

Petter Haugereid

Phrasal subconstructions

A constructionalist grammar design,
exemplified with Norwegian and English

Thesis for the degree of Philosophiae Doctor

Trondheim, June 2009

Norwegian University of Science and Technology

Faculty of Arts

Department of Language and Communication Studies



Norwegian University of
Science and Technology

NTNU

Norwegian University of Science and Technology

Thesis for the degree of Philosophiae Doctor

Faculty Arts

Department of Language and Communication Studies

© Petter Haugereid

ISBN 978-82-471-1629-6 (printed ver.)

ISBN 978-82-471-1630-2 (electronic ver.)

ISSN 1503-8181

Doctoral theses at NTNU, 2009:122

Printed by NTNU-trykk

Contents

1	Introduction	1
1.1	Theoretical assumptions	2
1.2	Five subconstructions	4
1.3	A construction-constraining mechanism	11
1.4	Syntactic structures	15
1.5	Layout of the thesis	19
I	Argument Structure	21
2	HPSG, LFG, CG, and GB/Minimalism	23
2.1	Introduction	23
2.1.1	Unergative and unaccusative verbs	24
2.1.2	Other alternations	25
2.1.3	Voice	27
2.2	HPSG	28
2.3	LFG and the Lexical Mapping Theory	31
2.4	Construction Grammar	37
2.5	GB/Minimalism	39
2.5.1	Passive in GB/Minimalism	39
2.5.2	Hale and Keyser's theory	40
2.5.3	First Phase Syntax	44
2.5.4	Minimalism - Borer's neo-constructionist approach	47
2.6	Comparison	49
2.7	Some methodological considerations	50
2.7.1	Remarks to HPSG	51

2.7.2	Remarks to LFG/LMT	51
2.7.3	Remarks to Construction Grammar	54
2.7.4	Remarks to Hale and Keyser's theory	55
2.7.5	Remarks to First Phase Syntax	56
2.7.6	Remarks to Borer's neo-constructionalist approach	57
2.8	Summary	57
3	Subconstructions	59
3.1	Some syntactic tests	59
3.2	Five subconstructions	62
3.2.1	ARG1	62
3.2.2	ARG2	63
3.2.3	ARG3	65
3.2.4	ARG4	66
3.2.5	ARG5	67
3.3	Levin's "English Verb Classes and Alternations"	68
3.3.1	The Causative/Inchoative Alternation	68
3.3.2	The Induced Action Alternation	68
3.3.3	The Substance/Source Alternation	69
3.3.4	Intransitive/Transitive Alternations	70
3.3.5	Conative and Preposition Drop Alternations	72
3.3.6	Dative and Benefactive Alternations	72
3.3.7	Locative and similar alternations	73
3.3.8	Delimiter Alternations	74
3.3.9	Other Alternations	76
3.4	BRRs and semantic representations	78
3.4.1	MRS	79
3.4.2	BRR/RMRS	80
3.5	A hierarchy of subconstructions	82
3.6	Summary	85
4	Valence	87
4.1	Valence in HPSG	88
4.2	The Grammar Matrix and Norsyg	91

4.2.1	The Grammar Matrix	91
4.2.2	Norsyg - some data	93
4.3	The linking types	94
4.3.1	Four valence features	94
4.3.2	A hierarchy of linking types	95
4.3.3	<i>Packing</i> of argument frames	96
4.3.4	Introductory remarks on the composition of subconstructions	98
4.4	Lexical types in Norsyg	100
4.5	Expansion of the lexicon	105
4.5.1	Adaptation of Norsk Ordbank	105
4.5.2	Unknown words	107
4.5.3	Lexicon acquisition	108
4.6	Construction-constraining mechanism vs. lexicalism	112
4.7	Comparison with the RASP system	117
4.8	Other Norwegian computational resources	120
4.8.1	TROLL	120
4.8.2	NorKompLeks	122
4.8.3	NorSource	124
4.8.4	NorGram	125
4.9	Summary	126
II The realization of argument structure in the syntax		129
5	Methodology	131
5.1	Preliminary analyses	131
5.2	Some remarks on syntactic structures	137
5.2.1	Introductory remarks on tree structures	137
5.2.2	Linguistic considerations	141
5.2.3	Cognitive considerations	144
5.2.4	Computational considerations	145
5.3	Summary	148
6	Basic syntactic structures	151
6.1	The valence rules	152

6.2	The filler rule	155
6.3	The force rules	156
6.4	Some simple analyses	157
6.4.1	Analysis of a transitive sentence	159
6.4.2	Analysis of a resultative sentence	163
6.5	The merge rule	168
6.6	Subordinate clauses and relative clauses	173
6.6.1	Subordinate clauses	174
6.6.2	Relative clauses	180
6.7	Infinitival clauses and small clauses	183
6.7.1	Unexpressed subjects	183
6.7.2	Analyses of infinitival clauses and small clauses	184
6.7.3	Raising and control	191
6.7.4	Remarks on raising	194
6.8	The modifier rules	197
6.9	Long distance dependencies	202
6.9.1	The <i>trace</i> approach	202
6.9.2	Reflection of extraction path	202
6.9.3	The lexical approach	204
6.9.4	Some problems	204
6.9.5	The approach taken in Norsyg	206
6.10	Summary	207
7	Passive and Presentation	209
7.1	Passive	209
7.1.1	Data	209
7.1.2	The passive types	211
7.1.3	Analysis	213
7.2	The presentational construction	216
7.2.1	Some data	216
7.2.2	The presentational rules	217
7.3	Summary	217

8	Coordination	219
8.1	Coordination of VPs	219
8.1.1	Data	219
8.1.2	Analysis	220
8.2	Coordination of Vs	223
8.2.1	Data	223
8.2.2	Analysis	225
8.3	Ellipsis	227
8.4	Pseudo-coordination	227
8.4.1	Sub-coordination	228
8.4.2	The Empty Object Construction	230
8.4.3	Analysis	231
8.5	Summary	238
9	Comparison with GB	241
9.1	GB as presented in Carnie 2007	242
9.2	A GB analysis based on Norwegian data	246
9.3	Three positions for verbs	249
9.3.1	The position corresponding to C	250
9.3.2	The position corresponding to T	252
9.3.3	The position corresponding to V	253
9.4	An account of basic clause structure in English	254
9.4.1	Blocking main verbs from appearing before the subject	255
9.4.2	Assuming an empty complementizer	256
9.5	Difference between Norsyg and GB	259
9.5.1	Difference in parsing strategy	260
9.5.2	Infinitival clauses and ‘skewed’ syntactic-semantic relations	262
9.6	Summary	264
10	Sentence adverbials	265
10.1	Data	266
10.1.1	Sentence adverbials in different clause types	267
10.1.2	Sentence adverbials and the arguments	268
10.2	A GB approach	272

10.3	The approach taken in Norsyg	274
10.3.1	Analysis of sentence adverbials in different clause types	275
10.3.2	Analysis of sentence adverbials and the arguments	276
10.4	Summary	279
11	Conclusion	281
A	Norsyg	287
A.1	Download	287
A.2	Short description	288
A.2.1	Composing argument structure in the syntax	288
A.2.2	Left-branching tree structures	289
A.3	Data	292
A.4	Coverage	293
A.5	NorKompLeks test sentences	294
A.6	Technical details about case and linking	297
A.6.1	The linking mechanism	297
A.6.2	Case	301
B	Demo grammars for English and German	305
B.1	English demo grammar	305
B.2	German demo grammar	308
C	Example sentences of the thesis	315
C.1	Norwegian example sentences	315
C.2	English example sentences	322
D	BRRs of example trees	331
D.1	BRRs of example trees in Chapter 4	331
D.2	BRRs of example trees in Chapter 5	332
D.3	BRRs of example trees in Chapter 6	335
D.4	BRRs of example trees in Chapter 7	342
D.5	BRRs of example trees in Chapter 9	344
D.6	BRRs of example trees in Chapter 10	347
D.7	BRRs of example trees in Appendix B	351

Acknowledgements

Doing a PhD can be compared to making a trip to a mountain top in the relatively flat landscapes of Lierne. You start out in high spirit, seeing the mountain top in the distance. You walk for a long while, through moss and belts of forest until you stop to make a break. You still see the mountains in a distance. After some more walking you start approaching the foot of the mountain, and you don't see the top any more. By then you are getting a little tired, but you think that the top cannot be very far away, so you climb energetically, only to find out that what you thought was the top turns out to be a hill next to the real mountain. At this point you have to decide whether to return or whether you want to get to the *real* top. So you rest for a short while, fill your bottle with water from a stream, and decide to climb the last rocky part until you finally reach the highest top. It turns out, as you get to the top and can see far into the distance, that the climbing of this rocky part is not the biggest effort. It is getting to and from the foot of the mountain that is the challenge. And for a PhD student, this is the part where you get help and support that you need from colleagues, friends, and family.

My supervisor during my period as a PhD student has been Lars Hellan. With patience and insight, he has helped me to understand what it really is that I wanted to do. In addition to being a mentor, he has included me in several of his projects and introduced me to a great network of people working with grammar engineering.

Through the projects DeepThought, ScanMatrix and especially LOGON, which employed me as a PhD student, I got the chance to present and discuss my ideas to a large group of people, including Lars Ahrenberg, Dorothee Beermann, Emily Bender, Felix Bildhauer, John Carrol, Berthold Crysmann, Ann Copestake, Luca Dini, Helge Dyvik, Andreas Eisele, Liv Ellingsen, Dan Flickinger, Dario Gonella, Hannes Hirzel, Per Anker Jensen, Lars Johnsen, Valia Kordoni, Daniela Kurz, Gunn Inger Lyse, Jan Tore Lønning, Giampaulo Mazzini, Paul Meurer, Stefan Müller, Torbjørn Nordgård, Stephan Oepen, Victoria Rosèn, Melanie Siegel, Anders Sjøgaard, Jesse Tseng, Hans Uszkoreit, Erik Velldal, and Ben Waldron. Thanks to all of you! Dan Flickinger has in particular been a great inspiration during my period as a PhD student. Whenever I have presented my work with him in the audience, he has, in his own gentle way, helped me to see how my analyses could be improved.

I have also benefited from discussions with my colleagues at Dragvoll. Thanks to

Jørn Almberg, Getahun Amare, Nana Amfo, Kaja Borthen, Jonathan Brindle, Heidi Brøseth, Rositsa Dekova, Tszvetana Dimitrova, Arne-Kjell Foldvik, Anne Sandø Frank, Thorstein Fretheim, Janicke Furberg, Snefrid Holm, Ola Huseth, Jacques Koreman, Ota Ogie, Rein Ove Sikveland, Siri Simonsen, Jostein Ven, and Tesfaye Wondwosen. During my stay in Saarbrücken February-August 2004 I enjoyed the company of Peter Dienes, Frederik Fouvry, Shravan Vasishth, Yi Zhang and Zhiping Zheng. Thanks to Martha Larson for inviting me to Bonn (and making me aware of Empty Object Constructions), thanks to Berthold Crysmann, who helped me to motivate my left-branching structures, and special thanks to Valia Kordoni and Hans Uszkoreit for accommodating me at CoLi.

I especially want to thank Emily Bender, Kaja Borthen, Anders Søgaard, Stephan Oepen, and Ben Waldron for reading drafts of the thesis and giving very valuable feedback. And also thanks to Dorothee Beermann, Ann Copestake, Stefan Müller, who in their initial report provided me with many helpful comments on my thesis.

Finally, I would like to thank my family for always believing in me, and a special thanks to my wife, Lilian, for her support and faith.

Chapter 1

Introduction

Current computational grammars designed within the HPSG and LFG frameworks suffer from an increasing amount of analyses of sentences parsed, and increasing processing time, as sentence length extends beyond that of 8-10 words. Such grammars do not purport to reflect the psychological reality of what happens in sentence processing, and so far, no theory adequately covers this area. I nevertheless feel it as a legitimate concern that the rather explosive processing demands witnessed in such grammars bear no intuitive similarity to what happens when we actually use sentences of normal length (which may well be 20-30 words). Part of the discrepancy can be attributed to pragmatics: much of the processing load hinges on substantive ambiguity of the words used, and in actual language use, we normally have no problem determining the relevant meaning of any lexical item uniquely. The account of this belongs to theories of discourse and pragmatics, and should not affect the design of computational grammars, which deal with modules of word combinatorics at sentence level. However, even with this aspect sorted away, processing demands remain having to do with non-locality of information, manifesting itself in multiple lexical entries even when no real ambiguity is in question, and cumbersome strategies and massive hypothesis-building in parsing.

In this thesis, I try, with departure point in formalisms as alluded to above, to define designs of lexicon building and syntactic analysis which will reduce the processing loads of a parsing mechanism significantly. I build a grammar of Norwegian to illustrate and verify my proposals.

This grammar model may seem unorthodox in many ways, but in presenting it, I

provide evidence and motivation that would be relevant in any standard analysis. No appeal to psychological reality is made throughout, except one particular paragraph where I relate to the issue. Thus, the model presented is to be evaluated as any standard analysis and implementation should be; only, the reader may bear in mind that what motivates the various sub-proposals being collected into this particular whole, is the intuition mentioned.

1.1 Theoretical assumptions

One of the differences between Construction Grammar (CG) on the one hand and lexicalist frameworks like HPSG and LFG on the other, is that in the analysis of verbal constructions, the former posit constructional frames as ‘primitive’ entities into which the individual verbs will accommodate their semantics, whereas in the latter frameworks, the corresponding type of entity is often referred to as ‘argument structure’, and is assumed to be propagated into the grammar through the specifications (‘lexical frames’, or ‘subcat restrictions’) of the individual verbs. In the analytic practice in such grammars, these lexical frames are distinguished as ‘lexical types’ or ‘macros’ and defined at an abstract level, and only in turn associated with the individual verbs; hence it might be questioned whether the difference originally mentioned is of mainly rhetorical significance rather than representing a difference in insights about the interplay between grammar and the lexicon. In the present thesis, I will try to show that the difference can indeed be modelled in such a way as to provide interestingly different designs of grammar. I will do this using the overall architecture of HPSG grammars, but inside of this architecture, develop a mechanism by means of which the over-all grammatical configuration in which a verb occurs, rather than its predefined lexical frame, is what induces its argument structure. I will show that this design provides a more efficient parsing grammar than one using the ‘lexicalist’ design, and argue that also on conceptual and empirical grounds, this design is advantageous.

In this enterprise, the grammar engineering aspect is the most important one, and is the area where I hope to be contributing something new by this thesis. However, the model I develop can be fully appreciated only on the background of my theoretical views of grammar.

My theoretical view of grammar makes a sharp distinction between ‘form’ and ‘content’, the former comprising morphology and morphologically and distributionally

validated aspects of what is called ‘syntax’. Grammar, in my view, is constituted only by these components, excluding semantics; as far as syntax is concerned, I thereby stand very much on the side of ‘autonomous syntax’, maintained by Chomsky all since Chomsky (1957). To avoid confusion with more inclusive conceptions of ‘syntax’ found in the literature, I will refer to my notion as one of ‘strict syntax’ when necessary.

My view of the Lexicon as connected to a grammar is that it should highlight those properties or parameters which are highlighted in the grammar, and only subsidiarily expose other properties of lexical items (thus quite unlike an encyclopedia, for instance). It follows that by my view of Grammar, in the lexicon, only those properties which reflect parameters of morphology and strict syntax should be represented. Valence properties of verbs are in my view mostly a reflection of their meaning, and therefore not a proper aspect of grammar: ‘argument structure’ is thus not part of strict syntax, and valence requirements should not be part of verb entries in the Lexicon.

However, I recognize that for most parsing grammars, a component of ‘valence’ or ‘argument structure’ may be desirable: a parsing grammar is, in many respects, more a ‘performance’ than a ‘competence’ construct, and thereby combining components which on a strict view should be kept apart. To the extent that ‘argument structure’ ought to be represented in the verb lexicon of a parsing grammar and reflected in the parsing mechanisms, I want to do that in such a way that in a lexical entry, this type of information is easily detachable, almost to be regarded as an ‘add-on’ property. This ‘add-on’ nature of argument structure specification is what models my constructional view of grammar, in that what ‘adds’ the specification in question is information provided by the environment of the verb, i.e., the construction in which the verb occurs.

The parameters of specification constituting argument structure are of the same type as those underlying the ‘Grammatical Relations’ of LFG, and relational primitives of Relational Grammar - see Section 1.2 below - and do not involve semantic properties such as ‘roles’ of participants and the like. Since the criterial basis for the Grammatical Relations are constructional environments, my formal term for grammatical relations is *subconstructions*. Subconstructions are realized by morpho-syntactic signs such as syntactic rules, inflections and function words.

My avoidance of semantic assumptions in syntax also has as a consequence that I omit the more standardly assumed levels of constituent structure representation (such as c-structure in LFG, and counterparts of this assumed in most HPSG grammars),

since I believe that the structures proposed to a large extent reflect assumptions about ‘logical form’. Thus, my assignment of ‘Grammatical Relations’ to a string will be based mostly on linear order, and not supposing any ‘constituent structure backbone’ previously assigned, as in standard LFG and HPSG grammars.

The grammar implementation I am providing is called Norsyg,¹ which has been developed since 2002. The grammar is a typed feature structure grammar, and it is implemented with the LKB system, which is a standard software for implementing typed feature structure grammars, typically HPSG grammars. I employ the over-all architecture of the ‘HPSG Grammar Matrix’, however only up to the point where the ‘constructional’ design is defined. At this point, what populates the mechanisms representing semantics in a standard HPSG/Matrix grammar such as MRS (see Section 3.4), is a display of *Grammatical Relations*, and thus, notionally, more on a par with an LFG f-structure rather than with an HPSG semantic structure. The feature geometry employed is similar to what is used in the HPSG literature, but a new mechanism for assigning and constraining the expected argument frames of verbs, involving a type hierarchy of construction and subconstruction types, will be presented.

1.2 Five subconstructions

A construction serves as a skeleton that open class lexical items fit into. On the view outlined above, the ‘argument structure’ of an open class lexical item is projected from the construction it occurs in. This grammatical configuration is a constellation of functional signs like inflections, function words (i.e., ‘strict syntax’) and (more abstractly) rules.² In order to get the relation between a construction and the individual functional signs that together express the construction, I assume that a construction can be decomposed into *subconstructions*.³

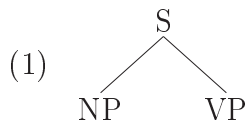
As anticipated above, a subconstruction is closely tied to the notion of ‘Grammatical Relation’. A Grammatical Relation is always realized through a syntactic constellation

¹See Appendix A.

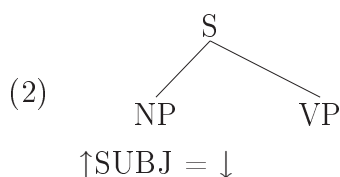
²In this thesis I make a distinction between what I refer to as *functional signs*, namely inflections, closed class lexical items, and syntactic rules on the one hand, and *open class lexical items*, which are uninflected adjectives, nouns and verbs.

³Since I assume that subconstructions are expressed by what I refer to as functional signs (see footnote 2), I sometimes refer to subconstructions as *phrasal* subconstructions, in order to separate them from what is referred to as *lexical* constructions (see Sag *et al.* (2003, Chapter 16), and Müller (2006)).

– for instance, ‘subject-of’ is realized through a constellation depictable as



in a language like Norwegian, and similarly for other functions. A constellation like that in (1), which may be called a local subtree, will here be referred to as a *subconstruction*, and a GR will be seen as corresponding to the set of subconstructions which realize it. Such a view on GRs relative to realizing constellations is similar to the way in which LFG correlates GRs with C-structure constellations, through, in the PS-rules, annotating these constellations for the GRs they induce. For instance, for the constellation (1), the PS-rule in an LFG grammar would provide the following annotation stating that the constellation realizes the ‘subject-of’ GR: (2)



The counterpart of this notation in the present work is outlined in this section, in Chapter 3, and in Section 6.1. A comparison between our representation of GRs and the ‘f-structure’ in LFG is given in Section 2.7.2.

I assume that there are five kinds of subconstructions, and that a construction can be a constellation of zero to five subconstructions. The subconstructions are called *arg1-sign*, *arg2-sign*, *arg3-sign*, *arg4-sign* and *arg5-sign*. These five subconstructions are signs with a syntactic expression and a semantic content. As mentioned, the syntactic expression is either a function word, an inflection, or a rule. The subconstructions are not expressed as open class lexical items like verbs, nouns, or adjectives.

The semantic content of the subconstructions are Parsons-style “underlying events.” Parsons (1990), argues that a transitive sentence like (3a) can be given the semantic representations in (3c) or (3d) rather than the traditional semantic representation in (3b). In (3c) the binary relation *Stabbed* has been given an “underlying event analysis” with three underlying events. The predicate is the first underlying event (*Stabbing*), the first argument is the second underlying event (*Subj*), and the second argument is the third underlying event (*Obj*). (3d) is a representation with thematic roles instead of functions, where the first argument is (*Agent*), and the second argument is (*Theme*).

- (3) a. Brutus stabbed Caesar.
 b. $(\exists e)[\text{Stabbed}(B,C)]$
 c. $(\exists e)[\text{Stabbing}(e) \ \& \ \text{Subj}(e,B) \ \& \ \text{Obj}(e,C)]$
 d. $(\exists e)[\text{Stabbing}(e) \ \& \ \text{Agent}(e,B) \ \& \ \text{Theme}(e,C)]$

While Parsons uses functional terms such as *Subj* and *Obj*, or thematic role names such as *Agent*, *Theme*, *Goal*, *Benefactive*, *Instrument* and *Experiencer* (Parsons, 1990, 71–72) for the underlying events, I will use the relation names *arg1-rel*, *arg2-rel*, *arg3-rel*, *arg4-rel* and *arg5-rel* for the underlying events. These represent underlying events that are not meant to correspond directly to thematic roles (Fillmore, 1968), but in combination with the meaning of the main verb and the arguments, they can be interpreted as thematic roles.

The underlying events provided by the syntactic elements (*arg1-rel–arg5-rel*) is as far as I will go into semantic decomposition. In order to get thematic role interpretation, or more elaborate semantic decomposition as in Jackendoff (1990), I assume that the underlying events will have to be interpreted in conjunction with the meaning of the verb and the meaning of the arguments. This is outside the scope of this thesis.

I do not have as an ambition to let my analysis yield *meanings* or *semantic representations* of sentences. According to Frege’s principle of compositionality, the meaning of a sentence is determined by the meaning of the constituents as well as the structure of the constituents. In this thesis, I will only look into the structure of the constituents. The meaning of the constituents will not be taken into consideration. So one of the two factors, which according to the principle of compositionality are needed to give a semantic representation of a sentence, is missing. This does not mean that the output is completely detached from meaning, only that it represents a *partial* meaning, namely the meaning provided by the structure. (The constituents will be represented as well, but only as unanalyzable predicates.) Given that representations produced by the grammar are assumed to give meaning only if interpreted in conjunction with the meaning of the constituents, I have chosen to refer to them as a *Basic Relation Representations* (BRRs).⁴ The BRR of (3a) is given in Figure 1.1. It represents the

⁴This term was suggested to me by Lars Hellan. I have also considered other terms such as *structural semantic representations* or *collections of Grammatical Relations*. However, these terms are potentially confusing or misleading. The term *structural semantic representation* may be seen as nonsensical if one does not adhere to the view that meaning is compositional. The term *collection of Grammatical Relations* may give the wrong impression that the representation of underlying events corresponds to

stabbing event decomposed into three underlying events *_stab_v_rel*, *arg1_rel*, and *arg2_rel*. The underlying events are linked by means of a *handle* (*h1*) (see Section 3.4 on semantic representations for more details). The indices of the two participants *Brutus* and *Caesar* are bound by the underlying events *arg1_rel* and *arg2_rel*, respectively. The binding of the indices implies that the representation is an *indexed* BRR.

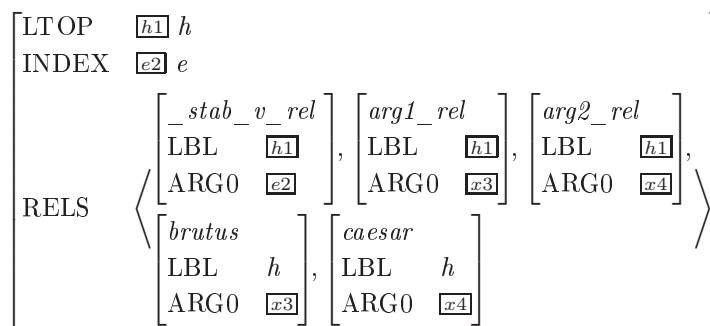


Figure 1.1: Indexed BRR of *Brutus stabbed Caesar*

The subconstructions can to some extent be illustrated by argument structure features used in LFG (see Bresnan (2001, 302–321), and discussion in Section 2.3). In LFG, argument structure is assumed to be lexically specified, and the semantic argument roles carry features, $[\pm o]$ and $[\pm r]$, which constrain the way the argument roles are mapped onto argument functions in f-structures. The feature $[-r]$ maps the argument role onto an *unrestricted* syntactic function, that is, either subject or object.⁵ Obliques and restricted objects are $[+r]$. The feature $[-o]$ maps arguments onto *non-objective* syntactic functions (subjects and obliques). The feature $[+o]$ maps arguments onto objects and restricted objects.

The subconstructions can more directly be illustrated by means of Grammatical Relations in Relational Grammar (Blake, 1990). In Relational Grammar, strata represent the grammatical relations of a verb by means of arcs labelled 1 (subject), 2 (direct object), and 3 (indirect object). In addition there are oblique relations (including benefactive, locative, and instrumental). The Initial Stratum shows the “deep” grammatical relations of a verb, and the Final Stratum shows the surface

a representation of surface grammatical relations like F-structure in LFG. The Grammatical Relation ‘Subject’ does for example not appear in the representations.

⁵These functions are referred to as unrestricted since they according to the theory do not need to have a semantic role. Raised and expletive arguments are presented as examples of syntactic functions with no semantic role. It should be noted that in this thesis, raised arguments are assumed to be arguments both of the raising verb and the controlled verb. (See Section 6.7.3.)

grammatical relations. The Initial Stratum may be identical to the Final Stratum of arcs. This is the case in active, transitive clauses, where the initial 1 is the final 1 and the initial 2 is the final 2. There may also be *revaluations* of arcs. In that case, the Final Stratum is different from the Initial Stratum. This is the case in a passive transitive clause where the initial 1 is demoted to *chômeur* and the initial 2 is promoted to 1. There may be more than one revaluation. The subconstructions assumed in this thesis correspond to arcs in the Initial Stratum in Relational Grammar.⁶

The *arg1-sign* is a subconstruction that corresponds to the realization of an external argument, or deep structure subject, in GB. It corresponds to the realization of an (agent) argument with the [-o] feature in LFG. It corresponds to the realization of an argument which has a 1-arc in the initial stratum in Relational Grammar. When this subconstruction is used, it implies that the event of the main verb has something that can be interpreted as a causer or initiator (an *arg1-rel* underlying event). The information that the event has an *arg1-rel* is assumed to come from the syntax, and *not* from the main verb. In an active main clause, the *arg1-sign* is expressed as a rule that links the subject to the head projection (see (4a)), and in a passive clause, this subconstruction is expressed as the passive auxiliary or the passive morphology (see (4b)). In an infinitival active clause, the *arg1-sign* is expressed as the infinitival marker (see (4c)).

- (4) a. **John** smashed the ball.
 b. The ball **was** smashed.
 c. (John tried) **to** smash the ball.

The *arg2-sign* is a subconstruction that corresponds to the realization of the direct object internal argument in GB. In LFG it corresponds to the realization of an (patient/theme) argument with the [-r] feature, or an (patient/theme) argument with the [+o] feature if there is another (beneficiary) argument with the [-r] feature. It corresponds to the realization of an argument which has an 2-arc in the initial stratum in Relational Grammar. The *arg2-sign* expresses that the event of the main verb has something that can be interpreted as a theme or patient argument (an *arg2-rel* underlying event). Again, the information that the event has an *arg2-rel* underlying event, comes from the syntax, and not from the main verb. The *arg2-sign* is usually

⁶It should also here be noted that in the approach presented in this thesis, raised arguments are assumed to be arguments both of the raising verb and the controlled verb.

realized as a rule that attaches the direct object to the head projection (see (5a)). In unaccusative and passive clauses, the rule attaches the subject to the head projection (see (5b) and (5c)). In an infinitival unaccusative or passive clause the *arg2-sign* may be realized as the infinitival marker (see (5d)).

- (5) a. John smashed **the ball**.
 b. **The boat** arrived.
 c. **The ball** was smashed.
 d. (The car needed) **to** be washed.

The *arg3-sign* is a subconstruction that corresponds to the realization of an indirect object internal argument in GB. In LFG it corresponds to the realization of a (beneficiary) argument with the [-r] feature. It corresponds to the realization of an argument with a 3-arc in the initial stratum in Relational Grammar. The *arg3-sign* expresses that the event happens in the (dis)favor of somebody (an *arg3-rel* underlying event). Also here, the information that the event has an *arg3-rel* underlying event is contributed by the syntax, and not by the main verb. The *arg3-sign* is usually realized as a rule that attaches the indirect object to the head projection (see (6a)), but if the clause is passive, it may be the subject that the rule attaches to the head projection (see (6b)). The *arg3-sign* may also be realized as the infinitival marker in a ditransitive passive clause (see (6c)).

- (6) a. John gave **Mary** a book.
 b. **Mary** was given the book.
 c. (Mary wanted) **to** be given a book.

The *arg4-sign* is a subconstruction that attaches a delimiter to the head projection. It corresponds to the realization of a goal/locative oblique in GB, LFG, and Relational Grammar. A delimiter is a goal phrase as in (7a) or a resultative as in (7b). The *arg4-sign* expresses that there is something that can be interpreted as an end point or end state for the argument realized by the *arg2-sign* (if realized) (an *arg4-rel* underlying event). It is important to notice that the information about there being an *arg4-rel* is assumed to come from the syntax, and not from the main verb (or from the delimiter itself). The *arg4-sign* is realized by a rule that attaches the delimiter to the head projection.

- (7) a. John smashed the ball **out of the room**.
 b. John hammered the metal **flat**.

The *arg5-sign* is a subconstruction that attaches PP arguments that are *not* delimiters, to the head projection. It corresponds to the realization of for example an instrument oblique in LFG and Relational Grammar. An *arg5-sign* can express that the event has an instrument as in (8).

- (8) John punctured the balloon **with a needle**.

Table 1.1 summarizes what argument realizations the subconstructions correspond to in GB, LFG,⁷ and Relational Grammar (RG).⁸

Subconstr.	GB	LFG	RG
arg1-sign	external argument	agent [-o]	initial 1-arc
arg2-sign	internal dir obj	patient/theme [-r]	initial 2-arc
arg3-sign	internal indir obj	beneficiary [-r]	initial 3-arc
arg4-sign	oblique	oblique	oblique
arg5-sign	oblique	oblique	oblique

Table 1.1: Subconstructions corresponding to argument realizations in GB, LFG, and Relational Grammar

The five subconstructions can be combined to form a wide range of constructions. An intransitive sentence like (9a), has only an *arg1-sign*. This means that it has an *arg1-construction*. A transitive sentence, like (9b), has two subconstructions, an *arg1-sign* and an *arg2-sign*. This means that it has an *arg12-construction*. An unaccusative sentence like (9c) only has an *arg2-sign*, which means that it has an *arg2-construction*. A ditransitive sentence like (9d) has three subconstructions, an *arg1-sign*, an *arg2-sign* and an *arg3-sign*. This means that it has an *arg123-construction*. A transitive clause with a PP complement like (9e) has an *arg1-sign*, an *arg2-sign* and an *arg4-sign* (the PP *to Mary* is a delimiter). This means that it has an *arg124-construction*.

⁷If there is a beneficiary argument with the [-r] feature, the argument realization corresponding to the *arg2-sign* is a patient/theme with the [+o] feature.

⁸The difference between an oblique realized as an *arg4-sign*, and an oblique realized as an *arg5-sign* can be understood by means of the distinction made between *subsequent* and *antecedent* roles in Croft (1991, 184–191). Croft refers to the roles *benefactive*, *malefactive*, *recipient*, and *result* as subsequent and the roles *instrumental*, *manner*, *means*, *comitative*, *passive agent*, *ergative*, and *cause* as antecedent. The subsequent roles are assumed to follow the object in the causal chain and the antecedent roles are assumed to precede them. In this thesis, the *arg4-sign* is assumed to realize a subsequent oblique, and the *arg5-sign* is assumed to realize an antecedent oblique.

- (9) a. John smiles. (arg1-construction)
 b. John smashed the ball. (arg12-construction)
 c. The boat arrived. (arg2-construction)
 d. John gave Mary a book. (arg123-construction)
 e. John gave a book to Mary. (arg124-construction)

Since the constructions are creations of the syntax, a lexical entry can be allowed to enter all possible constructions simply by not constraining it. A verb like *drip* is not tightly connected to a particular construction. The range of constructions that this verb can enter, can easily be accounted for. (I will present constructions that *drip* can enter in Chapter 3.)

1.3 A construction-constraining mechanism

The grammar I am presenting has a mechanism which makes it possible to constrain verbs in such a way that they only enter constructions that one would expect them to appear in. A verb like *eat* is normally allowed into an arg1-construction (see (10a)) and an arg12-construction (see (10b)). Given that these are the constructions one wants the verb to appear in, the verb can be provided with the lexical constraint *arg1-12*, which means that it is either allowed into the arg1-construction or the arg12-construction, but no other construction.

- (10) a. John eats. (arg1-construction)
 b. John eats an apple. (arg12-construction)

The construction-constraining mechanism involves 8 “top” types, one positive type and one negative type for each of the first four subconstructions.⁹ (The positive types are named *arg1+* (arg1 plus), *arg2+* (arg2 plus), *arg3+* (arg3 plus), and *arg4+* (arg4 plus), and the negative types are named *arg1-* (arg1 minus), *arg2-* (arg2 minus), *arg3-* (arg3 minus), and *arg4-* (arg4 minus).) The types indicate whether or not a subconstruction is present in a clause. By default, a clause is assigned the four negative types. For each subconstruction that applies in the clause, the negative type is switched

⁹The arg5-sign is not a part of the mechanism. The PPs realized by the arg5-sign are in the implemented grammar treated as adjuncts.

to a positive type. So if an *arg1* sign applies, the type *arg1-* is switched to *arg1+*. An intransitive clause like (10a) has one subconstruction, the *arg1*-sign, and it therefore has the types *arg1+*, *arg2-*, *arg3-*, and *arg4-*. The unification of these types gives the construction type *arg1*, as shown in Figure 1.2.¹⁰ A transitive clause like (10b) has two subconstructions, the *arg1*-sign and the *arg2*-sign, and has the types *arg1+*, *arg2+*, *arg3-* and *arg4-*.

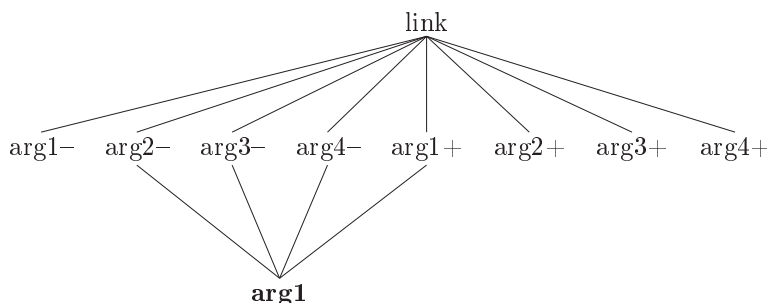


Figure 1.2: Supertypes of the construction type *arg1* in the *link* hierarchy

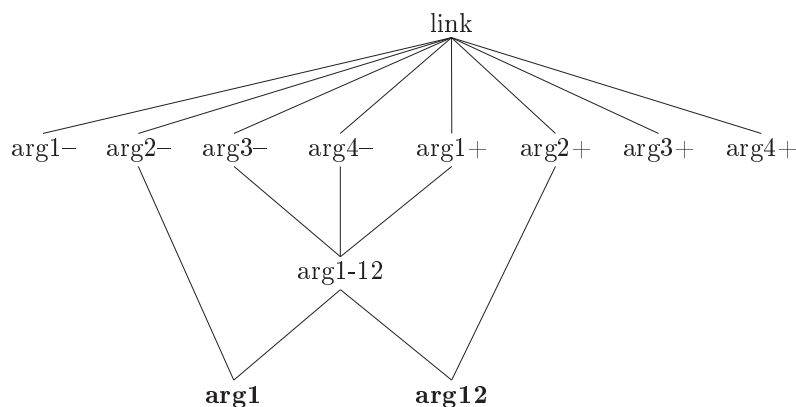
In order to limit the number of possible constructions a verb can enter, a set of “intermediate” types is introduced. The hierarchy in Figure 1.3 illustrates one such type, namely *arg1-12*. It inherits from *arg1+*, *arg3-*, and *arg4-*, and it has two subtypes, the construction types *arg1* (which inherits from *arg1-12* and *arg2-*) and *arg12* (which inherits from *arg1-12* and *arg2+*). The intermediate types represent lexical information associated with verbs, and they are unified with the four (positive or negative) subconstruction types of the clause. This forces a verb specified with the *arg1-12* type to occur in clauses with the *arg1*-construction or the *arg12*-construction.

The construction-constraining mechanism is not a part of strict syntax. Its function is to prevent *odd* sentences rather than *ungrammatical* sentences.¹¹ However, such a

¹⁰A more extended version of the hierarchy is given in Figure 4.9, p. 96. The full hierarchy is given in the file ‘nor.tdl’ in norsyg, under ‘valence types’ and includes 128 types.

¹¹I believe that there should be a distinction made between the ungrammaticality of examples like (xi) on the one hand, and the oddity of examples like (xii) on the other. While the examples in (xi) are unacceptable because of syntactic errors (in (xia) there is a past tensed verb in an infinitival clause, and in (xib) the determiner *a* does not agree with the noun *men*), the examples in (xii) are unacceptable because the main verbs enter constructions that they are not compatible with. The syntactic structures in the latter examples, I argue, are grammatical.

- (xi) a. *John tries to slept.
b. *A men smiles.

Figure 1.3: A partial *link* hierarchy

mechanism is necessary in order to keep the search space of a parser at a manageable level. When implementing a grammar, one has to attend to the grammar both as a linguistic theory and as a parser. This raises concerns that not always unite. For example, in principle I would like to allow all verbs (or maybe even all open class lexical items) to enter all constructions, but in a real implementation, this will make the parser too slow. The construction-constraining mechanism is designed for these concerns; see Chapter 4. (Some of the more technical aspects of the implemented grammar are also discussed in Appendix A.6).

To give an idea of the type of system I am proposing in these respects, imagine an LFG-like grammar differing from any actual LFG grammar in not obeying principles

-
- (xii) a. #John slept the car.
b. #John admires.

The common judgment of examples like (xiiia) and (xiiib) is that (xiiia) is grammatical, whereas (xiiib) is ungrammatical.

- (xiii) a. John filled the mouth with chocolate.
b. #John smiled the mouth with chocolate.

I argue that (xiiib) is not really ungrammatical, rather that it is very odd. (I will later in the thesis star “very odd” sentences like (xiiia), (xiiib) and (xiiib), even though I claim they are not ungrammatical.) It is possible to get some meaning out of (xiiib) by coercion. For example that John caused his mouth to be filled with chocolate by smiling. Or that John used chocolate to turn his mouth into a smile. The term ungrammatical I reserve for sentences like (xia) and (xib). These sentences could never be grammatical, irrespective of the meaning assigned to the open class lexical items. I will however use the term ungrammatical about sentences that are “very odd” later on, simply because that is the convention.

of Completeness and Coherence (see Bresnan (2001, 63)). In the lexical entries for verbs in such a grammar - being f-structure skeletons - there would be no GR-lists provided with the attribute ‘PRED’ (just lines like ‘PRED ‘kick’’). In such a grammar, due to phrase structure rules like (2), syntactic combination would still populate the f-structure with whatever GRs were encountered, and the resulting f-structure would provide a record of the GRs syntactically encoded in the construction parsed; however, without any mechanism checking whether such an assembly of GRs is accepted by the verb in question. This is in spirit how I would like a grammar to function. However, for concerns mentioned, we may want to include constraints in each lexical entry concerning admissible GRs. In the imaginary LFG grammar in question, one would then add the relevant specification inside the PRED value, e.g., ‘PRED ‘kick(Subj, Obj)’’. In my system, I similarly have one version of lexical entries where nothing is said about which arg-types a verb may combine with, and one line in which, for ‘kick’, for instance, I can insert the specification ‘arg12’ (cf. above). So far, though, this might seem just a pointless exercise of notational inventiveness. What are crucial contributions by my system are the following, however:

In the first place, in cases where a given verb has many environments, LFG and standard HPSG will posit as many entries for that verb as it has frames. My deployment of a type system as sketched, on the contrary, will allow me to have only one entry, which still accommodates all the frames. This will be shown in Sections 4.3 and 4.4. Secondly, this same type-design will allow me to use the actual parsing of a corpus as a way of incrementally defining the sum of frames in which a verb can enter, but as a resolution process working relative to the one single entry required. This will be shown in Section 4.5. Although the latter point has not yet been carried out on a large corpus, the mechanism is clear, and I see these two points as valuable technical contributions to parsing design and grammar engineering in general.

The way in which the unitary type definitions mentioned above depend on resolution by the syntactic environment, may raise the question whether this mechanism would apply also for a grammar where discriminants of multi-frame verbs involve semantic structure in addition to GFs. Of relevance are cases of non-isomorphy between semantic and syntactic structure. Having stated that I will not be concerned with semantics, it follows that I will not try to represent the ‘skewed’ syntax-semantics relationship of sentences like “I believe him to be sick” or “He seems sick.” By standard assumptions, the former will have a logical structure of the form ‘believe (I, he be sick)’, and the

latter ‘seem (he be sick)’, thus ‘believe’ here being logically a two-place predicate and ‘seem’ a one-place predicate. As far as arg-roles in my system are concerned, ‘believe’ will have three arg-roles and ‘seem’ two in these examples, since the analysis addresses syntactic structure exclusively. By these resolutions, I obviously will not get any semantic structure beyond what stands in a one-to-one relation to the GF structures. At least at its present stage of development, this can be seen as a limitation of my system, and I discuss what it may take for it to cope with these types of ‘skewed’ constructions in Section 6.7.4.

1.4 Exo-skeletal grammar and left-branching syntactic structures

I propose that the approach I am taking can be called an *exo-skeletal* approach in the sense of Borer (2005a, 15). This term is borrowed from zoology, where it is used to describe animals that have their skeleton on the outside. The opposite of *exo-skeletal* is *endo-skeletal*, which is used to describe animals with the skeleton inside the body, like humans. In an *exo-skeletal* grammar, the functional signs (function words, inflections and rules) are given more emphasis, while the role of the open lexicon (lexical entries of nouns, verbs and adjectives) is played down. In an *endo-skeletal* grammar, it is the lexemes that define what is outside, and the argument structure is fixed in the verb lexeme.

In an *exo-skeletal* grammar, the grammar can in principle only generate grammatical sentences even if the open class lexical items do not have any subcategorization constraints. This is an advantage that an *exo-skeletal* grammar has to a strictly *endo-skeletal* grammar, which crucially relies on the subcategorization constraints of open class lexical items. If the subcategorization constraints of the open class lexical items in an *endo-skeletal* grammar were left out, there would be nothing preventing ungrammatical sentences like (14) from being generated.

(14) *John eats an apple Mary that he smiles.

The ideas about *exo-skeletal* grammar that I present in this thesis, are implemented in the Norsyng grammar. The main objective of this grammar is this: I have wanted to make a grammar that does not make use of lexical rules or multiple lexical entries in

order to account for verbs with more than one construction. Thus, no matter how many argument frames a verb can occur in (and provided they are not distinct in terms of ‘strict syntax’), the lexicon will provide just one entry for the verb, and the multiplicity of frames will be induced from the different constructional environments solely.

