:

Hun unnlater å komme ’she fails to come’

The left side shows the grammatical and morphological rules taking part in the parsing of the sentence in question, and the right side the semantic representation generated for the sentence, its MRS. ’Arg1’ corresponds to ‘ACT1’, etc.

(The attribute paths for clausal embedding exemplified in the AVMs in Table 1 are matched in the MRS by independent-standing predications, tied together through a system of extra labels, making the formalism more amenable to computational parsing.)

Syntax	MRS
head-subject-rule	ltop=h0, index=e1
hun_perspron	h3:hun_pron_rel([arg0:x2])
hun	h4:_pronoun_q_rel([arg0:x2, rstr:h5, body:h6])
head-verb-complementized-s-comp-rule	h7:_unnlate_v-tr_rel([arg0:e1, arg1:x2, arg2:h8])
pres-infl_rule	h8:_inf-mark_rel([arg0:u9, arg1:h10])
unnlate_subj-equi_vlxm	h10:_melde_v-tr_rel([arg0:e11, arg1:x2, arg2:x12])
unnlater	h13:_3p_refl_rel([arg0:x12])
head-complementizer-comp-equinf-rule	h14:_reflpronoun_q_rel([arg0:x12, rstr:h15, body:h16])
å_inf-comp	h13:coreferential_rel([arg0:u17, arg1:x12, arg2:x2])
å	< qeq(h5,h3), qeq(h15,h13) >
head-verb-tame-comp-rule	e1, sort=verb-act-specification, sf=prop, e.tense=pres,
inf-const_infl_rule	e.mood=indicative, e.aspect=semsort
melde_tr-refl_vlxm	x2, wh=-, png.ng.num=sing, png.ng.gen=f, png.pers=thirdpers
melde	e11, sort=verb-act-specification, sf=prop, e.tense=tense,
seg_refl	e.mood=indicative, e.aspect=semsort
seg	x12, wh=-, png.ng.num=sing, png.ng.gen=f, png.pers=thirdpers

Further facilities

A further facility exposing valence information is *MultiVal*, an online service showing the verb valence specifications from four HPSG grammars, viz. for Norwegian, Ga, Spanish and Bulgarian (Hellan et al. 2014):

http://regdili.hf.ntnu.no:8081/multilanguage_valence_demo/multivalence

A small corpus annotated for valence (22,000 sentences), produced through NorSource, is shown at (Hellan et al. 2021):

https://typecraft.org/tc2wiki/Norwegian_Valency_Corpus

Representing *Meaning* beyond ‘Logical Form’

Representing Meaning

In investigating correlations between valence frames and *senses*, the latter has to be represented as succinctly and perspicuously as the former.

The question is ***how*** this is to be done. We explore this issue in a stepwise fashion.

First, it seems that we need to represent *sameness* of meaning.

A first area where *sameness* of meaning should be represented is in the lexicon. To illustrate, we indicate how the verb *forhøre* ‘interrogate’ and its corresponding noun *forhør* ‘interrogation’ can be represented in a lexicon, with specification of sameness of meaning across lexical entries and inside of entries.

forhøre

{

- *SENSEPOD*:{**[a]**'*interrogate*', **[b]**'*query*'}
- *VALPOD*: forhøre:{V__tr & V__tr-obRefl & V-om__trObl-oblINTERR

& V-om__trObl-oblN & V-om__trObl-obRefl-oblINTERR & V-

om__trObl-obRefl-oblN}

- *VAL-SENSES*:{< **[a],** V__tr> & <**[b]**, V__tr-obRefl> & <**[a],**

V-om__trObl-oblINTERR> & <**[a],** V-om__trObl-oblN> & <**[b]**,

V-om__trObl-obRefl-oblINTERR> & <**[b]**, V-om__trObl-obRefl-oblN>}

- *V-N-CORR*: ***forhør*** {
  - *VAL-SENSES*:{<**[a]**,N__> & <**[a]**,N-om__intrObl-oblN>, & <**[a]**,N-om__intrObl-oblINTERR>} }