To put this another way, Norsyg is different from lexicalist grammars in that open class lexical items are unconstrained by default. Restrictions can be made if there is a need for it. The common procedure in lexicalist grammars is to be very restrictive by default, that is, only to allow one construction on a lexical entry, and then create mechanisms that produce other possible constructions, mainly by means of multiple lexical entries or lexical rules.

Syntactic structures are assumed in general to be left-branching (see Figure 1.4), rather than mixed left- and right-branching (center-embedded) (see Figure 1.5), as assumed in HPSG and LFG, or right-branching (see Figure 1.6), as assumed in versions of GB/Minimalism using Larsonian shells (Larson, 1988; Culicover, 1997). With a left-branching structure, the first constituent will appear at the bottom of the tree (like the node *a* in Figure 1.4), and the last constituent will be the last daughter of the top rule (like the node *d*).

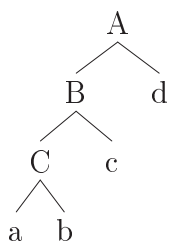


Figure 1.4: Left-branching tree

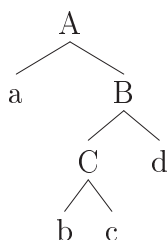


Figure 1.5: Mixed left- and right-branching tree

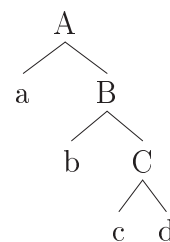


Figure 1.6: Right-branching tree

Left-branching syntactic structures make it possible to give an account of long distance dependencies where the *filler* appears at the bottom of the tree, and the *extraction site* c-commands the filler. That is, the position that the filler is assumed to be extracted from, is situated higher up the tree, as a sister of one of the ancestors of the filler. The information that there is a long distance dependency, passes through the nodes intervening between the filler and the extraction site. If there is a long distance dependency between the node *a* and *d* in the tree in Figure 1.4, this information will be local to the nodes *b* and *c* since it passes through their mothers (*C* and *B*). Given a

mixed left- and right-branching tree structure as shown in Figure 1.5, a long distance dependency between a and d will *not* be local to the nodes b and c , since it does not pass through their mother (C). In a right-branching tree structure as shown in Figure 1.6, the information that there is a long distance dependency between the nodes a and d , is again local to the nodes b and c , since it passes through their mothers B and C . In some languages (Sag (2005) mentions among other languages Chamorro and Irish), long distance dependencies are registered by verbs or complementizers. This indicates that such constituents have local access to long distance dependencies.

The left-branching structures allow for incremental parsing, with a bottom-up, left-to-right parsing strategy. The nodes of the tree in Figure 1.4 are then enumerated in the order shown in (15a). Also right-branching structures (often used in GB/Minimalism) allow for incremental parsing, if they are parsed with a left-corner parsing strategy. The nodes of the tree in Figure 1.6 are then enumerated in the order shown in (15b). Mixed left- and right-branching tree structures (used in LFG and HPSG) do not lend themselves to incremental parsing in the same way since these kinds of structures require storage proportional to the height of the tree. (I will return to parsing strategies in Section 5.2.)

- (15) a. a, b, C, c, B, d, A
 b. a, A, b, B, c, C, d

Given the left-branching structures assumed in this thesis,¹² the traditional notion of a syntactic constituent, is not applicable. What traditionally is conceived of as a syntactic constituent (a word or a phrase which can be replaced by a pronoun, which can be fronted, or which may be possible to coordinate) is rather reflected as a constituent in the Basic Relation Representation. Syntactic structures in this thesis are to a large part determined by the exo-skeletal nature of the grammar. A main verb may for example be regarded more as a modifier than as the syntactic head of a clause. A complementizer may form a constituent together with the matrix clause, rather than forming a constituent together with the rest of the subordinate clause. The syntactic structures reflect how words and phrases combine and form new constituents, but as mentioned, these constituents are not necessarily constituents in the traditional sense.

¹²There are some cases where the left-branching tree structures are not employed in the analyses, like in PPs and some cases of coordination, but these cases have not been the focus of my study. I will also make use of a STACK in order to account for embedded clauses. This implies that parsing will not be fully incremental.

Rather, they can be seen as the history of combinations of words and phrases of a sentence.

The grammar formalism I am presenting in this thesis borrows ideas from several grammatical theories, including HPSG, Construction Grammar, LFG, and GB. The fact that the grammar is a typed feature structure grammar and designed for bottom-up chart parsing (Kay, 1986), is due to the fact that it is implemented with the LKB system (Copestake, 2002). Since the formalism was developed from the Grammar Matrix (Bender *et al.*, 2002), the terminology used to represent grammatical objects is to a large degree taken from HPSG.

The idea of one lexical entry per stem (and no lexical rules) and that constructions have meaning independent of the words that appear in them is inspired by Construction Grammar, but while constructions in my grammar formalism can be decomposed into subconstructions, constructions in Construction Grammar are seen as entities that cannot be analyzed further (see Section 2.4).

As already mentioned, the grammatical relations assumed to hold between a predicate and its arguments can be compared to the grammatical relations used in LFG, but there is no one to one correspondence.

Apart from apparent similarities to HPSG, Construction Grammar, and LFG, the grammar formalism is maybe best conceived of as a monostratal variant of GB (Chomsky, 1986) where surface grammatical relations, deep grammatical relations, and movements are represented at one level. Movement to the specifier position of C (accounting for wh-movement/long distance dependencies in GB) is accounted for by means of the percolation of a feature SLASH as in HPSG (but as I will show in Section 6.9, the approach in this grammar formalism differs in several respects to the approaches in HPSG). Movement to an argument position as assumed in cases where an argument receives thematic role from one verb and case from another verb (accounting for raising constructions and small clauses in GB) is not possible. Instead, the grammar formalism allows for an argument to be realized twice in these cases. This corresponds to assuming an argument similar to PRO in GB. (See discussion in Sections 6.7.4 and 9.5.2.) Passive is accounted for by assuming that what corresponds to the external argument in GB is realized by the passive auxiliary or the passive morpheme (see Section 7.1). The formalism does not imply anything corresponding to head movement in GB (V to T and/or T to C movement), but certain positions correspond to C, T, and V, and the categories appearing in these positions are assumed to originate in these positions (see

Section 9.3).

The left-branching tree structures result in tree structures completely different from the right-branching structures known from GB (and from syntactic structures in any other theory, except perhaps from CCG), and constituents in the traditional sense are not formed. Still, given the different parsing strategies associated with the two approaches (a bottom-up, left-to-right parsing strategy in the approach presented in this thesis vs. a left corner parsing strategy argued to be appropriate for GB (see Section 5.2)), preterminals are enumerated in the same order. This will be demonstrated in Chapter 9.

1.5 Layout of the thesis

The first part of the thesis includes Chapters 2–4 and deals with argument structure. In Chapter 2, I introduce some central notions in the discussion around argument structure, such as unaccusativity and unergativity, valence alternations and voice. I discuss how HPSG, LFG, Construction Grammar, and three versions of Minimalism deal with argument structure. I look at how much argument structure information the theories assume is present in the lexicon, and how much they assume can be reduced to syntax, and I situate the theories on a scale lexicalist \leftrightarrow non-lexicalist (or endo-skeletal \leftrightarrow exo-skeletal). In Chapter 3, I go through a number of the valence alternations and constructions presented in Levin (1993), and show how these alternations can be accounted for syntactically with the five subconstructions that I am assuming. I will present the Basic Relation Representations (BRRs) that are employed in the grammar. In Section 3.5, I suggest four basic sign types which represent the realization of the first four subconstructions. I show how different syntactic instantiations of the subconstructions inherit from the basic signs. In Chapter 4, I show how valence can be represented in a grammar formalism where argument structure can be inferred from functional signs. I introduce four valence features, one for each of the first four subconstructions. These will carry positive and negative values, reflecting whether the argument is realized or not. Further, I introduce a hierarchy which allows me to give a compact representation of possible constructions for a lexeme. I give some examples of lexical entry types, and present methods for expanding the lexicon. Finally I compare my approach to a lexicalist version of the grammar, the Robust Accurate Statistical Parsing (RASP) system, and other Norwegian computational resources.

The second part of the thesis includes Chapters 5–10. In this part I show how an exo-skeletal grammar may be structured. I present analyses of a range of linguistic phenomena. Chapter 5 gives a preliminary introduction to the syntactic structures I am assuming. I will present some methodological considerations concerning linguistic, conceptual, and computational aspects of the approach. In Chapter 6, I present the basic syntactic interior of a grammar for Norwegian. I suggest six main kinds of rules. First, the *valence* rules, which realize the first four kinds of subconstructions. Second, the *filler* rule, which fills in the extracted constituent. Third, the *merge* rule, which for example combines a projection headed by a complementizer or an auxiliary with the main verb. Forth, the *subordination* rules, where embedded clauses are entered. Fifth, the *clause boundary* rules, which mark the boundary of the clauses. Sixth, the *modifier* rules, which let a modifier modify a head projection. The chapter gives analyses of main clauses, subordinate clauses, relative clauses and infinitival clauses. There is also a section on long distance dependencies. In Chapter 7, I present analyses of passive and presentation. In Chapter 8, I present four kinds of coordination in Norwegian, and argue that it is an advantage to use an exo-skeletal grammar in for example the analysis of coordination of Vs. In Chapter 9, I compare the analysis presented in Chapter 6 with GB, and use the comparison to illustrate how syntactic structures of basic clauses in English can be accounted for. In Chapter 10, I present an analysis of sentence adverbials in Norwegian in light of the analysis presented in Chapter 6.

Appendix A has information about the Norsyg grammar, where the analysis presented in this thesis is implemented. Appendix B has information about an English and a German demo grammar, which I have developed in order to illustrate how the analysis can be extended to other languages. All the Norwegian and English examples in this thesis are gathered in the files ‘ex.items’ and ‘eng-ex.items’, distributed with Norsyg, and the results of batch parses of these sentences with the Norsyg grammar and the English demo grammar are given in Appendix C. Basic Relation Representations (BRRs) of all analyses conducted with the Norsyg grammar and with the English and German demo grammars are given in Appendix D.

Part I

Argument Structure

Chapter 2

Argument structure in HPSG, LFG, Construction Grammar, and Minimalism

2.1 Introduction

In this chapter I will look at how HPSG (Head-Driven Phrase Structure Grammar), LFG (Lexical Functional Grammar), Construction Grammar, and three approaches within Minimalism treat argument structure and valence alternations. The three Minimalist approaches are Hale and Keyser’s Prolegomenon to a Theory of Argument Structure, Ramchand’s “First Phase Syntax” and Borer’s neo-constructionist approach. I have chosen three Minimalist approaches that span from a lexicalist approach to argument structure to a strict non-lexicalist approach to argument structure. I will present how the theories account for the most basic argument frames of intransitive verbs (both unergative and unaccusative), transitive verbs, and ditransitive verbs. I will also show how they do valence alternations like passive, the causative/inchoative alternation and resultative constructions.¹ I aim at situating the frameworks on a scale lexicalist – non-lexicalist by classifying them with regard to three main criteria:

¹Studies by Boguraev and Briscoe (1989) and Manning (2003) show that it is difficult to give good criteria for when valence alternations can apply. Corpus evidence presented in Bangalore and Joshi (1999) shows that lexical items on average are associated with as many as 47 supertags, which are bundles of phrase structure information and dependency information.

1. *Variable behavior verbs*² – Whether the alternation between unaccusativity and unergativity of the same verb is treated as part of the lexicon or as part of the syntax.
2. *Valence alternations* – Whether alternations such as the difference in arity,³ the causative/inchoative alternation, the dative alternation, the spray/load alternation and the resultative construction are accounted for lexically or syntactically.⁴
3. *Voice* – Whether active, passive and middle voice is treated lexically, or as a part of the syntax.

Generally speaking, frameworks like HPSG and LFG will be shown to classify mostly as lexicalist with regard to all three criteria. The Minimalist frameworks I will be considering differ with regard to the three criteria. Before I discuss the frameworks in detail, I will present some linguistic notions that I will use in this section. Much of the material I present is taken from or inspired by Levin (1993). I will consider argument frames that occur in Norwegian and English.

2.1.1 Unergative and unaccusative verbs

The difference between *unergative* and *unaccusative* verbs has been an issue in linguistics for a long time (see Jespersen (1924, 164-167), Fillmore (1968), Perlmutter (1978) and Levin and Hovav (1995)).

Unergative (or “real” intransitive) verbs are verbs like *smile*, *laugh* and *sing*. These verbs may passivize in Norwegian. They can not transitivize in the sense that a causer is added to the event. This is illustrated by (16) where (16a) is grammatical and (16b) is ungrammatical.

- (16) a. The man smiled.
 b. * Mary smiled the man. (On the interpretation that Mary caused the man to smile)

²I have taken this notion from Borer (2005b, 30-46).

³By difference in arity I mean whether a verb can shift between intransitive and transitive, and transitive and ditransitive.

⁴Variable behavior is not treated as part of *valence alternations* since variable behavior in some theories cannot be accounted for by means of one root/lexical item, while in other alternations it can. This makes the lexicalist – non-lexicalist distinction more fine-grained.

Unaccusative verbs on the other hand are intransitive verbs like *arrive*, *die* and *fall*. These verbs cannot passivize. An intuition behind this group of verbs is that their argument corresponds to the object of a transitive clause. If we include the intransitive versions of verbs like *break*, *widen*, and *crack* to the unaccusative verbs, we see that these verbs may transitivize by adding a causer, as illustrated in (17) where the causer *Mary* is added in (17b). The object of the causativized version correspond to the subject in the intransitive version. This phenomenon is often referred to as the *causative/inchoative alternation*.

- (17) a. The glass broke.
 b. Mary broke the glass.

It is possible for an unergative verb to have an object added while maintaining the semantic role of the subject as illustrated in (18a). An object like *a big smile* in (18a) is usually referred to as a *cognate object*. Unaccusative verbs on the other hand cannot have such objects, as (18b) illustrates. In order for (18b) to be grammatical, the subject cannot be the argument that is being broken, as it is in (17a).

- (18) a. Mary smiled a big smile.
 b. * The glass broke a crack. (On the interpretation that the glass is breaking)

Some verbs are ambiguous between an unaccusative and an unergative reading, like *drip* in (19). Either the subject is the source of the dripping, as in (19a) (unergative reading), or the subject is what is dripping, the theme, as in (19b) (unaccusative reading). These verbs, as said above, are called *variable behavior verbs*.

- (19) a. The roof drips.
 b. Water drips (from the roof).

Data such as those presented in examples (16)-(18) have made linguists propose that the syntactic subject of an unaccusative verb as in (17a) is really an underlying object or internal argument of the verb, since this argument functions as object if a causer is added as in (17b) (see for example Fillmore (1968); Perlmutter (1978)).

2.1.2 Other alternations

Transitive verbs and unaccusative verbs can have the *resultative construction*, as illustrated in (20) and (21). In the resultative construction a predicative element

(typically a PP or an adjective) predicates over the “underlying object”. In (20b), the predicative element predicates over the object of an active transitive verb, and in (21b), it predicates over the subject of an unaccusative verb. An unergative verb (which does not have an underlying object) can not express the resultative construction, as illustrated in (22).

(20) a. John hammered the metal.

b. John hammered the metal flat.

(21) a. The river froze.

b. The river froze solid.

(22) a. The man smiles.

b. * The man smiles happy. (On the interpretation where the man becomes happy)

Some overtly transitive verbs like *eat*, *read* and *paint* may have an *understood object* that may or may not be expressed, as illustrated with the pair in (23). This is one form of alternation in arity.

(23) a. John ate the apple.

b. John ate.

The *dative alternation* is an alternation between a ditransitive verb, as in (24a), and a transitive verb with a PP complement, as in (24b). The indirect object of the ditransitive verb (*Mary*) corresponds to the prepositional object of the transitive verb. The indirect object of the ditransitive verb must be something that can take the direct object into its possession. This interpretation is not necessarily present for the prepositional object of the transitive verb (see Pinker (1989, 48)).

(24) a. John gave Mary an apple.

b. John gave an apple to Mary.

The *spray/load alternation* is an alternation between two transitive verbs with a PP complement. In one variant the object is the argument whose location is changed, and the PP is the new location (see (25a)). In the other variant the object is the location and the prepositional object is the argument that has changed location (see (25b)).

- (25) a. John loaded hay onto the wagon.
 b. John loaded the wagon with hay.

2.1.3 Voice

English has *active*, *passive*, and *middle* voice, as illustrated in (26).⁵

- (26) a. The butcher cuts the meat.
 b. The meat was cut (by the butcher).
 c. The meat cuts easily.

The transitive verb *cut* can be the main verb of clauses with all three voices. (26a) is an *active* sentence. So far in this section all sentences have been active. (26b) is a passive sentence. Passive is usually either periphrastic, as in English (passive auxiliary + past participle) or morphological (marked with an affix on the main verb). When a clause is *passive*, as in (26b), the subject of the corresponding active clause (in this case *the butcher*) is expressed in an optional PP_{by}. Some other element is realized as the subject. In English, this will be the object that in active is closest to the verb (i.e. *the meat* in (26b)). Even though the agent may not be expressed, there is still a notion of some causer of the situation expressed. In this sense, passive sentences differ from sentences with unaccusative verbs (see (17a)) where there is no notion of a causer. ((17a) does not convey that the breaking event is caused by anyone or anything, it just happened.)

(26c) is a sentence with *middle* voice. A sentence with middle voice has no particular marking in English except that it usually contains an adverb like *easily* in (26c).⁶ The subject of the corresponding active clause (*the butcher*) is not expressed. Still there is a notion of causation, which is not present in the unaccusative clause. Compare for

⁵I here change perspective and present *voice* as a property of clauses, rather than a property inherent to verbs. I could also have taken the clause perspective for the alternations I have presented in the previous sections, but since most of the literature seems to treat these alternations as lexical alternations, rather than as syntactic alternations, I have used the lexical perspective.

⁶Norwegian does not have middle voice. Instead of middle, the sequence *let* + reflexive + main verb is used as in (xxvii). Languages like Spanish and Russian mark middle with a reflexive suffix.

(xxvii) Studiet lar seg lett kombinere med en jobb.
 study-DEF lets itself easily combine with a job

‘The study combines easily with a job.’

example *The cup broke* with *The cup broke easily*. In the latter example there is a notion of something external to the cup that made it break, while this notion is not available in *The cup broke*.

Having sketched the intuitions behind verb alternations and voice, I now proceed to a discussion of different theoretical frameworks and how they relate to the phenomena I have presented.

2.2 HPSG

In HPSG, the argument frame of a verb is to a large extent determined when the verb enters the syntax. A lexical item is a sign consisting of phonological, syntactic and semantic information, as illustrated in Figure 2.1.⁷

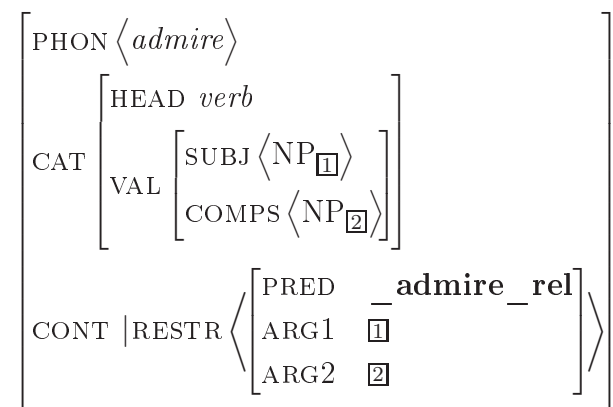


Figure 2.1: Lexical entry for the verb *admire*

The phonological information is usually represented as a list of strings (see the value of PHON in Figure 2.1) The syntactic information is represented as a feature structure as value of the feature CAT. The semantic information is represented as the value of CONT.⁸ The verb *admire* is transitive, and this is reflected on the valence lists SUBJ and COMPS.⁹ The SUBJ list contains an NP (the subject) and the COMPS list contains

⁷There are different naming conventions for features in HPSG. I will be using the ontology of features that is used in Pollard and Sag (1994), Chapter 9. These features are also used in the English Resource Grammar (ERG) Flickinger (2000).

⁸In parts of the literature the features SYN and SEM are used instead of CAT and CONT.

⁹In my presentation of HPSG I use the valence features SUBJ and COMPS as in Borsley (1996). In parts of the HPSG literature, there is only one valence list, SUBCAT (e.g. Pollard and Sag (1994) (Chapter 1-8), Müller (2002)), while in other parts of the literature the feature ARG-ST (or ARG-S) has as value the concatenation of the SUBJ list and the COMPS list Manning (1996) and Sag *et al.* (2003).

an NP (the object). They are co-indexed with the first and the second argument of the predicate respectively.^{10 11}

An intransitive verb like *smile* has an empty COMPS list, and only one semantic argument as shown in Figure 2.2.

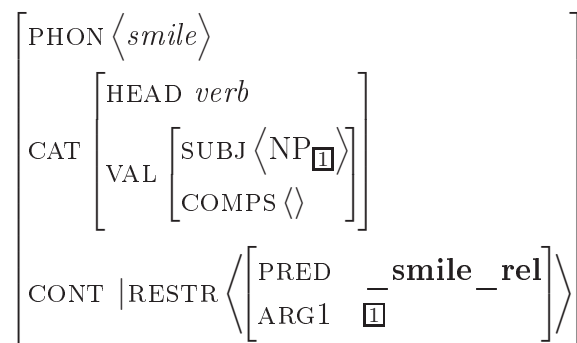


Figure 2.2: Lexical entry for the verb *smile*

A ditransitive verb like *give* has two elements on the COMPS list and three semantic arguments as shown in Figure 2.3. The first element on the COMPS list is the indirect object and the second element on the COMPS list is the direct object. The direct object is linked to the second argument and the indirect object is linked to the third argument.

Passive is usually accounted for with a lexical rule (Pollard and Sag (1994), Sag *et al.* (2003)). In Figure 2.4, I show a simplified version of what the passive lexical rule may look like. What comes before the arrow, is the input to the lexical rule and what comes after, is the output. As can be seen, the first complement of the input lexeme ($\boxed{1}$) is the subject of the output. The rest of the complement list ($\boxed{2}$) of the input lexeme becomes the complement of the output. This means that a passive lexeme is derived from an active lexeme.

Alternatives to this approach are suggested for German in Kathol (1994), Pollard (1994) and Müller (2007, 272-273), where the passive auxiliary determines the realization of the arguments of the past participle, and there is no need for lexical

¹⁰It is an HPSG convention that lowered subscripts, as those attached to the NPs in Figure 2.1, abbreviate a link to the semantic index.

¹¹There are different conventions for displaying semantic information. In some approaches features like ADMIRER and ADMIREE are used (e.g. Pollard and Sag (1994) and Sag *et al.* (2003)), and in other approaches thematic roles like AGENT, THEME and EXPERIENCER are used (Müller (2002)). I will follow the convention in Copestake *et al.* (2005) with argument names like ARG1, ARG2, ARG3 and ARG4. To a certain degree, these argument names correspond to the syntactic relations that are expressed by the subconstructions assumed in this thesis.

$$\left[\begin{array}{l} \text{PHON } \langle \textit{give} \rangle \\ \text{CAT } \left[\begin{array}{l} \text{HEAD } \textit{verb} \\ \text{VAL } \left[\begin{array}{l} \text{SUBJ } \langle \text{NP}_{\mathbf{1}} \rangle \\ \text{COMPS } \langle \text{NP}_{\mathbf{2}}, \text{NP}_{\mathbf{3}} \rangle \end{array} \right] \end{array} \right] \\ \text{CONT | RESTR } \left\langle \begin{array}{l} \text{PRED } \textit{_give_rel} \\ \text{ARG1 } \mathbf{1} \\ \text{ARG2 } \mathbf{3} \\ \text{ARG3 } \mathbf{2} \end{array} \right\rangle \end{array} \right]$$

Figure 2.3: Lexical entry for the verb *give*

$$\left[\begin{array}{l} \text{CAT | VAL } \left[\begin{array}{l} \text{SUBJ } \langle \text{NP} \rangle \\ \text{COMPS } \langle \mathbf{1} \rangle \oplus \mathbf{2} \end{array} \right] \end{array} \right] \Rightarrow \left[\begin{array}{l} \text{CAT | VAL } \left[\begin{array}{l} \text{SUBJ } \langle \mathbf{1} \rangle \\ \text{COMPS } \mathbf{2} \end{array} \right] \end{array} \right]$$

Figure 2.4: Passive lexical rule

rules.

Other verb alternations are accounted for with lexical rules (Sag *et al.* (2003, 262-263), Müller (2002, 240-247) and Davis (2001, 274)). The lexical rule for deriving a transitive resultative verb from an intransitive unergative verb may look as in Figure 2.5.¹²

$$\left[\begin{array}{l} \text{CAT | VAL } \left[\begin{array}{l} \text{SUBJ } \langle \mathbf{1} \rangle \\ \text{COMPS } \langle \rangle \end{array} \right] \\ \text{CONT | RESTR } \left\langle \left[\text{ARG1 } \mathbf{2} \right] \right\rangle \end{array} \right] \Rightarrow \left[\begin{array}{l} \text{CAT | VAL } \left[\begin{array}{l} \text{SUBJ } \langle \mathbf{1} \text{ NP}_{\mathbf{2}} \rangle \\ \text{COMPS } \langle \mathbf{2} \text{ NP}_{\mathbf{3}}, \text{AP/PP}_{\mathbf{4}} \rangle \end{array} \right] \\ \text{CONT | RESTR } \left\langle \begin{array}{l} \text{ARG1 } \mathbf{2} \\ \text{ARG2 } \mathbf{3} \\ \text{ARG3 } \mathbf{4} \end{array} \right\rangle \end{array} \right]$$

Figure 2.5: Resultative lexical rule

What is displayed in Figure 2.5 is that an NP and a PP or AP are added to the COMPS list of the output verb, and that two semantic arguments are added as well. The result state is linked to the third argument.

¹²The lexical rule in Figure 2.5 is based on the resultative lexical rule for unergatives in Müller (2002, 241).

Sag *et al.* (2003, 262-263) suggest to account for also dative alternation and locative alternations with lexical rules.

The HPSG literature also has approaches to valence alternations that make less use of lexical rules. In case of verbs like *eat*, that may have unexpressed objects, the object may be considered optional, as suggested in Flickinger (2000, 22-24).

Riehemann (2001, Chapter 7) employs a type hierarchy with a type *stem* on the top and possible versions of stems as subtypes. At the bottom of the hierarchy are types for fully inflected linguistic objects (see Riehemann (2001, 264)). Between the type *stem* and the linguistic object types, are generalizations over linguistic objects. The approach claims to make it possible to avoid the use of lexical rules. Instead, a stem can undergo complex type constraints as it is forced down the hierarchy. The type constraints can be recursive so that more than one affix can be added. Type resolution makes sure that linguistic objects are bottom types in the hierarchy. Riehemann's approach relies on complex type constraints and type resolution, which are powerful mechanisms and not available in the LKB system. It is difficult to see whether this approach is better than a lexical rule approach since this approach seems to have the same complexity in the type system as an ordinary HPSG approach has in the lexical rules. Since the approach uses type resolution, words must be fully specified when they are combined with other words/phrases. So there is no way to delay the decision of which argument frame a word has in case of valence alternations where no inflection is involved (e.g. the dative alternation in English). In the approach taken in this thesis, the type hierarchy is also playing a crucial role, but while Riehemann uses the type hierarchy to allow for underspecified lexical entries and forces words to be fully specified, I allow both for underspecified lexemes and underspecified words, and let the syntax help constrain the argument frame. This delays the decision on which argument frame a word has until the syntactic context has been made available to the word. Also the formal apparatus differs. In the approach taken in this thesis, complex type constraints and type resolution are not employed.

2.3 LFG and the Lexical Mapping Theory (LMT)

In this section I will sketch the theory for mapping semantic arguments onto syntactic functions proposed in Bresnan (2001, 302-321) and Dalrymple (2001, 195-215). This mapping takes place in the lexicon and gives an account of valence alternations without

using lexical rules or multiple lexical entries.

LFG assumes an argument structure (a-structure) which consists of a predicator and its argument roles. These roles are associated with a feature $[\pm r]$ or $[\pm o]$ and ordered with regard to the thematic hierarchy in (28):

(28) **Thematic Hierarchy:**

agent \succ beneficiary \succ experiencer/goal \succ instrument \succ patient/theme \succ locative

According to the Lexical Mapping Theory the verb *pound* has the a-structure in (29).

$$(29) \quad \begin{array}{c} \textit{pound} < & x & y & > \\ & [-o] & [-r] & \end{array}$$

Here, *pound* is the predicator, and *x* and *y* are its two argument roles. The *x* is the agent role, and comes first in the a-structure since *agent* is the most prominent role in the Thematic Hierarchy. The *y* is the patient role.

The $[\pm r]$ and $[\pm o]$ features determine what syntactic function the argument roles get. $[\pm r]$ says whether the syntactic function is restricted or not. $[\pm o]$ says whether a syntactic function is objective or not. With these two features the syntactic functions can be grouped into four classes, SUBJ, OBJ, OBJ_Θ and OBL_Θ:

	$-r$	$+r$
(30) $-o$	SUBJ	OBL _Θ
$+o$	OBJ	OBJ _Θ

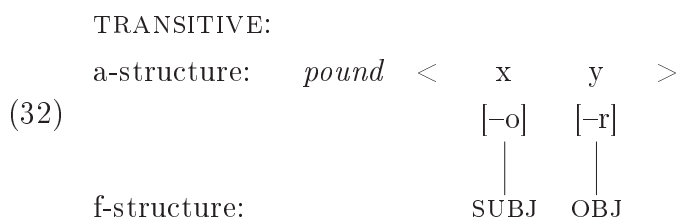
SUBJ is the subject of the clause. In English, OBJ is the first object of the clause (the direct object in a transitive clause or the indirect object in a ditransitive clause). OBJ_Θ is in English the second object of a clause (the direct object of a ditransitive clause). OBL_Θ is an argument which is not a subject and not an object, for example a PP complement.

As can be seen in the a-structure of *pound*, the *x* and the *y* have only one feature instantiated. The *x* is $[-o]$ and the *y* is $[-r]$. So the syntactic functions are not yet determined. This is done with the help of a couple of *mapping principles*. The first mapping principle says (i) that the most prominent role in the a-structure, marked with $[-o]$, becomes the subject. (31a) is an example of this. But (ii) if there is no such $[-o]$ role, a non-agentive role marked with $[-r]$ will become the subject. (31b) is an example

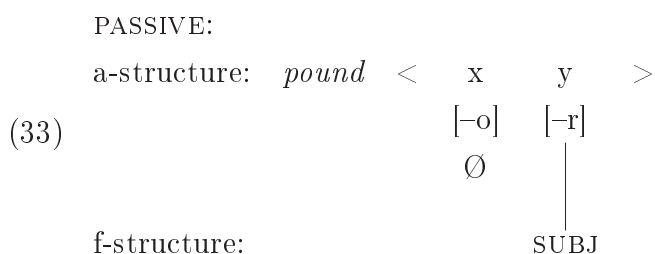
of this. The second principle deals with the mapping of the rest of the arguments. I will not go further into how this is done here (see Bresnan (2001, 309-311)).

- (31) a. John pounds the metal.
 b. The metal was pounded.

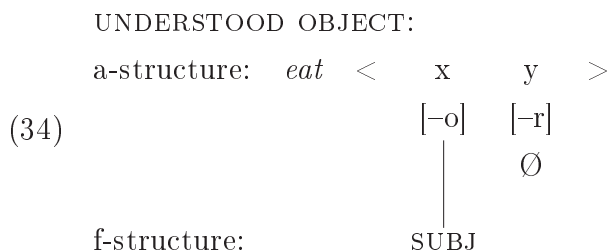
Given the a-structure of *pound* and the first mapping principle, we see that the first argument role x will be mapped to SUBJ since it is the most prominent role and has the $[-o]$ feature. The second principle will map the second argument role y onto the OBJ function. Syntactic functions are represented in f-structure, which serves as a link between the argument structure and expression structure (c-structure) (Bresnan, 2001, 9-10).



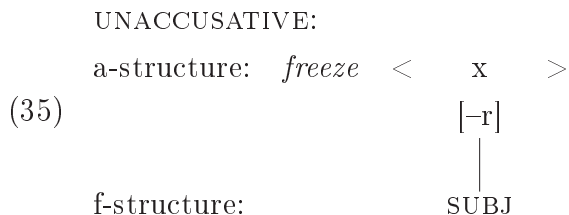
However, if a semantic argument is not marked with a positive restricted/ objective feature, it may be possible to “suppress” it. This happens in passive, where the most prominent role is suppressed. As (33) shows, the role with the $[-r]$ feature will be realized as subject (due to the second part of the first mapping principle).



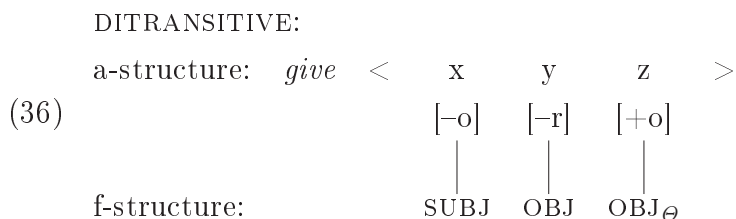
In alternations where an understood object is not realized (“understood object alternations”), an argument role (patient or theme) marked with $[-r]$ is suppressed:



Unaccusatives are assumed to have one semantic argument which has the $[-r]$ feature. This argument will be mapped to the SUBJ function due to the second part of the first mapping principle and the fact that every predicator must have a subject, as shown in (35).

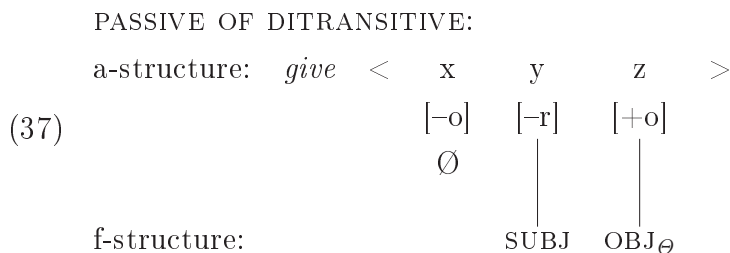


Ditransitive verbs have a mapping as in (36).

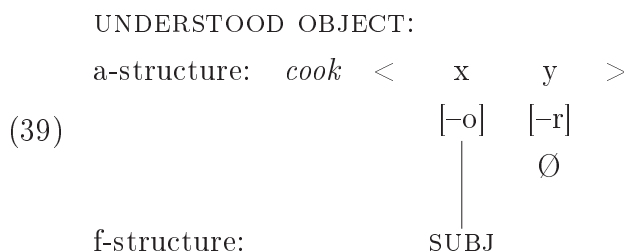
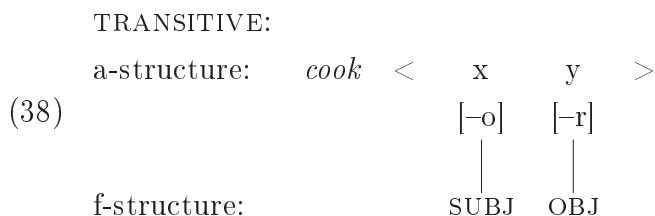


As (36) shows, ditransitives have three argument roles. In English, only one role can have the $[-r]$ feature, and it is given to the primary patient-like role. In (36), this is the recipient y . The lower patient role (according to the thematic hierarchy in (28)) z gets the feature $[+o]$.

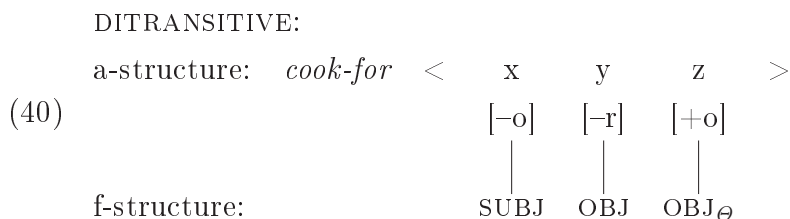
In passive, the semantic role with the $[-r]$ feature is mapped to the SUBJ function (second part of the first mapping principle). This is illustrated in (37). This prevents the direct object of a corresponding active ditransitive verb to become the subject in passive, which is usually judged as ungrammatical in English.



It is possible for one form can be both intransitive, transitive, and ditransitive. Bresnan (2001) uses the verb *cook* as an example. When used transitively and intransitively the verb has the a-structures in (38) and (39). As is shown, the intransitive variant has an a-structure with two roles where one argument y is suppressed.



When *cook* is used ditransitively, it gets another predicator *cook-for*, and the number of argument roles increases with one as illustrated in (40).



This means that the verb *cook* needs two a-structures. Since a-structures are projected from the lexical semantics, this seems to suggest that there are two concepts *cook*.

(41) has examples of active and passive ditransitives in Norwegian. In active, the agent role is linked to SUBJ (see (41a)), while in passive both the most prominent patient-like role (active indirect object) (see 41b) and the less prominent patient-like role (active direct object) (see (41c)) can be mapped to SUBJ. This is also pointed out in Lødrup (1995, 323–325). In addition an expletive *det* may function as subject (see (41d)). In order to allow both the patient-like roles to be mapped to SUBJ one could let both of them have the $[-r]$ feature required by the second part of the first mapping principle as illustrated in (42).

- (41) a. Jon overrekker Kari to bananer.
 Jon hands Kari two bananas
 ‘Jon hands Kari two bananas.’

- b. Kari blir overrakt to bananer.
Kari becomes handed two bananas
'Kari is handed two bananas.'
- c. To bananer blir overrakt Kari.
Two bananas becomes handed Kari
'Kari is handed two bananas.'
- d. Det blir overrakt Kari to bananer.
It becomes handed Kari two bananas
'Kari is handed two bananas.'

DITRANSITIVE:

- (42) a-structure: *overrekke* < x y z >
[-o] [-r] [-r]

But this could cause problems, since it now should be possible to suppress the less prominent patient role, and we could generate clauses with an agent role and a recipient role, which would be very odd or ungrammatical, as illustrated in (44).

DITRANSITIVE:

- (43) a-structure: *hand* < x y z >
[-o] [-r] [-r]
∅

- (44) ??/* Jon overrekker Kari. (On the interpretation that *Kari* is a recipient)
Jon hands Kari

Another possibility would be to change the second part of the first mapping principle so that it also allowed for [+o] argument roles to be mapped to SUBJ. This would be a bit strange since [+o] means *objective*.

Lødrup (2000) and Lødrup (2004, 10-11) points out that in Norwegian it is possible to have a presentational construction with an expletive (*det*) functioning as subject if there is no *agent* role mapped to SUBJ (see (45a)). He also shows that an agent role can function as object (see (45b)). This is a challenge to the first lexical mapping principle that requires that an agent is mapped to SUBJ, and if there is no agent, the most prominent patient role is mapped to SUBJ.

- (45) a. Det forsvant en mynt i gresset. (theme [-r])
 it disappear-Past a coin in grass-Def
 ‘A coin disappeared in the grass.’
- b. Det lekte noen barn i gresset. (agent [-r])
 it play-Past some kid-Pl in grass-Def
 ‘Some kids played in the grass.’

2.4 Construction Grammar (CG)

While frameworks like HPSG and LFG are mainly lexicalist, Construction Grammar (Fillmore *et al.*, 1988; Kay and Fillmore, 1999; Goldberg, 1995) lets the syntax play a more important role. Goldberg (1995) gives a number of phrasal constructions that independent of the lexical meaning of the words can be said to have a meaning. Examples of such constructions are:

- i) *The English Ditransitive Construction* (see (46)), which has the following syntactic active structure: [SUBJ [V OBJ OBJ2]],
- ii) *The English Caused-Motion Construction* (see (47)), which has the following syntactic active structure: [SUBJ [V OBJ OBL]],
- iii) *The English Resultative Construction* (see (48)), which has the following syntactic active structure: [SUBJ [V OBJ OBL]], and
- iv) *The Way Construction* (see (49)), which has the following syntactic active structure: [SUBJ_{*i*} [V [POSS_{*i*} way] OBL]]

(46) Sally baked her sister a cake. (Goldberg, 1995, 141)

(47) They laughed the poor guy out of the room. (Goldberg, 1995, 152)

(48) He talked himself blue in the face. (Goldberg, 1995, 189)

(49) Frank dug his way out of the prison. (Goldberg, 1995, 199)

Typical for verbs appearing in these constructions is that their argument frames are not necessarily predictable from the verb's semantics. In Construction Grammar, the argument frames can be contributed by the constructions, and the meaning is composed by the verb's semantics and the construction it appears in. There is no need to assume

several verb meanings for the same stem in order to account for a verb with more than one possible argument frame.

The notion of *non-compositionality* is central in Construction Grammar. Constructions as the ones just mentioned are argued to be semantic entities that cannot be analyzed further. Goldberg (1995, 4), gives the definition in (50) of a construction.

(50) C is a CONSTRUCTION iff_{def} C is a form-meaning pair $\langle F_i, S_i \rangle$ such that some aspect of F_i or some aspect of S_i is not strictly predictable from C's component parts or from other previously established constructions.

This seemingly goes against the assumption made in this thesis, namely that events can be decomposed into underlying events. This is however not the case. The two approaches focus on different issues. While the underlying event analysis assumed in this thesis allows for further interpretation of the event, settling on thematic roles, or in the case of idiomatic expressions, arriving at the meaning of the idiomatic construction (both of which would be out of the scope of this thesis), the Construction Grammar approach seems to get directly at the most specific meaning. This means that the Basic Relation Representation assumed in this thesis is more abstract than the semantics assumed in Construction Grammar. The fact that one interpretation of an abstract construction is unanalyzable, does not mean that the abstract construction itself cannot be decomposed.

The relation between the abstract constructions assumed in this thesis and the Construction Grammar constructions illustrated by the examples (46)–(49) can be conceived of in terms of a hierarchy as shown in Figure 2.6. In the approach taken in this thesis, the examples belong to two constructions types, the *arg123-construction* and the *arg124-construction*. The English Ditransitive Construction can be said to be an instance of the *arg123-construction*, and the English Caused-Motion Construction, the English Resultative Construction, and the Way Construction can be said to be instances of the *arg124-construction*. (Construction types like the *arg123-construction* and the *arg124-construction* were briefly mentioned in Section 1.2. I will return to construction types and how they are composed in Chapter 3)

It is not quite clear how the constructions are realized in Construction Grammar. In Goldberg (1995, 192) a resultative construction is realized as a ternary branching rule (*V OBJ OBL*), and in Goldberg and Jackendoff (2004), the resultative construction is a phrase structure rule *V NP AP/PP*. However, in Sign-Based Construction Grammar

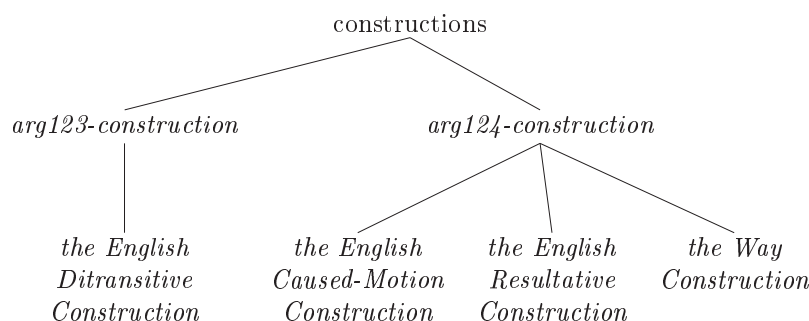


Figure 2.6: Norsyg and Construction Grammar construction types

(see Sag *et al.* (2003, Chapter 16)), a distinction is made between lexical and phrasal constructions, where lexical constructions correspond to lexical rules in HPSG, and phrasal constructions correspond to phrases in HPSG. According to Michaelis (2005), the Caused-Motion construction does not specify the function of the agent and the theme, since the Active or the Passive construction have to apply before the function of these roles are settled. The Caused-Motion construction has to apply before the Active or the Passive construction. Since in some languages, passive is marked by means of inflection, the construction would need to be a lexical construction, and not a phrasal construction, as suggested by Goldberg.

2.5 GB/Minimalism

2.5.1 Passive in GB/Minimalism

Before I present the different GB/Minimalist frameworks, I will take a brief look at how passive is treated in GB/Minimalism. As in HPSG there are two directions, one lexical and one syntactic (assuming that the analyses of passive in German that I mentioned in Section 2.2, are syntactic).