- ***forhør***

{

- GENDER: neuter;
- INFLECTION_PARADIGM: sg.indef. -*Ø,* sg.def. -*et,* pl.indef. -*Ø,* pl.def. -*ene*;
- N-TYPE: s01dur0&&soc&fjam0;
- VALPOD : N & N-av__intrObl-oblN & N-om__intrObl-oblINTERR & N-om__intrObl-oblN ;
- SENSEPOD: :{**[a]**'*interrogate*'} ;
- ENGLISH_GLOSS: 'interrogation';
- *VAL-SENSES*:{<**[a]**,N__>, & <**[a]**,N-av__intrObl-oblN-oblParticipant> & <**[a]**,N-om__intrObl-oblN-oblMatter> & <**[a]**,N-om__intrObl-oblINTERR- oblMatter>;
- *V-N-CORR*: ***forhøre;***
- *LVC-GOVERNORS*: {***foreta, gjennomgå***}

}

Sameness of meaning as a basis for establishing *cross-linguistic valence resources*

We indicate two possible procedures, both based on the assumption that the assembly of a verb’s valence environments reflects essential aspects of the verb’s meaning, so that translation plays a key role.

The first is based on sameness of *verb* senses, the other on sameness of *meanings of minimal sentences*.

- 1. Identify general differences between the ‘source’ language S and the ‘target’ language L in their grammatical encoding of argument structure.
- 2. Modulo these differences, map the frame types defined for S – let’s call it *FrameSet_S* - to a putative set of frame types for L, viz. *FrameSet_L,* thus*,* constructing a *list* of correspondence pairs

  FrameSet_S *FrameSet_L*

  - FrameS-1 - FrameL-1;
  - FrameS-2 - FrameL-2; ......
- 3. Establish a correspondence of basic verb synonyms between S and L, thus, a list of verb pairs:
  - V_S-1 - V_L-1,
  - V_S-2 - V_L-2, ...
- 4. Assign to each verb V_L-*n* the valpod of its corresponding verb V_S-*n* (i.e., a valpod where each lexval has a frame in *FrameSet_L* established in the mapping in point 2).

- 1. Identify general differences between the ‘source’ language S and the ‘target’ language L in their grammatical encoding of argument structure.
- 2. Modulo these differences, map the frame types defined for S – let’s call it *FrameSet_S* - to a putative set of frame types for L, viz. *FrameSet_L,* thus, constructing a *list* of correspondence pairs

FrameSet_S		*FrameSet_L*
• FrameS-1	-	FrameL-1;
• FrameS-2	-	FrameL-2; …...

- 3. Establish a correspondence of *Minimal Sentence synonyms* between S and L, thus, a list of minimal sentence pairs:
  - MS_S-1 - MS_L-1,
  - MS_S-2 - MS_L-2, …
- 4. Assign to the head verb V_L of each minimal sentence MS_L-*n* the lexval corresponding to its valence environment in MS_L-*n* (the frame of that lexval being defined in *FrameSet_L*), assemble all lexvals thus created for V_L, and from this set assemble the valpod of V_L.

Representing *Situation Types*

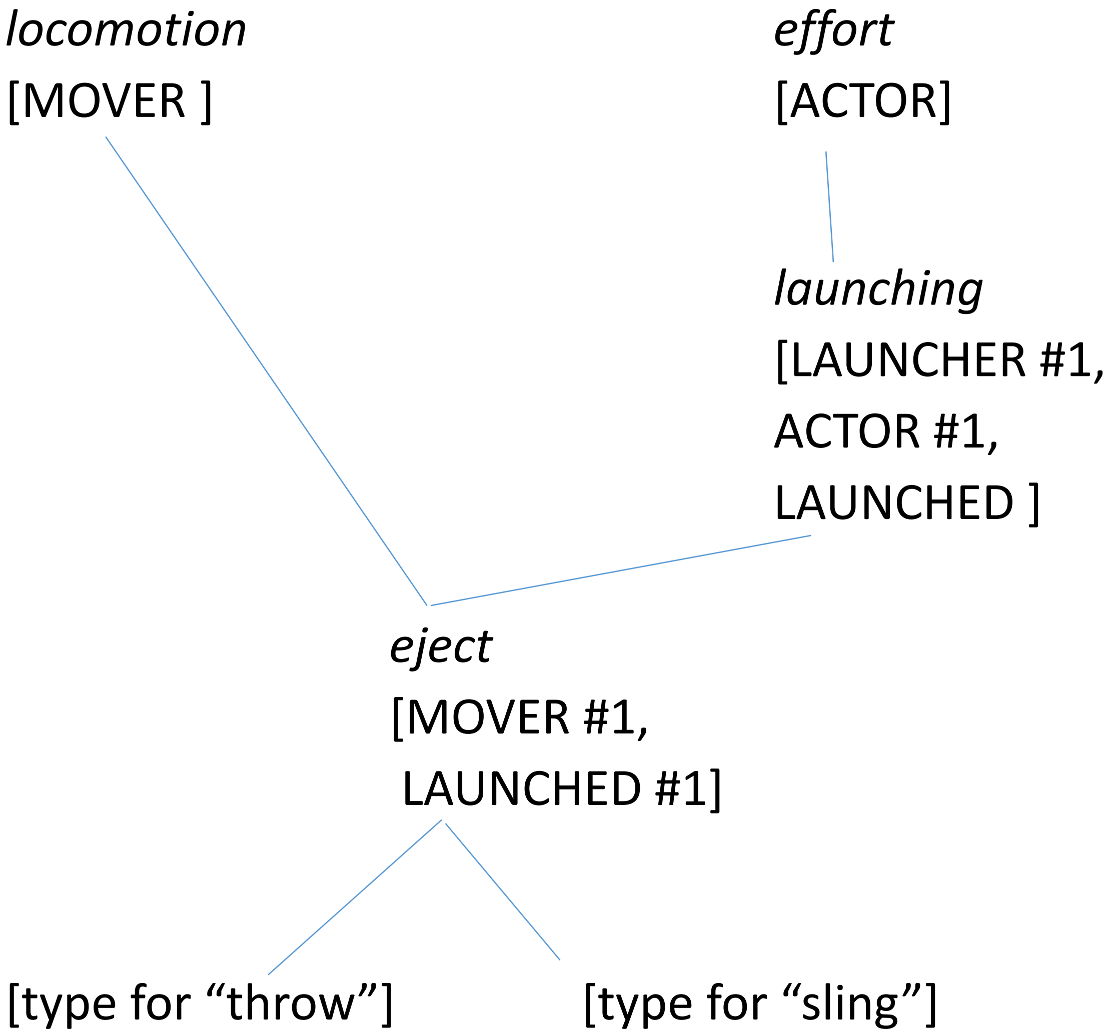
The notion 'transitive' is defined in TFS terms as:

$$tr\begin{bmatrix} \text{GF} \begin{bmatrix} \text{SUBJ } sign\begin{bmatrix}\text{INDX } \boxed{1}\end{bmatrix} \\ \text{OBJ } sign\begin{bmatrix}\text{INDX } \boxed{2}\end{bmatrix} \end{bmatrix} \\ \text{ACTNT}\begin{bmatrix} \text{ACT1 } \boxed{1} \\ \text{ACT2 } \boxed{2} \end{bmatrix} \end{bmatrix}$$