According to Chomsky (1981, 117-127) passive is a lexical process. When a verb gets passive morphology the subject's theta-role is absorbed, and (in most cases) one of the arguments inside the VP is not assigned Case. This forces the argument that did not get Case inside the VP to move to the subject position. In English, the participle form is considered as passive morphology. The participle *killed* in *John was killed* assigns Case but not theta role to the subject and theta role but no Case to the object: [_S [_{NP}

e] [$_{VP}$ kill NP*]].¹³ That forces the NP that gets the internal theta role, *John*, to move to subject position in order to receive Case.

The consequence of this approach is that there are five versions of the participle *overrakt* of the Norwegian ditransitive verb *overrekke* ('hand'). First, there is the active form: [$_S$ NP [$_{VP}$ overrekke NP NP]]. Second, there is a passive version where the indirect object does not receive Case (corresponding to (41b)): [$_S$ [$_{NP}$ e] [$_{VP}$ overrekke NP* NP]]. Third, there is a passive version where the direct object does not receive Case (corresponding to (41c)): [$_S$ [$_{NP}$ e] [$_{VP}$ overrekke NP NP*]]. Fourth, there is a version where both objects receive Case (corresponding to (41d)): [$_S$ [$_{NP}$ e] [$_{VP}$ overrekke NP NP]]. And fifth, there is an adjectival form which I will not go into here.

An alternative to this approach is to treat passive as an argument of the verb (see Jaeggli (1986), Baker (1988) and Åfarli (1992)). An element PASS is then assumed to take the external argument role of the verb. The external argument is, when present, the argument that is assigned nominative Case. But the PASS element does not take Case. So since the verb still has to assign nominative Case, some other element, that is not an external argument, has to take the subject position. This will be a syntactic process, and not a lexical process as in Chomsky (1981). If PASS is a verb internal argument, as suggested in Jaeggli (1986), Baker (1988) and Åfarli (1992), there will be one *active* participle and one *passive* participle. Given that the passive argument has its origin in the syntax, as suggested in Åfarli (2006), there only has to be one version of the participle *overrakt* ('handed').

2.5.2 Hale and Keyser's theory

According to Hale and Keyser (1993) and Hale and Keyser (2002), argument structure can be represented as a tree structure that is composed by certain substructures. These substructures are given in Figures 2.7-2.10.¹⁴ Examples come below.

The structure in Figure 2.7 represents a head that takes a complement, but no specifier. In English, these structures are associated with the category V (verb). The structure in Figure 2.8 shows a head that takes both a complement and a specifier. In English, these structures are usually associated with the category P (preposition). The structure in Figure 2.9 shows how a complement *Comp* licenses a specifier *Spec* on the

¹³[$_{NP}$ e] means that Case, but no theta role is assigned, and NP* means that a theta role, but no Case is assigned.

¹⁴The structure in Figure 2.7 is the abbreviated version from page 159 in Hale and Keyser (2002).

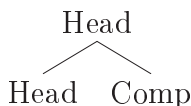


Figure 2.7: Only Comp

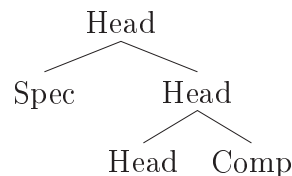


Figure 2.8: Comp and Spec

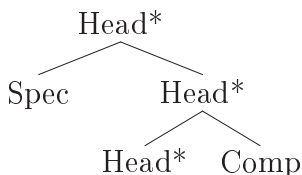


Figure 2.9: Adding Spec to a structure with Comp

Head

Figure 2.10: No Comp or Spec

head that takes *Comp* as a complement. In English, these structures usually appear when an adjective is taken as complement. The atomic structure in Figure 2.10 takes neither complements, nor specifiers. These structures come with nouns. Argument structures are constructed by the substructures in Figure 2.7-2.10.

A transitive verb like *make* in *He made a fuss*, has the structure in Figure 2.11.¹⁵ Here the structure from Figure 2.7 is employed with V as the head and DP as the complement. The DP complement *a fuss* becomes the object in the active clause.¹⁶ The subject *he* is not represented in the argument structure since it is an external argument.

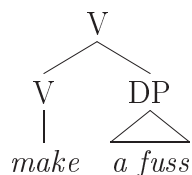


Figure 2.11: Argument structure of *make*

For intransitive verbs it is a bit different. Here, the same structure is employed as with transitive verbs, but instead of having a DP as a complement, the root (R) of the verb becomes the complement.

The argument structure for the unergative verb *bark* is illustrated in Figure 2.12. Here the structure in Figure 2.7 is working, the structure where a head takes a

¹⁵I include terminal strings in the tree representations in order to make them easier to read.

¹⁶Hale and Keyser leave it open whether a verb is specified for voice or not when it enters the syntax.

complement. The R is the complement. What will become the subject of *bark* is an external argument and is not represented in the argument structure.

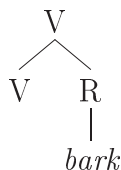


Figure 2.12: Argument structure of *bark*

R decides whether the structure in Figure 2.9 may be employed or not. This structure describes a situation where a complement *Comp* licenses a specifier on the head that takes it as a complement. The *Comp* can be said to be parasitic on the head that takes it as a complement. In this way a monadic *Comp* structure (Figure 2.7) can combine with a *Spec* structure (Figure 2.9) to form a dyadic structure (Figure 2.8). Roots of unaccusative verbs like *break* enforce such structures, while roots of unergative verbs like *bark* do not enforce them. So when the root *break* becomes the complement of a V, it licenses a specifier on the V, as illustrated in Figure 2.13. The argument structure of the intransitive version of *break* is a combination of the structure in Figure 2.7 and the structure in Figure 2.9.

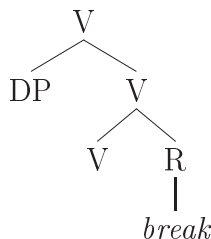


Figure 2.13: Intransitive argument structure of *break*

This difference in the root of *bark* and *break* accounts for the different syntactic environments that these two verbs can occur in. Because of the structure enforced by the root *break*, the verb now has an internal argument (in Figure 2.13 the DP), while *bark* does not. An internal argument is required for a V projection to be taken as complement of another V projection. Since *break* has an internal argument, a V projection may take it as a complement, as illustrated in Figure 2.14. This extra projection makes *break* transitive. *bark* does not have this option since it does not have an internal argument.

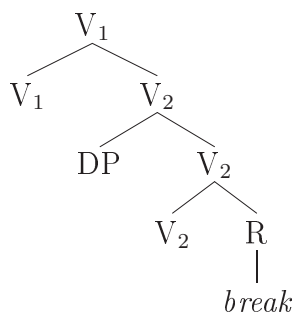


Figure 2.14: Transitive argument structure of *break*

Unaccusative verbs like *freeze* and *break* can have the resultative construction. Instead of having the R as complement as in Figure 2.15, they may instead take an adjective as complement as in Figure 2.16. The adjective has the same ability as the root of unaccusative verbs to require a specifier on the projection that takes it as a complement. That is why *the liquid* here becomes an internal argument. And since there is a structure with a specifier, the structure may get encapsulated inside another verb projection which transitivizes the verb (see Figure 2.17). The structure in Figure 2.16 will realize the internal argument as subject in English, as in (51a), while the structure in Figure 2.17 will realize the internal argument as object, as in (51b) if the sentence is active.

- (51) a. The liquid froze solid.
- b. John froze the liquid solid.

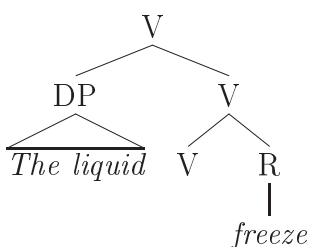


Figure 2.15: Unaccusative intransitive *freeze*

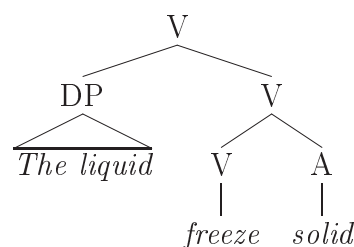
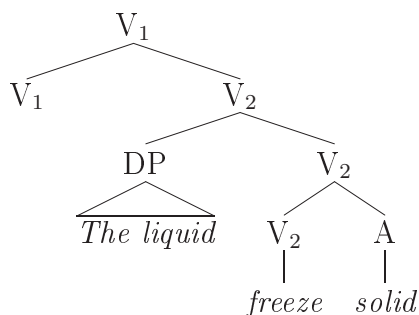
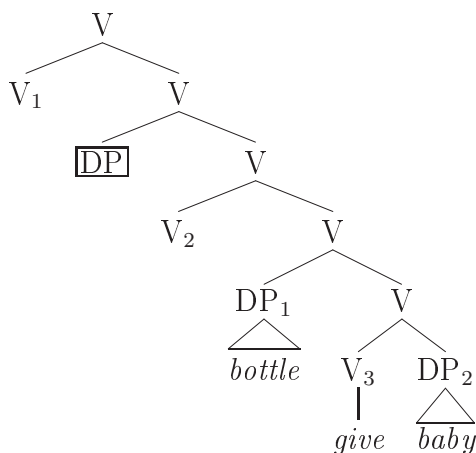


Figure 2.16: Intransitive resultative *freeze*

The argument structure of ditransitive verbs consists of three substructures, two of the kind shown in Figure 2.8 and one of the kind shown in Figure 2.7. The result is a structure with three verb projections and three internal argument positions. The verb moves to V_1 and DP_2 moves to the framed DP.

Figure 2.17: Transitive resultative *freeze*Figure 2.18: Ditransitive *give*

2.5.3 First Phase Syntax

Ramchand (2008) advocates a more flexible lexicon which does not have the lexical structures assumed by Hale and Keyser, but rather some “selectional information that constrains the way lexical items can be associated with syntactic structure” (Ramchand (2008, 3)). One is not supposed to make generalizations over argument structure in the lexicon, but rather in the syntax.

A lexical item is a bundle of phonological, encyclopedic and syntactic information. The syntactic information on the lexical item serves as the interface between the phonological/encyclopedic information and the syntax. The fact that some verbs are constrained with regard to what kind of complements they take and what kind of alternations they can enter, Ramchand sees as an argument for having this syntactic information in the lexical item.

Ramchand argues that an event can be decomposed into three subevents, namely a

process, which is the core of the event, a causation event, which initiates the process, and a result event, which comes as a result of the process. An event may consist of one or more subevents, but the process must always be present. Each of these subevents have a specifier as indicated in Figure 2.19. Here the INITIATOR is the specifier of the cause/initiation subevent (*init*). The UNDERGOER is the specifier of the process subevent (*proc*). And the RESULTEE is the specifier of the result subevent (*res*).

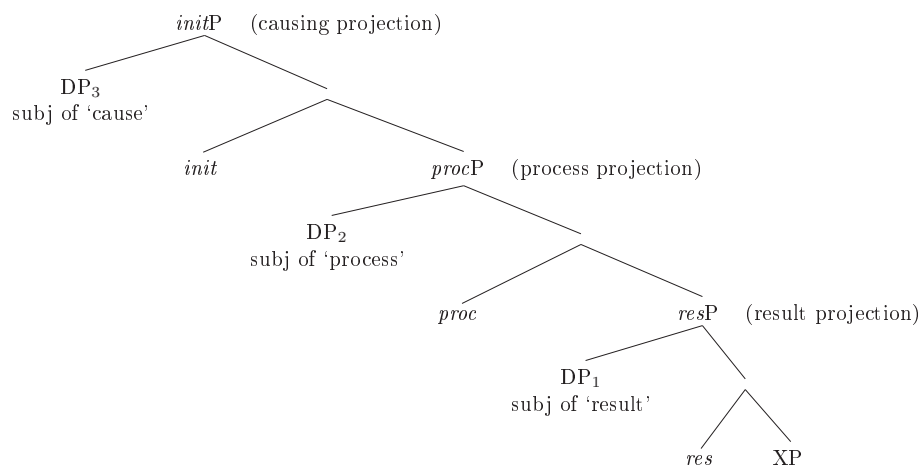
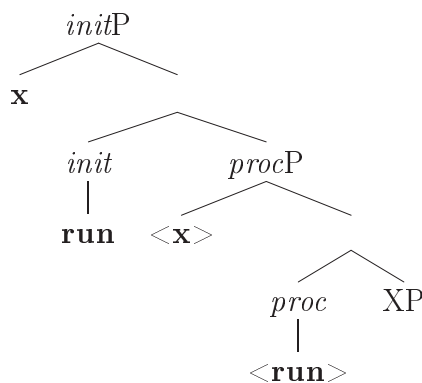


Figure 2.19: First Phase Syntax

It is possible for one referent to be associated with several roles. The intransitive *run* for example has the same referent for both the INITIATOR and the UNDERGOER role as shown in Figure 2.20.

Figure 2.20: *x run*

The syntactic information in the lexical entry for *run* is $[\text{init}_i, \text{proc}_i]$. Since the two subevents are co-indexed, there can only be one argument. A transitive verb like *kick*

does not have this co-indexation, so there are two arguments. The lexical entry of a transitive verb has the syntactic information [init, proc].

An unaccusative verb like *melt* does not have the initiator subevent, only the process subevent, so the lexical entry has the syntactic information [proc], and the syntactic structure it is associated with is given in Figure 2.21.

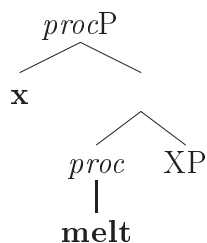


Figure 2.21: *x melted*

When verbs that do not have an initiating subevent specified in the lexical entry, are causativized (like *melt* in Figure 2.21), an invisible verb with an initiator as specifier takes the non-causative verb as complement, as in Figure 2.22.

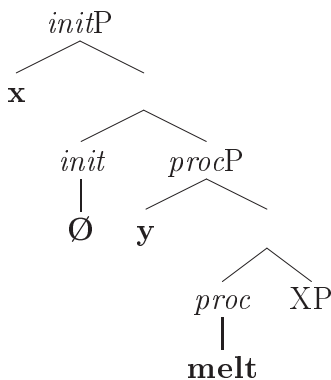


Figure 2.22: *x melted y*

The difference between causativization in this framework and Hale and Keyser's framework is that here the causativization is a syntactic process, while in Hale and Keyser's framework it is a lexical process. Since causativization is treated as a syntactic process in Ramchand's framework, passive must also be a syntactic process, since the external role of verbs like *break* is not projected from the lexicon. In Hale and Keyser's framework, however, passive can either be a syntactic or lexical process, since it is determined in the lexicon whether a verb can have an external role or not.

Something similar to causativization happens when a result is added to a verb that does not have the result subevent like *run*. In a clause like *Ariel run her shoes ragged*, the adjective *ragged* introduces the result subevent (*res*). The *res* head in this subevent is null (in English). The syntactic structure is given in Figure 2.23.

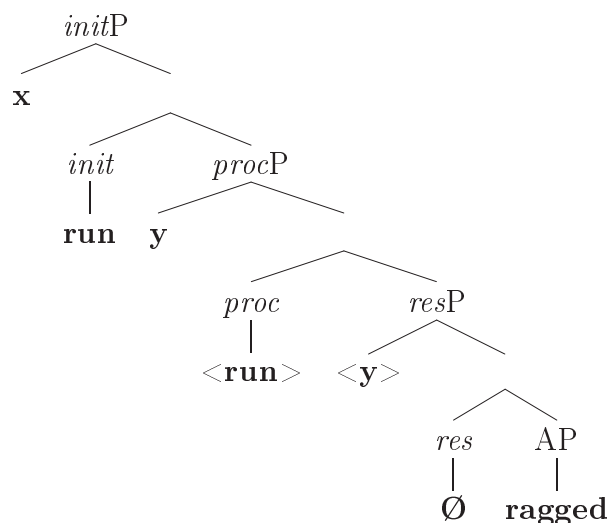


Figure 2.23: *x ran y ragged*

2.5.4 Minimalism - Borer's neo-constructionist approach

Unlike the approaches mentioned so far Borer's Exo-Skeletal approach (Borer (2005a) and Borer (2005b)) does not assume any syntactic information present in open lexical items like nouns, verbs and adjectives. They are only seen as modifiers of an event that is created by the syntax.

The ability of certain word forms to occur in a range of syntactic positions is the motivation behind the approach. She shows how for example most nouns can be transformed into verbs and how verbs may enter many different argument frames by coercion. She contrasts this flexibility with the grammatical strictness that comes with closed word class items and grammatical formatives. If you use a determiner, the category of the element the determiner is attached to is fixed to *noun*. And if you use a past tense suffix, you have a verb. Borer suggests that there are three cognitive modules involved in the use of language:

1. A conceptual system which has non-grammatical concepts that are created from

perception and conceptualization. These can be seen as small conceptual packages with a phonological index, but no grammatical content.¹⁷

2. A grammar component which consists of structures and formal properties of functional items. This component produces a grammatical structure that has an interpretation.¹⁸
3. A component Borer refers to as ‘making sense’, where the conceptual packages are matched with the interpretation you get from the grammatical structure. If the output from module 1 and 2 match, then it is grammatical, and if not, it is ungrammatical.

Since all open class lexical items come without syntactic information, the distinction between unergative and unaccusative verbs is due to different syntactic structures. In this way it is possible to account for all the uses of *drip* in (52) with only one lexical entry, simply because the lexical item *drip* comes from the lexicon with no syntactic information.

- (52) a. It drips.
 b. The roof drips.
 c. The roof drips water.
 d. John drips medicine in the glass.
 e. John drips himself medicine.
 f. John drips himself medicine in the glass.
 g. Water drips.
 h. Water drips into the bucket.

In cases where a lexical item enters a syntactic frame which does not match the concept it encodes, Borer prefers to talk about oddity rather than ungrammaticality. So if one for example replaces *drip* with *smile* in (52), the result is a set of odd rather than ungrammatical sentences as in (53).

¹⁷This component corresponds to the (ideal) Lexicon in my approach. However, in my application, I have included some grammatical content in the lexical entries in order to keep the search space at a reasonable level.

¹⁸This component corresponds to the notion of ‘strict syntax’ in my approach.

- (53) a. It smiles.
 b. The roof smiles.
 c. The roof smiles water.
 d. John smiles medicine in the glass.
 e. John smiles himself medicine.
 f. John smiles himself medicine in the glass.
 g. Water smiles.
 h. Water smiles into the bucket.

2.6 Comparison

I have shown that the approaches discussed above situate themselves differently with regard to how much information about argument structure is present on a lexical item when it enters the syntax. On one side of the scale we have LFG's Lexical Mapping Theory and HPSG. In LFG and HPSG one assumes not only that the lexicon specifies a verb's arity, but also that the lexicon contains information about resultatives,¹⁹ suppressed arguments and voice (active/passive).²⁰ Hale and Keyser's approach is more moderate in that it appears to leave the active/passive alternation and the decision about what is realized as subject, to syntactic processes, but information about causativization, resultative constructions and ditransitivity is still present in the lexicon. In Construction Grammar, phrasal constructions such as the English Ditransitive Construction and the English Resultative Construction are assumed to have meaning independent of the lexical meaning of the words, and words may be underspecified with regard to whether they enter these constructions or not. Similarly, Ramchand's approach lets the lexical items carry little syntactic information when they are entered into the syntax. Verbs that have the causative/inchoative alternation are underspecified with regard to whether they have a causative argument. Verbs that may or may not have the resultative construction are underspecified with regard to this,

¹⁹This was not made clear in Section 2.3, but Bresnan (2001, 313) mentions that a resultative predicate alters the a-structure and adds a resultative argument.

²⁰As I mentioned on page 29, there are some HPSG approaches to passive in German where the passive auxiliary determines the realization of the arguments of the past participle, and the passive lexical rule is not needed.

and verbs that may be ditransitive are also underspecified with regard to this. Still information about unergativity/unaccusativity is assumed to be present in the lexical item. Finally, Borer’s *neo-constructionist* approach claims that (open class) lexical items do not have any syntactic information present at all. This makes it possible for one lexical item not only to enter all possible argument frames as a verb, but it can also end up as a noun or an adjective.

In Table 2.1, I have categorized the different frameworks with regard to whether *passive* is a lexical process, whether *other alternations* such as arity alternations, the causative/inchoative alternation, the dative alternation, the spray/load alternation and the resultative construction are treated as lexical processes, and whether *variable behavior* is specified in the lexicon.²¹

	Passive	Other alternations	Variable behavior
HPSG		+	+
LFG (LMT)	+	+	+
Hale and Keyser		+	+
CG (Goldberg)		–	
Ramchand	–	–	+
Borer	–	–	–

Table 2.1: Overview of alternations that are represented lexically in different frameworks

2.7 Some methodological considerations

In the approach to argument structure taken in this thesis, I assume that the argument structure can be reduced to grammatical relations. One motivation for doing this is to avoid the use of multiple lexical entries or lexical rules in order to account for the different argument frames that a verb can enter. If one makes use of multiple lexical entries or lexical rules, one may end up with a large set of words with the same form, each having their specialized argument frame that fits with the syntactic environment. The fact that there is no morphological evidence to support the hypothesis that the argument frame of a word is fixed in the lexicon (one form can occur in several frames), suggests that argument structure is *not* fixed in the lexicon. Or at least, that a certain degree of freedom is allowed with regard to the choice of argument structure.

²¹I have left the field open when it may be unclear whether the phenomenon is lexically specified.

I also assume that syntactic structures are binary, and that there are no constraints on trees of depth greater than one. This makes it possible to account for phenomena such as scrambling, modifier attachments, and complex predicates with a small set of rules. If syntactic structures are assumed to be flat (or if constraints on trees are allowed to reach further than one node down), the rules may become too tightly connected to particular word orders, and the amount of rules may become unmanageable.

2.7.1 Remarks to HPSG

A methodological problem with the non-inflecting lexical rules assumed in HPSG is that there always has to be one lexical entry (with a particular argument frame) that other lexical entries can be derived from. Since there is no inflection, there is no way to tell which lexical entry that was first. In case of the dative alternation, for example, one has to decide whether *give* in *John gave a flower to Mary* is derived from *give* in *John gave Mary a flower* or the other way around. To choose one instead of the other seems to be just a stipulation.²²

2.7.2 Remarks to LFG/LMT

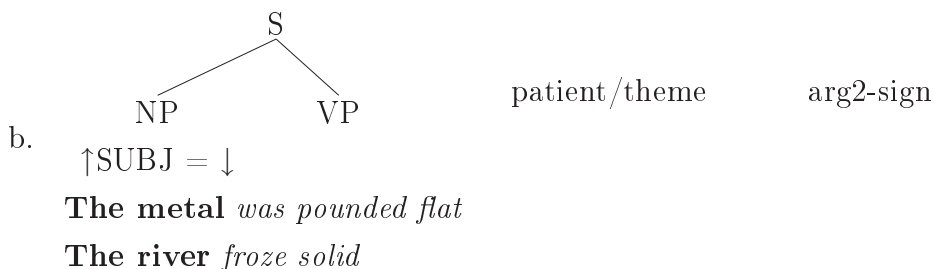
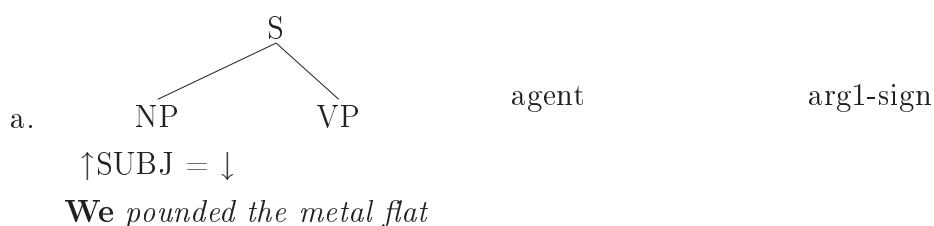
The Lexical Mapping Theory is suggested as an alternative to lexical rules in LFG. With the Lexical Mapping Theory, valence alternations can be accounted for by employing *relation changes* (see Bresnan (2001, 25-40)). The suppressions of argument roles in a-structures are examples of such relation changes (see (33) and (34)). However, it is a bit difficult to see the difference between using lexical rules and the employment of relation changes. A lexical rule may alter the conditions a lexeme puts on its syntactic environment. A relation change can apply to a relation and thereby alter the conditions that a lexeme with this relation finally puts on its syntactic environment. Although in a lexical rule, the conditions on the syntactic environment are changed more directly, the result is the same. One ends up with two versions of a word either way. For example, in the case of passive in English, there is a distinction between a past participle and a passive participle. But the form (for example *cooked*) is exactly the same. So even

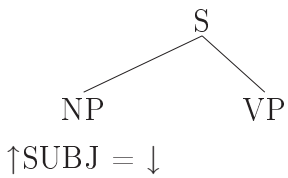
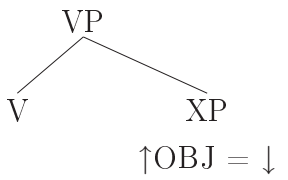
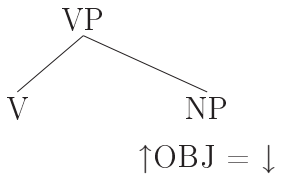
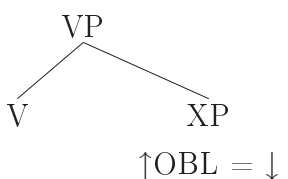
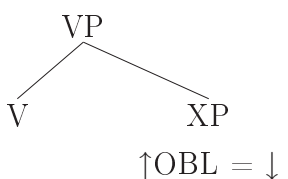
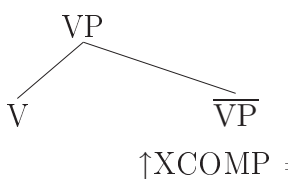
²²Since the version with two NP objects is semantically more restricted (see Pinker (1989, 48)), one could argue that it is derived from the semantically less restricted version with one NP object. Another argument in favour of a lexical rule where the version with two NPs is the output, is that Bantu languages employ an applicative affix to derive a verb that takes two NP complements from a verb that takes an NP and a PP complement (see Baker (1988)).

though there is no proof for it (such as different morphological marking), the LFG/LMT theory predicts two distinct words, exactly as a framework that employs lexical rules would do.

The realizations of Grammatical Relations in LFG have certain similarities to the subconstructions in the present work (see Section 1.2, Chapter 3 and Section 6.1). This is shown in (54) where LFG ‘sub-trees’ mapping arguments to Grammatical Relations are compared to subconstructions. Each comparison is illustrated with one or more examples where the mapped argument is printed in boldface. In (54a), the mapping of an agentive argument to the Subject GR in LFG corresponds to an arg1-sign. In (54b), the mapping of a patient/theme argument to the Subject GR in LFG corresponds to an arg2-sign. In (54c), the mapping of a beneficiary argument to the Subject GR in LFG corresponds to an arg3-sign. In (54d), the mapping of a patient/theme argument to the Object GR in LFG corresponds to an arg2-sign. In (54e), the mapping of a beneficiary argument to the Object GR in LFG corresponds to an arg3-sign. In (54f), the mapping of a goal argument to the Oblique GR in LFG corresponds to an arg4-sign. In (54g), the mapping of an instrument argument to the Oblique GR in LFG corresponds to an arg5-sign. And in (54h), the mapping of a propositional argument to the XCOMP GR in LFG corresponds to an arg2-sign.

(54) **Realization of LFG Gram-** **LFG** **Argument** **Corresponding**
matical Relation **Role** **subconstruction**



- c.  beneficiary arg3-sign
 ↑SUBJ = ↓
The children were cooked supper
- d.  patient/theme arg2-sign
 ↑OBJ = ↓
We pounded the metal
- e.  beneficiary arg3-sign
 ↑OBJ = ↓
To bananer blir overrakt Kari
 'Kari is handed two bananas'
- f.  goal arg4-sign
 ↑OBL = ↓
The glass was put on the table
- g.  instrument arg5-sign
 ↑OBL = ↓
The ball was hit with a stick
- h.  proposition arg2-sign
 ↑XCOMP = ↓
He seems to agree

The argument roles referred to in (54) are LFG argument roles. The table shows what subconstructions certain grammatical realizations of argument roles in LFG correspond to.

2.7.3 Remarks to Construction Grammar

Given that the constructions in Construction Grammar are realized as phrasal constructions as presented in Goldberg (1995), the theory faces certain challenges, pointed out in Müller (2006). In order to account for resultatives in connection with permutations of SUBJ, OBJ and OBL, verb initial/verb final position, passive, middle, modal infinitives and free datives in German, 218 constructions are required. This leaves out the treatment of adjuncts and complex predicates, which could make the number of constructions needed infinite. Müller's criticism presupposes that the phrasal constructions are either flat or that they involve constraints on trees of depth greater than one. For the German subordinate clauses in (55), he assigns the structures in (56):

(55) a. daß so grün selbst Jan die Tür nicht streicht
 that that green even Jan the door not paints
 ‘that not even Jan would paint the door that green’

b. daß so grün die Tür selbst Jan nicht streicht
 that that green the door even Jan not paints

c. daß Jan so grün selbst die Tür nicht streicht
 that Jan that green even the door not paints

d. daß eine solche Tür so grün niemand streicht
 that a such door that green nobody paints
 ‘that nobody paints such a door that green’

(56) a. [OBL SUBJ OBJ V]

b. [OBL OBJ SUBJ V]

c. [SUBJ OBL OBJ V]

d. [OBJ OBL SUBJ V]

In the approach taken in this thesis, where constructions are decomposed into subconstructions (see Section 1.2 and Chapter 3), this criticism does not hold. With decomposed phrasal constructions, it is possible to maintain binary structures and at the same time have a phrasal approach to constructions. The examples in (55) can be given the (binary) structures in (57), where COMPL is the complementizer. Analyses of the German clauses are given in Appendix B.2, p. 309.

- (57) a. [[[[COMPL ARG4] ARG1] ARG2] V]
 b. [[[[COMPL ARG4] ARG2] ARG1] V]
 c. [[[[COMPL ARG1] ARG4] ARG2] V]
 d. [[[[COMPL ARG2] ARG4] ARG1] V]

The left-branching tree structures assumed here were briefly introduced in Section 1.4, and will be discussed in more detail in Chapter 5. In addition to allowing for phrasal (sub-)constructions, binary left-branching tree structures open for incremental parsing of sentences (see Section 5.2). This could be seen as a development of CG, which would make the theory less hit by Müller’s criticism, but as mentioned in Section 2.4, the analysis with phrasal subconstructions presupposes abstract constructions that can be decomposed, and not unanalyzable constructions, as assumed in CG.

2.7.4 Remarks to Hale and Keyser’s theory

As in HPSG and LFG, also in Hale and Keyser’s theory the argument structure is assumed to be fixed in the lexicon before it enters the syntax.²³ This forces one to assume several lexical entries for one form in the case of verb alternations. In the causative/inchoative alternation, for example, the two alternates are associated with different argument structures (see Figure 2.13 (p. 42) and 2.14 (p. 43)). That implies that *break* in *The glass broke* and *break* in *John broke the glass* are two different lexemes (which still share the same root). If a clause has a secondary predicate, this is represented in the argument structure as well. So *hammer* in *He hammered the metal* and *hammer* in *He hammered the metal flat* are also different lexemes. In some verb alternations it seems like the alternates are not even able to have the same root. (The root is determining whether a verb is unergative or unaccusative.) As mentioned in Section 2.1.1, the verb *drip* in (18), repeated here as (58), is ambiguous. It may mean that something is the source of the dripping, as in (58a), or it means that something is the theme of the dripping, as in (58b).

- (58) a. The roof drips.

²³Hale and Keyser make it clear that these structures are projected from the lexicon: “We use the term *argument structure* to refer to the syntactic configuration projected by a lexical item. It is the system of structural relations holding between heads (nuclei) and their arguments within the syntactic structures projected by nuclear items. While a lexical entry is more than this, of course, argument structure in the sense intended here is nothing other than this.” (Hale and Keyser, 2002, 1).

b. Water drips (from the roof).

On the first interpretation, the verb can be characterized as an unergative and has the structure in Figure 2.12 (p. 42). On the other interpretation, the verb is an unaccusative and has the structure in Figure 2.13 (p. 42). The reason why these structures are different is that the root of an unaccusative verb requires a specifier, while the root of an unergative does not. So unless there is a way to underspecify the requirements of the root, there must be two different roots for *drip*. This is unfortunate if the root is supposed to be the lowest common denominator for all argument frames. That would exclude any generalizations over the unergative *drip* and the unaccusative *drip*, for example that some dripping is taking place.

The verb *drip* can enter a large number of argument frames, as illustrated in (52). If one wants to account for all these frames in the framework of Hale and Keyser, one is forced to assume two roots and seven different argument frames. It seems to be only the examples in (52a) and (52b) that can share lexical entry for the verb *drip* since the subjects in these examples are external arguments.

2.7.5 Remarks to First Phase Syntax

Unlike the frameworks mentioned so far in this section, Ramchand manages to separate argument structure from lexical items in such a way that one lexeme can be associated with a range of argument frames. As I have shown, the verb *kick* can be both transitive, ditransitive and enter a resultative construction without having to posit several lexical entries, as the case was in Hale and Keyser's framework. Also the causative/inchoative alternation is accounted for without using more than one lexical entry per verb.

Apart from the fact that the syntactic structures are right-branching, this framework is quite similar to the approach taken in this thesis. Phrasal subconstructions allow a lexeme to be associated with several argument frames, and the syntactic structures are binary and they are not center-embedded.

One problem with this approach is that it presupposes the use of unpronounced words. This seems to be implied by the right-branching trees (as is typical for the GB/Minimalist analyses). First, there is an unpronounced cause-verb that accounts for causativization of verbs that do not have the causative sub-relation in the lexical entry (see Figure 2.22, p. 46). Second, there is an unpronounced resultative item that adds a resultative sub-relation when adjectives serve as resultatives (see Figure 2.23, p. 47).

Although Ramchand manages to account for most of the alternations I have considered so far in Section 2.5.3 without employing several lexical entries or lexical processes of any kind, I am not quite sure how verbs such as the *drip* in (58), can be accounted for with only one entry. Analyzed as an unergative, *drip* will have the lexical entry $[v_i, V_i]$, while analyzed as an unaccusative it must have the lexical entry $[V]$. So it seems like some verbs still need two lexical entries in this approach.

2.7.6 Remarks to Borer's neo-constructionalist approach

The main problem with Borer's neo-constructionalist approach may be that it leaves it up to the 'making sense' component to determine whether a sentence is well-formed or not. There does not seem to be a clear understanding of how this component works, and the chance of overgeneration seems to be bigger than in the other frameworks discussed. At least in parsing, a lot of structures will be build before they eventually are rejected in 'making sense'. Since the approach does not commit itself to a particular syntactic theory, it is not quite clear whether it needs to posit unexpressed words in the way that Ramchand does.

Goldberg (2006, 210–211) mentions three problems with neo-constructionalism. First, the meanings of the noun *dog* and the verb *dog* in English are different. According to the neo-constructionalist approach, the lexical meaning of these words should be the same. Second, the theory fails to account for idiosyncrasy with regard to obligatory arguments of certain words like the verbs *eat*, *dine*, and *devour*. *Dine* is intransitive, *eat* may be either intransitive or transitive, and *devour* is obligatorily transitive. Third, the assumption that the external argument is an agent fails to account for transitive examples where the subject is not an agent, like sentences with the verbs *undergo*, *receive*, *fill*, *frighten*, *cost*, and *weigh*.

2.8 Summary

I have presented six approaches to argument structure, HPSG, LFG/LMT, Hale and Keyser, Construction Grammar, First Phase Syntax, and Borer's neo-constructionalism. Three of the frameworks are lexicalist (HPSG, LFG, and Hale and Keyser) and three of them are constructionalist (Construction Grammar, First Phase Syntax, and neo-constructionalism).

I have pointed out problems with each of the approaches. HPSG, LFG/LMT, and Hale and Keyser create several lexical items for the same phonological form. For each alternation a verb has, there is a particular lexical item. This procedure is problematic when there is no morphological evidence for more than one lexical item. Construction Grammar assumes flat syntactic structures, which may result in an unmanageable amount of rules. Ramchand's First Phase Syntax approach has to assume several unpronounced words in order to be able to have only one lexical item per phonological form, and Borer's approach may have a problem with overgeneration.

The frameworks presented in this chapter differ in regard to how to approach argument structure. They span from strict lexicalist approaches to argument structure to pure non-lexicalist approaches to argument structure. They also differ with regard to whether argument structure can be composed by substructures or whether it is a primitive. The approach I am going to present in the remaining chapters is a non-lexicalist (or constructionalist) approach to argument structure where argument structure can be composed by substructures. In order to achieve that, I employ what I refer to as *phrasal subconstructions*. As in the constructionalist approaches, I will assume that the argument structure of a verb is determined by the grammatical configuration in which the verb occurs, rather than by a lexically specified frame. That is, the construction is a *phrasal* construction. And, as in frameworks such as First Phase Syntax and Hale and Keyser's theory, I assume that argument structure can be decomposed into substructures. That is, a construction can be decomposed into *subconstructions*. In principle, open lexical entries will be assumed to have no syntactic information, as proposed in Borer's neo-constructionalist approach, but of practical reasons, I will introduce a mechanism that allows me to constrain a verb to occur in the argument frames one would expect it to occur in. In the next chapter, I will discuss how information about possible argument frames can be represented on verb lexemes.

Chapter 3

A subconstructional approach to Argument Structure

In this Chapter I will present an alternative constructional approach where phrasal constructions are decomposed into five subconstructions. (I have already introduced the subconstructions in Section 1.2.) I will present a number of alternations and constructions discussed by Levin (1993), and common in the linguistics literature. Some of the alternations and constructions, like the resultative construction, the understood object alternation, the dative alternation, and the spray/load alternation I have already mentioned in the previous chapter. For each alternation or construction that I go through, I will show how the alternate argument frames can be accounted for by means of the five subconstructions. The approach will make it possible to have binary structures and at the same time have a phrasal approach to constructions, without positing constraints on trees of depth greater than one.

3.1 Some syntactic tests

The five subconstructions are general in nature, and will be reflected in each language according to the grammar of the language. In Norwegian, they are reflected in the following phenomena: *passive*, *presentation*, *topicalization*, and *resultative* constructions. On the Norwegian data, I employ a passive test and a presentational test from Áfarli and Eide (2003, 226-239) to determine whether an argument is internal or external. I use a topicalization test to determine whether a PP is an argument or an

adjunct, and I use a resultative test to determine whether an argument is a delimiter.

Passive is used to determine whether a verb may be in a clause with an arg1-sign (or put in GB terms, whether a verb may have an external argument). If a verb can be the main verb in a passive clause, it is compatible with the arg1-sign. In an active version of the clause, the subject is realized by the arg1-sign. But the fact that a verb may be the main verb in a passive clause, does not imply that the verb always appears in clauses with the arg1-sign,¹ and passive is also no prerequisite for having an external argument.²

Presentation is used to determine whether a verb may be in a clause with an arg2-sign (or put in GB terms, whether a verb may have an “direct object internal argument”). In Norwegian, presentational constructions may be used in cases where the subject is not realized by the arg1-sign, as in unaccusative clauses like (59a) and passive clauses like (59b). If a verb can be the main verb in a clause with a presentational construction, and the clause has a direct object (the presented NP), then this object is realized by the arg2-sign. But the test does not say that the verb *always* has an object realized by the arg2-sign.³

- (59) a. Det kommer en mann.
 it comes a man
 ‘There is a man coming.’
- b. Det blir sendt en pakke.
 it becomes sent a packet
 ‘A packet is being sent.’

Åfarli and Eide (2003, 235) show that the tests may reveal that an intransitive verb can have either an external argument or an internal argument, i.e. that the verb can be both unergative and unaccusative. Example (60a) with the verb *arbeide* (‘work’), has a passive version (60b), and according to this, it is unergative. But it also has a presentational version as shown in (60c), which means that it is unaccusative. This verb is therefore considered to be a variable behavior verb.

¹Variable behavior verbs may passivize when they are transitive, but when they are unaccusative they do not passivize.

²Source subjects are assumed to be external arguments even though sentences with source subjects do not passivize (see Sections 3.2.1 and 3.3.3).

³It may not be expressed, or the verb may be a variable behavior verb with an unaccusative and an unergative variant.

- (60) a. En mann arbeider på åkeren.
 A man works on field-DEF
 ‘A man is working on the field.’
- b. Det blir arbeidet på åkeren.
 It becomes worked on field-DEF
 ‘The field is being worked on.’
- c. Det arbeider en mann på åkeren.
 it works a man on field-DEF
 ‘A man is working on the field.’

Topicalization is used to determine whether a PP is an argument of the verb or an adjunct. If the complement of the PP can be topicalized and leave the preposition behind, as in (62a), the PP is treated as an argument. If this is not possible, as in (62b) the PP is treated as an adjunct.⁴

- (62) a. Marit snakker Jon med.
 Marit talks Jon with
 ‘Marit Jon talks to.’
- b. * Mandag kommer Jon på.
 Monday comes Jon on

Resultative is used to determine whether an argument is a delimiter. (A delimiter is a resultative or a goal phrase.) I use this test in Section 3.2.5 and 3.3.7 where I deal with alternations like the *spray/load* alternation. The idea is that a clause can have only one delimiter. That means that if a resultative (which is a delimiter) can be added, then the variant without the resultative does not have a delimiter. And if a resultative cannot be added, then this is an indication that the clause already has a delimiter.

⁴It may be objected to the topicalization test that it is possible to extract from spatial adjuncts, as shown in lxi. This kind of spatial expressions will be considered as arguments, rather than adjuncts, in this approach. As argued in Sections 1.2 and 2.4, the arguments assigned to a verb by the syntax do not need to be predictable from the meaning of the verb.

- (lxi) Den broen ble det funnet et lik under.
 that bridge was it found a body under
 ‘A body was found under that bridge.’

3.2 Five subconstructions

In this section I revisit the five subconstructions introduced in Section 1.2, *arg1-sign*, *arg2-sign*, *arg3-sign*, *arg4-sign*, and *arg5-sign*. I use the syntactic tests from the previous section to determine what subconstructions a clause has.

3.2.1 ARG1

The *arg1-sign* is the realization of what in GB is referred to as the “external argument”. In Ramchand’s terms the external argument corresponds to the INITIATOR. This argument can be syntactically realized as subject, as in (64a), or as a passive auxiliary, as in (64b). The *arg1-sign* cannot be the realization of the direct object or the indirect object. When the *arg1-sign* is the realization of the subject, the subject is an NP. The argument realized by the subconstruction can semantically be interpreted as an agent, as in (64), or a source, as in (65).⁵

(64) a. John smashed the ball.

b. The ball was smashed.

(65) The roof drips water.

Most clauses with an *arg1-sign* realized as subject, like (66a), do not have a presentational variant in Norwegian, as illustrated in (66b). However, as I have already shown in (60) with the variable behavior verb *arbeide* (‘work’), this is not always the case.

(66) a. En spiller smasheet.

a player smashed

‘A player smashed.’

⁵The reason why I treat source arguments as realizations of *arg1-signs*, is that they cannot function as objects in presentational constructions as (lxiii) illustrates. In order to have a presentational construction, the source has to function as a prepositional object as in (lxiiib). See also discussion in Section 3.3.3.

(lxiii) a. * Det utstråler en sol varme.
it radiates a sun heat

b. Det utstråler varme fra sola.
It radiates heat from sun-DEF

‘Heat radiates from the sun.’

- b. * Det smashet en spiller.
 it smashed a player

3.2.2 ARG2

The arg2-sign corresponds to the realization of what I have referred to as a “direct object internal argument”. In Hale and Keyser’s framework it will be the “internal argument”. In Ramchand’s terminology it corresponds to the UNDERGOER in a transitive clause. The argument may be realized as direct object as in (64a) and (67a), but if the clause does not have an arg1-sign or if the sentence is passive, then the argument realized by the arg2-sign may function as subject, as in (64b) and (67b). In a clause where it is possible to realize the arg2-sign as a subject, as in (67b), the clause also has a presentational variant in Norwegian. Then the expletive *det* (‘it’) functions as the subject. This is illustrated in (67c). Formally the arg2-sign can be an NP (like *ice cream* in (68a)), an infinitival clause (like *to compete* in (68b)) or a subordinate clause (like *that it rains* in (68c)). Usually the subconstruction can be interpreted semantically as a theme, patient or undergoer, but as showed in (60c), it may also be interpreted as an agent.

- (67) a. En spiller smashet en ball.
 a player smashed a ball
 ‘A player smashed a ball.’
- b. En ball ble smashet.
 a ball became smashed
 ‘A ball was smashed.’
- c. Det ble smashet en ball.
 it became smashed a ball
 ‘A ball was smashed.’
- (68) a. The man likes ice cream.
 b. The man likes to compete.
 c. The man says that it rains.

If the verb is ergative, the argument realized by the arg2-sign can either function as subject, as in (69a), or as direct object in a presentational construction, as in (69b).

- (69) a. En avis brenner.
 a newspaper burns
 ‘A newspaper is burning.’
- b. Det brenner en avis.
 it burns a newspaper
 ‘A newspaper is burning.’

A verb that can undergo so called “adjective conversion” (see Bresnan (2001): 30-37), links the argument of the arg2-sign to what it modifies. This is illustrated in (71a). If the verb is not likely to have an arg2-sign, like *shout* in (71b), the past participle cannot be an adjective. There are some verbs that cannot undergo the adjective conversion, like *come* in (71c). Bresnan (ibid.) points out that there is a semantic restriction on past participles that convert to adjectives, namely that the verb has to have an inherent result state. This accounts for the ungrammaticality of (71c), where *come* does not have an inherent result state. (71d), on the other hand, is grammatical since *arrive* has an inherent result state.⁶

- (71) a. a punctured ball
 b. * a shouted man
 c. * a come man
 d. an arrived message

⁶Bresnan mentions some intransitive unergative verbs (*well-prepared*, *confessed*, *recanted*, *(un)declared*, *practiced*, and *unbuilt*) which can undergo the adjective conversion (*a well-prepared teacher*). In Norwegian, only one of these verbs *forberede* (‘prepare’) can undergo the adjective conversion. But this verb is not intransitive in Norwegian. It requires an object, like the reflexive pronoun in (lxxa). Otherwise the sentence is ungrammatical, as illustrated in (lxxb).

- (lxx) a. Læreren forberedte seg godt.
 teacher-DEF prepared REFL well
 ‘The teacher prepared well.’
- b. * Læreren forberedte godt.
 teacher-DEF prepared well

Also *konsentrere* (‘concentrate’) behaves in the same way. As a verb in Norwegian it requires an object, and it may undergo adjectival conversion, while the English *concentrate* may be intransitive. My suggestion is that these verbs are associated with an arg2-sign, the realization of which must be expressed syntactically in Norwegian. Maybe it is not required to express this arg2-sign as an object (or as a subject in passive) in English.

3.2.3 ARG3

The arg3-sign is usually the realization of the indirect object, like *John* in (72a). If the clause is passive, then the arg3-sign can be the realization of the subject (see (72b)). Formally the argument of the arg3-sign is an NP. The subconstruction can semantically be interpreted as a receiver or benefactive/malefactive.

- (72) a. Mary gave John a book.
 b. John was given a book.

The verb *gi* ('give') in (75) has three subconstructions, an arg1-sign, an arg2-sign, and an arg3-sign. (75a) is active and (75b)-(75e) are passive. (75b) shows that the arg3-sign can be the realization of a subject. (75c) illustrates that an expletive can be subject in passive. The contrast in grammaticality between (75c) and (75d) illustrate that the direct object must be indefinite when the clause has a presentational construction. The presentational construction does not have any such influence on the arg3-sign.⁷

- (75) a. Jon gir Kari en bok.
 Jon gives Kari a book
 'Jon gives Kari a book.'
- b. Kari blir gitt en bok.
 Kari becomes given a book
 'Kari is given a book.'
- c. Det blir gitt Kari en bok.
 It becomes given Kari a book
 'Kari is given a book.'

⁷The restriction on the direct object in presentational constructions is not quite as straightforward as I present it here. There are examples of definite direct objects in presentational constructions, as (lxxiii) and (lxxiv) illustrate. See Faarlund *et al.* (1997, 836) for more examples.

- (lxxiii) Det fins ikke matbiten i huset.
 it is not food-piece in house-DEF
 'There is not any food in the house.'
- (lxxiv) Det står navnet ditt på døra.
 it is-written name-DEF yours on door-DEF
 'Your name is written on the door.'

- d. * Det blir gitt Kari boka.
 It becomes given a girl book-DEF
- e. En bok blir gitt Kari.
 A book becomes given Kari
 ‘Kari is given a book.’

3.2.4 ARG4

The arg4-signs are realizations of delimiters like resultative and goal phrases. The syntactic argument of an arg4-sign is a PP or adverb, as in (76a), or an adjective, as in (76b). It can also be an NP, as pointed out in Rothstein (1985, 81-95) (see 76c).

- (76) a. John put the glass on the table.
 b. John kicked the ball flat.
 c. He sprayed his new car a brilliant shade of green.

Semantically the arg4-sign expands the event, by telling the location or state where the arg2-sign argument is ending up.

In (76b) *flat* is ambiguous between the resultative reading and the adjunct reading. Either John kicked the ball into a flat state (resultative reading), or he kicked the ball while it was flat (adjunct reading). In (77) the function of *flat* is disambiguated when a goal phrase *out of the room* is added. Then only the adjunct reading of *flat* is accessible. Since goal phrases are delimiters and resultatives are delimiters, this suggests that there can only be one delimiter/arg4-sign in a clause.

- (77) John kicked the ball flat out of the room.

Winkler (1997, 375) makes similar observations with regard to resultative secondary predications (RSPs). Simpson (2006, 154–155) points out that change of location attributes and change of state attributes cannot apply at the same time, and if “a verb attributes a change of location of some argument, it is not possible to have a secondary predicate attributing a change of state involving that same argument.” While Simpson proposes that the incompatibility of change of location and change of state on the same verb is as a semantic constraint, I claim that it is also a syntactic constraint, since both are interpretations of the arg4-sign, and a clause only can have one arg4-sign.

3.2.5 ARG5

Arg5-signs are realizations of PP complements that are *not* delimiters. In (78a), *about flowers* is realized by the arg5-sign. In (78b), *to Sandy* is assumed to be realized by the arg4-sign. (78c) shows that the arg4-sign (the realization of *to Sandy*) can come together with the arg5-sign (the realization of *about flowers*). (78d) has an arg1-sign (the realization of *Mary*), an arg2-sign (the realization of *John*), an arg4-sign (a resultative) (the realization of *to sleep*), and an arg5-sign (the realization of *about flowers*).

- (78) a. Mary talks about flowers.
 b. Mary talks to Sandy.
 c. Mary talks to Sandy about flowers.
 d. Mary talks John to sleep about flowers.

The spray/load alternation exemplifies the distinction between the arg4-sign and the arg5-sign. In (79a) *on the wall* is assumed to be realized by the arg4-sign, while in (79b) *with paint* is realized by the arg5-sign.

- (79) a. Jack sprayed paint on the wall.
 b. Jack sprayed the wall with paint.

The test I use to determine whether an argument is realized by an arg4-sign or an arg5-sign is to add a possible delimiter like *wet* in (80). When *wet* must be interpreted as an adjunct, this means that the clause already has a delimiter, as in (80a) (*on the wall*). (80a) cannot mean that the paint ended up wet and ended up on the wall. It must mean that the paint was wet as it ended up on the wall. So *on the wall* must be realized by an arg4-sign. In (80b) on the other hand, *wet* is interpreted as a resultative, and since there can only be one arg4-sign, *with paint* cannot be realized by an arg4-sign, and therefore is realized by an arg5-sign.

- (80) a. Jack sprayed the paint wet on the wall.
 b. Jack sprayed the wall wet with paint.

3.3 Alternations in Levin’s “English Verb Classes and Alternations”

In this section I will go through most of the verb alternations described for English in Chapter 1 and 2 in Levin (1993)⁸ and describe them as alternations of argument frames or constructions using the five subconstructions *arg1*, *arg2*, *arg3*, *arg4*, and *arg5*. A construction with only an *arg1*-sign, as in the sentence *John smiles* will be called an *arg1-construction*. A construction with an *arg1*-sign and an *arg2*-sign as in *John admires Mary* will be called an *arg12-construction*. And so on.

3.3.1 The Causative/Inchoative Alternation (2-12 Alternation)

In the causative/inchoative alternation there is one unaccusative intransitive variant ((81a)) and one transitive variant ((81b)). The object of the transitive variant (*the glass*) is the subject of the intransitive variant.

- (81) a. The glass broke.
 b. John broke the glass.

The unaccusative intransitive variant has an *arg2*-construction, which means that there is only an *arg2*-sign (the realisation of *the glass*). The transitive variant has an *arg12*-construction, which means that there is one *arg1*-sign (the realization of *John*) and one *arg2*-sign (the realization of *the glass*).

3.3.2 The Induced Action Alternation (14/24-124 Alternation)

In the induced action alternation, the subject of a clause, in this case (82a), can be the object of another clause, as illustrated in (82b). The latter clause has an agent that causes the event expressed by the first clause.

⁸Some of the alternations are variants of a general kind of alternation. There are for example eight unexpressed object alternations, and they are all alternations of the same kind in my approach. Some alternations, like the middle alternation, are not applicable for Norwegian. And some alternations, like the body-part possessor ascension alternation, are not relevant for the present study. (The body-part possessor ascension alternation *Margaret cut Bill’s arm* vs. *Margaret cut Bill on the arm* is in my approach simply an alternation between a transitive (*arg12-construction*) and a transitive with a PP argument (*arg124-construction*.) I will therefore *not* consider the following alternations in Levin (1993): Middle alternations, alternations that have to do with reflexives and reciprocals, the last seven of the eight unexpressed object alternations, search alternations, body-part possessor ascension alternation, the five possessor-attribute factoring alternations and the *as* alternation.

- (82) a. The horse jumped over the fence.
 b. Sylvia jumped the horse over the fence.

Norwegian has the same alternation, as (83a) and (83b) demonstrate. I assume that the intransitive variants ((82a) and (83a)) either has an arg14-construction or an arg24-construction. The reason why I allow two constructions in these examples is that (83a) passes both the passive test (see (83c)) and the presentation test (see (83d)), and can be considered to be a variable-behavior verb. The transitive examples (82b) and (83b) are assumed to have arg124-constructions.

- (83) a. Bilen kjører inn i garasjen.
 car-DEF drives into garage-DEF
 'The car drives into the garage.'
- b. Marit kjører bilen inn i garasjen.
 Marit drives car-DEF into garage-DEF
 'Marit drives the car into the garage.'
- c. Det kjører en bil inn i garasjen.
 it drives a car into garage-DEF
 'A car drives into the garage.'
- d. Det kjøres inn i garasjen.
 it drive-PASS into garage-DEF
 'Something drives into the garage.'

3.3.3 The Substance/Source Alternation (25-12 Alternation)

In the substance/source alternation, the subject of a clause with a complement PP, as in (84a), can be the object of another clause, as in (84b). The subject of this other clause is what corresponds to the object of the preposition in the first clause.

- (84) a. Water drips from the roof.
 b. The roof drips water.

This alternation is illustrated for Norwegian in (85). The intransitive variants (84a) above and (85a) below are assumed to have arg25-constructions. The PP with the

source *from the roof* is realized by an arg5-sign. An argument for not assuming that the PP is realized by an arg4-sign, is that it is possible to add a goal phrase, which will be realized by an arg4-sign (see (86)) (see also the discussion in Section 3.2.5).

- (85) a. Vann drypper fra taket.
 Water drips from roof-DEF
 ‘Water drips from the roof.’
- b. Taket drypper vann.
 roof-DEF drips water
 ‘The roof drips water.’
- c. Det drypper vann fra taket.
 it drips water from roof-DEF
 ‘Water drips from the roof.’
- d. * Det drypper et tak vann.
 it drips a roof water

(86) Water drips from the roof into the bucket.

The reason for assuming that the subject is realized by an arg2-sign in the intransitive variants (84a) and (85a), is that the Norwegian example (85a) has a presentational variant (see (85c)). The transitive variants (84b) and (85b) are assumed to have arg12-constructions. One reason for this is that example (85b) does not have a presentational variant (see (85d)). That means that the subject of (85b) *taket* (‘the roof’) cannot be realized by an arg2-sign, but should be realized by an arg1-sign.⁹

3.3.4 Intransitive/Transitive Alternations (1-12 Alternations)

In the intransitive/transitive alternations there is one intransitive variant (see (88a), (89a), and (90a)) and one transitive variant ((88b), (89b), and (90b)). The intransitive

⁹A problem with letting a source be realized by an arg1-sign, is that it does not pass the passive test. In (lxxxvii) it is not possible to get the source reading for *the roof*. It must be interpreted as an agent.

(lxxxvii) # Water is dripped (by the roof).

variant is unergative and has an arg1-construction. The transitive variant has an arg12-construction. The subject of the intransitive variant and the subject of the transitive variant have the same relation to the verb (arg1-relation).

(88) *Unexpressed object*

- a. John eats.
- b. John eats a cake.

(89) *Cognate object*

- a. Sarah smiled.
- b. Sarah smiled a charming smile.

(90) *Reaction Object*

- a. She mumbled.
- b. She mumbled her adoration.

Norwegian also has the intransitive/transitive alternation. This is illustrated with *spise* ('eat') in (91a) and (91b). Both variants can be passivized, as illustrated in (91c) and (91d). And the intransitive (active) variant cannot have the presentational construction, as (91e) shows. The positive passive tests and the negative presentation test indicate that the subject is realized by an arg1-sign in both the transitive and the intransitive variant.¹⁰

- (91) a. Jon spiser.
 Jon eats
 'Jon eats.'
- b. Jon spiser en kake.
 Jon eats a cake
 'Jon eats a cake.'
- c. Det spises.
 it eat-PASS
 'Eating is going on.'

¹⁰If the adjunct *her inne* ('in here') is added to (91e), the sentence is grammatical. This indicates that *spise* is a variable behavior verb like *arbeide* ('work') (see (60)) when an adjunct is added.

- d. Kaker spises.
cakes eat-PASS
'Cakes are eaten'
- e. * Det spiser en mann.
it eats a man

3.3.5 Conative and Preposition Drop Alternations (12-14 Alternations)

The 12-14 alternation is an alternation between a transitive variant ((92a) and (93a)) and an intransitive variant with a PP argument ((92b) and (93b)) where the object in the transitive variant corresponds to the object of the preposition of the intransitive variant. The transitive variants have arg12-constructions and the intransitive variants have arg14-constructions.

(92) *Conative Alternation*

- a. John cut the meat.
b. John cut in the meat.

(93) *Preposition Drop Alternations*

- a. Martha climbed the mountain.
b. Martha climbed up the mountain.

3.3.6 Dative and Benefactive Alternations (123-124 Alternations)

The 123-124 alternation is an alternation between a ditransitive variant (see (94a) and (95a)) and a transitive variant with a PP (see (94b) and (95b)). The indirect object of the ditransitive variant corresponds to the object of the PP in the transitive variant. The ditransitive variants have arg123-constructions and the transitive variants have 124-constructions.

(94) *Dative Alternation*

- a. John gave Mary the book.

- b. John gave the book to Mary.

(95) *Benefactive Alternation*

- a. Martha carved the baby a toy.
- b. Martha carved a toy for the baby.

3.3.7 Locative and similar alternations (124-125 Alternations)

In the alternations I present in this section, I argue that there is an alternation between an arg1₂₄ construction and an arg1₂₅ construction. In the *a* examples below there is an arg1-sign, an arg2-sign, and an arg4-sign. In the *b* examples there is an arg1-sign, an arg2-sign, and an arg5-sign. See Section 3.2.5 for the motivation behind this distinction.

(96) *Locative Alternation*

- a. Jack sprayed paint on the wall.
- b. Jack sprayed the wall with paint.

(97) *Creation and Transformation*

- a. Martha carved the piece of wood into a toy.
- b. Martha carved a toy out of the piece of wood.

(98) *With/Against Alternation*

- a. Brian hit the stick against the fence.
- b. Brian hit the fence with the stick.

(99) *Through/With Alternation*

- a. Alison pierced the needle through the cloth.
- b. Alison pierced the cloth with a needle.

(100) *Blame Alternation*

- a. Mira blamed the accident on Terry.
- b. Mira blamed Terry for the accident.

The next two alternations are a bit different. The objects in the *b* examples seems to have a possession relation to the prepositional object that is not present in the earlier examples in this section. This could be an indication that the objects in the *b* examples below are realized by arg3-signs rather than arg2-signs. An other indication is that it is hard to have a resultative in the *b* examples below, while this can be done in most of the *b* examples above. It is also impossible to add an indirect object in the *b* examples below, which could be an indication that they already have an arg3-sign. On the other hand, these differences from the alternations above may also result from differences in lexical meaning.

(101) *Fulfilling*

- a. The judge presented a prize to the winner.
- b. The judge presented the winner with a prize.

(102) *Image Impression Alternation*

- a. The jeweler inscribed the name on the ring.
- b. The jeweler inscribed the ring with the name.

3.3.8 Delimiter Alternations (arg4 alternations)

The alternations in this section are alternations between a variant without a delimiter (arg4-sign) and a variant with a delimiter. If the arg4-sign in a clause realizes a resultative, the clause must also have an arg2-sign.

(103) is an alternation between an unergative intransitive (arg1-construction), illustrated by (103a), and a transitive with a resultative (arg124-construction), illustrated by (103b). Since the subject is an arg1 argument, an arg2-sign must be added in order to have the adjective resultative. Simply having an arg14 construction is not possible here, as (103c) illustrates.

(103) *Resultative Construction (1-124 Alternation)*

- a. The guests drank.
- b. The guests drank the teapot dry.
- c. * The guests drank dry.

(104) shows an alternation with a transitive variant (arg12-construction), illustrated by (104a), and a transitive variant with a resultative (arg124-construction), illustrated by (104b).

(104) *Resultative Construction, Transitive (12-124 Alternation)*

- a. Pauline hammered the metal.
- b. Pauline hammered the metal flat.

(105) is an alternation between an intransitive unaccusative (arg2-construction), illustrated by (105a) and an intransitive unaccusative with a resultative (arg24-construction), illustrated by (105b).

(105) *Resultative Construction, Intransitive (2-24 Alternation)*

- a. The river froze.
- b. The river froze solid.

The alternation in (106) is assumed to have an unergative intransitive variant (arg1-construction), illustrated in (106a), and an intransitive unaccusative variant with a goal phrase (arg24-construction), illustrated in (106b).

(106) *Directional phrases with non-directed Motion verbs (1-24 Alternation)*

- a. The car rumbled.
- b. The car rumbled into the driveway.

The reason why the variant without the delimiter is assumed to be unergative is that it does not have a presentational variant in Norwegian (see (107a) and (107b)). The variant with the delimiter on the other hand can have the presentational variant (see (107c) and (107d)).

(107) a. En bil skramlet.

- a car rumbled
- 'A car rumbled.'

- b. * Det skramlet en bil.
- it rumbled a car

- c. En bil skramlet inn oppkjørselen.
 a car rumbled in driveway-DEF
 ‘A car rumbled into the driveway.’
- d. Det skramlet en bil inn oppkjørselen.
 it rumbled a car in driveway-DEF
 ‘A car rumbled into the driveway.’

3.3.9 Other Alternations

The alternation in (108) is an alternation between an arg12-construction and an arg23-construction.

(108) *12-23 Alternation*

- a. We awaited their report.
 b. Their report awaited us.

As the Norwegian data in (109) show, the arg12-construction (109a) has a passive variant (109b), which predicts that the subject is realized by an arg1-sign. The arg23-construction (109c) has a presentational variant (109d), which predicts that the subject is realized by an arg2-sign.¹¹

- (109) a. Vi ventet en overraskelse.
 We awaited a surprise
 ‘We awaited a surprise.’
- b. Det ble ventet en overraskelse.
 it was awaited a surprise
 ‘A surprise was awaited.’
- c. En overraskelse ventet oss.
 a surprise awaited us
 ‘a surprise awaited us.’
- d. Det ventet oss en overraskelse.
 it awaited us a surprise
 ‘A surprise awaited us.’

¹¹Hellan (1991) has a discussion of similar data.

Clauses like (110) with no other argument than an expletive subject have an arg0-construction. A clause with an arg0-construction does not have any subconstructions.

(110) It drips.

But *drip* can also have an arg1234-construction as illustrated in (111).

(111) John drips himself water into the eyes.

I sum up this section by listing all possible constructions including the subconstructions arg1-sign, arg2-sign, arg3-sign, arg4-sign, and arg5-sign (or none of them).

Argument frame	Example
arg1	John smiles.
arg14	John talked to Mary.
arg15	John sprayed with paint.
arg145	John sprayed onto the wall with paint.
arg12	John admires Mary.
arg124	John washed the car clean.
arg125	John sprayed the wall with paint.
arg1245	John sprayed the wall wet with paint.
arg123	John gave Mary an ice cream.
arg1234	John dripped himself two drops of water into the eyes.
arg12345	John dripped himself two drops of water into the eyes with a drop counter.
arg2	The glass broke.
arg24	The river froze solid.
arg23	A surprise awaited him.
arg0	It rains.
arg4	It drips into the bucket.

Figure 3.1: Possible constructions

A verb like *drip* can (more or less successfully) have all these constructions except from the arg23-construction, as illustrated in Figure 3.2. One aim of the grammar I am going to present in the next sections, is to account for verbs like *drip* with only one lexical entry and no lexical rules.

Argument construction	Example
arg1	The roof drips.
arg14	The doctor drips into the eyes.
arg15	The doctor drips with water.
arg145	The doctor drips into the eyes with water.
arg12	The roof drips water.
arg124	The roof drips water into the bucket.
arg125	The doctor dripped the eyes with water.
arg1245	The doctor dripped into the eyes with water.
arg123	John dripped himself two drops of water.
arg1234	John dripped himself two drops of water into the eyes.
arg12345	John dripped himself two drops of water into the eyes with a drop counter.
arg2	Water dripped.
arg24	Water dripped into the bucket.
arg0	It drips.
arg4	It drips into the bucket.

Figure 3.2: Possible constructions with *drip*

3.4 Basic Relation Representations (BRRs) and semantic representations

Before I start discussing how the subconstructions are accounted for in Norsyg, I give a brief presentation of the Basic Relation Representations (BRRs) that are returned by the Norsyg grammar. A BRR is similar to a semantic representation produced by an HPSG grammar. The main difference between a BRR and a semantic representation is that a BRR consists of the Grammatical Relations of an utterance *plus* the words of the utterance, represented as analyzable predicates. A BRR is assumed to have meaning only when interpreted in conjunction with the meaning of the words. The semantic representations in HPSG on the other hand represent the meaning of an utterance directly.

There are different formalisms for representing semantic information in implemented HPSG grammars. MRS (Minimal Recursion Semantics) (Copestake *et al.*, 2005) and LRS (Lexical Resource Semantics) (Penn and Richter, 2004) are the two most well-known. I present MRS, which is used in the Grammar Matrix, in Section 3.4.1, and then, in Section 3.4.2, I compare it to a flatter semantic representation (RMRS) (Copestake,

2003), which is the basis for the BRRs returned by the Norsyg grammar.

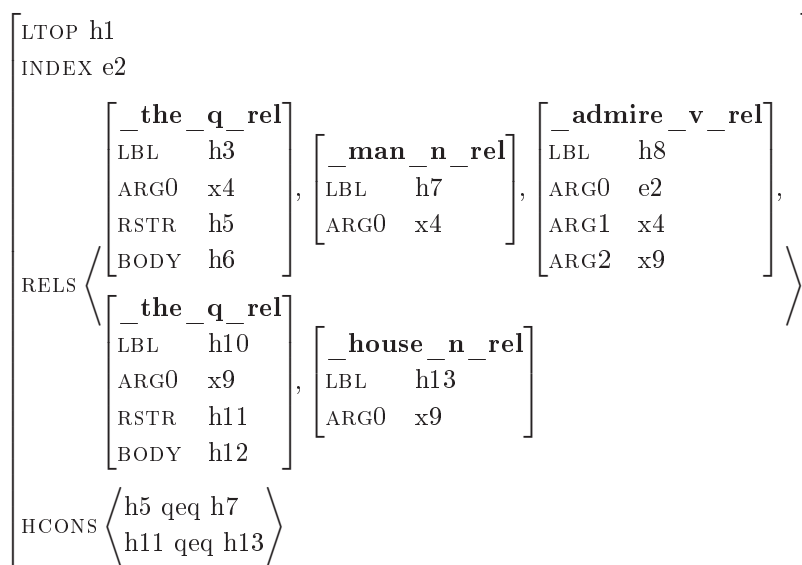
3.4.1 MRS

Grammars that are implemented with the LKB system (the English Resource Grammar (Copestake and Flickinger, 2000), the German Grammar (Crysmann, 2003), the Japanese HPSG grammar (Siegel and Bender, 2002), the Korean Resource Grammar (Kim and Yang, 2003), the Greek HPSG grammar (Kordoni and Neu, 2003), and NorSource (Hellan and Haugereid, 2004)) usually use the MRS formalism to represent semantic information. When a string of words is parsed with an LKB grammar, the semantic information contributed by the lexemes, words, and phrases is gathered in the type *mrs* (the value of `CONT`). An MRS representation relates to the value of the type *mrs* of the top node of a derivation, and displays the semantic information gathered in *mrs*. An MRS representation has the attributes `LTOP`, `INDEX`, `RELS`, and `HCONS`. The `LTOP` feature has as value the top handle. A handle is a tag assigned to a relation, and the relation with the top handle has the widest scope. The `INDEX` feature has as value the index of the string that is parsed. In a clause this will be an event index, which is the index of the main verb. The `RELS` feature has as value a list with all the relations contributed by the constituents of the sign, and the `HCONS` feature has as value a list with handle constraints, which represent pairs of handles that are equal (but not unified). The handle constraints carry information about which relations outscope which (see Copestake *et al.* (2005)). The MRS of *the man admires the house* is given in Figure 3.3.¹²

In Figure 3.3 the verb relation *_admire_v_rel* has two argument features, `ARG1` and `ARG2`.¹³ The `ARG1` is linked to the `ARG0` of the first quantifier relation and the *_man_n_rel* relation (x7). The `ARG2` is linked to the second quantifier relation and the *_house_n_rel* relation (x10). So the first argument of the admire-relation is the man and the second argument of the admire-relation is the house. Each of the quantifiers

¹²It is a convention to begin relation names that are language specific like *_admire_v_rel* with an underscore, while relation names that are not language specific, like *proper_q_rel* (proper noun quantifier) do not begin with an underscore. Another convention is to let the category be reflected in the relation name, so a noun has the infix *_n_*, a verb has the infix *_v_*, and a quantifier has the infix *_q_*. The November 2007 version of the ERG does not show the illocutionary force of a sentence as a separate relation, but rather as a value of the feature `SF` on the index of the verb.

¹³The semantic arguments `ARG1`, `ARG2`, `ARG3`, and `ARG4` used in MRS representations should not be confused with the valence features `ARG1`, `ARG2`, `ARG3`, and `ARG4` that I will use in the rest of this thesis. (The four valence features are introduced in Section 4.3.)

Figure 3.3: MRS of *the man admires the house* from the ERG

equals its RESTR value with the handle of the noun relation that they share index (i.e. ARG0 value) with via the two handle constraints. This means that the noun relations are in the restriction of the quantifiers. The scope (BODY) of the quantifiers is left underspecified. The LKB system provides a scope resolving mechanism that can produce all possible scope resolved readings of the MRS. The MRS in Figure 3.3 gives two scope resolved readings, as illustrated in (112). In (112a) the quantifier of *the man* outscopes the quantifier of *the house*, and in (112b) the quantifier of *the house* outscopes the quantifier of *the man*. These kinds of scope resolved readings are supposed to account for ambiguities of well-known linguistic examples such as *Every dog chased a cat*.

- (112) a. $\text{the}(x4, \text{man}(x4), \text{the}(x9, \text{house}(x9), \text{admire}(e2, x4, x9)))$
 b. $\text{the}(x9, \text{house}(x9), \text{the}(x4, \text{man}(x4), \text{admire}(e2, x4, x9)))$

3.4.2 BRR/RMRS

The grammar implementation platform that Norsyg is implemented with (the LKB system) is designed for producing MRS representations. The BRRs returned by Norsyg (which is an LKB grammar) deviate from standard MRS representations in two respects (in addition to the fact that BRRs are not *real* semantic representations). First, the BRRs do not have scope features like RESTR and BODY, and handle constraints are left

out. And second, the relations that have more than one argument position (the ARG0 position) are decomposed, so that an *arg12*-relation like the *admire* relation in Figure 3.3 becomes three relations as illustrated in Figure 3.4. The decomposition of relations into a Parsons style notation (Parsons, 1990) is taken from Copestake (2003, 9) which uses decomposed semantics in RMRS (Robust Minimal Recursion Semantics). RMRS is a style of semantic representation designed for shallow parsers where for example the arity of a predicate is not specified.¹⁴

$$\left[\begin{array}{l} \textit{arg12-relation} \\ \text{PRED } _ \textit{admire_v_rel} \\ \text{LBL } \textit{handle} \\ \text{ARG0 } \textit{event} \\ \text{ARG1 } \textit{individual} \\ \text{ARG2 } \textit{individual} \end{array} \right] \Rightarrow \left[\begin{array}{l} \textit{arg0-relation} \\ \text{PRED } _ \textit{admire_v_rel} \\ \text{LBL } \textit{h1} \\ \text{ARG0 } \textit{event} \end{array} \right], \left[\begin{array}{l} \textit{arg1-relation} \\ \text{PRED } \textit{arg1_rel} \\ \text{LBL } \textit{h1} \\ \text{ARG0 } \textit{individual} \end{array} \right], \left[\begin{array}{l} \textit{arg2-relation} \\ \text{PRED } \textit{arg2_rel} \\ \text{LBL } \textit{h1} \\ \text{ARG0 } \textit{individual} \end{array} \right]$$

Figure 3.4: Translation from one *arg12*-relation to three subrelations

The translation in Figure 3.4 shows how one *arg12*-relation can be decomposed into three subrelations. The unity of the three subrelations are accounted for by letting them share LBL value. The first subrelation has the same PRED value as the *arg12*-relation. The second subrelation has the PRED value *arg1_rel*, and the third subrelation has the PRED value *arg2_rel*. The values of the ARG0 feature of the second and the third subrelation correspond to the values to the features ARG1 and ARG2 in the *arg12-relation*. The semantic representation of *mannen beundrer huset* ('the man admires the house') is given in Figure 3.5.

The semantic representation in Figure 3.5 is intended to have the reading $def(x4) \wedge man(x4) \wedge def(x6) \wedge house(x6) \wedge admire(e3, x4, x6)$.

¹⁴The choice of a decomposed representation in Norsyng is necessitated by the treatment of for example coordinated verbs in Chapter 8, where I argue that there are several predicates (one for each verb), but only one argument frame. This can only be achieved by detaching the argument roles from the predicate. (Figure 8.8, p. 226 shows the BRR for the sentence *Marit fanger, steker og spiser fisken* ('Marit catches, fries, and eats the fish').) The choice of semantic representation is also motivated by the approach taken to 'packed' argument structure information (which I will come back to in Section 4.3.3), since in this approach, the amount of semantic argument roles is not fixed in the lexical entry. The alternative would have to be a hierarchy of semantic relations of the same complexity as the hierarchy of linking types in Figure 4.9 (p. 96).

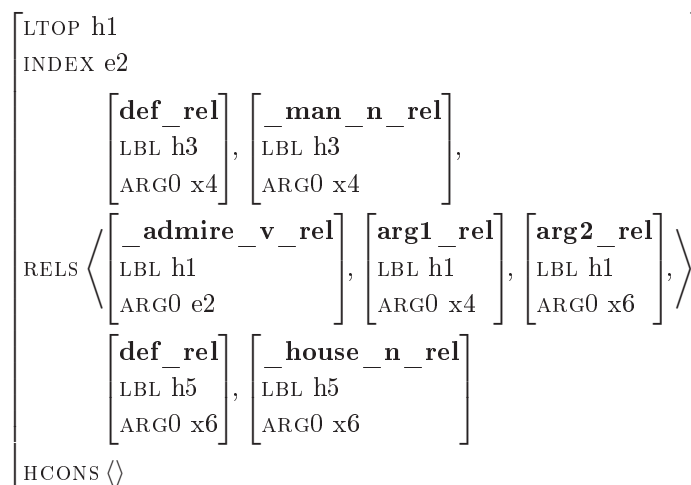


Figure 3.5: BRR of *Mannen beundrer huset* ('the man admires the house') from Norsyng

3.5 A hierarchy of subconstructions

As I showed in Section 3.2, the five subconstructions have different kinds of morpho-syntactic realizations. They can be realized as syntactic rules, inflections, and function words. In this section I will show how a type hierarchy of signs can be used to capture generalizations over these kinds of expressions. The term *sign*, which is central in the HPSG literature, is used in the Saussurean sense with the combination of form and meaning. The kinds of signs that I will discuss here are lexemes, words, suffixes, and phrases. I assume that morpho-syntactic entities expressing the different kinds of subconstructions are associated with meanings. These meanings are argued to be abstract meanings which can get more specific interpretations, as argued in Section 1.2 for individual subconstructions and in Section 2.4 for constructions. The more specific interpretations of the constructions are assumed to be a result of the combination of the abstract meaning of the construction with the meaning of the main verb and the meaning of the arguments. But, as mentioned in Section 1.2, what this more specific interpretation is, and how it is arrived at, is outside the scope of this thesis.

In order to generalize over the different means of expression, I employ a hierarchy of subconstructions. A subconstruction is a subtype of *sign* and introduces the features IN, OUT, and MEANING, as illustrated in Figure 3.6. The value of IN is the syntactic information that the sign takes as input. The value of OUT is the syntactic information that the sign outputs. The changes made from IN to OUT represent the syntactic expression of the sign, and the value of MEANING is a relation that represents the

meaning of the sign.¹⁵ The argument of the MEANING relation (\mathbb{I}) is linked to the index of the ARGUMENT of the input. The feature ARGUMENT generalizes over the different valence features, and is a pointer to the syntactic argument of the subconstruction.

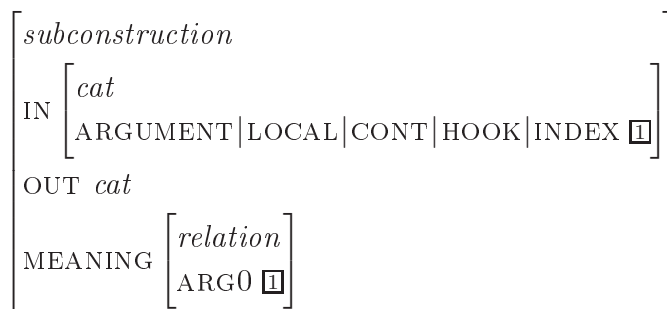


Figure 3.6: The type *subconstruction*

Four of the immediate subtypes of *subconstruction* are *arg1-sign*, *arg2-sign*, *arg3-sign*, and *arg4-sign* (see Figure 3.8).¹⁶ These signs have a formal side, namely switching a linking type from *positive* in IN to *negative* in OUT, and a meaning side, which is the relation that is the value of MEANING.

The definition of *arg1-sign* is given in Figure 3.7.¹⁷ Formally, the type *arg1-sign* switches the ARG1|LINK value from *arg1+* (‘the arg1 subconstruction is expressed’ (from a top-down perspective)) in IN to *arg1-* (‘no arg1 subconstruction is expressed so far’ (from a top-down perspective)) in OUT.¹⁸ This expresses that the arg1-sign is realized. The other valence features stay unchanged. As for meaning, the type has an *arg1-relation*. Note that ARGUMENT is unified with ARG1. This ensures that the argument

¹⁵The reason why I do not use the feature CONT to represent the meaning of the sign is that lexemes have their meaning in CONT, while rules have their meaning in C-CONT (constructional content). (I am here discussing relative to the Matrix system (see Section 4.2).) The feature MEANING is introduced in order to generalize over CONT and C-CONT. The same holds for IN and OUT. In the most cases, IN will point to the head daughter, and OUT will point to the mother, but in the case of the passive auxiliary, IN will point to an auxiliary valence feature since a lexeme does not have a daughter. (See Sections 6.1, 6.6.1, and 7.1 for more discussion.) The features IN and OUT do not imply that the signs are lexical rules.

¹⁶As I will show in Section 6.4.2, The *arg4-sign* does not inherit all constraints from *subconstruction*. Instead of unifying the *index* of the ARGUMENT with the argument of the *arg4-relation*, the *arg4-sign* unifies the LTOP value of the ARGUMENT with the argument of the *arg4-relation*.

¹⁷The introduction to the valence features ARG1, ARG2, ARG3, and ARG4 used in Figure 3.7 is given in Section 4.3.

¹⁸The root node in a parse tree has only negative linking types. As the subconstructions work (from a top-down perspective), the negative linking types are switched to positive linking types. In this way, the subconstructions that have worked will be recorded in the word that heads the clause. I will return to this linking mechanism in Sections 4.3.3 and 5.1, and in Chapter 6.

that is linked in the supertype *subconstruction* is the value of ARG1. The types *arg2-sign*, *arg3-sign*, and *arg4-sign* have definitions similar to *arg1-sign*, where the arg1s are exchanged with arg2s, arg3s, and arg4s, respectively.

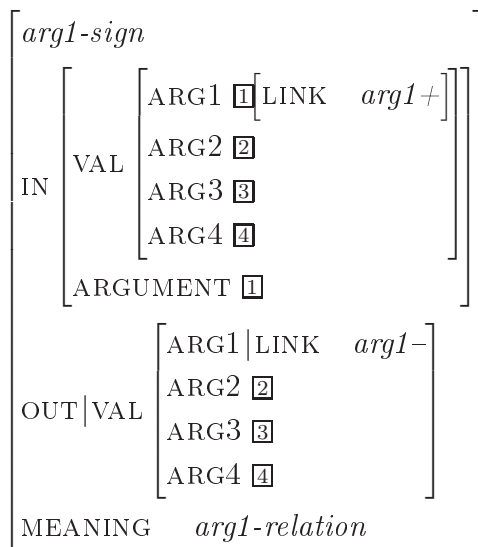
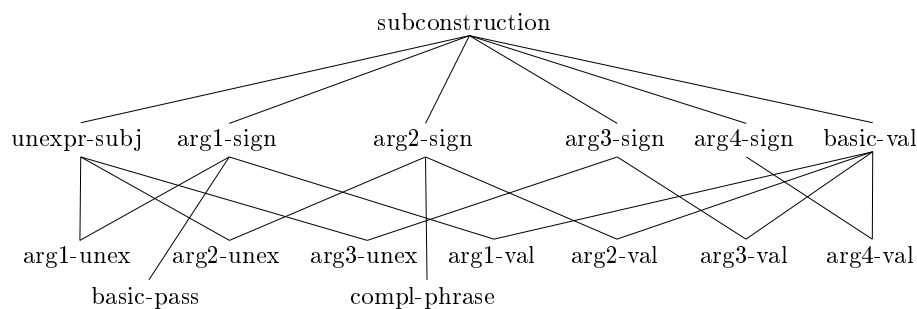


Figure 3.7: Definition of *arg1-sign*

A hierarchy of subconstruction types is shown in Figure 3.8. The type *subconstruction* has six immediate subtypes. Four of the six types are the types *arg1-sign*, *arg2-sign*, *arg3-sign*, and *arg4-sign* discussed above. The type *basic-val* is a type for subconstructions that link arguments that are expressed. These arguments are either realized in the canonical position by binary valence rules, or they are realized in a non-canonical position. They are then extracted by unary extraction valence rules. I will return to valence rules in Section 6.1. The last immediate subtype of *subconstruction* is *unexpr-subj*. This is a type for the realization of unexpressed subjects. It is a unary rule that takes the infinitival complementizer, the small clause construction or the imperative inflection as input. I will return to unexpressed subjects in Section 6.7.1. As shown in the hierarchy, *basic-val* is cross-classified with *arg1-sign*, *arg2-sign*, *arg3-sign*, and *arg4-sign*, and *unexpr-subj* is cross-classified with *arg1-sign*, *arg2-sign*, and *arg3-sign*.

In addition to the valence types and the unexpressed subject types in the subconstruction hierarchy, there is one type for passive, *basic-pass*, and one type for subordinate clause complements, *compl-phrase*. *basic-pass* inherits from *arg1-sign*, and is a supertype of the passive auxiliary as well as the passive *s*-morpheme in Norwegian. I

Figure 3.8: Type hierarchy below *subconstruction*

will return to passive in Section 7.1. *compl-phrase* is a subtype of *arg2-sign*. It is a type for rules that introduce subordinate clauses (both with and without complementizers). I will return to the treatment of subordinate clauses in Section 6.6.1.

3.6 Summary

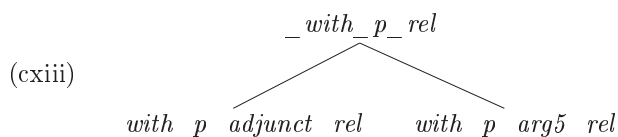
In this chapter I have introduced five basic subconstructions ARG1 – ARG5, and shown how constellations of these subconstructions constitute syntactic frames accommodating verb alternations such as the Causative/Inchoative alternation, the Induced Action alternation, the Substance/Source alternation, the Intransitive/Transitive alternations, the Conative and Preposition Drop Alternations, the Locative alternation, and other alternations. I have presented the Basic Relation Representations returned by the grammar (BRR). Finally, a type hierarchy of subconstructions has been presented where the linking between syntactic and semantic information is done. This hierarchy will be the basis of the syntactic analyses presented in Part II of the thesis. Before I get to the syntax part, I will show how valence information is represented on lexical entries in Chapter 4.

Chapter 4

Valence

In this chapter I will show how the information about possible argument frames that I discussed in the previous chapter, can be represented on verb lexemes. The central idea is that there are four valence features (ARG1, ARG2, ARG3 and ARG4), one corresponding to each of the first four subconstructions.¹ A type hierarchy of *linking types* (types that reflect whether a subconstruction is realized or not) allows for constraining verbs with regard to which constellations of subconstructions (argument frames) they can enter. A strategy for expanding the lexicon will be presented. I will also present a comparison of the Norsyg grammar and a lexicalist version of the Norsyg grammar, where verbs are given one lexical entry for each argument frame it can enter. Finally, I will compare the approach taken in Norsyg with the RASP system (a shallow

¹I do not include a separate valence rule for the arg5-role. The reason for this is that I want to keep the number of parses to a minimum. All PPs that get the arg5-role, can also be analyzed as adjuncts. If I decide to include the valence rule for the arg5-roles in addition to the modifier rules, which easily can be done, the number of parses with a PP attaching to a VP will at least double. Instead of introducing separate arg5 valence rules, I suggest that the arg5-role can be interpreted as a specialization of the prepositional predicate, as shown in (cxiii).



The only cases where the arg5-role would be possible to distinguish from an adjunct role would be in cases of topicalization of the complement of a PP, as discussed in Section 3.1, where I suggested that the possibility for topicalizing the complement of a PP can be regarded as a test for whether a PP is an argument or an adjunct. The Norsyg grammar does however not at present account for topicalization of the complement of PPs, so the interpretation of prepositional predicates as arg5-roles has not been implemented. In the present implementation, selectional restrictions about the arg5-role are specified via the ARG4 valence feature.

parser for English) and some other Norwegian computational lexicons/grammars. But first I will have a brief look at how valence is treated in HPSG.