A subtype of this notion, for the expression of 'ejection' like in *throw*, can be formalized expanding the AVM with an attribute SIT (see figure), for 'SITuation structure'; its value *eject* is a type defined in a hierarchy of *situation types*, and as a CL label, *EJECT* represents that same type:

$$tr-EJECT\begin{bmatrix} \text{GF} \begin{bmatrix} \text{SUBJ } sign\begin{bmatrix}\text{INDX } \boxed{1}\end{bmatrix} \\ \text{OBJ } sign\begin{bmatrix}\text{INDX } \boxed{2}\end{bmatrix} \end{bmatrix} \\ \text{ACTNT}\begin{bmatrix} \text{ACT1 } \boxed{1} \\ \text{ACT2 } \boxed{2} \end{bmatrix} \\ \text{SIT}\begin{bmatrix} eject\begin{bmatrix} \text{ACTOR } \boxed{1} \\ \text{LAUNCH-MECH} \\ \text{LAUNCHED } \boxed{2} \\ \text{MOVER } \boxed{2} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Partial hierarchy of situation types



Both of the verbs *throw* and *sling* express acts that may be called ‘ejections’. Salient differences are that throwing may involve a better aiming than slinging, and that the curved arm movement typically associated with ‘throw’, by which the kinetic energy behind the ejection is built up typically unfolds in a vertical plane , while with ‘sling’ that plane is more horizontal. As type labels for the latter difference one can use *ejectArmVerticurve* and *ejectArmHorizcurve*, respectively, and enter these as the separate subtype branches under *eject.*

The CL counterparts can in turn appear in lexval specifications as follows, where the final part is the situation type:

```
kaste__tr-EJECTarmVerticurve
slenge__tr-EJECTarmHorizcurve
```

The CL specifications can in principle reflect any node on inheritance lines like in the above diagram, reflecting the degree of specificity represented by the meaning encoded, or the degree of semantic specification that one is in a position to make.

A valence catalogue where significant steps have been taken in using CL notation for semantic specification is the lexicon for the West African language Ga in Dakubu (2010, 2011), where altogether 498 verb lexemes are represented through 1980 lexvals with parameters including 130 situation types and a set of semantic roles. The verb *ba* ‘come’, for instance, is represented by 18 different lexvals; an example of one of its frame types involving semantic specification is given below, to be read as ‘a verb-headed intransitive syntactic frame where the subject carries an agent role and the situation expressed belongs to the type ‘MOTIONDIRECTED’’:

```
v-intr-suAg-MOTIONDIRECTED
```

In this format one can easily search for correspondences between grammatical aspects of a valence frame and its situation type, which is one possible representative of what one can generally call its *sense*.

This can be illustrated by an assortment of lexvals on the basis of the situation type MOTIONDIRECTED, of which there are about 50 lexvals; the semantic roles are in principle deducible from the SIT specification; the English glosses are part of the specifications:

- ba_3 := v-tr-obPostp-suAg_obLoc-MOTIONDIRECTED (English gloss: 'come')
- shi_1336 := v-tr-obPostp-suAg_obLoc-MOTIONDIRECTED (English gloss: 'forget')
- tee_1497 := v-tr-obPostp-suAg_obLoc-MOTIONDIRECTED (English gloss: 'go')
- tsi_1677 := v-tr-obPostp-suAg_obLoc-MOTIONDIRECTED (English gloss: 'push away')
- tsɔ_1729 := v-tr-obPostp-suAg_obLoc-MOTIONDIRECTED (English gloss: 'go before')
- ya_1912 := v-tr-suIDobSpec_obPostp-suAg_obLoc-MOTIONDIRECTED (English gloss: 'go')

Final word

The present Norwegian resource has taken about 35 years of consistent (if not constant) development; fully supplementing it with semantics along the lines just indicated may well take another decade. From a theoretical viewpoint, this will be interesting at any rate as an exploration in extending the coverage of a TFS architecture. From a practical viewpoint, having a clearly defined formalism is a prerequisite for undertaking such an extension, however long a haul.