4.1 Valence in HPSG

In Section 2.2 I showed how HPSG represents the valence information in a lexical entry of a verb. A transitive verb has the information in Figure 2.1, repeated here as Figure 4.1.

$$\left[\begin{array}{l} \text{PHON } \langle \textit{admire} \rangle \\ \text{CAT } \left[\begin{array}{l} \text{HEAD } \textit{verb} \\ \text{VAL } \left[\begin{array}{l} \text{SUBJ } \langle \text{NP}_{\boxed{1}} \rangle \\ \text{COMPS } \langle \text{NP}_{\boxed{2}} \rangle \end{array} \right] \end{array} \right] \\ \text{CONT | RESTR } \left\langle \begin{array}{l} \text{PRED } \quad \underline{\textit{admire_v_rel}} \\ \text{ARG1 } \quad \boxed{1} \\ \text{ARG2 } \quad \boxed{2} \end{array} \right\rangle \end{array} \right]$$

Figure 4.1: Lexical entry for the verb *admire*

The complements of a word are realized with the Head-Complement Rule (Pollard and Sag, 1994, 362–363) (see Figure 4.2). This rule has a head daughter, with one or more elements on the COMPS list. The elements on the COMPS list are realized as non-head daughters in the phrase.

$$\left[\begin{array}{l} \textit{phrase} \\ \text{CAT } \left[\begin{array}{l} \text{HEAD } \boxed{1} \\ \text{VAL | COMPS } \langle \rangle \end{array} \right] \end{array} \right] \Rightarrow \left[\begin{array}{l} \textit{word} \\ \text{CAT } \left[\begin{array}{l} \text{HEAD } \boxed{1} \\ \text{VAL | COMPS } \langle \boxed{2}, \dots, \boxed{n} \rangle \end{array} \right] \end{array} \right], \boxed{2} \dots \boxed{n}$$

Figure 4.2: Head-Complement Rule

This rule has as many non-head daughters as there are complements. It requires that the head daughter is a word and that the COMPS list of the mother is empty. An obvious problem with such a rule is that it does not allow adjuncts to be realized before or in between the complements as in (114), where *yesterday* comes in between the two

complement PPs.²

(114) I talked to her yesterday about John.

An alternative is to have a binary Head-Complement Rule that realizes one complement at a time, and that does not require that the head-daughter is a word (see Sag *et al.* (2003, 97)). This procedure is common in implemented grammars like the English Resource Grammar.

$$\left[\begin{array}{l} \textit{phrase} \\ \text{CAT} \left[\begin{array}{l} \text{HEAD } \boxed{1} \\ \text{VAL} \mid \text{COMPS } \boxed{3} \end{array} \right] \end{array} \right] \Rightarrow \left[\text{CAT} \left[\begin{array}{l} \text{HEAD } \boxed{1} \\ \text{VAL} \mid \text{COMPS } \langle \boxed{2} \rangle \oplus \boxed{3} \end{array} \right] \right], \boxed{2}$$

Figure 4.3: Binary Head-Complement Rule

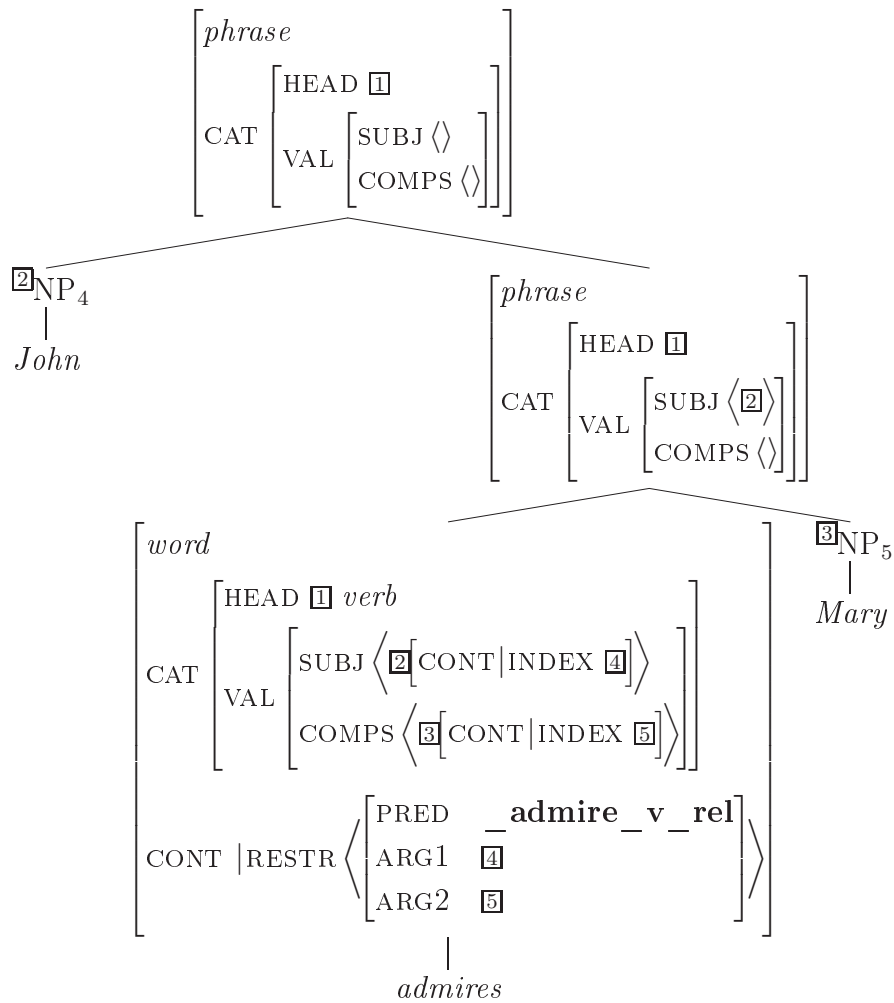
The binary Head-Complement Rule is given in Figure 4.3. The head daughter of the rule is underspecified with regard to whether it is a word or a phrase. And it only realizes the first element on the COMPS list. The rest of the list is reentered in the mother ($\boxed{3}$). If the complement list contains more than one element, the Head-Complement Rule will work repeatedly until the COMPS list is empty. By assuming such binary structures, it is easier to account for adjuncts that come in between the complements, since a Head-Modifier Rule can be allowed to work in between two Head-Complement Rules.

The subject of a clause is realized with the Head-Subject Rule (see Figure 4.4). This rule has as its head daughter a word or phrase that has an empty COMPS list and an element on the SUBJ list ($\boxed{2}$). The element on the SUBJ list is realized as the non-head daughter, and the SUBJ list of the mother is empty. An analysis of a transitive clause is given in Figure 4.5.

$$\left[\begin{array}{l} \textit{phrase} \\ \text{CAT} \left[\begin{array}{l} \text{HEAD } \boxed{1} \\ \text{VAL} \left[\begin{array}{l} \text{SUBJ } \langle \rangle \\ \text{COMPS } \langle \rangle \end{array} \right] \end{array} \right] \end{array} \right] \Rightarrow \boxed{2}, \left[\text{CAT} \left[\begin{array}{l} \text{HEAD } \boxed{1} \\ \text{VAL} \left[\begin{array}{l} \text{SUBJ } \langle \boxed{2} \rangle \\ \text{COMPS } \langle \rangle \end{array} \right] \end{array} \right] \right]$$

Figure 4.4: Head-Subject Rule

²Müller (2006) gives a good illustration of problems one may encounter with adjuncts in flat syntactic structures (see also Section 2.7.3).

Figure 4.5: HPSG analysis of *John admires Mary*

The analysis in Figure 4.5 illustrates the application of the Head-Complement Rule and the Head-Subject Rule. The word *admire* has one element on the SUBJ list ([2]) and one element on the COMPS list ([3]). The rule that combines the verb *admires* with the proper noun *Mary* is the Head-Complement Rule. It unifies the element on the COMPS list with the non-head daughter. Since the COMPS list of *admire* has only one element, the COMPS list of the mother is empty. (The rest of a list with one element is an empty list.) The rule that applies at the top of the tree is the Head-Subject Rule. It unifies the element on the SUBJ list of the head daughter with the non-head daughter. The SUBJ list of the mother is now empty.

The tree in Figure 4.5 also illustrates how linking works. The verb *admire* links the arguments of its predicate to the indices of the elements on the SUBJ and COMPS

lists (see Figure 4.1). When the Head-Complement rule and the Head-Subject Rule unify the elements on the valence lists with the syntactic arguments *John* and *Mary*, the indices of these words become the arguments of the predicate `_admire_v_rel`.

4.2 The Grammar Matrix and Norsyg

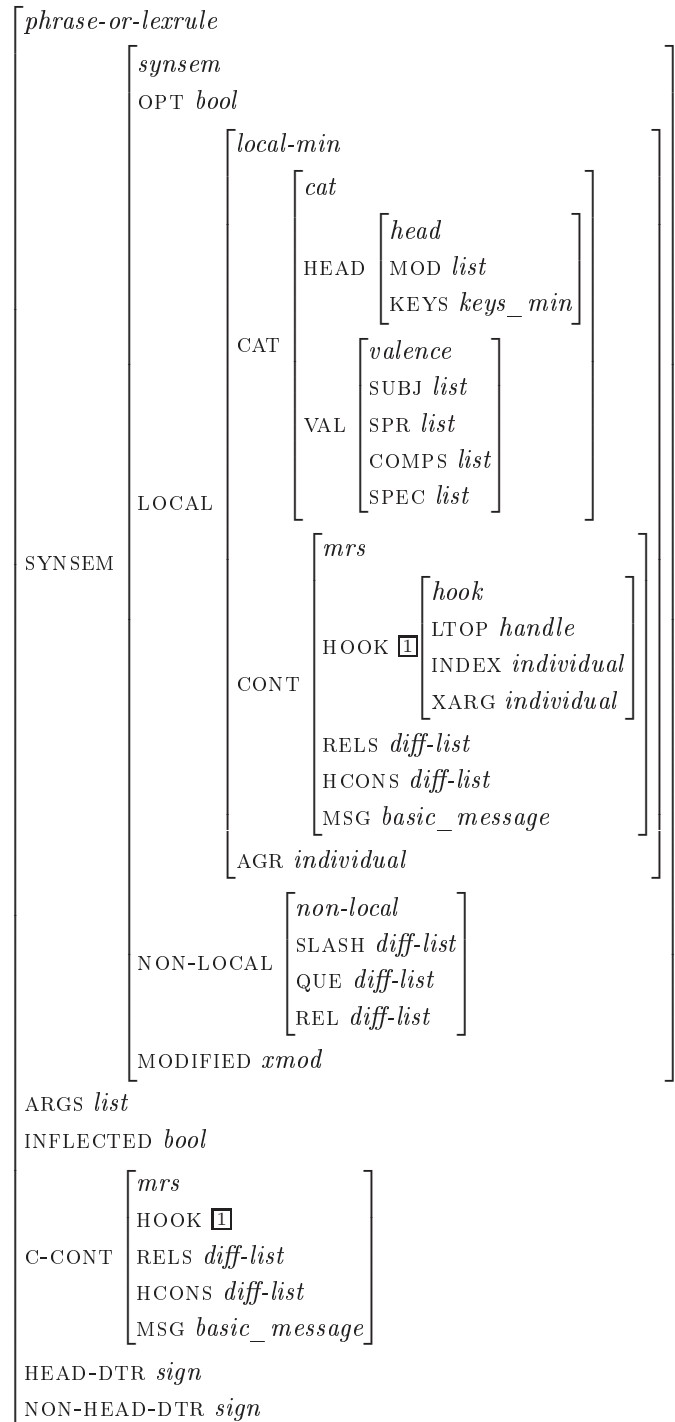
I will now present an alternative way of doing linking in HPSG which I have used in my grammar for Norwegian *Norsyg* (Norwegian syntax-based grammar). Norsyg is implemented with the LKB system (Copestake, 2002), which is a grammar development environment for implementing typed feature structure grammars.³ The grammar has adopted many of the types and part of the feature geometry from the Grammar Matrix (version 0.6) (Bender *et al.*, 2002). Some of the lexical entries stem from NorSource of January 2004 (Hellan and Haugereid, 2004).

4.2.1 The Grammar Matrix

The Grammar Matrix is a starter kit for HPSG grammar development. The 0.6 version has 203 types (664 lines of code) containing general information that can be used in grammar writing. The Grammar Matrix has general types for lexical items and phrases. The lexical types can be used to make lexical rules and add inflection. The phrasal types include types for Head-Subject Rules, Head-Complement Rules and Head-Modifier Rules. There are also types for extraction of arguments and filling in of arguments. These rules are underspecified with regard to whether they are head initial or head final. Implicit in the types of the Grammar Matrix is an architecture of features that is more or less adopted in Norsyg. A sign that is a phrase or a lexical rule in the Grammar Matrix (potentially) has the features in Figure 4.6.

The AVM in Figure 4.6 shows that the type *phrase-or-lexrule* may have six features: SYNSEM, ARGS, INFLECTED, C-CONT, HEAD-DTR and NON-HEAD-DTR. SYNSEM has

³Typed feature structures (Carpenter, 1992) have been employed in grammar development since the 80's. Flickinger (1987) employs type hierarchies in order to make generalizations over lexical entries and lexical rules. Later, also generalizations over phrases were done by means of type hierarchies (see Sag (1997)). The English Resource Grammar (ERG) (Flickinger (2000)), which has been developed since 1994, employs type hierarchies to make generalizations over lexemes, words, phrases and all other kinds of linguistic information. The ERG is developed with the LKB system (Copestake, 2002). The LKB system does not allow for relational constraints with complex antecedents, type resolution, disjunction or negation, which are often presupposed in the theoretical HPSG literature.

Figure 4.6: The type *phrase-or-lexrule*

as value the type *synsem*, which again has the following features: OPT, LOCAL, NON-LOCAL and MODIFIED. The function of the feature OPT is to say whether an element

on one of the valence lists is optional or not. The complement of the transitive verb *eat* is optional, and therefore marked as OPT +, (see Flickinger (2000, 22–24)). LOCAL has as value the type *local-min* which has the features CAT, CONT and AGR. The value of CAT, *cat*, has the syntactic features HEAD and VAL, while CONT, with the value *mrs* has the semantic information. The application of MRS semantics in the Grammar Matrix is explained in Flickinger *et al.* (2003). (MRS semantics is introduced in Section 3.4). The function of the feature AGR is agreement. NON-LOCAL keeps track of non-local dependencies. MOD tells whether a sign is modified or not (and from which direction). The feature ARGS has as value a list that contains the daughters of the sign. The feature INFLECTED tells whether a sign is inflected or not. The feature C-CONT has as value *mrs*, just as the feature CONT. The function of C-CONT (constructional content) is to let non-terminal signs enter semantic information. A sign can also have the features HEAD-DTR and NON-HEAD-DTR. In a binary head initial phrase, the value of HEAD-DTR is unified with the first sign on the ARGS list and the value of NON-HEAD-DTR is unified with the value of the second sign on the ARGS list. In a head final phrase it is the other way around. The Grammar Matrix makes certain theoretical assumptions. Some of these assumptions, like the Head Feature Principle, are adopted in Norsyg, whereas others, like the existence of valence lists like SUBJ and COMPS, are not adopted in Norsyg.

4.2.2 Norsyg - some data

425 of the original 664 lines of code in the Grammar Matrix are changed or deleted in Norsyg. Norsyg is a grammar with 1215 types, 1530 hand-built lexical entries, 144 161 lexical entries derived from Norsk Ordbank, 52 syntactic rules, 46 inflectional rules and 0 lexical rules (approximately 4200 lines of code (excluding lexicon)). In comparison, the English Resource Grammar (version Nov-07) has 3260 types, 31675 lexical entries (excluding 13620 proper nouns used in the Handon project), 175 syntactic rules, 17 inflectional rules and 26 lexical rules (26687 lines of code (excluding lexicon)). More information about Norsyg is given in Appendix A.

4.3 The linking types

In the approach taken in Norsyg, the linking happens in the syntax rather than in the lexical types. Instead of assuming that a lexical entry has detailed information about a certain syntactic frame, which is crucial in an approach that does linking in the lexicon (see Figure 4.1), I assume that a lexical entry by default has little information about its syntactic environment. The syntactic frames are not projections of the lexicon. They are rather constructions made up of what I refer to as *functional signs*, that is inflections, closed class lexical items, and syntactic rules. These signs do the linking of the arguments of the open class lexical items that enter the syntactic frames. In order to avoid overgeneration, the open class lexical items may be specified with information that restricts the number of argument frames they can enter. The fact that constraints are put on open class lexical items in order not to be compatible with all frames can be said to go against one of the assumptions in Chapter 1, namely that also what I refer to as “odd” sentences are grammatical (strict syntax). Still, of practical reasons it is necessary to put some constraints on the open class lexical items in order to make the implemented grammar work. In this section I will show the mechanism used in Norsyg for restricting the possible constructions verbs can enter.

4.3.1 Four valence features

In the implementation of a grammar that does linking by means of functional signs realizing subconstructions, I make use of four valence features (ARG1, ARG2, ARG3 and ARG4), corresponding to the four first subconstructions discussed in Chapter 3.⁴ They have *synsem* as value. The type *synsem* is given the feature LINK. The value of the LINK feature is the type *link*. In addition, there is a feature ARGFRAME with the value *link*. It is via this feature that a lexeme may put restrictions on what types of constructions it can enter. There is also a feature PART which allows a lexeme to select for particles. The type *valence* now has the definition in Figure 4.8, rather than the definition with the SUBJ and COMPS lists as presented in Figure 4.7.

⁴As for the *arg5*-signs, I do not have a separate valence feature for them in the current implementation. (See footnote 1, page 87.)

$$\left[\begin{array}{l} \textit{valence} \\ \text{SUBJ} \quad \textit{list} \\ \text{SPR} \quad \textit{list} \\ \text{COMPS} \quad \textit{list} \\ \text{SPEC} \quad \textit{list} \end{array} \right]$$
Figure 4.7: *valence* in the Grammar Matrix
$$\left[\begin{array}{l} \textit{valence} \\ \text{ARGFRAME} \quad \textit{link} \\ \text{ARG1} \quad \left[\begin{array}{l} \textit{synsem} \\ \text{LINK} \quad \textit{link} \end{array} \right] \\ \text{ARG2} \quad \left[\begin{array}{l} \textit{synsem} \\ \text{LINK} \quad \textit{link} \end{array} \right] \\ \text{ARG3} \quad \left[\begin{array}{l} \textit{synsem} \\ \text{LINK} \quad \textit{link} \end{array} \right] \\ \text{ARG4} \quad \left[\begin{array}{l} \textit{synsem} \\ \text{LINK} \quad \textit{link} \end{array} \right] \\ \text{PART} \quad \left[\begin{array}{l} \textit{synsem} \\ \text{SAT} \quad \textit{bool} \end{array} \right] \end{array} \right]$$
Figure 4.8: *valence* in Norsyg

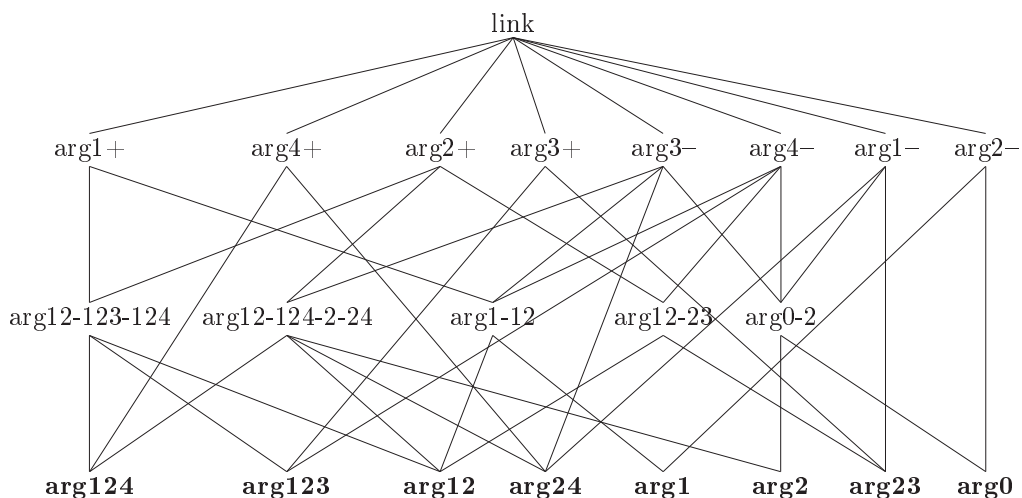
4.3.2 A hierarchy of linking types

As mentioned in Section 1.3, the type *link* has a hierarchy below it. First, there are eight types, one positive and negative type for each of the valence features in Figure 4.8 (see Figure 4.9).⁵ So there is one *arg1+*, one *arg1-*, one *arg2+*, one *arg2-* and so on.

Each of the types in the bottom of the hierarchy inherit from four of the top types. These types represent the different argument frames that I discussed in Chapter 3. For instance, the type *arg123* represents an *arg123*-construction, which is the frame type for ditransitive verbs like *handed* in *John handed Mary a book*. The type *arg124* is the type for transitive verbs with delimiters, like *hammer* in *John hammered the metal flat*. The type *arg1* is the type for unergative intransitive verbs like *smile* in *John smiled*. If we study the hierarchies above the bottom types, we see that *arg123* is a subtype of *arg1+*, *arg2+*, *arg3+*, and *arg4-*. The type *arg124* is a subtype of *arg1+*, *arg2+*, *arg3-*, and *arg4+*, and the type *arg1* is a subtype of *arg1+*, *arg2-*, *arg3-*, and *arg4-*.

⁵The hierarchy in Figure 4.9 is not complete. Several intermediate and bottom types are left out in order not to make the illustration too complex. The complete hierarchy can be found in Norsyg in the file *nor.tdl* under “Valence types”.

The *e_{part}* feature is not a part of the linking mechanism.

Figure 4.9: The *link* hierarchy

4.3.3 Packing of argument frames

The intermediate types in the hierarchy are inserted in order to allow something that can be thought of as *packing* of argument frames.⁶ These types have two or more bottom types as subtypes. So a verb that is specified in the lexicon with an intermediate link type will be compatible with all the frames that correspond to the subtypes of the intermediate link type. The verb *give* can occur with three valence frames, as illustrated in (115).⁷

- (115) a. John gave a book.
 b. John gave Mary a book.
 c. John gave a book to Mary.

In (115a) *give* has an *arg12*-frame, in (115b) an *arg123*-frame, and in (115c) an *arg124*-frame. In order to allow the verb to enter all these argument frames, it is given the ARGFRAME value *arg12-123-124* in the lexicon. *arg12-123-124* inherits from *arg1+* and *arg2+*, but is underspecified with regard to *arg3* and *arg4*. It has three subtypes, namely *arg12*, *arg123*, and *arg124*, which means that *give* can enter the relevant argument frames.

⁶The term *packing* was suggested to me by Lars Hellan.

⁷Passive and presentational variants of the examples I am using in this section are not assumed to alter the argument frame, so I do not mention them here. I come back to passive and presentation in Sections 7.1 and 7.2.

A verb like *break* can enter the frames illustrated in (116).

- (116) a. John broke the cup.
 b. John broke the cup to pieces.
 c. The cup broke.
 d. The cup broke to pieces.

(116a) has a transitive frame (arg12-construction), (116b) has a transitive + resultative frame (arg124-construction), (116c) has an unaccusative frame (arg2-construction) and (116d) has an unaccusative + resultative frame (arg24-construction). In order to allow *break* in all these frames, it is specified with the intermediate link-type *arg12-124-2-24*, which has the four subtypes *arg12*, *arg124*, *arg2* and *arg24*.

A verb like *smile* can have the argument frames in (117).

- (117) a. John smiles.
 b. John smiles a big smile.

(117a) has an unergative intransitive frame (arg1-construction) and (117b) has a transitive frame (arg12-construction). The verb *smile* is specified with the ARGFRAME value *arg1-12*, which has the two subtypes *arg1* and *arg12*.

A verb like *rain* can enter the argument frames illustrated in (118).

- (118) a. It rains.
 b. It rains money.

(118a) has an arg0-construction and (118b) has an arg2-construction, and in order to allow *rain* in both these frames, it is given the ARGFRAME value *arg0-2*. *arg0-2* has the two subtypes *arg0* (*arg0* inherits from *arg1-*, *arg2-*, *arg3-*, and *arg4-*) and *arg2*.

As I argued in Section 3.3.9, the verb *await* has two argument frames, as illustrated in (108), repeated here as (119). (119a) has an arg12-construction and (119b) has an arg23-construction. It is given the ARGFRAME value *arg12-23*.

- (119) a. We awaited their report.
 b. Their report awaited us.

The alternations I have mentioned here are just a few of the alternations I allow in Norsyng. I did not include all of them here because it would make the hierarchy in Figure 4.9 too complex for a display (128 types). Below are some of the sets of construction types that I did not mention:

- arg0-1-12-123-1234-124-14-2-24-4: *dryppe* ('drip')
- arg1-12-123-124-14: *kaste* ('throw')
- arg1-12-124-14: *snakke* ('talk')
- arg1-12-124: *male* ('paint')
- arg1-12-123: *love* ('promise')
- arg12-124: *verdsette* ('estimate/appreciate')
- arg12-2: *ankomme* ('arrive')

Some verbs only allow one frame:

- arg123: *frata* ('deprive of')
- arg1: *le* ('laugh')

4.3.4 Introductory remarks on the composition of subconstructions

Figure 4.10 gives a simplified illustration of how the information about realized subconstructions in the syntax and argument structure information specified on the main verb is represented.⁸ As the Figure shows, each valence rule switches a negative LINK value in the mother to a positive LINK value in the daughter. The top node has only negative LINK values. In this way, the LINK values in the bottom of the tree reflect what subconstructions are realized higher up in the tree. The argument structure

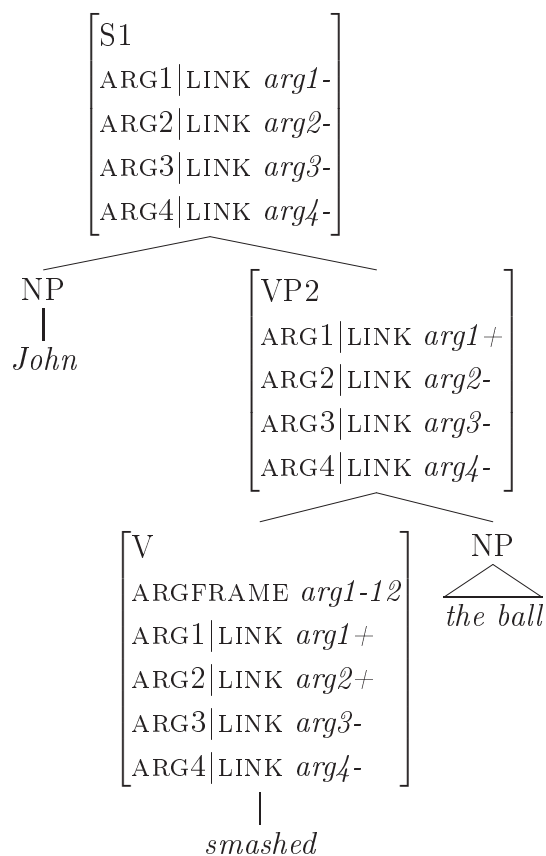


Figure 4.10: Information about realized subconstructions (BRR: D.1, p. 331)

information specified on the main verb is given as value of the feature ARGFRAME (*arg1-12*).

The type *uni-link* (see Figure 4.11) unifies the LINK values with the argument structure information specified on the main verb (the value of ARGFRAME). This type applies to constituents at the bottom of the tree where the linking information is available.⁹ In the analysis of a transitive sentence like that in Figure 4.10, the types *arg1+*, *arg2+*, *arg3-*, *arg4-*, and *arg1-12* will be unified. This gives the type *arg12* (see Figure 4.9).

⁸This tree does not reflect the fact that syntactic structures are assumed to be left-branching (see Figure 1.4, page 16). A left-branching structure implies that the initial constituent appears at the bottom-left, like *a* in Figure 1.4.

The initial constituent of a clause (or the rule that realizes the first constituent of a clause) is given a special role in the grammar, namely to unify the LINK values. A presentation of how the unification of the LINK values is done is given in Appendix A.6.1.

⁹This unification is left out in Figure 4.10 in order to show how the linking types end up at the bottom of the tree.

$$\left[\begin{array}{l} \textit{uni-link} \\ \text{ARGFRAME} \sqcup \textit{arg12} \\ \text{ARG1} \mid \text{LINK} \sqcup \\ \text{ARG2} \mid \text{LINK} \sqcup \\ \text{ARG3} \mid \text{LINK} \sqcup \\ \text{ARG4} \mid \text{LINK} \sqcup \end{array} \right]$$

Figure 4.11: Unification of LINK values and ARGFRAME value

The type *arg1-12* is also compatible with the types *arg1+*, *arg2-*, *arg3-*, *arg4-*, the unification of which gives the type *arg1*. This means that the verb *smash* can also enter a construction with only an *arg1*-sign. I would like to emphasize that the restrictions put on lexical entries via the VAL feature with regard to what argument frames they enter is *not* supposed to be seen as a part of the general theory, but rather as a way to implement restrictions, which in a practical implementation is unavoidable.

4.4 Lexical types in Norsyg

In this section I present a selection of the 100 handwritten and 288 automatically derived lexical entry types for verbs in Norsyg.¹⁰

The lexical type for a transitive verb with an optional NP object, like *eat* is presented in Figure 4.12. The feature ARGFRAME is given the value *arg1-12*, which means that the verb is compatible with both the unergative intransitive frame (*arg1*-construction) and the transitive frame (*arg12*-construction). The HEAD value of the (optional) ARG2 of the verb is specified to be *nominal*. Since I express optionality with the argument frame type, there is no need for the feature OPT on syntactic arguments. The PART|SAT value is *plus*, which means that the verb is not a particle verb.¹¹

The lexical type for a transitive verb that has an object that can either be an NP or a subordinate clause like *admire*, has the lexical type shown in Figure 4.13. The ARGFRAME value is *arg12*, which means that the two roles are obligatory, and the HEAD

¹⁰The complete list of lexical entry types for verbs can be found in the files ‘nor.tdl’ in Norsyg under “Lexical entry types for verbs” and in the file ‘oble.tdl’, which has lexical entry types automatically derived from Norsk Ordbank (see Section 4.5.1).

¹¹From now on, unless something else is stated, the value of the PART|SAT feature in the lexical entry types will be *plus*. That is, they are not particle verb types.

$$\left[\begin{array}{l} \text{arg1-12_np_le} \\ \text{SS|LOC|CAT|VAL} \left[\begin{array}{l} \text{ARGFRAME } \text{arg1-12} \\ \text{ARG2|LOCAL|CAT|HEAD } \textit{nominal} \\ \text{PART|SAT } + \end{array} \right] \end{array} \right]$$

Figure 4.12: The *arg1-12_np_le*

value of the ARG2 argument is *atcompl-noun*,¹² which means that both a subordinate clause headed by the complementizer *at* ('that') and an NP are accepted as the internal argument.

$$\left[\begin{array}{l} \text{arg12_cp-np_le} \\ \text{SS|LOC|CAT|VAL} \left[\begin{array}{l} \text{ARGFRAME } \text{arg12} \\ \text{ARG2|LOCAL|CAT|HEAD } \textit{atcompl-noun} \end{array} \right] \end{array} \right]$$

Figure 4.13: The *arg12_cp-np_le* type

The lexical type for unaccusative verbs like *fall*, which selects for an optional ARG4 PP, is given in Figure 4.14. The value of ARGFRAME is *arg2-24*. This means that the ARG4 argument is optional. The ARG2 argument is an NP (hence the HEAD value *nominal*). The ARG4 argument has two constraints, namely that the HEAD value is *prep*, and that the ARG2|LINK value is *arg2-*. This means that the verb selects for a satisfied preposition projection (a PP). (Prepositions are lexically specified as *arg2+*, and therefore they must realize their argument in order to become *arg2-*.)

The lexical type for unaccusative verbs that can be causativized, like *burn*, and for variable behavior verbs, like *arrive*, is given in Figure 4.15. The ARGFRAME value is specified to be *arg12-2*, which accounts for the alternation between unaccusative and transitive. The HEAD value of ARG2 is specified to be *nominal*, which constrains the internal argument to be an NP.

The lexical type for transitive verbs that require a reflexive object, like the Norwegian verb *ombestemme* ('reconsider') in (120), is given in Figure 4.16. The

¹²The grammar has a hierarchy of *head* types that makes it possible to restrict the head value of a sign to particular sets of categories. In general, a *head* type that has subtypes reflects which subtypes it has in the type name. So the type *atcompl-noun* in Figure 4.13 is the supertype of *atcompl* and *nominal*.

$$\left[\begin{array}{l} \text{arg2-24_np_pp_le} \\ \text{SS|LOC|CAT|VAL} \left[\begin{array}{l} \text{ARGFRAME } \text{arg2-24} \\ \text{ARG2|LOCAL|CAT|HEAD } \textit{nominal} \\ \text{ARG4|LOCAL|CAT} \left[\begin{array}{l} \text{HEAD } \textit{prep} \\ \text{VAL|ARG2|LINK } \text{arg2-} \end{array} \right] \end{array} \right] \end{array} \right]$$

Figure 4.14: The *arg2-24_pp_le* type

$$\left[\begin{array}{l} \text{arg12-2_np_le} \\ \text{SS|LOC|CAT|VAL} \left[\begin{array}{l} \text{ARGFRAME } \text{arg12-2} \\ \text{ARG2|LOCAL|CAT|HEAD } \textit{nominal} \end{array} \right] \end{array} \right]$$

Figure 4.15: The *arg12-2_np_le* type

ARGFRAME value is specified to be *arg12*, which means that both ARG1 and ARG2 are obligatory. The HEAD value *refl* on ARG2 ensures that the internal argument is the reflexive *seg*.

- (120) Jon ombestemmer seg.
 Jon reconsider REFL
 ‘John reconsidered.’

$$\left[\begin{array}{l} \text{arg12_refl_le} \\ \text{SS|LOC|CAT|VAL} \left[\begin{array}{l} \text{ARGFRAME } \text{arg12} \\ \text{ARG2|LOCAL|CAT|HEAD } \textit{refl} \end{array} \right] \end{array} \right]$$

Figure 4.16: The *arg12_refl_le* type

The lexical type for verbs like *paint*, which can be both intransitive, transitive and transitive resultative, is given in Figure 4.17. The ARGFRAME value is specified as *arg1-12-124*, which means that it can enter an unergative frame, a transitive frame, and a transitive frame with a delimiter. The HEAD value of ARG2 is specified to be *nominal*, and the HEAD value of ARG4 is specified to be *adj*. This ensures that the

internal argument is an NP, and that the delimiter is an adjective.¹³

$$\left[\begin{array}{l} \text{arg1-12-124_np_ap_le} \\ \text{SS|LOC|CAT|VAL} \left[\begin{array}{l} \text{ARGFRAME } \text{arg1-12-124} \\ \text{ARG2|LOCAL|CAT|HEAD } \textit{nominal} \\ \text{ARG4|LOCAL|CAT|HEAD } \textit{adj} \end{array} \right] \end{array} \right]$$

Figure 4.17: The *arg1-12-124_np_ap_le*

The lexical type for intransitive particle verbs like *let* in (121) is given in Figure 4.18. The ARGFRAME is *arg1*, which means that it must appear in a clause that realizes an arg1-sign. The particle will be unified with the value of the PART feature. Each lexical entry of this type will select the particle(s) they can have via the ALTKEYREL feature. In the case of *let* in (121), the value of ALTKEYREL|PRED is *_up_p_rel*.

(121) The rain let up.

$$\left[\begin{array}{l} \text{arg1_part_le} \\ \text{SS} \left[\begin{array}{l} \text{LOC|CAT|VAL} \left[\begin{array}{l} \text{ARGFRAME } \text{arg1} \\ \text{PART|LKEYS|KEYREL|PRED } \mathbb{1} \end{array} \right] \\ \text{LKEYS|ALTKEYREL|PRED } \mathbb{1} \end{array} \right] \end{array} \right]$$

Figure 4.18: The *arg1_part_le* type

The lexical type for a verb like *throw*, which can be intransitive, transitive and ditransitive (see (122a)–(122c)), intransitive or transitive with a PP argument (see (122d)–(122e)), intransitive or transitive with a particle (see (122f)–(122g)), and even intransitive or transitive with a particle and a PP argument (see (122h)–(122i)), is given in Figure 4.19. The ARGFRAME constraint makes sure that the verb can enter the five possible constellations of ARG1, ARG2, ARG3 and ARG4. Underspecification of whether the particle is realized or not accounts for the presence/absence of the particle. All the argument frames in (122) are accounted for.

(122) a. John throws.

¹³An analysis of a resultative sentence is given in Section 6.4.2.

- b. John throws the ball.
- c. John throws Mary the ball.
- d. John throws to Mary.
- e. John throws the ball to Mary.
- f. John throws out.
- g. John throws out the ball.
- h. John throws out to Mary.
- i. John throws out the ball to Mary.

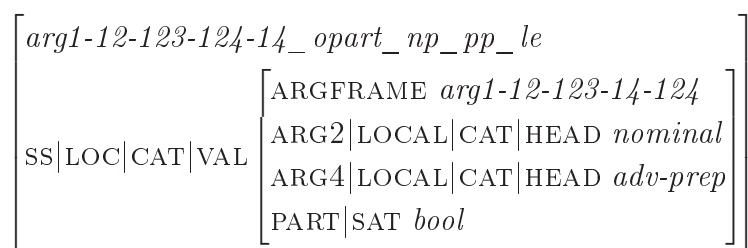


Figure 4.19: The *arg1-12-123-124-14_opart_np_pp_le* type

Verbs that select for particular prepositions or adverbs to head their ARG4 argument, are constrained to select for the predicate of that preposition/adverb. This procedure is adopted from the ERG.¹⁴ It is illustrated in Figure 4.20, where the verb *fokusere* (‘focus’) selects for the KEY value *_på_p_rel* (‘on’) on its ARG4 argument. The PRED value of prepositions and adverbs are unified with the feature KEYS|KEY that is situated in *head*. In this way, the PRED value of the preposition that heads a PP, is visible in the head value of the PP. So when *fokusere* selects for the KEY value *_på_p_rel* as in Figure 4.20, then the PRED value of the preposition that heads the PP complement must be compatible with it.

A verb that selects for a certain set of prepositions or particles, is accounted for by a type hierarchy of PRED values. The verb selects for a supertype of those PRED values that are acceptable.¹⁵

¹⁴The ERG constrains an element on the COMPS list, and not the ARG4 argument.

¹⁵This type hierarchy becomes quite complex when all the verbs in Norsk Ordbank (see Section 4.5.1) are taken into consideration. The script that converts Norsk Ordbank into a Norsyg-compatible lexicon creates a hierarchy consisting of 1805 predicate types.

$$\left[\begin{array}{l} \text{arg12-124-14_np_pp_le} \\ \text{STEM} \langle \text{“fokusere”} \rangle \\ \text{SS|LOC} \left[\begin{array}{l} \text{CAT|VAL|ARG4|LOCAL|CAT|HEAD|KEYS|KEY_p\hat{a}_p_rel} \\ \text{CONT|RELS} \langle \text{[PRED “_fokusere_v_rel”]} \rangle \end{array} \right] \end{array} \right]$$

Figure 4.20: The lexical entry for *fokusere* (‘focus’)

Given the means I have described for restricting the syntactic environment of verbs in Norsyg, the ARGFRAME values, the HEAD values of the ARG2 and ARG4 arguments, the KEY value of the ARG4 argument, and the PRED value of the particles, one is free to give very specific constraints, only allowing one particular argument frame, or one can let the constraints be less specific, so that the verb can enter more frames.

4.5 Expansion of the lexicon

4.5.1 Adaptation of Norsk Ordbank

Norsyg is adapted to Norsk Ordbank,¹⁶ which is a fullform lexicon for Norwegian with more than 1.1 million entries. I have converted Norsk Ordbank into a lexicon with 144161 uninflected lexical entries, where 8229 entries are verbs. The verbs in Norsk Ordbank are annotated with the argument frame information from the NorKompLeks project (see Section 4.8.2). The program that converts the lexicon¹⁷ gathers the argument frame information about each verb and creates the corresponding type if this type does not exist already. This is often necessary if a verb can enter many argument frames. The lexical types for verbs have five kinds of information. First, they specify what kind of constructions the verb can enter. If the verb can enter the arg1-construction, the arg12-construction, and the arg124-construction, it is assigned the ARGFRAME value *arg1-12-124*. Second, they specify the HEAD value of the ARG2 argument (if applicable). If the ARG2 is either an NP or a subordinate clause, the new verb lexical entry type inherits from the type *arg2_cp-np*. Third, the ARG3 value is specified to be a reflexive (if applicable). Forth, the new verb lexical types specify the

¹⁶<http://www.edd.uio.no/prosjekt/ordbanken/>

¹⁷‘convlex.py’ is distributed with Norsyg (see Appendix A).

ARG4 value (if applicable). If the ARG4 value is a PP, the type inherits from the type *arg4_pp*. Fifth, the new verb lexical entry type specifies whether the verb is a particle verb. If it is a particle verb, it inherits from the type *part-verb*, and if not, it inherits from *non-part-verb*. Other information, like the PRED values of selected particles and prepositions, is specified on each individual lexical entry. Based on the argument frame information specified on verbs in NorKompLeks, the lexicon conversion program builds 288 new types for verb lexical entries in addition to the 100 lexical entry types for verbs that already exist (see ‘oble.tdl’ in the norsyg directory). An example of an automatically created verb lexical type is given in (123).

```
(123) arg12-124-2_part_np_pp_le := arg2_np & arg4_pp & part-verb &
      [ SYNSEM.LOCAL.CAT.VAL.ARGFRAME arg12-124-2 ].
```

(123) is the type for the verbs *etse* (‘corrode’), *helle* (‘pour’/‘slope’), *hive* (‘throw’), *kippe* (‘flip up’), and *knalle* (‘crack’). What these verbs have in common, is that they can enter the arg12-construction, the arg124-construction, and the arg2-construction, hence the ARGFRAME value *arg12-124-2*. The verbs are particle verbs, so the type inherits from *part-verb*. The verbs require an NP as value of ARG2 and a PP as value of ARG4 (if applicable), so the type inherits from *arg2_np* and *arg4_pp*.

The entry of the infinitival form of *helle* in Norsk Ordbank is given in (124), where the fields in angle brackets show what argument frames the verb can enter, <intrans2>, <adv6>, and <part1/ut>.¹⁸

```
(124) 27112 helle helle verb inf <intrans2> <adv6> <part1/ut> 021 1
```

These argument frame specifications are translated into the type in 123 according to a table distributed with the Norsyg grammar (‘nkl2lkb.txt’). When appearing alone, <intrans2> translates into the type *arg2_np_le* (the type for intransitive unaccusative verbs), <adv6> translates into the type *arg124_np_pp_le* (the type for transitive verbs with PP complements), and <part1/ut> translates into the type *arg12_part_np_le* (the type for transitive particle verbs (the PRED value of the particle *ut* (‘out’) is specified on the lexical entry)). When these three argument frames appear on the same lexical entry, the type *arg12-124-2_part_np_pp_le* is created, as shown above. It

¹⁸This argument frame information stems from the NorKompLeks project (see Section 4.8.2).

accommodates all the frames just mentioned.¹⁹ The lexical entry of *helle* in the Norsyg grammar is given in (125).

```
(125) helle-v := arg12-124-2_part_np_pp_le &
      [ STEM <"helle">,
        INFLECTION v1,
        SYNSEM.LKEYS.ALTKEYREL.PRED _ut_p_rel,
        SYNSEM.LKEYS.KEYREL.PRED "_helle_v_rel" ].
```

4.5.2 Unknown words

Unknown words pose a challenge to deep linguistic grammars when they are used to parse unknown text. In an evaluation of a large-scale grammar referred to in Fouvry (2003), 89% of the total number of failed parses failed (possibly partly) because of unknown words. A lexicon will never be “complete” since new words are created all the time. One approach to the unknown word problem is to make use of the syntactic environment to “recognize” an unknown word (see for example Erbach (1990); Horiguchi *et al.* (1995); Barg and Walther (1998)). The syntactic environment then imposes constraints on the unknown word, which is an underspecified entry. The information about the unknown word from the syntactic environment is collected and refined.

Norsyg is employed in a similar fashion. If a word is not recognized by the grammar, it is assigned the lexical type *unknown-word* shown in Figure 4.21. The type is given the HEAD value *adj-noun-verb*, which means that it is either an adjective, a noun, or a verb. The semantic relation is underspecified. The type is specified as INFLECTED +, which means that it is fully inflected. This prevents inflectional rules from applying to it.

The syntactic rules that apply to the unknown word will determine the category of the unknown word. If the unknown word is a verb, also the argument frame will be settled. That is, an unknown intransitive verb will be assigned the ARGFRAME value *arg1* if the verb enters an arg1-construction, and an unknown transitive verb will be assigned the ARGFRAME value *arg12* if the verb enters an arg12-construction. Also

¹⁹One weakness of the frame packing procedure described here, is that not only the arg12-construction may appear with a particle, but also the arg124-construction and the arg2-construction may appear with a particle, even though that is not specified in the original lexicon. This makes the lexicon less precise.

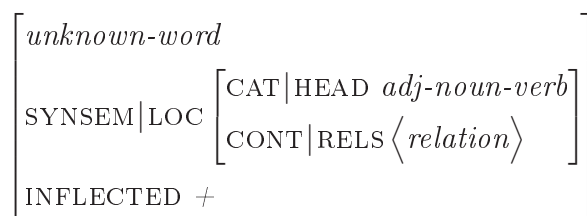


Figure 4.21: Partial representation of the type *unknown-word* in Norsyg

the HEAD values of the ARG2, ARG3, and ARG4 valence requirements will be settled.²⁰ There is no need for additional mechanisms to make the parser recognize unknown words. Given the exo-skeletal design of the grammar and the fact that the formalism is unification-based and uses typed feature structures, the unknown word recognition comes for free.²¹

4.5.3 Lexicon acquisition

The unknown word mechanism can be used for lexicon acquisition. The use of a large-coverage unification-based grammar (the ERG) for lexicon acquisition is presented in Fouvry (2003). With the help of a statistical Part-of-Speech (PoS) tagger, a selection of the 463 possible lexical types are assigned to the unknown word, each as a separate entry. The possible definition of the unknown word can be derived from the successful parse(s). The procedure suggested for Norsyg differs from the procedure shown in Fouvry (2003) in that only one underspecified entry is entered into the parse chart, rather than one entry per (probable) lexical type. This is possible due to the exo-skeletal nature of the grammar.

One way to use Norsyg to do automatic acquisition of argument frames would be to let the grammar parse a corpus, and let the subcat requirements of the verbs in the lexicon be underspecified. The grammar would then build syntactic trees dependent

²⁰When a sentence with several unknown words is parsed, and the unknown words are assigned the type *unknown-word*, the number of edges in the parse chart may become too big for the parser to handle. I therefore use a more constrained type *uk-noun-phrase* when I parse unknown text. The head value is in this type specified to be *nominal* since most of the unknown words are proper nouns or nouns. This means, however, that sentences with unknown verbs and adjectives will not get the correct analysis. (In Appendix A.4, I estimate that 24.6% of the sentences taken from a Wikipedia article, that the grammar parses, do not get the correct analysis.)

²¹One would however need a mechanism for refining the recognized unknown words, since the constraints specified on the unknown words are often too specific. Verbs are for example specified with number information about their subjects and objects, which is information one does not want to represent on verbs (at least not in a language like Norwegian).

on the context of the verbs. The constraints imposed by the syntactic trees onto the lexical entries of the verbs would be gathered and stored, and a program similar to the lexicon conversion program mentioned in Section 4.5.1, would create the necessary lexical types according to different sets of constraints imposed by the syntax in all the successful analyses. In order to restrict the mechanism so that the constraints of highly unlikely analyses were left out, the statistical data of a treebank similar to the LinGO Redwoods Treebank (Oepen *et al.*, 2004a) could be used to select only the most probable parses for each parsed item.²²

Given that a verb like *feire* ('celebrate') was assigned the valence constraints in (126) by different syntactic contexts ((126a) in an intransitive clause, (126b) in a transitive clause with an NP as ARG2 value, and (126c) in a transitive clause with a subordinate clause as ARG2 value) the lexical type in (127) could be created for the lexical entry of *feire*.

(126) a. $\left[\begin{array}{l} \text{ARGFRAME } arg1 \\ \text{ARG1} | \text{LOCAL} | \text{CAT} | \text{HEAD } noun \end{array} \right]$

b. $\left[\begin{array}{l} \text{ARGFRAME } arg12 \\ \text{ARG1} | \text{LOCAL} | \text{CAT} | \text{HEAD } noun \\ \text{ARG2} | \text{LOCAL} | \text{CAT} | \text{HEAD } noun \end{array} \right]$

c. $\left[\begin{array}{l} \text{ARGFRAME } arg12 \\ \text{ARG1} | \text{LOCAL} | \text{CAT} | \text{HEAD } noun \\ \text{ARG2} | \text{LOCAL} | \text{CAT} | \text{HEAD } atcompl \end{array} \right]$

(127) `arg1-12_cp-np_le := arg2_cp-np & non-part-verb &`
`[SYNSEM.LOCAL.CAT.VAL.ARGFRAME arg1-12] .`

The type in (127) subsumes the different valence constraints given in (126) since it i) is compatible with both the *arg1*-construction and the *arg12*-construction by specifying the ARGFRAME value to be *arg1-12*, and ii) constrains the ARG2 value to be either an NP or a subordinate clause by inheriting from the type *arg2_cp-np*. It also specifies that it is not a particle verb type by inheriting from the type *non-part-verb*. A lexical entry that inherits from *arg1-12_cp-np_le* will be compatible with the three kinds of syntactic context in (126), and no other kinds of syntactic context.

²²At present, there is no HPSG treebank for Norwegian.

I made a test where I removed all subcat information on the main verbs. Instead of letting them inherit from the lexical types specifying argument frame information, as shown in Section 4.4, I let them inherit from the type *main-verb-lxm*, which is the general type for all main verbs except raising and control verbs. I constrained the type *main-verb-lxm* so that it did not take nominals as value of *arg4*, since the class of verbs that take nominals as predicatives is very small in Norwegian. (See Figure 4.22.) I also removed the selectional restrictions (lexical constraints of the ARG4 value and the PART value). This allowed all main verbs to enter all possible constructions, except from the raising and control constructions, and the predicative constructions with nominal predicates.



Figure 4.22: The (slightly altered) *main-verb-lxm* type

The alternative grammar was tested on a corpus consisting of 8272 5 to 10 word sentences from Norwegian Wikipedia.²³ In order to reduce the number of errors, I made sure that all the words of the selected sentences were listed in Norsk Ordbank, which the Norsyg lexicon is derived from, (see Section 4.5.1). As Table 4.1 shows, the alternative grammar parsed 54.7% of the items (4521). The average number of parses for each parsed sentence is 111.62. This number is relatively low, mainly due to the fact that 2270 of the items had the copula verb *er*, and that this verb has kept its original constraints. In addition, 1794 parses failed because the edge limit was exhausted. (The chart size limit was set to 10000 nodes.) If the chart size limit had been raised, the average number of parses would have gone up since more ambiguous sentences would also have been analysed.

I also tested the original grammar on my Norwegian Wikipedia corpus of short sentences, and the results are shown in Table 4.2. The grammar parses 64.8% of the items and the average number of parses is 27.14. 142 parses failed because the edge limit (10000) was exhausted.

²³One 4 word sentence was also included in the corpus. I did not realize this before all the tests were finished.

Aggregate	total items #	positive items #	word string \emptyset	lexical items \emptyset	distinct analyses \emptyset	total results #	overall coverage %
i-length in [10 .. 15]	1569	1569	10.00	44.92	146.47	519	33.1
i-length in [5 .. 10]	6702	6702	7.65	35.07	107.10	4002	59.7
i-length in [0 .. 5]	1	1	4.00	0.00	0.00	0	0.0
Total	8272	8272	8.10	36.42	111.62	4521	54.7

(generated by [incr tsdb()] at 29-oct-08 (10:21))

Table 4.1: Coverage of the Norsyg grammar with ‘open’ verb lexical entries on Wikipedia corpus of short sentences.

Aggregate	total items #	positive items #	word string \emptyset	lexical items \emptyset	distinct analyses \emptyset	total results #	overall coverage %
i-length in [10 .. 15]	1569	1569	10.00	48.34	46.54	915	58.3
i-length in [5 .. 10]	6702	6702	7.65	37.54	23.14	4443	66.3
i-length in [0 .. 5]	1	1	4.00	0.00	0.00	0	0.0
Total	8272	8272	8.10	39.53	27.14	5358	64.8

(generated by [incr tsdb()] at 29-oct-08 (13:01))

Table 4.2: Coverage of the original Norsyg grammar on Wikipedia corpus of short sentences.

The initial test of the grammar with underspecified subcat constraints on verbs shows that the grammar can to some extent be used to parse short sentences when the subcat constraints of the main verbs are removed. Given the statistics from a tree bank, it would be possible to extract subcat information of the highest ranked analyses involving a certain main verb, and use this information to arrive at a possible lexical type in the manner outlined for *feire* above.

4.6 Comparison of the construction-constraining mechanism and a lexicalist approach

In order to test how the constructionalist approach performs compared to a lexicalist approach on real data, I created a version of Norsyg where valence alternations are accounted for by means of multiple lexical entries rather than using the construction-constraining mechanism (see Sections 1.3, 4.3, and 4.4). A verb that has the type *arg1-12_np_le* in Norsyg is in the alternative version given two lexical entries, one of the type *arg1_le* and one of the type *arg12_np_le* (one for each of the argument structure codes assigned by the original NorKompLeks lexicon (see Section 4.8.2)). 5009 of the verbs from the NKL lexicon are listed with only one frame, and are therefore given only one lexical entry in the new lexicon, while 3439 verbs are listed with more than one argument frame and are given the corresponding amount of lexical entries. (The verb *få* was given 12 lexical entries.) This gave me a lexicon with 13201 lexical entries for verbs, rather than the original 8448 lexical entries for verbs, an increase of 4753. I added 38 new types for verb lexical entries.

I used the alternative lexicalist version of the Norsyg grammar and the original Norsyg grammar to parse the Wikipedia corpus of 5 to 10 word sentences mentioned in Section 4.5.3. I compared the results of the batch parses and selected the sentences that were given the same number of analyses by the two grammars. Sentences that did not parse were not included. I also excluded sentences with the copula verb *er/var* ('is'/'was') and the verb *har* ('has') since they seemed to be overrepresented in the data.²⁴ I ended up with a set of 544 sentences. I excluded the sentences that differed with regard to the number of parses in order to make the comparison of the two grammars as good as possible.

I let the two grammars parse the new set of sentences and compared the results. Table 4.3 shows that the two grammars, as expected, have the same coverage (100%), and that they produce the same amount of analyses (15.02 on average). The table also illustrates the difference in lexical ambiguity of the two grammars. The lexicalist grammar ('(g)old') has a lexical ambiguity of 6.31, while Norsyg ('new') has a lexical ambiguity of 4.65.

²⁴Typical short sentences in the Wikipedia data are sentences like *Lesotho er et land i Afrika*. ('Leshoto is a country in Africa.') and *I dag har selskapet rundt seksti ansatte*. ('Today, the company has about sixty employees').

Aggregate	(g)old				new			
	lexical ∅	analyses ∅	in ∅	out ∅	lexical ∅	analyses ∅	in ∅	out ∅
i-length in [10 .. 15]	6.38	17.71	100.0	100.0	4.64	17.71	100.0	100.0
i-length in [5 .. 10]	6.29	14.56	100.0	100.0	4.65	14.56	100.0	100.0
Total	6.31	15.02	100.0	100.0	4.65	15.02	100.0	100.0

(generated by [incr tsdb()] at 11-nov-08 (21:52))

Table 4.3: Comparison of competence. Gold = one lexical entry per argument frame. New = packed argument frames.

Table 4.4 shows that the original Norsyg has a better performance than the lexicalist version of the grammar. The number of tasks is 27% smaller, parsing time is reduced by 34.9%, and space is reduced by 40.2%.²⁵

Aggregate	(g)old			new			reduction		
	tasks ∅	time ∅	space ∅	tasks ∅	time ∅	space ∅	tasks %	time %	space %
i-length in [10 .. 15]	2789	0.97	108396	2129	0.66	72660	23.7	31.5	33.0
i-length in [5 .. 10]	1644	0.58	78498	1183	0.37	45569	28.0	35.9	41.9
Total	1810	0.64	82840	1321	0.42	49503	27.0	34.9	40.2

(generated by [incr tsdb()] at 11-nov-08 (21:49))

Table 4.4: Comparison of performance. Gold = one lexical entry per argument frame. New = packed representations.

The complexity of the two grammars and the different approaches to argument frame alternations in the two grammars make it difficult to achieve equal coverage on all the data for the two grammars. I chose to exclude most of the sentences, where the two

²⁵It could of course be objected to this test that a grammar without the packing of argument structure information maybe could be implemented in a different way, that would make parsing more efficient. (In languages with fixed word order like Norwegian and English, one could for example enter all the arguments on a single subcat list and use 2 rather than 8 valence rules to realize the arguments; one binary valence rule and one valence rule for extracted arguments.) This comparison is only done for testing the impact of the packing of argument structure information in a grammar that is implemented similar to Norsyg.

grammars do not have the same amount of analyses, rather than attempting to track down the reason for the difference in behavior. The result of this is that many of the cases with more ambiguity are not included. (The average number of analyses of all the sentences in the 5–10 word Wikipedia corpus is 27.14 (see Table 4.2), while in the new set, which was behind the numbers shown in Tables 4.3 and 4.4, the average number of analyses is 15.02.) If more ambiguous examples had been included, one could expect a bigger difference in performance between the two grammars, since verbs with more alternations would be part of the test. The difference in competence and performance of the two grammars on the whole 5–10 words Wikipedia corpus is shown in Tables 4.5 and 4.6.

Aggregate	(g)old				new			
	lexical ∅	analyses ∅	in ∅	out ∅	lexical ∅	analyses ∅	in ∅	out ∅
i-length in [10 .. 15]	5.85	73.98	52.7	100.0	4.59	46.54	58.3	100.0
i-length in [5 .. 10]	6.57	46.48	65.0	100.0	4.84	23.14	66.3	100.0
i-length in [0 .. 5]	1.50	6.00	100.0	100.0	1.50	0.00	0.0	100.0
Total	6.40	50.86	62.6	100.0	4.78	27.14	64.8	100.0

(generated by [incr tsdb()] at 12-nov-08 (13:56))

Table 4.5: Comparison of competence (8272 sentences). Gold = one lexical entry per argument frame. New = packed argument frames.

As Table 4.6 shows, the reduction is bigger when all the sentences are considered (35% difference in tasks, 41.2% difference in time, and 45.4% difference in space.) However, in this comparison, the number of analyses produced by the two grammars differs. (See Table 4.5.) In the lexicalist version, the average number of analyses is 50.86, while in the original version, the average number of analyses is 27.14. The high number of analyses in the lexicalist version is probably due to the fact that the grammar is less constrained. (It was constructed with the single purpose of being a comparison to the original Norsyg grammar.)

The parse charts in Figures 4.23 and 4.24 illustrate how the work load of the two grammars may differ for a short sentence like *Jon presset appelsinen* (‘Jon

Aggregate	(g)old			new			reduction		
	tasks ∅	time ∅	space ∅	tasks ∅	time ∅	space ∅	tasks %	time %	space %
i-length in [10 .. 15]	4739	1.81	265510	3141	1.08	152926	33.7	40.4	42.4
i-length in [5 .. 10]	2821	1.04	148698	1792	0.60	78510	36.5	42.0	47.2
i-length in [0 .. 5]	1655	0.43	44256	0	0.00	0	100.0	100.0	100.0
Total	3154	1.17	168984	2040	0.69	92212	35.3	41.2	45.4

(generated by [incr tsdb()] at 12-nov-08 (13:50))

Table 4.6: Comparison of performance (8272 sentences). Gold = one lexical entry per argument frame. New = packed representations.

pressed the orange’) when a verb with many alternations appear in the sentence.²⁶ The verb *presse* (‘press’) can enter 8 argument frames. In the original Norsyg grammar it has one lexical entry of the type *arg12-124-14_part_np_pp+ip2_le*, and in the lexicalist version of the grammar it has 8 lexical entries of the types *arg124_np_pp_le*, *arg12_np_le*, *arg124_np_pp+ip2_le*, *arg12_refl_le*, *arg12_part_np_le*, *arg124_refl_pp_le*, *arg14_pp_le*, and *arg124_np_pp_le*.

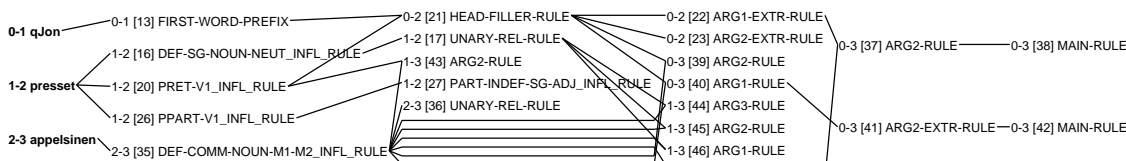


Figure 4.23: Parse chart for *Jon presset appelsinen* (‘Jon pressed the orange’) in the original Norsyg grammar.

The parse chart of the original Norsyg grammar shown in Figure 4.23 has 46 edges, and the parse chart of the lexicalist version shown in Figure 4.24 has 154 edges.²⁷ Both grammars give two analyses to the sentence.

²⁶The reason why the first word *Jon* in the parse charts is given the prefix *q* is explained in Appendix A.6.1.

²⁷The morphological rules were not displayed in the parse charts.

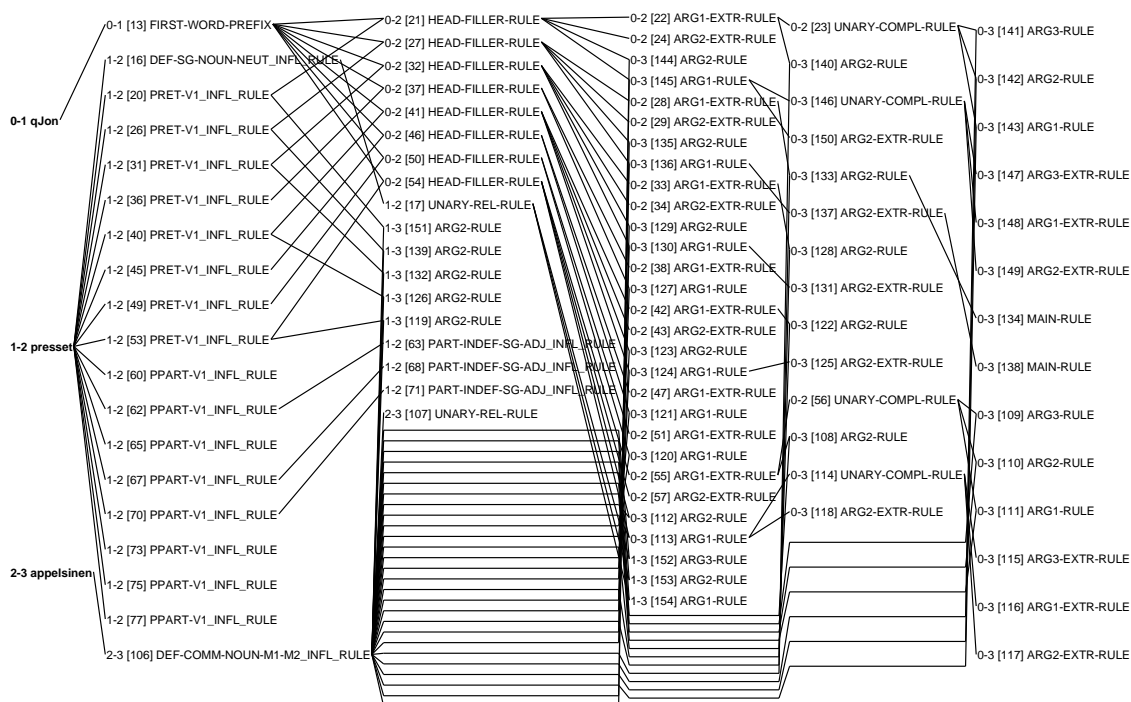


Figure 4.24: Parse chart for *Jon presset appelsinen* ('Jon pressed the orange') in the lexicalist version of the Norsy grammar.

4.7 Comparison with the RASP system

The design of the Norsyng grammar has certain abstract similarities with the Robust Accurate Statistical Parsing (RASP) system (Briscoe *et al.*, 2006), which is a so-called ‘shallow’ parser. A shallow parser is more robust and efficient than a ‘deep’ parser. Typically, a shallow parser has no or very limited access to fine-grained lexical information. It typically includes PoS tagging, chunking, and Relation Finding. Shallow parsers often parse sentences into partial trees (chunks), and find relations that hold between the parts of the sentences (subject, object, and so on). They are designed to be robust, and they will parse also ungrammatical input. A deep parser on the other hand gives complete analyses, and analyzes in principle only grammatical input.

The RASP system is an advanced shallow parser in that it returns full tree analyses, although the analyses do not include phenomena such as long distance dependencies and raising (see below). Also, the RASP system is somewhat atypical as a shallow parser, in that it utilizes a hand-written syntactic grammar, albeit assuming only very coarse-grained lexical categories (which are obtained by PoS tagging). Given the sentence *Mary likes John*, the RASP system outputs the tree structure in Figure 4.25.

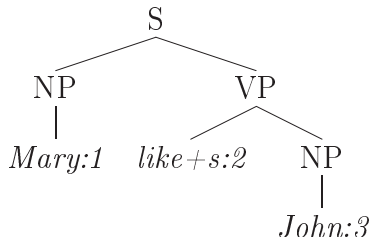


Figure 4.25: RASP tree structure for *Mary likes John*

It also outputs the Grammatical Relations holding between predicates and arguments that the system can recover (Briscoe *et al.*, 2006, 79). This is shown for *Mary likes John* in (128), where *Mary* is identified as the subject of *like*, and *John* is identified as the direct object. The tree structures can also be used to extract RMRSs (Ritchie, 2004).

(128) (|ncsubj| |like+s:2_VVZ| |Mary:1_NP1| _)
 (|dobj| |like+s:2_VVZ| |John:3_NP1|)

The Norsyng grammar and the RASP system have in common that they both allow for underspecified argument frames on verbs. As I showed in Sections 4.3, 4.4, and 4.5,

Norsyg assumes only one lexical entry per verb, also in cases where the verb can have more than one argument frame. Constraints entered on each lexical entry together with a type hierarchy of linking types restrict the possible number of argument frames. In the RASP system, all the verbs have an underspecified VSUBCAT feature, and they can be given any of the 31 possible subcat frames for verbs (Briscoe, 2006, 9). One of the initial applications of the RASP system was to extract Grammatical Relations (Carroll and Briscoe, 2001). This is also a possible application of the Norsyg grammar. But there are significant differences between a shallow parser like the RASP system and a deep parser like Norsyg.

The RASP system has 678 phrase structure rules which provide tree analyses of English sentences. The relatively high number of rules is due to detailed specifications of the daughters, and syntactic structures that are not strictly binary. For example, the sentence *Mary gives him an apple* receives the structure in (4.26) where the ternary rule ‘V1/v_np-pro_np’ (VP goes to verb, pronoun and NP) forms a VP from the verb *give*, the pronoun *him*, and the NP *an apple*. The system also pays a lot of attention to punctuation.

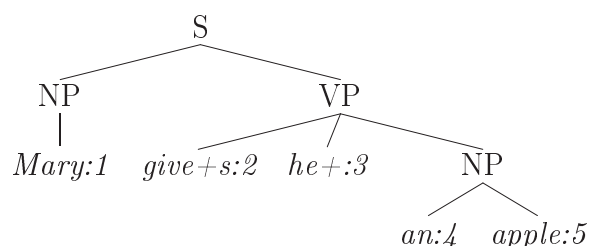


Figure 4.26: RASP tree structure for *Mary gives him an apple*

The use of rules with detailed specifications of the daughters together with a preference for flat structures, would result in a very large number of rules in a language like German if precise analyses involving scrambling and adjunct attachment were to be given, and such an approach would not be feasible for a deep grammar. (See remarks to Construction Grammar in Section 2.7.3.) The Norsyg grammar on the other hand uses far less rules (52) and employs binary structures. This makes it possible, in principle, to account for the German data without changing the fundamentals of the design (see Appendix B.2.).

The RASP system does not account for long distance dependencies like Wh-movement (see Briscoe (2006, 15)), as illustrated for *Who do you think Mary likes?*

in (129). Here, the system outputs an object relation between *who* and *think*, while there should have been an object relation between *who* and *like*.

```
(129) (|obj| |think:4_VV0| |Who:1_PNQS|)
      (|aux| |think:4_VV0| |do:2_VD0|)
      (|ncsubj| |think:4_VV0| |you:3_PPY| _)
      (|ccomp| _ |think:4_VV0| |like+s:6_VVZ|)
      (|ncsubj| |like+s:6_VVZ| |Mary:5_NP1| _)
```

A deep grammar like Norsyg on the other hand, can account for long distance dependencies (see Section 6.9).

The treatment of raising in the two grammars has certain similarities. In both grammars the “raised” argument is assumed to be an argument of the control verb.²⁸ However, while in Norsyg it is assumed that the argument is also an argument of the controlled verb (see Section 6.7.3), the RASP system only assigns a grammatical relation to the raised argument from the control verb. The tree in Figure 4.27 is the RASP analysis of *Mary seems to eat apples*. The Grammatical Relations extracted from that tree are given in (130). It shows that the raised argument is the subject of the raising verb *seem*, and not the controlled verb *eat*.

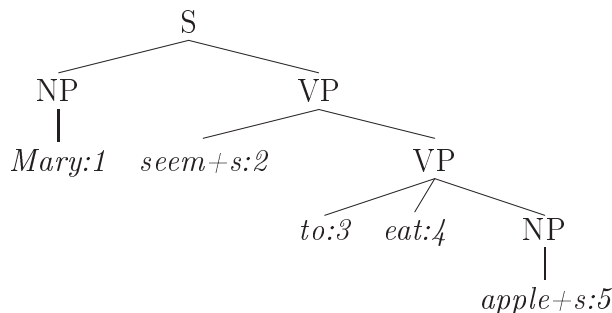


Figure 4.27: RASP tree structure for *Mary seems to eat apples*

```
(130) (|ncsubj| |seem+s:2_VVZ| |Mary:1_NP1| _)
      (|xcomp| |to| |seem+s:2_VVZ| |eat:4_VV0|)
      (|dobj| |eat:4_VV0| |apple+s:5_NN2|)
```

²⁸This goes against the general assumption that the argument is raised from the controlled clause. See remarks in Section 6.7.4.

A final difference between the RASP system and Norsyg is that the RASP system outputs surface Grammatical Relations similar to the functions in LFG, while Norsyg outputs deep Grammatical Relations corresponding to the Initial Stratum in Relational Grammar and the deep structure in GB. (See Section 1.2.)

4.8 Norsyg compared to other Norwegian computational resources

In this section I will compare Norsyg to four other Norwegian computational resources, the lexicon projects TROLL (The Trondheim Linguistic Lexicon Project) and NorKompLeks (Norsk Komputasjonelt Leksikon), and the grammar projects NorSource (Norwegian Resource Grammar) and NorGram (Norsk komputasjonell grammatikk).

4.8.1 TROLL (The Trondheim Linguistic Lexicon Project)

TROLL (Johnsen *et al.*, 1989) is an HPSG-like computational lexicon for Norwegian in the spirit of Hellan (1988). It has 27 basic templates for Norwegian verbs. These templates can undergo derivational valence-changing rules.

There are templates for for example intransitive verbs like *jump*, ergative verbs like *roll*, experiencer intransitive verbs like *freeze*, transitive verbs like *kick*, and ditransitive verbs like *give*. The templates contain information about the thematic role, syntactic function and category of the arguments. The transitive template has the following definition:

SAF: <ag,np,ea>, <th,np,gov>

Statement: tv

SAF stands for Syntactic Argument Frame, and in case of the transitive template, it lists two arguments. The first argument on SAF, ‘<ag,np,ea>’, has the thematic role *agent* (‘ag’), the category is *noun phrase* (‘np’) and the syntactic function is *external argument* (‘ea’). The second argument, ‘<th,np,gov>’, has the thematic role *theme* (‘th’), the category *noun phrase* (‘np’) and the syntactic function *governed* (‘gov’).

The derivational rules in TROLL are like HPSG lexical rules. The derivation in Figure 4.28 shows how a passive transitive particle verb is derived from the

intransitive verb *skyte* ('shoot'). There are four derivational rules applying in this example ('InhObj' (Cognate object alternation), 'TV_smallcl_AdvP' (Resultative construction), 'PredicMvt' (Predicative preposing) and 'Pass' (passive)) applying in a fixed order.

Derivation	Syntactic Argument Frame:	Example
Pass	<ag,rp,implarg>,<tvscsu,np,gov>, <_ ,_ ,preposed _predic>	... ble skutt bort kulene ... was shot away bullets-the
↑		
PredicMvt	<ag,np,ea>,<tvscsu,np,gov>, <_ ,_ ,preposed _predic>	Per skyter bort kulene Per shoots away bullets-the
↑		
TVsmallclAdvP	<ag,np,ea>,<tvscsu,np,gov>, <_ ,_ ,predic>	Per skyter kulene bort Per shoots bullets-the away
↑		
InhObj	<ag,np,ea>,<inherobj,np,gov>	Per skyter kuler Per shoots bullets
↑		
basic	<ag,np,ea>	Per skyter Per shoots

Figure 4.28: Lexical derivations in TROLL

The result of the derivation in Figure 4.28 is a lexeme with the Syntactic Argument Frame ' $\langle \text{ag,rp,implarg} \rangle, \langle \text{tvscsu,np,gov} \rangle, \langle _ ,_ ,\text{preposed_predic} \rangle$ ', which is the argument frame for the passive transitive particle verb. ' $\langle \text{ag,rp,implarg} \rangle$ ' means that the verb has an argument that has the thematic role *agent*, which is implicit ('rp' means that it has the empty category 'Referential Phrase'). ' $\langle \text{tvscsu,np,gov} \rangle$ ' means that the verb has the thematic role 'small clause subject' with a transitive verb, which is realized as a governed NP. ' $\langle _ ,_ ,\text{preposed_predic} \rangle$ ' means that the verb has a preposed particle. The derivational rules have means to restrict the input, in order to avoid overgeneration.

The core idea with TROLL is to have a restricted number of basic lexical templates, from which certain sets of other lexical templates can be derived. By associating a verb with a particular basic template, one can derive all possible syntactic argument frames by means of the derivational rules.

In a sense I try to achieve the same with Norsyg, except that instead of letting a lexeme derive all possible syntactic argument frames in the lexicon (a transitive verb in TROLL has 85 argument frames), I let the lexeme have only one specification, which

will make it compatible with the syntactic structures that can be expected for that verb.

Norsyg does not have the thematic roles that TROLL has, and it also does not have the possibility to merge a verb and a particle by means of a derivational rule. Norsyg can account for all the syntactic structures that are predicted in the ‘xtemplates’ Appendix of Johnsen *et al.* (1989).

4.8.2 NorKompLeks (Norsk Komputasjonelt Leksikon)

NorKompLeks (NKL) is a Norwegian computational lexicon with information about inflectional patterns and phonological representations. The lexicon also has information about argument structure frames for verbs. There are 105 different argument structure frames in NKL. In contrast to TROLL, which operates with basic templates from which all surface structures are derived, NorKompLeks operates with direct descriptions of the surface argument structure. All argument structures that a verb can have are represented as lists of codes in the lexical entry of the verb. The argument structure representations contain information about the thematic role, syntactic function and category of the arguments, adapted from TROLL’s templates.

The definition of the code for an unergative intransitive argument structure is given in (131), and the definition of the code for a transitive argument structure is given in (132). (131) has the code name ‘intrans1’, and its single argument (‘arg1’) is marked functionally as *subject* (‘su’), its thematic role is *agent* (‘ag’), and its category is *noun phrase* (‘np’). (132) has the code name ‘trans1’, and it has two arguments. The first is an agent subject NP (‘su::ag::np’), and the second is a theme object NP (‘obj::th::np’).

(131) `arg_code(intrans1, [arg1:su::ag::np])`

(132) `arg_code(trans1, [arg1:su::ag::np, arg2:obj::th::np])`.

An example of a verb that can be both intransitive and transitive is listed as *akkompagnere* (‘accompany’) in (133), where the two argument structure codes ‘intrans1’ and ‘trans1’ are listed. If the verb has certain selectional restrictions, this is marked in the lexical specification, as illustrated in (134), where the verb *agitere* (‘agitate’) selects for a PP headed by *for* (‘for’). Verbs that can enter many argument frames, like *få* (‘get’), are specified with many argument structure codes, as illustrated in (135).

(133) `w(akkompagnere,459,[intrans1,trans1]).`

(134) `w(agitere,372,[intrans1,trans11([for]))).`

(135) `w(få,18819,[
 trans14,
 part6([tilbake,igjen,fram,frem]),
 predik7,
 part1([til,gjennom,igjennom,med,bort,vekk,unna,fram,frem,igang]),
 ditrans5([til,fra]),
 ditrans6([til]),
 refl13([til]),
 refl14([med]),
 refl6,
 trans20([med]),
 trans11([i]),
 aux1([perf_part,inf]))).`

Argument frame specifications for verbs that cannot passivize are marked with ‘-passiv’. This is illustrated in (136), which shows the definition of the code *intrans2* for unaccusatives.

(136) `arg_code(intrans2,[arg1:su::th::np,-passiv]).`

Norsyg does not have the specification of thematic roles that NKL has. All the syntactic argument frames specified in NKL are accounted for in Norsyg. This is illustrated in Appendix A.5, where each sentence corresponds to an argument structure frame in NKL.²⁹ The table shows how many analyses was assigned to each sentence by Norsyg, and also how many edges there were in each of the parse charts.³⁰ Most of the example sentences are taken from Hellan (2002).

²⁹A few frames like *part5* and *predic11*, *trans2* and *trans18*, *trans3* and *trans19*, *refl12* and *refl18*, *adv2* and *adv13* share one example. *part3* and *refl14* share two examples. *adv15* and *refl10* each correspond to two examples, and *aux1* corresponds to three examples.

³⁰The file that contains these test sentences, ‘nkl.items’, is distributed with Norsyg.

4.8.3 NorSource (Norwegian Resource Grammar)

NorSource is an implemented HPSG grammar for Norwegian (see Beermann and Hellan (2004), Hellan and Beermann (2005) and Hellan (2005)).³¹ The grammar gives detailed semantic representations. The grammar has approximately 80000 lexical entries (of which 13144 are lexical entries for verbs), 178 rules, 61 inflectional rules, and 34 lexical rules.

The grammar accounts for many more argument structure frames than assumed in NorKompLeks, mainly by means of lexical entry types. A verb that can enter more than one argument frame is given several lexical entries. The verb *gi* ('give'), for example, has 11 entries. Passive is accounted for by means of lexical rules.

This procedure for capturing the different argument frames a verb can enter is different from the procedure in Norsyg, where a single lexical entry is given information that allows it to enter all the frames that are expected.

NorSource has a number of lexical entry types for verbs that are equipped with fine-grained semantic information and restrictions on the syntactic environment. Norsyg does not have any such specifications.

In addition to the VAL features SUBJ, SPR, SPEC, COMPS and ICOMPS,³² NorSource has the QVAL (qualitative valence) features SUBJECT, DOBJECT (direct object), IOBJECT, PREDIC, OBL1 and OBL2 (see Hellan and Haugereid (2004)). The QVAL features function as 'pointers' to elements on the valence lists, as illustrated in Figure 4.29. The QVAL features make it possible to refer to for example the direct object irrespective of its position on the COMPS list. The linking between syntactic arguments and semantic arguments is done via the QVAL features in particular types, as illustrated in Figure 4.30.

While Norsyg has a fixed correspondence between the valence features ARG1, ARG2, ARG3 and ARG4 on the one hand, and the basic relations *arg1-relation*, *arg2-relation*, *arg3-relation* and *arg4-relation* on the other (see Section 3.5), there is no direct correspondence in NorSource between QVAL features and the semantic attributes ARG1 etc. For example, the direct object in a presentational construction is linked to the ARG1 role of the verb's relation, while the direct object otherwise is linked to the ARG2 role.

³¹The grammar's homepage is <http://www.ling.hf.ntnu.no/forskning/norsource/>.

³²ICOMPS (interspersable complements) is a list of complements that can be preceded by an adverbial.

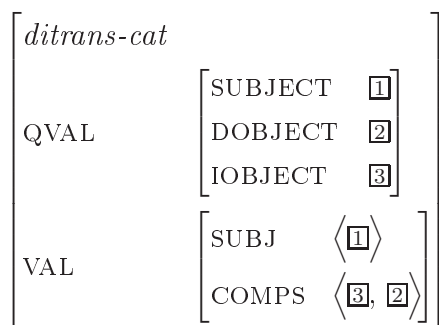


Figure 4.29: QVAL pointers to elements on the VAL lists in NorSource



Figure 4.30: Linking of the direct object index to the semantic ARG2 in NorSource

The QVAL feature PREDIC corresponds to the ARG4 valence feature in Norsyg. The OBL1 feature corresponds to what would be the ARG5 valence feature in Norsyg.³³ As for the other QVAL features, SUBJECT, DOBJECT and IOBJECT, there is no one-to-one correspondence to Norsyg.

4.8.4 NorGram (Norsk komputasjonell grammatikk)

NorGram is a broad coverage computational LFG grammar for Norwegian (both Bokmål and Nynorsk) developed at the university of Bergen by Helge Dyvik and Victoria Rosén. It is implemented with XLE (Crouch *et al.*, 2007), which is a combination of linguistic tools developed at PARC and Grenoble XRCE. The grammar is used as the analysis component in the LOGON translation system (Oepen *et al.*, 2004b).

NorGram consists of approximately 15000 lines of code (excluding lexicon) and has about 940 templates (generalisations over linguistic expressions), 230 phrase structure rules,³⁴ and approximately 80000 lexical entries. As in Norsyg, the argument structure information of the verbs is based on the NKL lexicon.

³³As stated in footnote 1, page 87, selectional restrictions about the arg5-role are specified via the ARG4 valence feature in the present implementation.

³⁴This number is according to Helge Dyvik (personal communication) not very informative since the complexity of these rules varies a lot, and many of them contain many disjunctions, which means that they can be expanded into almost 50000 phrase structure rules.

The approach to argument frame alternations is similar to the lexicalist variant of Norsyg presented in Section 4.6 which has one lexical entry per argument frame. In NorGram, a verb like *filme* (‘film’), which is both intransitive and transitive, has the definition in (137). The orthographic form *filme* is here assigned a disjunction of two lexical macros (V-SUBJ-OBJ and V-SUBJ).

(137) `filme V XLE { @(V-SUBJ-OBJ filme filme)
| @(V-SUBJ filme filme) }; ETC.`

The verb *presse* (‘press’) mentioned in Section 4.6 has the definition in (138). The orthographic form *presse* is here assigned a disjunction of 12 lexical frames.

(138) `presse V XLE { @(V-SUBJ-OBJ-PXCOMP presse presse til)
| @(nkl_adv7 presse presse)
| @(V-SUBJ-POBJ presse presse på)
| @(V-SUBJ-OBJrefl-POBJ presse presse på)
| @(V-SUBJ-OBJ-POBJ presse presse for)
| @(V-SUBJ-OBJ-POBJ presse presse av)
| @(V-SUBJ-PRT-OBJ presse presse igjennom)
| @(V-SUBJ-PRT-OBJ presse presse ut)
| @(V-SUBJ-PRT-OBJ presse presse inn)
| @(V-SUBJ-PRT-OBJ presse presse ned)
| @(V-SUBJ-OBJrefl presse presse)
| @(V-SUBJ-OBJ presse presse) }; ETC.`

The grammar has a coverage on unknown newspaper text of 50% (+ 30% with fragmented analysis), and the corresponding numbers for sentences shorter than 15 words are 65% (+ 30%). This means that the grammar has 95% coverage on sentences shorter than 15 words when fragmented analyses are included. There is ongoing work on treebanking, but still no numbers that show the number of parsed sentences that get the intended analysis.

4.9 Summary

This chapter has dealt with the specification of argument structure information in the lexicon. I started out by showing how argument structure information is specified

in HPSG, where syntactic arguments are listed on valence features like SUBJ and COMPS. This approach implies that the lexical entries have detailed information about the syntactic argument frame. Then I presented an alternative approach employed in the *Norsyg* grammar, where the valence lists are exchanged with four valence features, ARG1, ARG2, ARG3 and ARG4. I also introduced a type hierarchy of linking types, which makes it possible to capture the possible constellations of arguments that a verb can have (disregarding the category of the arguments) in one single type. I further showed how also the category of the arguments could be restricted, and gave several examples of types for verb lexical entries. I discussed different ways to expand the lexicon and compared the approach to the RASP system. In the last sections I compared the *Norsyg* grammar with the RASP system and the Norwegian projects TROLL, NorKompLeks, and NorSource.

This concludes the first part of the thesis, which has been focusing on argument structure and the representation of argument structure information in the lexicon. In the next part I will focus on syntactic structures, and how argument structure can be reduced to grammatical relations emerging from functional signs. The central idea is that the four subconstructions arg1-sign, arg2-sign, arg3-sign, and arg4-sign are realized by functional signs. These signs are a) valence rules (each role has a separate rule), b) function words (the passive auxiliary and the infinitival marker), and c) clitics (pronoun clitics) and inflections (the passive morpheme *-s*). I will lay out in detail how the argument structure information on these functional signs is represented, and how the information is checked with regard to the lexical requirements of the verb. This includes an explicit account of the basic syntactic structures in Norwegian, and completes the account of the strategy of argument frame packing.

Part II

The realization of argument structure in the syntax

Chapter 5

Methodology

In this second part of the thesis, I will show how argument structure can be realized by functional signs. By functional signs I mean syntactic rules, closed class lexical items, inflections and clitics. As for rules, I assume that there are syntactic rules associated with each of the subconstructions. I argue that the passive auxiliary and the infinitival marker are closed class lexical items that express subconstructions. I assume that the passive *s*-morpheme in Norwegian realizes a subconstruction, and I also assume that light pronouns express subconstructions.

Before I start discussing the implementation of these ideas in Norsyg in Chapter 6, I give an informal introduction to the general idea of how functional signs realize subconstructions and thereby form the argument frame of the clause. I give some simplified analyses of English sentences, where I argue that the argument frames emerge from the syntactic structures. The syntactic structures used in Section 5.1 are structures one would expect from an HPSG grammar, with mixed left- and right-branching (center-embedded) trees. In Section 5.2, I will present some motivation for purely left-branching tree structures, and in the remaining chapters syntactic structures will be assumed to be left-branching.

5.1 Preliminary analyses

In this section I present some preliminary analyses involving subconstructions. I assume four kinds of valence rules, one for each of the first four subconstructions.¹ The tree

¹As for the *arg5*-role, see footnote 1, page 87.

in Figure 5.1 reflects an analysis of a transitive sentence. The tree exposes two valence rules indicated by digits on the node labels. The rule VP2 combines the verb and the direct object. This rule realizes the arg2-role of the sentence.² The rule S1 combines the VP with the subject and realizes the arg1-role. By virtue of an arg1-role and an arg2 role being realized, the sentence has an arg12-frame.

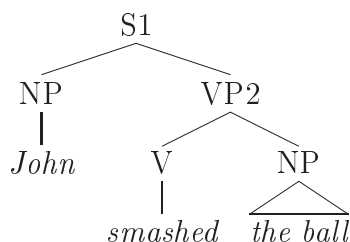


Figure 5.1: Analysis of a transitive active clause (BRR: D.1, p. 331)

Figure 5.2 shows an analysis of a ditransitive sentence. Here, three valence rules apply, the arg1-rule, combining the subject with the upper VP, the arg2-rule, combining the direct object with the lower VP, and the arg3-rule, combining the indirect object with the verb. This gives the sentence an arg123-frame.