For cross-linguistic extension or comparison of such systems, with large lexical coverage, the practical time perspectives are presumably similar, and likewise for the theoretical perspectives. And while the dimension of semantic equivalence is necessary as a basis for comparative studies, it is all too well known how much languages differ in the semantics of their verbal systems. Still, with a clearly laid-out formal design, one can start.

References

References

- Creissels, Denis. 2016. Transitivity, valency, and voice. Ms. European Summer School in Linguistic Typology. Porquerolles.
- Dakubu 2011. https://typecraft.org/tc2wiki/Ga_Valence_Profile
- Dakubu, M.E. Kropp and Lars Hellan. 2017. A labeling system for valency: linguistic coverage and applications. In Hellan, L., Malchukov, A., and Cennamo, M (eds). 2017. *Contrastive studies in Valency*. Amsterdam & Philadelphia: John Benjamins Publ. Co.
- Hellan, Lars. 2019. Construction-Based Compositional Grammar. *Journal of Logic Language and Information.* DOI: 10.1007/s10849-019-09284-5.
- Hellan, Lars. 2020. Interoperable Semantic Annotation. LREC workshop ISA-16, 6th Joint ACL-ISO Workshop on Interoperable Semantic Annotation
- Hellan, Lars, 2022. A Valence Catalogue for Norwegian. (n Loukanova (ed) Natural Language Processing in Artificial Intelligence - NLPinAI 2021. Springer. https://doi.org/10.18710/8U3L2U.
- Hellan, Lars and M.E. Kropp Dakubu. 2010. Identifying verb constructions cross-linguistically. In *Studies in the Languages of the Volta Basin 6.3*. Legon: Linguistics Department, University of Ghana. Accessible at: https://typecraft.org/tc2wiki/Verbconstructions_cross-linguistically_-_Introduction
- Hellan, L., D. Beermann, T. Bruland, M.E.K. Dakubu, and M. Marimon. 2014. MultiVal: Towards a multilingual valence lexicon. In Calzolari et al. (eds) Proceedings of LREC 2014 (web demo: https://typecraft.org/tc2wiki/Multilingual_Verb_Valence_Lexicon and http://regdili.hf.ntnu.no:8081/multilanguage_valence_demo/multivalence).

References

- Hellan, Lars and Tore Bruland. 2015. A cluster of applications around a Deep Grammar. In: Vetulani et al. (eds) Proceedings from The Language & Technology Conference (LTC) 2015, Poznan, web server version at http://regdili.hf.ntnu.no:8081/linguisticAce/parse.
- Hellan, Lars, Dorothee Beermann, Tore Bruland, Tormod Haugland, Elias Aamot. 2020. Creating a Norwegian valence corpus from a deep grammar. In *Human Language Technology. Challenges for Computer Science and Linguistics. 8th Language & Technology Conferene, LTC2017*. Springer 2020 ISBN 978-3-030-66526-5; web demo: https://typecraft.org/tc2wiki/Norwegian_Valency_Corpus).
- Malchukov, Andrej L. & Comrie, Bernard (eds.) (2015) *Valency classes in the world's languages.* Berlin: Mouton De Gruyter.
- Pollard, Carl and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar.* Chicago University Press, Chicago.
- Tesnière, Lucien. 1959. *Éleménts de syntaxe structurale*. Paris: Klincksieck.

Valence resources for other languages

- English: FrameNet; VerbNet; PropBank, with URLs : https://framenet.icsi.berkeley.edu/fndrupal, http://verbs.colorado.edu/~mpalmer/projects/verbnet.html.
- German: Evalbu; https://grammis.ids-mannheim.de/verbvalenz.
- Chech: Vallex; http://ucnk.ff.cuni.cz.
- Polish: Walenty; http://clip.ipipan.waw.pl/Walenty.
- Bilingual: CzEngVallex; https://ufal.mff.cuni.cz/czengvallex.
- Multilingual: ValPaL; http://valpal.info/ (cf. Malchukov and Comrie 2015).