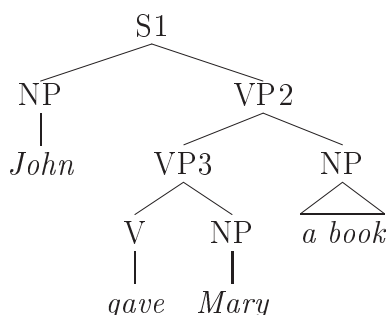


Figure 5.2: Analysis of a ditransitive active clause (BRR: D.2, p. 332)

Figure 5.3 shows an analysis of a transitive sentence with a delimiter (A delimiter is a resultative or a goal phrase. See Sections 3.1 and 3.2.4). Here, the arg1-rule (S1), the arg4-rule (VP4) and the arg2-rule (VP2) apply. That gives the sentence an arg124-frame.

Figure 5.4 shows an analysis of an unaccusative clause. In contrast to the previous analyses, the rule that combines the subject with the verb projection is an arg2-rule,

²The numbers on the nodes indicate that a syntactic entity expresses a subconstruction. When the arg1-role is realized, the node will have '1' attached to it, when the arg2-role is realized, the node will have '2' attached to it, and so on.

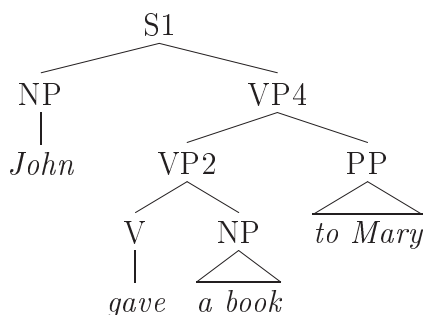


Figure 5.3: Analysis of a transitive active clause with a PP object (BRR: D.3, p. 332)

and not an arg1-rule. This illustrates that the valence rules are not necessarily linked to the grammatical function of the argument.

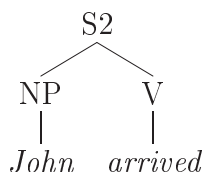


Figure 5.4: Analysis of an unaccusative clause (BRR: D.4, p. 332)

In passives, I assume that the passive auxiliary realizes the arg1-role, as illustrated in Figure 5.5. Here, the AUX1 (the passive auxiliary) realizes the arg1-role, and the S2, which combines the VP and the subject, realizes the arg2-role. As a result, the sentence has an arg12-frame, just like the active version in Figure 5.1.

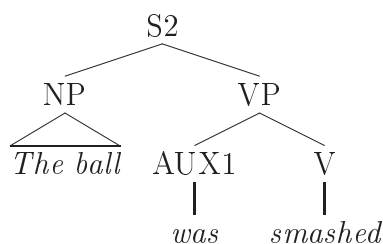


Figure 5.5: Analysis of a transitive passive clause (BRR: D.5, p. 333)

In infinitival clauses, I assume that the infinitival marker realizes a subconstruction. The subconstruction can be either the arg1-role, the arg2-role or the arg3-role.³ The

³The fact that the infinitival marker can realize different subconstructions, means that I have to assume three infinitival markers, or, alternatively, three unary rules that apply on the infinitival marker. This is discussed in Section 6.7.1. Assuming three infinitival markers instead of one may be seen as a

analysis in Figure 5.6 illustrates how the subconstructions in an active transitive infinitival clause are realized. Here, the infinitival marker realizes the arg1-role, and the rule that combines the verb with the direct object, realizes the arg2-role.

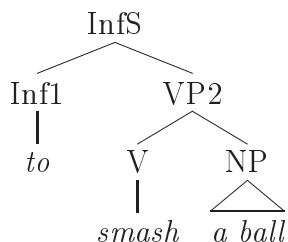


Figure 5.6: Analysis of an infinitival active clause (BRR: D.6, p. 333)

If the infinitival clause is a transitive passive clause, the infinitival marker realizes the arg2-role and the passive auxiliary realizes the arg1-role. This is illustrated in Figure 5.7.

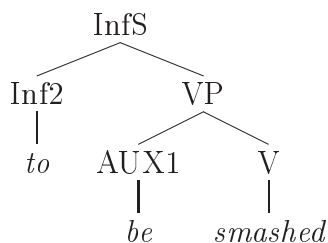
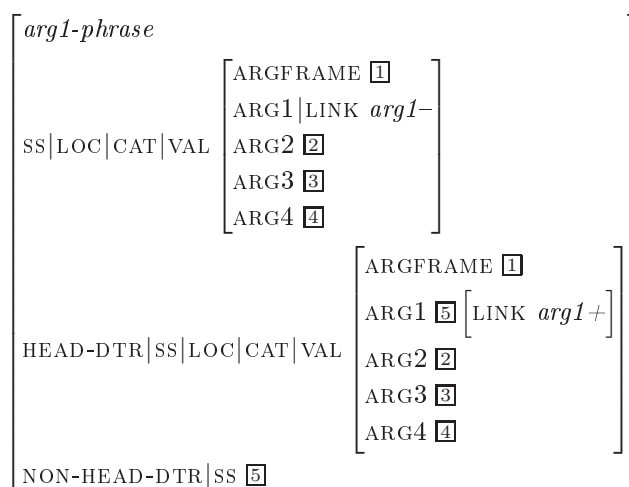


Figure 5.7: Analysis of an infinitival passive clause (BRR: D.7, p. 333)

Each of the syntactic items that realize a subconstruction will mark this by changing the LINK value of the relevant valence feature from $+$ to $-$. In the valence rules the head daughter has the positive value and the mother has the negative value. This is illustrated in Figure 5.8 where the arg1 valence rule shifts the ARG1|LINK value from $arg1+$ in the head daughter to $arg1-$ in the mother. The rest of the valence features are kept the same. (The linking types were introduced in Sections 3.5 and 4.3)

As for the passive auxiliary and the infinitival marker, they do not have a head daughter that they can relate their valence features to. Instead, it is assumed that they

drawback of the theory, similarly to the assumption of 8 valence rules rather than 2. It is a result of the exo-skeletal design of the system where it is the functional signs (including the infinitival marker) that build up the argument frame, and not the open lexical items. Adding complexity to the functional signs, rather than entering it in the open lexical items, is a deliberate choice. The number of functional signs is limited, while there is, in principle, no limit to the number of open lexical items. The result of a more complex open lexicon was shown in Section 4.6 in terms of parsing performance.

Figure 5.8: Valence constraints on the *arg1-phrase*

relate their VAL features to some VAL-B features, as illustrated for the passive auxiliary in Figure 5.9. The feature VAL-B is introduced in order to make it possible for a lexeme to be a subconstruction, even though it does not have a daughter. Instead of relating its valence values to its head daughter’s valence values, as valence rules do, a lexeme which is a subconstruction can relate its valence features to the values of VAL-B. A similar technique is employed by Riehemann (2001, 263–275), which in her account of derivational morphology lets a word relate its valence features (and also content) to the value of a feature MORPH-B, which functions as some sort of unrealized daughter.⁴ In Norsyg, it is only the passive auxiliary that is both a lexeme and a subconstruction at the same time (see Section 7.1). The function of the feature VAL-B is discussed in Sections 6.5, 7.1, and A.6.

I assume that all the LINK values are negative in the top node of a clause. This is enforced in the start symbols (*force-rules* (see Section 6.3)) and by all contexts for embedded clauses (*pop-rule* (see Section 6.6)). As the valence rules and the other syntactic items that express subconstructions apply, the negative link values are switched to positive values (from a top-down perspective). When all the syntactic items have applied, the valence information is gathered. In an active main clause, the information about realized subconstructions is available in the finite verb, as illustrated in Figure 5.10. Here, ARG1|LINK is switched from *arg1-* to *arg1+* from S1 to VP2. The

⁴The name VAL-B was chosen in order to show the analogy to Riehemann’s MORPH-B.

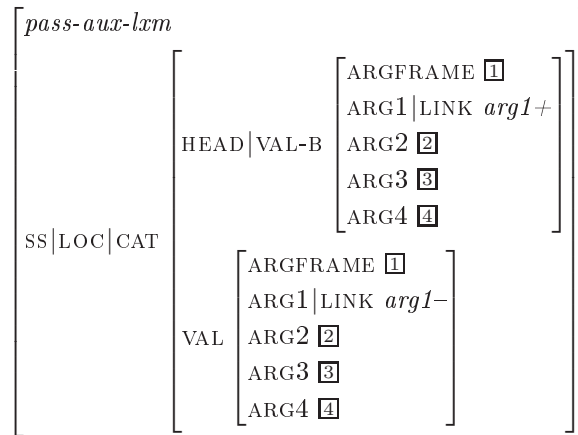


Figure 5.9: Valence constraints on the passive auxiliary

ARG2|LINK value is switched from *arg2-* to *arg2+* from VP2 to V.

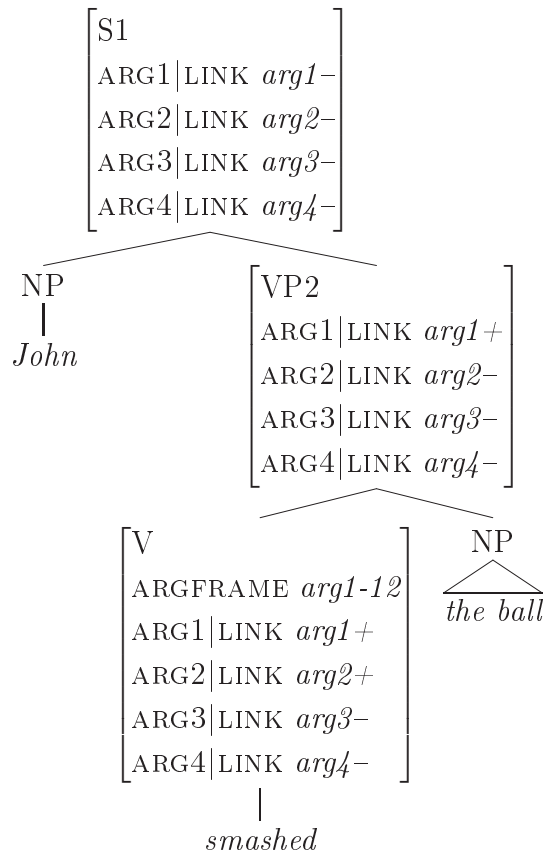


Figure 5.10: Information about realized subconstructions (repeated) (BRR: D.1, p. 331)

A mechanism, which I present in detail in Section A.6.1, makes sure that the values of the LINK features are unified and checked against the ARGFRAME value of the main verb. In the case of the sentence in Figure 5.10, the four values *arg1+*, *arg2+*, *arg3-* and *arg4-* in *V* are unified. This results in the argument frame type *arg12* (see Figure 4.9 p. 96).

5.2 Some remarks on syntactic structures

The syntactic structures that are assumed in this thesis are different from the structures standardly assumed in HPSG, LFG and GB/Minimalism. While the structures in these frameworks have the presupposition that the main verb is a head of a VP, the structures assumed in this thesis do not have this presupposition. Rather, the main verb may function more as a modifier of a syntactic structure headed by a functional element such as a complementizer or the infinitival marker. There are several considerations that motivate the structures assumed: Linguistic, cognitive and computational considerations. In the following sections, I will very briefly discuss these in turn.

5.2.1 Introductory remarks on tree structures

Before I get to considerations that motivate the syntactic structures assumed in this thesis, I will give some introductory remarks on syntactic tree structures. A tree structure reflects the way words combine into phrases and how phrases combine with words or phrases to form new phrases. A linguistic theory is to some extent reflected in how tree structures are built up. The tree in Figure 5.11 is uncontroversial, and is usually the kind of structures taught in introductory courses in linguistics (see eg. Borsley (1999, 38–51) and Carnie (2007, 63–80)). It employs two rules, one which combines the subject NP with the VP and forms a sentence ($S \rightarrow NP VP$), and one which combines the two complement NPs with the verb and forms a VP ($VP \rightarrow V NP NP$).

An alternative to syntactic tree structures as the one shown in Figure 5.11 are binary branching tree structures as shown in Figure 5.12 (Chomsky, 1981, 171). Here, the ternary rule from the tree in Figure 5.11 ($VP \rightarrow V NP NP$) is exchanged with binary rules. Binary structures are used in (later versions of) X-bar theory (Kayne,

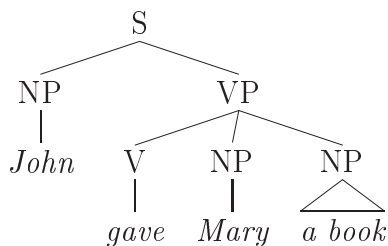


Figure 5.11: Conventional structure of ditransitive sentence

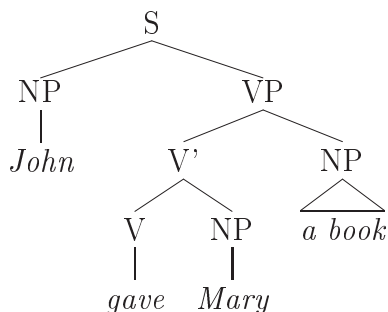
1984).⁵

Figure 5.12: Binary structure of ditransitive sentence

The binary structures also has a right-branching variant as the one illustrated in Figure 5.13. As the tree shows, such structures may have several V nodes. A motivation for assuming trees like these is that they can be processed incrementally, that is, word for word from left to right. They can also give better accounts of binding phenomena (see Culicover (1997, 364–373) and Carnie (2007, 375–380)). It is in particular data such as in (139) and (140) (from Culicover (1997, 365)) that motivate the right-branching structures. The examples show that an anaphoric direct object can be bound by the indirect object, but not the other way around.

- (139) a. I showed $Mary_i$ $herself_i$.
 b. * I showed $herself_i$ $Mary_i$.
- (140) a. I showed every worker $_i$ her $_i$ paycheck.
 b. * I showed its_i owner every paycheck $_i$.

Binding is accounted for by means of c-command in Principles and Parameters Theory. The data in (139) and (140) suggest that the indirect object c-commands

⁵Binary structures were not an assumption in the 70s, when X-bar theory came about.

the direct object. However, in structures as shown in Figure 5.12, the direct object is c-commanding the indirect object, and so they do not give the correct prediction. So-called Larsonian shells (Larson, 1988; Culicover, 1997) present a solution to the problem. They allow for several V nodes inside the VP, and the indirect object ends up c-commanding the direct object as shown in Figure 5.13. The verb is here assumed to have moved from the lower V to the upper V.⁶

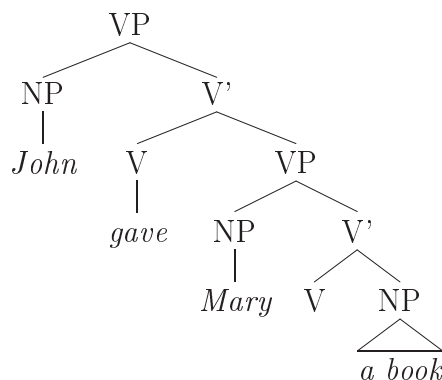


Figure 5.13: Right-branching tree structure

Tree structures in this thesis are assumed to be uniformly left-branching, as illustrated in Figure 5.14. The subject combines with the verb before the complements and the adjuncts in a bottom-up left-to-right fashion.⁷

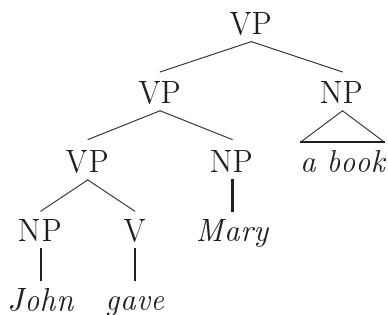


Figure 5.14: Left-branching tree structure (BRR: D.2, p. 332)

⁶The desired c-command may also obtain in Figure 5.11, where the indirect object and the direct object are sisters. This would however require extra order constraints to prevent the direct object from c-commanding the indirect object.

⁷The node label VP simply means that the syntactic head is a verb and that it is a phrase. It is not a VP in the sense of constituting a verb and the complements of the verb. As I will come back to in Chapter 6, the start symbol is one of three unary rules. It is not included in the tree in Figure 5.14, and so the top node is a VP, and not an S.

As mentioned, the exo-skeletal nature of the analyses allows for syntactic structures where the main verb may function as a modifier of a syntactic structure headed by an auxiliary, a complementizer, or an infinitival marker.⁸ The result is that subordinate clauses have the complementizer as a syntactic head, and that the arguments in the clause combine with the complementizer projection instead of the verb projection. A clause with a subordinate clause complement has the structure shown in Figure 5.15.

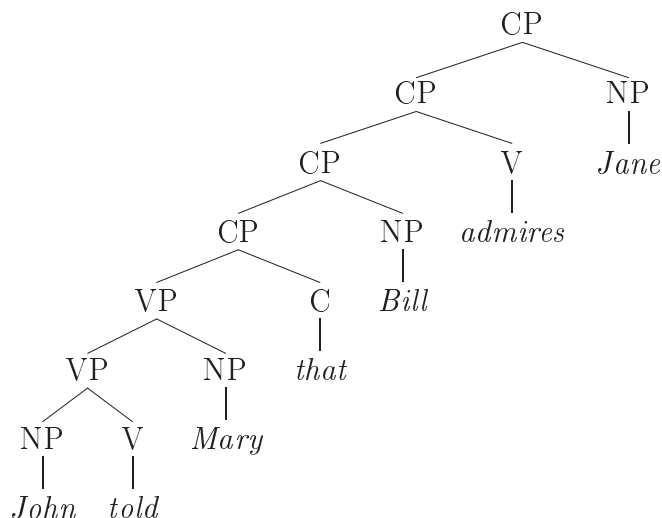


Figure 5.15: Left-branching tree with a subordinate clause (BRR: D.8, p. 334)

Here, the complementizer is the syntactic head of the upper part of the tree. It attaches to the phrase ‘John told Mary’ and forms a phrase where the complementizer is the syntactic head (CP). This is done by means of the *binary complementizer rule* (see Section 6.6, and Figure 6.37, p. 176 in particular). The binary complementizer rule attaches a complementizer to a matrix clause constituent preceding it, and initiates a subordinate clause, headed by the complementizer. The arguments *Bill* and *Jane* and the verb *admires* combine with the projection of the complementizer. The arguments are combined by means of *valence rules* (see Section 6.1), and the verb is combined by means of the *merge rule*, which combines non-head verbs to the head projection (see Section 6.5).

⁸What I refer to as the head in this thesis is the *syntactic* head, and not the *semantic* head. What corresponds to the semantic head is the value of the feature `HOOK`. `HOOK` is a bundle of features that is used to access the top handle, the index, and the external argument of a constituent. (See Copetake *et al.* (2005, 16-29).) This is illustrated in the analyses shown in Figures 6.16, 6.22, and 6.27, where the `HOOK` value of the main verbs is projected to the top of the clauses.

The structures of these trees resemble certain structures in CCG, where type-shifting of NP subjects together with backwards formation allows a subject to combine with the verb before the object (see Steedman (2000, 43–49)). However, while CCG allows for several possible surface structures for a sentence (and applies mechanisms such as type-raising and backwards formation to arrive at the left-branching structure), there is only one possible structure in the analysis presented in this thesis.

The rules employed in the trees in this section have all been *phrase structure rules*. A phrase structure rule is a rule of the form $A \Rightarrow B C$, which says that the constituent A can be separated into the subconstituents B and C . *Phrase structure rules* and configurations of them are closely connected to the GB tradition, where they have several theoretical implications such as the existence of a VP (a constituent consisting of the main verb and its complements), and structural relations holding between structural heads and their arguments (government) and between antecedents and anaphors (binding). Even though phrase structure rules can be reduced to a mechanic tool for syntactic combination, I have avoided using the term in this thesis because of the theoretical connotations. Instead I use the term *syntactic rules*.

5.2.2 Linguistic considerations

Basic clause structure and sentence adverbials

In the previous section I presented the assumption that a verb in a subordinate clause does not head a VP, but that it rather attaches to a complementizer projection and functions like an (obligatory) modifier. This gives a uniform treatment of the position of sentence adverbials in main clauses and subordinate clauses in Norwegian. Norwegian is generally assumed to have two clause patterns, one for main clauses and one for subordinate clauses. In main clauses, sentence adverbials appear after the finite verb (see (141a) and (141b)), and in subordinate clauses the sentence adverbials appear before the finite verb (see (141c)).

- (141) a. Jon ser ikke Kari.
 Jon sees not Kari
 ‘Jon doesn’t see Kari.’

- b. Jon har ikke sett Kari.
 Jon has not seen Kari
 ‘Jon hasn’t seen Kari.’
- c. at Jon ikke har kommet
 that Jon not has come
 ‘that Jon hasn’t come’

In HPSG and LFG, one is forced to assume separate modifier rules for the two clause patterns. This is because the theories presuppose that the finite verb is the head, and that the verb cannot move. So since the sentence adverbial occurs after the finite verb in main clauses and before the verb in subordinate clauses, two rules are needed.⁹ In Principles and Parameters, the verb can move to a position preceding the sentence adverbial in main clauses, and there is only one position for the sentence adverbial (see Åfarli and Eide (2003, 71–77)). In the analysis presented in Chapter 10, the exo-skeletal approach makes it possible to account for the position of sentence adverbials with one rule (and no movements). The sentence adverbial is assumed to attach to the head of the clause from the right. Since the head is the complementizer in subordinate clauses and the finite verb in main clauses, only one rule is needed. The analysis also includes a treatment of light pronouns in Norwegian.

Long distance dependencies

The left-branching structures, where the first constituent appears at the bottom left corner of the tree are motivated by some data involving long distance dependencies. As pointed out in Bouma *et al.* (2001), a large range of languages have elements that intervene the filler and the gap in a long distance dependency, and access the information that a constituent is extracted (see Section 6.9). These elements occur only on an *extraction path* Bouma *et al.* (2001, 1). Since the filler rule is at the top of the tree in HPSG, LFG and Principles and Parameters, the information that a constituent is extracted is actually only available in parts of the structure that do *not* intervene the gap and the filler.¹⁰ One is forced to introduce additional mechanisms that let verbs

⁹A version of HPSG that uses Schemata rather than phrase structure rules (Pollard and Sag, 1994) could use a single Schema with no Linear Precedence constraints, which would allow the adverbial to appear on either side of the verb.

¹⁰This claim does not hold for right-branching structures as shown in Phillips (2003).

have access to the gaps of their arguments (and adjuncts) (see Bouma *et al.* (2001)), so that the elements that reflect that they occur on the extraction path also have access to the information. In the analysis that I will present in Section 6.9, I will assume that the filler rule is at the bottom of the tree, rather than on the top. By having the filler rule at the bottom of the tree, the information that a constituent is extracted, will be accessible locally to the elements that reflect that they occur on the extraction path and nowhere else.

Inversion

Topicalization and yes-no-questions involve inversion, which means that the subject is realized to the right of the finite verb. This is illustrated in (142a) (topicalization) and (142b) (yes-no-question). In both examples the subject *Kari* is realized after the finite verb *leste*. In HPSG and LFG, inversion is accounted for either by means of special subject rules that realize the subject to the right, or by means of a lexical rule that moves the subject from the SUBJ list to the COMPS list. Neither of these operations seem to be motivated by other phenomena. In P&P, inversion is accounted for by means of verb movement, i.e. the (finite) verb moves out of the VP to receive tense, and the subject stays behind in the spec of V.

- (142) a. I går leste Kari en bok.
 In yesterday read Kari a book
 ‘Yesterday Kari read a book.’
- b. Leste Kari en bok?
 Read Kari a book
 ‘Did Kari read a book?’

Norwegian is a V2 language, and in the approach presented in this thesis, the element that comes before the finite verb in main clauses is assumed to always be extracted. This assumption also holds for sentence-initial subjects.¹¹ As a result, argument rules

¹¹The constituent that comes before the finite verb in main clauses has had a particular status in Scandinavian syntax since Diderichsen’s field analysis (Diderichsen, 1946), who refers to it as “Fundamentet” (The Fundament). Fundamentet is, according to Diderichsen, “usually the entity from which the sentence originates, or upon which it is built,” and “almost any constituent (except the finite verb) can take this position.” Diderichsen (1946, 185) (my translation). In GB it is generally assumed that the constituent occurring in the position before the finite verb in a main clause has moved to this position (Spec of C) (see Holmberg and Platzack (1995)).

are always head-initial. By assuming that sentence-initial arguments are extracted and that argument rules are head-initial, the inverted structures come as a consequence. For topicalization, if some constituent other than the subject is extracted, the subject must appear after the finite verb (which is the head), since the argument rules are head-initial. For yes-no-questions, there is no extraction taking place, so all arguments have to be realized after the finite verb. There is no need for extra rules or verb movement.¹²

5.2.3 Cognitive considerations

The notion of *incremental processing* is standard in the psycholinguistic literature (see, for example, Kempen and Hoenkamp (1987) and Levelt (1989)), and evidence is presented that shows that humans process language incrementally, that is, in the order in which linguistic material is heard or read. The assumption made in this thesis that the filler is at the bottom of the tree, and that arguments attach in a bottom-up fashion (from left to right) is compatible with the notion of incremental processing.

Another important notion is that of *syntactic flexibility* (Ferreira, 1996). Ferreira demonstrates that verbs that can appear in several syntactic argument frames (exhibit syntactic flexibility) like the verb *give* in (143) and (144) (taken from Ferreira (1996, 725)) are not more difficult to produce than verbs that are less flexible like *donate* in (145) and (146) (Ferreira, 1996, 726).

(143) Sheila gave the toys to the children.

(144) Sheila gave the children the toys.

(145) Sheila donated the toys to the children.

(146) *Sheila donated the children the toys.

Ferreira presents two models. His first model, *the competitive model*, has one lemma for each syntactic structure in cases of syntactically flexible verbs. It predicts that sentences with this kind of verbs are more complex, and therefore more difficult to produce. The second model, *the incremental model*, lets the syntactic structure be built while the utterance is produced. Incremental theories imply that syntactic structures are not set from the outset, but rather that the syntactic structures are selected as

¹²The assumption of no extraction in yes-no-questions (as well as conditional clauses with subject inversion and imperative clauses) corresponds to the assumption of an empty Fundament field by Diderichsen (1946, 191).

the utterance is produced. If the speaker has the choice between two constructions, the argument that is most active is used first, and the syntactic structure that is compatible with this choice is selected (Ferreira, 1996, 728). The incremental model predicts that utterances with flexible verbs are easier to produce than non-flexible verbs. In three experiments he shows that utterances with flexible verbs like *give*, with no conditions on what syntactic structure to use, are easier to produce than utterances with non-flexible verbs like *donate*, and offer support to the incremental model.¹³

Although the topic in Ferreira’s article is language production and not parsing, the central question is the same: Is syntactic structure present in words, that is, do we have to select a particular syntactic structure when we parse a word (the ‘competitive’ model), or is the syntactic structure something that is selected as an utterance is parsed (the ‘incremental’ model)? In this thesis I show that verbs can be lexically very flexible (see e.g. the verb *drip* on page 78), and I argue in correspondence with Ferreira’s incremental theory “that syntactic structures are slots that are *available* to be filled, rather than *active* plans that influence non-syntactic processing” (Ferreira, 1996, 728).

5.2.4 Computational considerations

In the LKB grammar engineering system (Copestake, 2002), which the grammar presented in this thesis is implemented with, and the vast majority of current unification-based parsing research, search strategies work predominantly bottom-up. Several authors argue that pure bottom-up parsers are psychologically implausible since they cannot parse incrementally (see Abney (1989) and Crocker (1996)). In a bottom-up parser, the lowest node is parsed first, and given a right-branching tree structure, which

¹³In the first two experiments, participants were instructed to form sentences that contained alternator verbs like “give” and non-alternator verbs like “donate” with some selected arguments. In half of the cases, the order of the arguments was constrained, either by adding a preposition, which excludes the use of the double object construction (experiment 1), or by using a pronoun which cannot be the theme of a double object construction (experiment 2). The results were measured with regard to number of errors and latency. In experiment 1 the participants produced sentences with alternator verbs, where the order of the arguments was not constrained, with reliably fewer errors than sentences with non-alternator verbs and sentences with obligatory prepositions. The unconstrained cases were produced reliably faster than the order-constrained cases. Experiment 2 showed that the participants produced sentences with flexible conditions reliably faster than sentences with non-flexible conditions. In experiment 3 the participants produced active and passive sentences. Case marking was used to add constraints (non-flexibility) on the possible productions in some of the tests. The experiment showed that syntactic flexibility made the production of passive sentences more efficient. All the results from the experiments give support to the incremental model.

in Phillips (2003) is presented as the best way to do incremental parsing, and a parser that works in a left-to-right fashion, which is compatible with incremental processing, the whole string has to be read before processing can begin. This is because the last word will be the lowest node. The argumentation does not hold if tree structures are assumed to be left-branching, as I do in this thesis. Then the first word, and not the last, will be at the bottom of the tree, and incremental parsing is possible in principle.

Crocker (1996) characterizes pure bottom-up parsers as psychologically implausible since “adjacent constituents may be left on the stack for an arbitrary long period” (page 14). He exemplifies this with the NP in a rule $S \rightarrow NP VP$, where the NP cannot attach to the VP before the whole VP is parsed.¹⁴ A top-down parser may be conceived of as psychologically more plausible since it allows for incremental parsing. However, the top-down method also has problems, namely that it “attempts to construct large portions of the tree before even looking at the words in the sentence” (page 14). This makes the parser do lots of hypothesizing about possible structures before it reaches the input. Left-recursive rules (eg. $VP \rightarrow VP PP$) will for example make naive top-down parsers enter infinite loops. So, while the bottom-up parser is input-driven but non-incremental, the naive top-down parser is non-input-driven but incremental. Crocker presents the “Left-Corner Algorithm” (see Johnson-Laird (1983, 296–309)) as the psychologically plausible alternative to the pure bottom-up or top-down algorithms. It combines features from both bottom-up and top-down parsing and is incremental and data-driven at the same time. Crocker writes: “The central intuition behind the left-corner algorithm is to use the ‘left-corner’ of a phrase structure rule (the left-most symbol on the right-hand side of the rule, i.e. the left-most daughter of a category), to project its mother category (the left-hand side of the rule), and predict the remaining categories on the right, top-down” (page 15). Given a right-branching tree, this yields a data-driven incremental parser. The method is however not guaranteed to be incremental. If the structure is not completely right-branching, the parser will delay building a completely connected structure.

The application of the Left-Corner Algorithm on right-branching structures can be compared with the approach taken in this thesis where left-branching structures are parsed bottom-up. Given that bottom-up parsers work in a left-to-right fashion as outlined in Steedman (2000, 229–246), both approaches can be said to be data-driven

¹⁴This is, as already mentioned, not applicable to the analyses presented in this thesis, since the filler is realized at the bottom of the tree, rather than at the top as Crocker presupposes.

and incremental.¹⁵ One difference is the predictive top-down aspect of the Left-Corner Algorithm which presupposes that the root node is known from the outset. In the left-branching bottom-up approach, this requirement is not present. The resulting tree can be an NP, an S, or any other structure that is licenced by the grammar. A pure bottom-up parser does not have any top-down restrictions, and so subtrees that do not become a constituent of a sentence can be built. While this property is often of practical benefit in language engineering, its theoretical status can hardly be discussed conclusively without reference to a complete theory of sentence processing (and its specific assumptions), an endeavour well beyond the scope of this thesis.

It has been pointed out by Resnik (1992) that the *type-raising* mechanism in CCG (see Steedman (1990, 13–14) and also Steedman (2000, 43–49)) shows some resemblance with a left-corner parser. In both approaches constituents are created, which are still to realize something. In an approach which assumes a right-branching syntax and uses a left-corner parser, a constituent can be formed that consists of the subject and the verb, and that has the arguments that belong under VP on its stack. If the verb is transitive, the stack will contain an NP (see Johnson-Laird (1983, 308)). In CCG, the subject NP can be type-raised and then form a constituent with the verb by backward formation. The new constituent will have the same rightwards saturation requirements as the verb, and the leftward (subject) requirement will be gone. So if the verb is transitive, the new constituent will require an NP to its right in order to become an S (see Steedman (2000, 45)). Constituents formed by, for example, the subject and the verb in the approach presented in this thesis are not “incomplete” in the way that the structures in left-corner parsing and CCG are, where a part of the constituent is yet to be parsed. In the approach taken in this thesis, the subject and the verb are assumed to be a “regular” constituent (given that the clause is a main clause with canonical word order).

The syntactic structures that are assumed in this thesis, have the topicalized element at the bottom of the tree, and it will always be the case that the extraction site dominates the filler. This, in addition to the fact that the syntactic structures are left-branching, means that a constituent will always be explicit with regard to whether it appears on the extraction path. The extraction is done by means of unary extraction

¹⁵If NPs consisting of more than one word are assumed to be constituents (and they are in this thesis), the Left-Corner Algorithm will have to stack more than one category when non-final NPs are parsed. Similarly, a bottom-up parser working in a left-to right fashion will have to build edges that are intermediately unconnected, when non-initial NPs are parsed.

rules, which simulate the existence of a trace. In other HPSG implementations, the filler is realized at the top of the tree, and unary extraction rules are tried out at every node that could be an extraction site whether a constituent actually is extracted or not. This creates many sub-trees in the parse chart that never lead to a result. In the approach taken in this thesis, the extraction rules will *only* apply when a long distance dependency evidently is taking place, that is, when a constituent is filled in at the bottom of the tree, or when a relative pronoun (possibly empty) has introduced a relative clause. This is especially beneficial in terms of computation when applied to V2 languages like Norwegian.

In this thesis I make the assumption that flexible verbs have the potential for entering several syntactic structures (the incremental model) (see Chapter 4), rather than equipping verbs with ready-made syntactic structures from the beginning, that is, using multiple lexical entries or lexical rules to make the syntactic structure explicit at lexeme level (the competitive model). This reduces the number of nodes in the parse chart considerably (see Section 4.6, in particular Figure 4.4, page 113). In the competitive model, a large range of subtrees will be built that build on lexical entries that are rejected before the parse is complete. This does not happen in the incremental model (apart from cases of *real* ambiguity), which posits only one lexical entry per word.

5.3 Summary

In this chapter I have presented preliminary outlines of basic syntactic structures, and I have discussed left-branching and right-branching tree structures. (I also mentioned mixed left- and right-branching (center-embedded) tree structures.) I have presented linguistic, cognitive, and computational motivation for using left-branching tree structures.

Abstracting away from parsing techniques, the approach I am presenting in this thesis has certain similarities to (Scandinavian) P&P, as I will discuss further in Chapter 9. First, the constituent that appears in the position before the finite verb in matrix clauses, has ‘moved’ there from its canonical position.¹⁶ Second, both approaches have

¹⁶In my approach, constituents do not move for real. A long distance dependency between the ‘moved’ constituent and the canonical position is represented by means of unification of constraints on the ‘moved constituent’ with constraints on the unary extraction rules (see e.g. the tree in Figure

syntactic structures that allow for incremental parsing (P&P analyses with Larsonian shells).

One main difference between the two approaches is that there is only one kind of ‘movement’ in the approach presented in this thesis, namely what in P&P is movement to Spec of C in matrix clauses and relative clauses. No other movements are necessary.

The rest of the thesis will focus on a grammar formalism where argument structure is reduced to grammatical relations realized by functional signs. I show in detail how this can be accomplished for Norwegian in the grammar implementation Norsyg. I have chosen to be explicit to such a degree that a moderately experienced grammar writer should be able to implement a grammar in the same fashion.¹⁷

6.41, p. 178).

¹⁷It is possible to download Norsyg and parse example sentences with it while reading this part of the thesis. Download instructions are given in Appendix A. The grammar directory contains files with test sentences. The files ‘ex.items’ and ‘eng-ex.items’ contain the Norwegian test sentences and the English test sentences in the thesis (see also Appendix C.1).

Chapter 6

Basic syntactic structures in Norwegian

In this chapter I will present an account of the basic syntactic structures of Norwegian. I will take up the thread from Section 3.5, where the hierarchy of subconstructions was introduced, from Section 4.3, where valence in Norsyg was introduced, and from Section 5.2, where left-branching tree structures were argued for.

HPSG grammars usually operate with a Head-Subject Rule, a Head-Complement Rule, a Head-Modifier Rule and a Head-Filler Rule to account for the basic structures of clauses. In Norsyg I employ rules that are not associated with the function of the non-head daughter in the way that the Head-Subject and the Head-Complement rules are. In order to account for the basic structures of Norwegian clauses, six kinds of rules are central:

1. The **valence** rules, which realize arguments and link them to the predicate.
2. The **filler** rule, which fills in the extracted constituent.
3. The **merge** rule, where (non-head) verbs merge their information with the head projection.
4. The **subordination** rules, where embedded clauses are entered.
5. The **clause boundary** rules, which mark the boundary of the clauses.
 - (a) The *force* rules for main clauses.

(b) The *pop* rule for embedded clauses.

6. The **modifier** rules.

I will explain the rules and show how they together account for clause structure in main clauses, yes-no questions, subordinate clauses, relative clauses, infinitival clauses and small clauses. I will show how the subconstructions presented in Section 3.5 relate to the different rules, and how linking is achieved. I will also discuss long distance dependencies, modification, and raising and control verbs.

6.1 The valence rules

In Section 3.5 the type *subconstruction* was introduced with some of its subtypes, including *basic-val* (see Figure 3.8, p. 85). The definition of *subconstruction* is repeated in Figure 6.1. In this section, I will look at the subtypes of *basic-val*, which are the valence rules of the grammar.

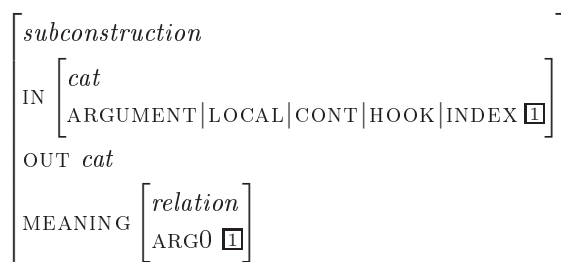


Figure 6.1: The type *subconstruction*

The type *basic-val* (see Figure 6.2) is a general type for valence phrases. It unifies the value of IN with the value of CAT of the head daughter. The value of OUT is unified with the value of CAT of the mother. The value of MEANING is unified with the element on the C-CONT|RELS list. The handle of the relation is unified with the LTOP value.

When the constraints from the supertype *subconstruction* are added, the type *basic-val* has the constraints shown in Figure 6.3.

Figure 6.3 shows that valence rules introduce a relation in C-CONT which links the argument to the predicate. The LBL value of the relation in C-CONT is unified with the value of LTOP, which again will be linked to the relation introduced by the main verb. The ARG0 value of the relation in C-CONT is unified with the index of the argument.

$$\left[\begin{array}{l} \textit{basic-val} \\ \text{SS|LOC} \left[\begin{array}{l} \text{CAT} \boxed{1} \left[\text{HEAD} \boxed{2} \right] \\ \text{CONT|HOOK|LTOP} \boxed{3} \end{array} \right] \\ \text{HEAD-DTR|SS|LOC|CAT} \boxed{4} \left[\text{HEAD} \boxed{2} \right] \\ \text{C-CONT|RELS} \left\langle \boxed{5} \left[\text{LBL} \boxed{3} \right] \right\rangle \\ \text{IN} \quad \boxed{4} \\ \text{OUT} \quad \boxed{1} \\ \text{MEANING} \quad \boxed{5} \end{array} \right]$$

Figure 6.2: Definition of *basic-val*

$$\left[\begin{array}{l} \textit{basic-val} \\ \text{SS|LOC} \left[\begin{array}{l} \text{CAT|HEAD} \boxed{1} \\ \text{CONT|HOOK|LTOP} \boxed{2} \end{array} \right] \\ \text{HEAD-DTR|SS|LOC|CAT} \left[\begin{array}{l} \text{HEAD} \boxed{1} \\ \text{ARGUMENT} \left[\begin{array}{l} \textit{synsem} \\ \text{LOC|CONT|HOOK|INDEX} \boxed{3} \end{array} \right] \end{array} \right] \\ \text{C-CONT|RELS} \left\langle \left[\begin{array}{l} \text{LBL} \boxed{2} \\ \text{ARG0} \boxed{3} \end{array} \right] \right\rangle \end{array} \right]$$

Figure 6.3: Constraints on the type *basic-val*

As I pointed out in Section 3.5, the subconstructions *arg1-sign* – *arg4-sign* have a formal contribution (switching a LINK value from + to –) and a meaning contribution (a Parsons-style underlying event). The definition of *arg1-sign* is repeated in Figure 6.4 (without the unification of the other valence features).

$$\left[\begin{array}{l} \textit{arg1-sign} \\ \text{IN} \left[\begin{array}{l} \text{VAL|ARG1} \boxed{1} \left[\text{LINK} \quad \textit{arg1+} \right] \\ \text{ARGUMENT} \boxed{1} \end{array} \right] \\ \text{OUT|VAL|ARG1|LINK} \quad \textit{arg1-} \\ \text{MEANING} \quad \textit{arg1_rel} \end{array} \right]$$

Figure 6.4: Abbreviated definition of *arg1-sign*

When the constraints of *arg1-sign* and *basic-val* are unified in *arg1-val*, we get a sign with the constraints shown in Figure 6.5. Here, the mother has the LINK value *arg1-*

and the head daughter has the LINK value $arg1+$. The C-CONT has the $arg1$ -relation. There is a basic valence rule type for each subconstruction ($arg1$ -val – $arg4$ -val, see the hierarchy in Figure 3.8, p. 85).

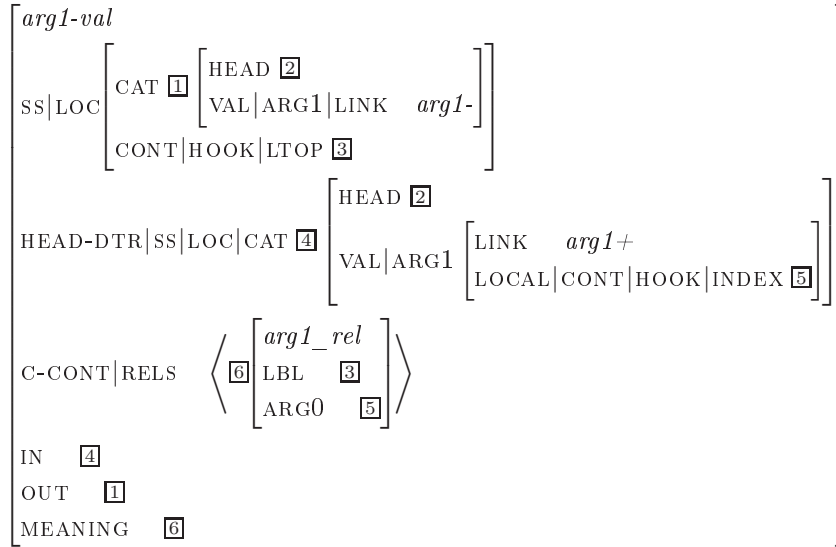


Figure 6.5: Constraints on $arg1$ -val

Each valence rule type has a binary variant and a unary extraction variant. The hierarchy of valence phrases is given in Figure 6.6.

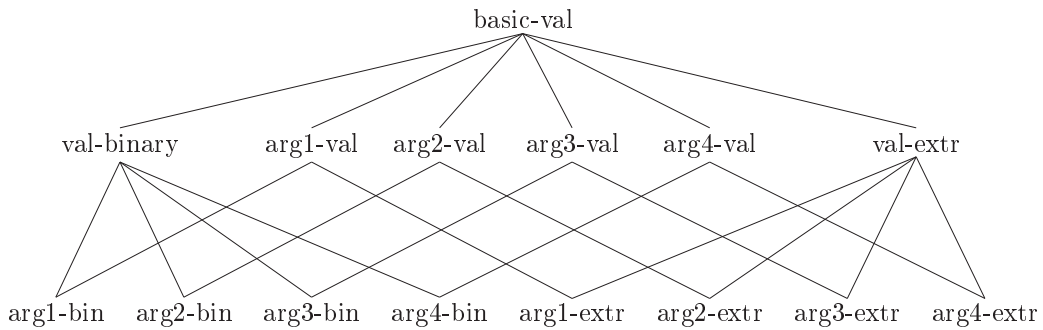


Figure 6.6: Hierarchy of valence phrase types

The top type in the hierarchy in Figure 6.6 is $basic$ -val and it has six immediate subtypes, val -binary, $arg1$ -val, $arg2$ -val, $arg3$ -val, $arg4$ -val and val -extr. The bottom types are cross-classifications of the types $arg1$ – $arg4$ -val with the types val -binary and val -extr.

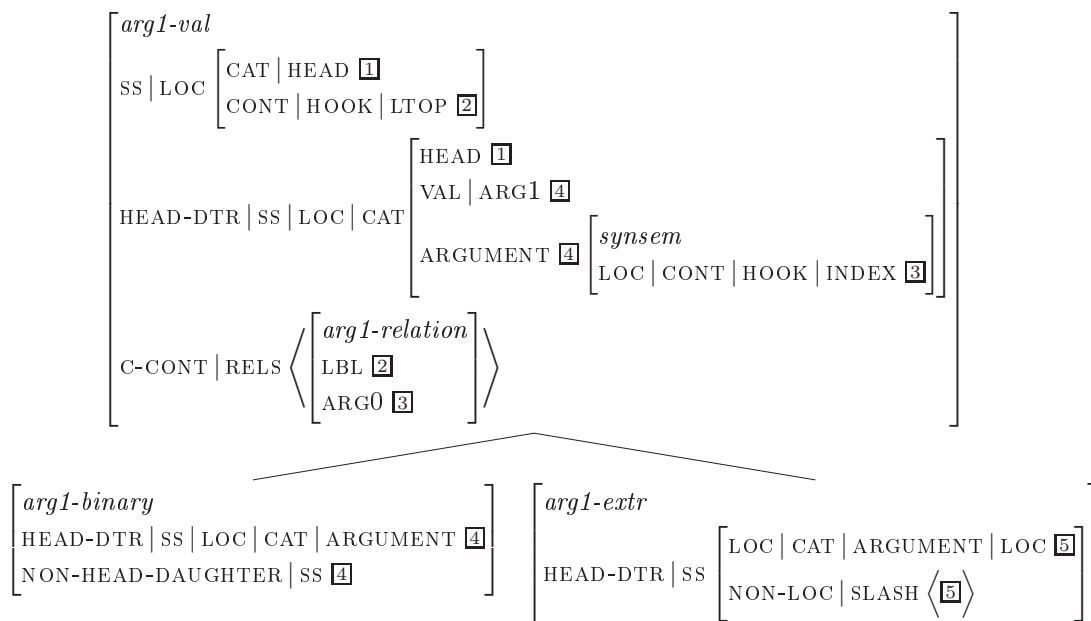
Figure 6.7: The *arg1-val* hierarchy

Figure 6.7 illustrates how *arg1-val* generalizes over the *arg1-binary* and the *arg1-extr* phrases. Only information specific to the types is specified in the subtypes. In the *val-binary* type the ARGUMENT value is unified with the NON-HEAD-DTR|SYNSEM and in the *val-extr* type the ARGUMENT|LOCAL value is unified with the element on the SLASH list.¹

6.2 The filler rule

The filler rule is the rule that fills in the extracted element of a main clause. It is a head-final rule which applies at the bottom of the tree. Given the left-branching tree structures in this approach, the filler rule will get the extracted constituent from above. The rule is illustrated in Figure 6.8.

As Figure 6.8 shows, the head filler rule unifies the element on the SLASH list of the

¹The SLASH list is a list that keeps track of extracted elements. If for example an NP is extracted, syntactic and semantic information about this NP (represented in the type *local*) enters the SLASH list. Then this information is transported down the tree until a filler rule realizes the topicalized NP. I come back to a detailed account of long distance dependencies in Section 6.9. Note that the value of SLASH is a *list*, and not a *difference list*, as in other grammars based on the Grammar Matrix (Bender *et al.*, 2002).

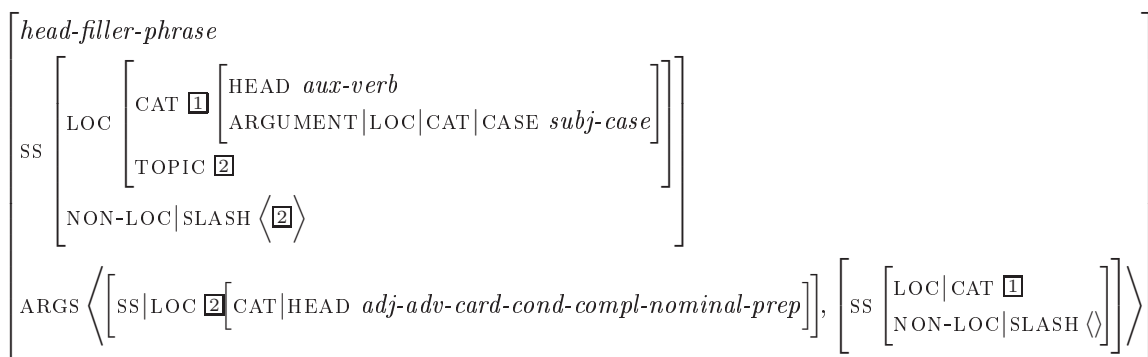


Figure 6.8: Constraints on the head filler rule

mother with the value of LOCAL of the first daughter. It also unifies the slashed element with the value of TOPIC.² The head value of the phrase is *aux-verb*, which means that it is either an auxiliary or a main verb. The head value of the filler is *adj-adv-card-cond-compl-nominal-prep*, which means either adjective, adverb, cardinal, conditional, complementizer, nominal or preposition. The SLASH list of the head daughter is empty.

6.3 The force rules

The next set of rules are the *force rules* which are used for marking the boundary of the sentence and constraining the event to say what kind of sentence it is. They are unary rules that apply at the top of the tree. I here present three force rules:

1. The **main-rule** constrains the event to be a proposition or a wh-question.
2. The **yes-no** rule constrains the event to be a yes-no-question.
3. The **imperative** rule constrains the event to be a command.

The information specified on the force rules is given the type hierarchy in Figure 6.9. Notice that all the valence features of the daughter are specified to have negative linking types. This means that all arguments of the sentence must be realized when the force rules apply. The function of the features MERGE and STACK I will return to in Section 6.5 and Section 6.6, respectively.

²The function of the feature TOPIC is to have a pointer to the extracted element. This is necessary in my analysis of coordinated VPs (see Section 8.1).

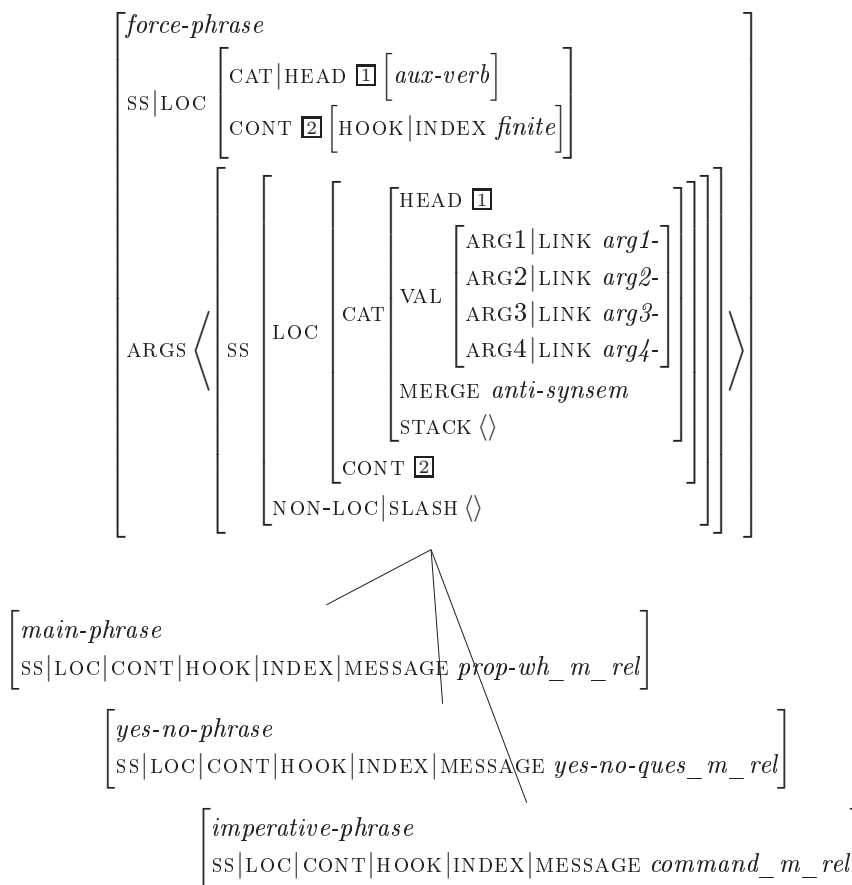


Figure 6.9: Hierarchy of force phrases

6.4 Some simple analyses

Given the rules introduced in Sections 6.1, 6.2 and 6.3 we can start analyzing simple sentences. In the analyses of Norwegian clauses I assume that the valence rules apply in a fixed order dependent on the case of the argument. The argument with subjective case will always come first. Then they appear in the order $\text{arg1} > \text{arg3} > \text{arg2} > \text{arg4}$.³ The way this order is fixed is described in Appendix A.6.2. In main declarative clauses and wh-questions I assume that the sign preceding the finite verb is always extracted.⁴

³The ARG4 argument may appear before the ARG2 argument, in particular if the ARG2 argument is a subordinate clause as in *Han foreslo for meg at jeg kunne studere medisn* ('He suggested to me that I could study medicine').

⁴This procedure is discussed for HPSG in Pollard and Sag (1994, 381) and is applied for Norwegian in Ellingsen (2003). (See also footnote 11, page 143.)

So in the case of an intransitive sentence the structure is as in Figure 6.10.⁵ The filler rule applies first (VP/NP), then the extraction rule (VP1), and finally, the force rule (S). In yes-no questions, which have the finite verb in the first position, there is no extraction (see Figure 6.11). The subject is realized after the main verb (VP1), and the *yes-no-rule* applies on the top (S).

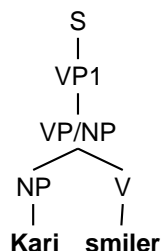


Figure 6.10: Intransitive main clause (BRR: D.9, p. 335)

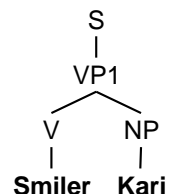


Figure 6.11: Intransitive yes-no clause (BRR: D.10, p. 335)

Transitive and ditransitive main clauses are analyzed as in Figure 6.12 and 6.13 with the verbs *beundre* ('admire') and *gi* ('give'). In a main clause with unmarked word order, the subject is extracted before the other arguments are combined.

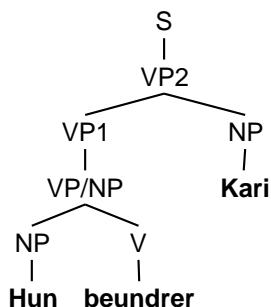


Figure 6.12: Transitive main clause (BRR: D.11, p. 335)

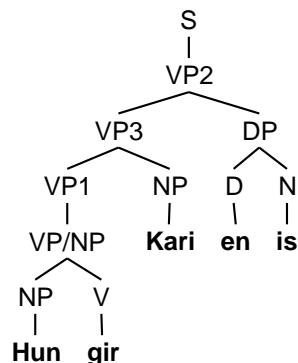


Figure 6.13: Ditransitive main clause (BRR: D.12, p. 336)

⁵The trees with boldface terminals are parsed with the LKB system loaded with Norsyg (except from the tree in Figure 6.24, where the ERG has been used). The labels reflect the head value, i.e V or VP if the head value is *verb*. If there is an element on the SLASH list, this is represented with for example VP/NP if the slashed element is an NP. If a rule realizes a subconstruction, this is shown with a number indicating what kind of subconstruction it is. So a valence rule that has the HEAD value *verb* and realizes an arg1 subconstruction is represented as VP1.

6.4.1 Analysis of a transitive sentence

The linking information in a transitive sentence is illustrated in detail in Figure 6.15. The head daughter (VP2) of the main rule at the top of the tree is constrained to have only negative linking types. Then for each valence rule that applies, the corresponding linking type is shifted from negative to positive. So the head daughter of VP2 (VP1) has the type *arg2+* as value of ARG2|LINK. The head daughter of VP1 (VP/NP, the head filler rule) has the type *arg1+* as value of ARG1|LINK. The head filler rule unifies its linking information with the head daughter (V). (The unification of the linking types is left out here. See Appendix A.6.1 for a presentation of how the unification of the linking types is done.) Now the filler rule has the linking types *arg1+*, *arg2+*, *arg3-* and *arg4-*, and the ARGFRAME type *arg1-12*. When these types are unified we get the greatest lower bound, which is the type *arg12* (see Figure 4.9 (p. 96)).

The tree in Figure 6.16 shows how the semantic composition works. The verb *leser* introduces an underlying event *_lese_v_rel* with a label and an index. The label is linked to CONT|HOOK|LTOP, and the event index is linked to CONT|HOOK|INDEX. The value of HOOK goes up to the top of the tree. The *arg1-extr-phrase* introduces an underlying event *arg1-relation*. The label of the underlying event is unified with CONT|HOOK|LTOP, and the argument of the underlying event is linked to the index of the extracted argument (*Jon*). The *arg2-phrase* introduces an underlying event *arg2-relation*. The label of the underlying event is unified with CONT|HOOK|LTOP, and the argument of the underlying event is linked to the index of the argument (*avisen*). The semantic representation of the sentence is given in Figure 6.17.

A sentence with a topicalized object as shown in Figure 6.14, realizes the subject after the verb (VP1/NP) and extracts the object (VP2). The object is filled in at the bottom of the tree by the filler rule (VP/NP).

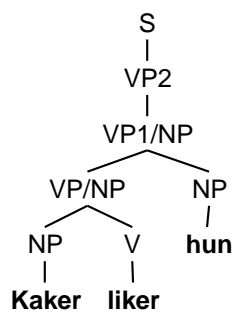
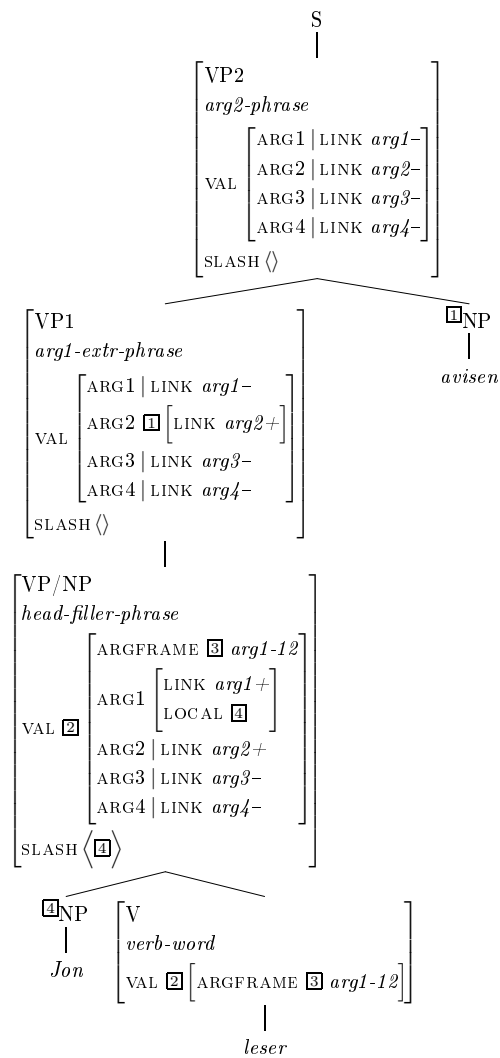


Figure 6.14: Transitive main clause with topicalized object with the verb *liker* ('likes') (BRR: D.13, p. 336)

Figure 6.15: Linking in a transitive main clause with the verb *lesen* ('reads')

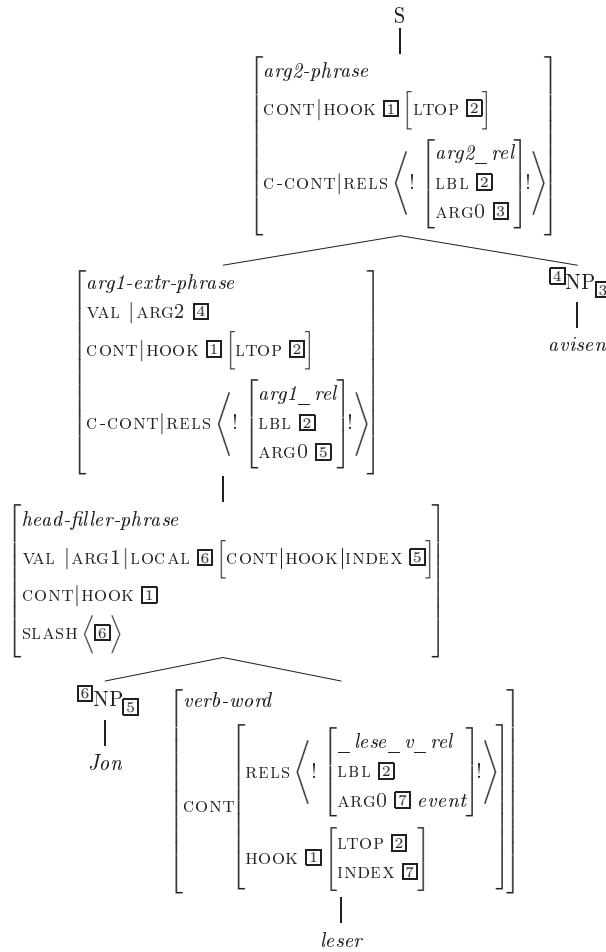


Figure 6.16: Semantic composition in a transitive main clause

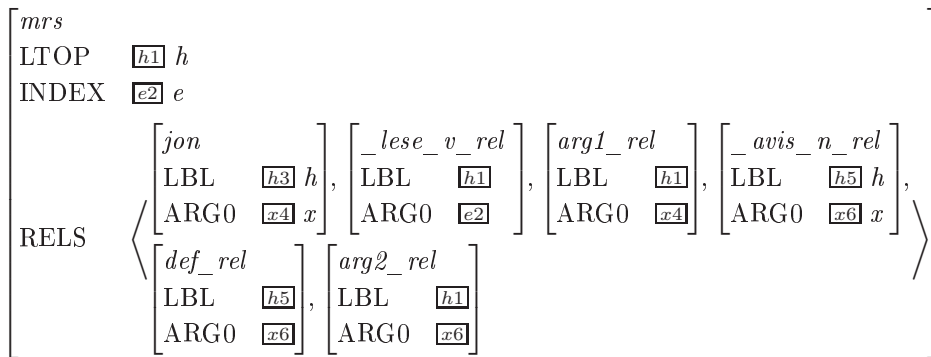


Figure 6.17: BRR of *Jon leser avisen* ('Jon reads the paper')

6.4.2 Analysis of a resultative sentence

The linking in clauses with delimiters is accounted for both by means of constraints on the *arg4* rules as well as constraints on the words that head the delimiter constituents (adjectives, adverbs and prepositions).

The type for the binary *arg4* rule, which realizes delimiters (see Section 3.2.4) in their canonical position, is given in Figure 6.18. It introduces an *arg4-relation* underlying event. The handle of the *arg4-relation* is unified with the LTOP of the rule, and the ARG0 of the *arg4-relation* is unified with the LTOP of the delimiter. Also, the INDEX of the ARG2 is unified with the XARG of the delimiter. This means that the argument that the delimiter predicates over (the value of XARG in the delimiter) is linked to the argument that is realized by the *arg2*-sign of the clause.

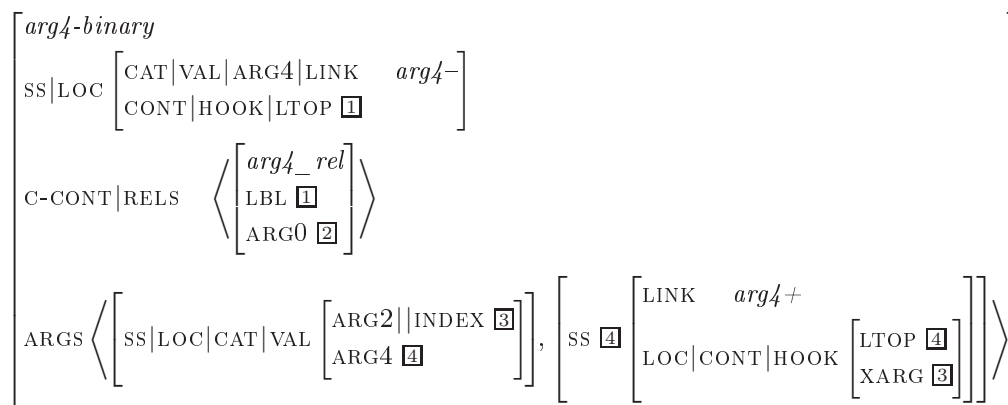
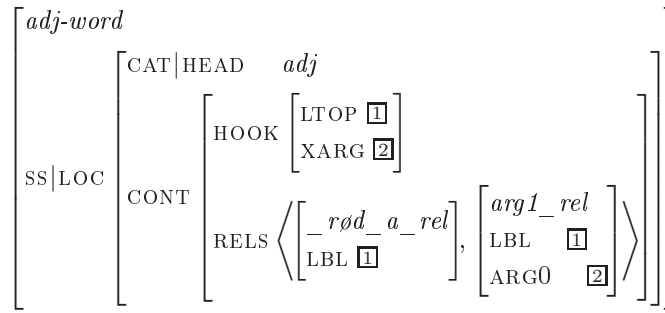


Figure 6.18: Constraints on *arg4-binary*

Adjectives, adverbs, and prepositions are assumed to introduce an *arg1-relation* underlying event, which is linked to the underlying event expressing the predicate as shown for the adjective *rød* in Figure 6.19. The argument of the *arg1-relation* underlying event is reentered as the value the feature XARG.⁶

The trees in Figures 6.21 and 6.22 show analyses of the resultative sentence *Jon maler veggen rød* ('Jon maler veggen rød'). Figure 6.21 shows the linking types. (The unification of linking types is left out in the head filler rule.) Figure 6.22 shows how the semantics is composed. The verb realizes an underlying event *_male_v_rel*, and three

⁶This goes against the general assumption that the *arg1-arg4*-relations are Grammatical Relations (strict syntax). An *arg1-relation* underlying event should strictly speaking not be introduced here. However, in order to make the predication obvious, I allow them to be introduced. (See more discussion on the relation between Grammatical Relations and the semantics of sentences in Section 6.7.4)

Figure 6.19: Constraints on the adjective *rød*

underlying events realized by the rules are linked to it, *arg1-relation*, *arg2-relation*, and *arg4-relation*. The argument of the *arg1-relation* is the index of the NP *Jon*. The argument of the *arg2-relation* is the index of the NP *veggen* ('the wall'). The argument of the *arg4-relation* is the handle of the adjective *rød* ('red'). The adjective introduces an *arg1-relation* underlying event, which handle is unified with the handle of the *_rød_a_rel*. The argument is linked to the NP *veggen* as a result of the linking constraints in the *arg4* binary rule.

The BRR of the sentence *Jon maler veggen rød* ('Jon maler veggen rød') is given in Figure 6.20.⁷

⁷It may seem like the *arg4-relation* underlying event is superfluous since the relation between the delimiter and the object is expressed through the *arg1-relation* introduced by the adjective. Still, the *arg4-relation* is introduced, first, because it is a Grammatical Relation, and second, because it is necessary in cases where there are delimiters but no *arg2*-sign, like in *Jon kaster til Kari* ('John throws to Kari'), where *til Kari* is a delimiter and *Jon* is realized by an *arg1* subconstruction.

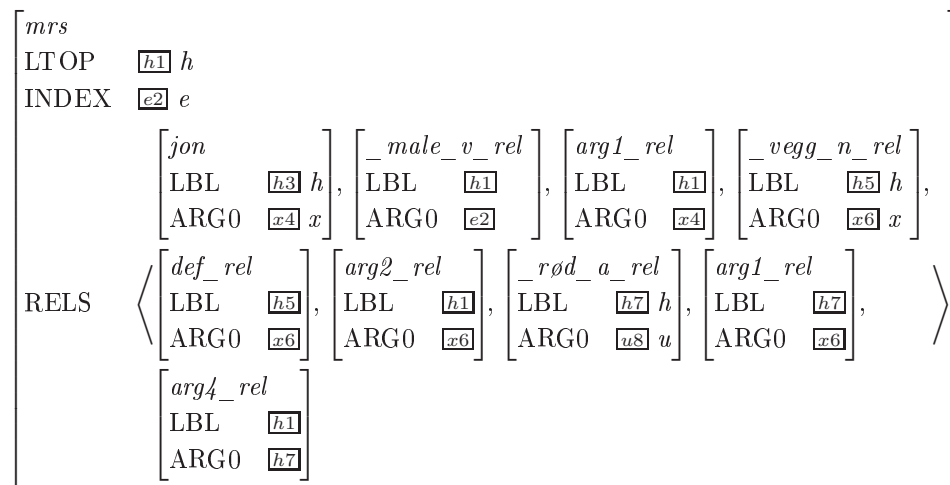


Figure 6.20: BRR of *Jon maler veggen rød* ('Jon paints the wall red') (Trees: 6.21, p. 166 and 6.22, p. 167)

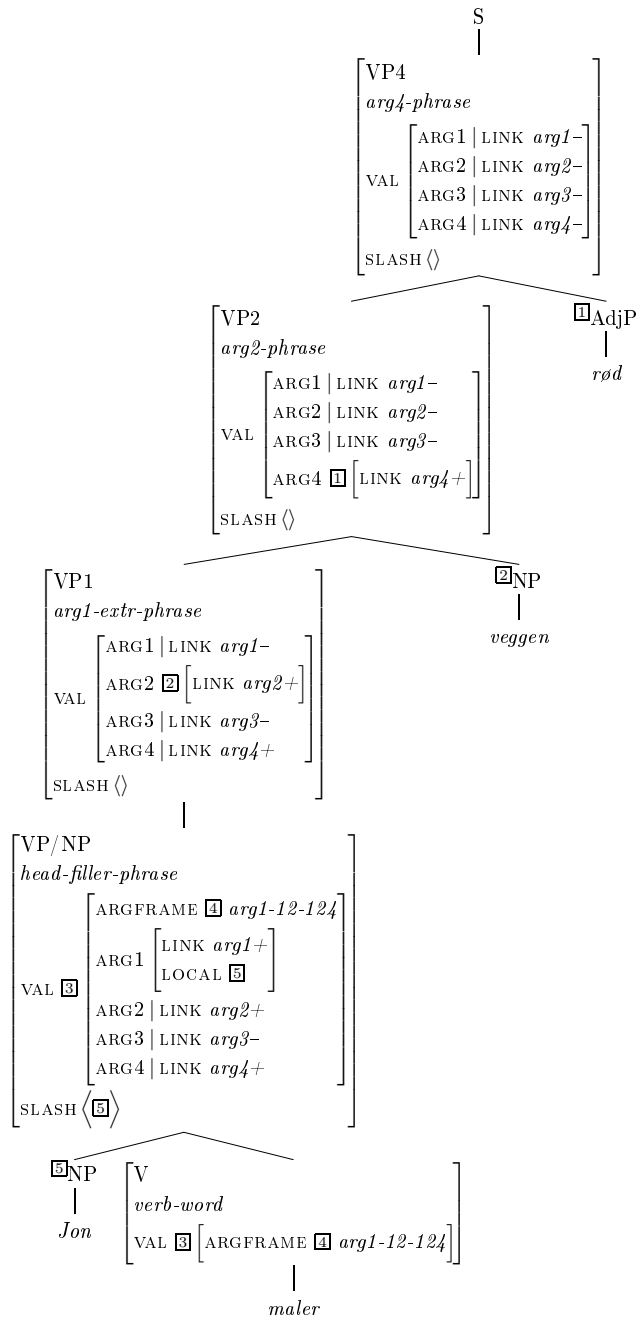


Figure 6.21: Linking types in a resultative main clause with the verb *maler* ('paint') (BRR: 6.20, p. 165)

6.5 The merge rule

In order to account for clauses with auxiliaries and complementizers I assume a rule that combines the projection of the auxiliary/complementizer with other verbs. I call it the *merge* rule. The rule opens for the first auxiliary or complementizer of a clause to be the head of the clause and realize the arguments. Before I go into the details of the merge rule, I show how a tree structure with a merge rule looks in Figure 6.23. The clause is given in (147).

- (147) at han beundrer Marit
 that he admires Marit
 ‘that he admires Marit’

In Figure 6.23 the merge rule is the node *CP*. Its first daughter (the head daughter) is the projection of the complementizer (*CP1*), and its second daughter is the main verb (*V*). The rule that realizes the object (*CP2*) applies after the merge rule. I will return to subordinate clauses in Section 6.6.1.

In clauses with auxiliaries and/or complementizers I assume that the first auxiliary or complementizer is the syntactic head and that the subject is attached to this element. The ERG has a similar analysis. In a clause like *John claims that Mary smiles*, the complementizer takes the subject *Mary* and the VP *smiles* as complements (see Figure 6.24). In the clause *John has smiled* the auxiliary takes the VP *smiled* as its complement.

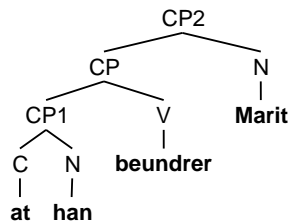


Figure 6.23: Subordinate clause in Norsyg (BRR: D.14, p. 336)

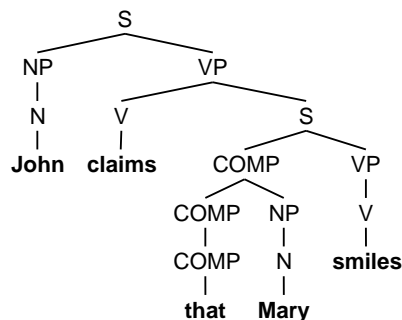


Figure 6.24: Sentence with subordinate clause complement in the ERG

Since the ERG does not do linking in rules like the head complement rule, the complementizer is dependent on having access to the subject and the VP in the lexicon

(or via a unary rule). This is achieved by having the subject and the VP on the COMPS list of the complementizer.

In Norsyg, linking is done in the rules, rather than in the lexicon, and so there is no need for the COMPS list in these cases. Instead of using the head complement rule in analyses involving auxiliaries and complementizers, I use the merge rule, illustrated in Figure 6.25.

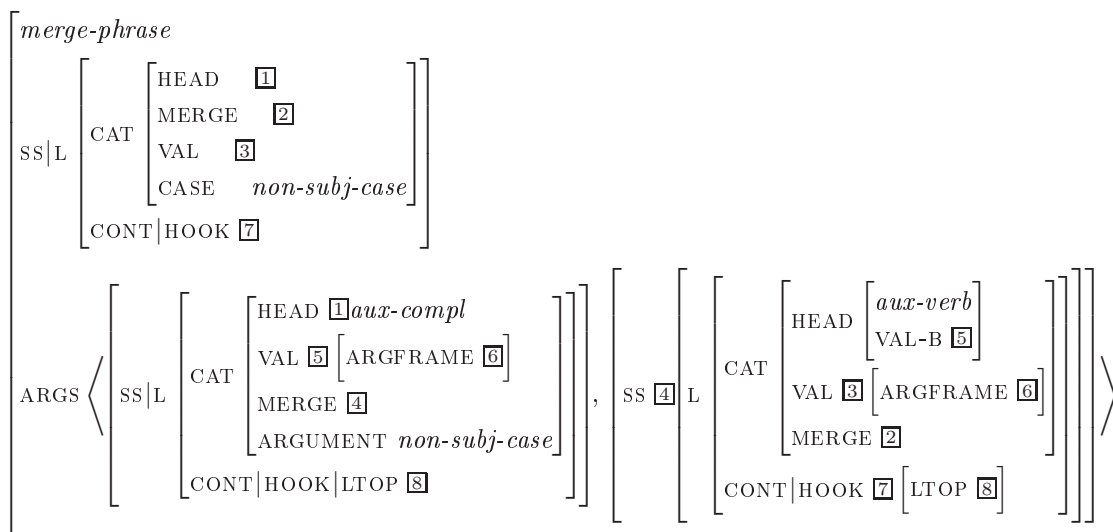


Figure 6.25: The merge rule

The merge rule has two daughters. The first daughter, which is the head daughter, has the head value *aux* or *complementizer*. When this projection enters the merge rule as the head daughter it has already realized the subject of the clause. This is ensured by constraining the ARGUMENT value to be *non-subj-case*. The second daughter has the head value *aux* or *verb*. The merge rule merges the valence information of the first daughter with the VAL-B feature of the second daughter.⁸ This makes it possible for the second daughter of the merge rule to have a subconstruction, and therefore have different values of VAL and VAL-B. I will get back to this possibility in Section 7.1 on passive. As long as the second daughter of the merge rule is not the passive auxiliary or a verb morphologically marked as passive, the valence features of the daughters and the mother in the merge rule will be unified. The function of the merge rule in a subordinate clause with regard to valence is illustrated in Figure 6.26.

⁸See explanation of the VAL-B feature in Section 5.1 and Appendix A.6.1.

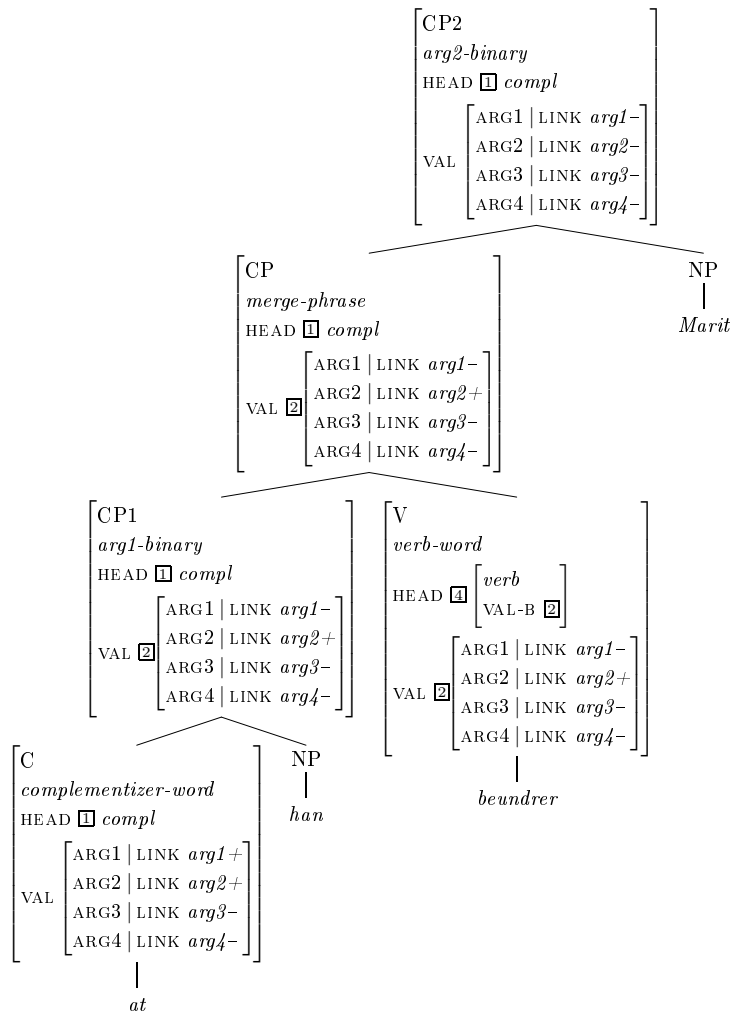


Figure 6.26: Linking types in *at han beundrer Marit* (‘that he admires Marit’) (BRR: D.14, p. 336)

By following the tag [2] in Figure 6.26 it is possible to see how the main verb in a subordinate clause unifies its valence information with the complementizer projection. All verbs, except from the passive auxiliary, unifies their VAL with their VAL-B.⁹

The HOOK value of the mother of the merge rule is unified with the HOOK value of the second daughter, and the LTOP value of the second daughter is unified with the LTOP value of the first daughter. The semantic composition of the subordinate clause *at han beundrer Marit* (‘that he admires Marit’) is illustrated in Figure 6.27. It shows how the merge rule unifies its HOOK value with that of its second daughter, the verb *beundrer*.

⁹The unification of the LINK features is left out in the complementizer word for expository reasons.

The LTOP of the verb is unified with the LTOP of the first daughter. This means that the underlying events *arg1-relation* and *arg2-relation*, which both are realized on the complementizer projection and make links to the subject *han* and the object *Marit*, share handle with the underlying event introduced by the verb, *_beundre_v_rel*.

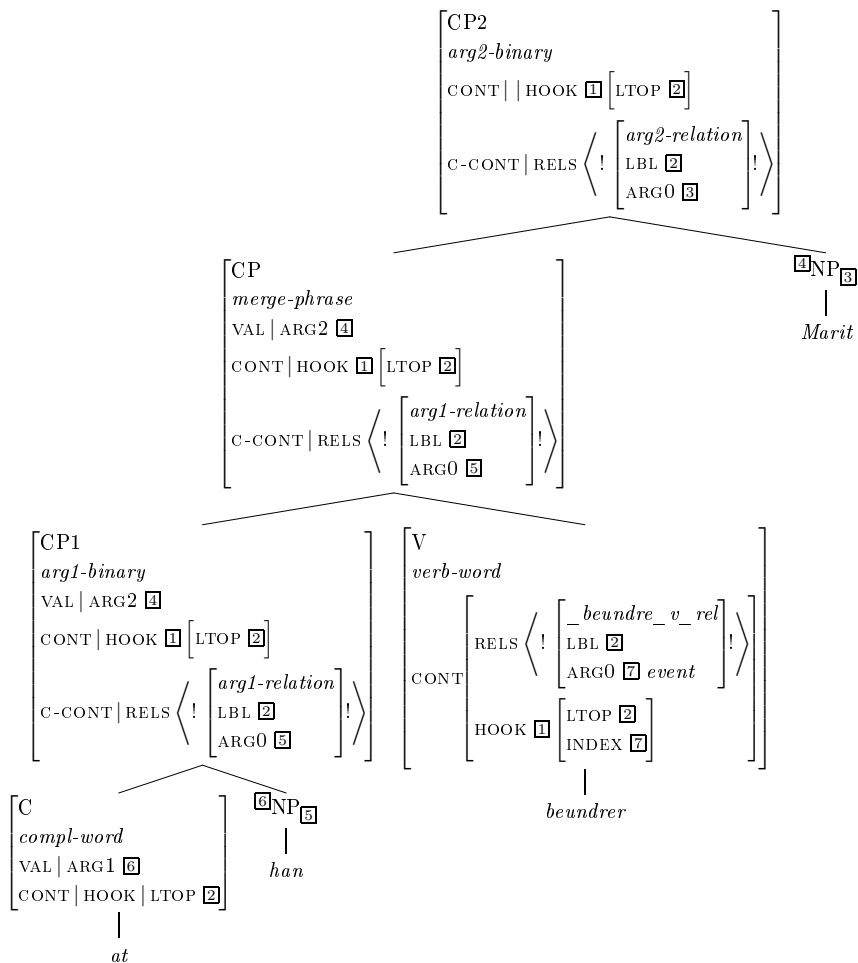


Figure 6.27: Semantic composition in *at han beundrer Marit* (‘that he admires Marit’) (BRR: D.14, p. 336)

The function of the CASE feature on the merge rule is to express whether a constituent is in a field where the subject is realized. When the merge rule has applied, the CASE value is set to *non-subj-case*. This implies that the subject cannot be realized after the merge rule. The CASE value of the first daughter of the (first) merge phrase will be *subj-case*, since the subject is realized before the (first) merge rule applies. In this way, the (first) merge rule marks a boundary between the field where the subject

is realized and the field where it cannot be realized. The feature is necessitated by the analysis of sentence adverbials, which are assumed to attach to a constituent in the field where the subject is realized. I will discuss this in more detail in Chapter 10.

The trees in Figure 6.28 and 6.29 show how Norsyg analyzes sentences with auxiliaries. In Figure 6.28 the *arg1* extraction rule (AUXP1) applies on the filler rule (AUXP/NP) and extracts the subject. The merge rule (AUXP) combines the auxiliary projection with the main verb verb (V). Then the second argument is realized (AUXP2), before the filler rule applies (S). If there is more than one complementizer or auxiliary, the merge rule will apply several times as in Figure 6.29, where three auxiliaries apply before the main verb.

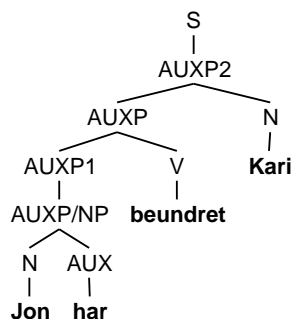


Figure 6.28: Sentence with auxiliary (BRR: D.15, p. 337)

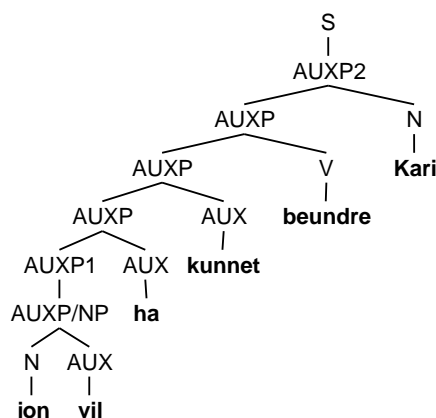
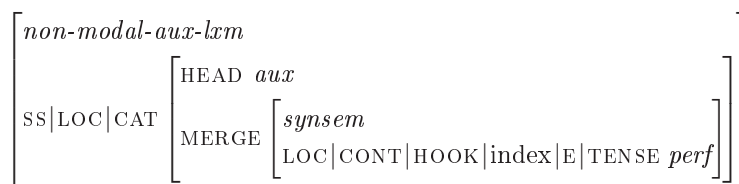
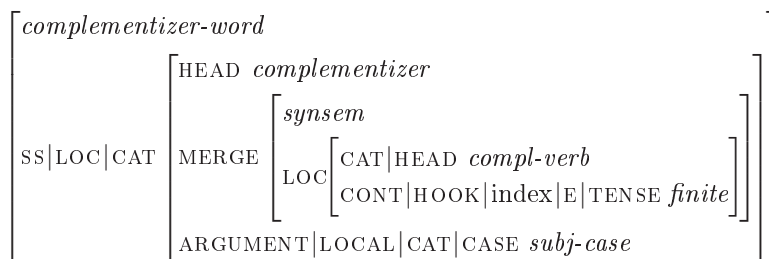


Figure 6.29: Sentence with three auxiliaries (BRR: D.16, p. 337)

Complementizers and auxiliaries have the feature *MERGE* with the value *synsem*. In the merge rule the *MERGE* value of the head daughter is unified with the *synsem* of the second daughter. This makes it possible for the complementizers or auxiliaries to constrain the tense of the verb (main verb or auxiliary) they are merging with. The auxiliary *ha* ('have') has the lexical information in Figure 6.30. It constrains the *TENSE* value of the verb that it merges with to be *perf*. An auxiliary appearing in a string of verbs, as *ha* and *kunnet* in Figure 6.29 constrains the tense of the following verb. Main verbs block the possibility of merging with other verbs by having the *MERGE* value *anti-synsem*, which is not compatible with the type *synsem*.

Complementizers have the constraints shown in Figure 6.31. Via the *MERGE* feature they constrain the tense of the verb they merge with to be *finite*.

Figure 6.30: The auxiliary *ha* ('have')Figure 6.31: Constraints on *complementizer-word*

6.6 Subordinate clauses and relative clauses

One consequence of an analysis where the extraction site dominates the filler, is that valence rules applying in embedded clauses (subordinate clauses, relative clauses and infinitival clauses) need to dominate the filler. This leads to a radically new analysis of embedded clauses where they are not necessarily analyzed as constituents.¹⁰ In the new analysis I am proposing here, the subordinating conjunction (here meaning complementizers, the relative pronoun and the infinitival marker) may attach to the projection of the matrix clause (or a nominal, in the case of relative clauses), and turn it into an embedded clause, which it heads. The matrix clause projection (or nominal) is put on stack until the embedded clause is parsed. Then it is popped from the stack, and the matrix clause projection (or nominal) takes over again. This is illustrated for subordinate clauses in Figure 6.32, for infinitival clauses in Figure 6.33,¹¹ and for relative clauses in Figure 6.34.

In each of the analyses in Figure 6.32–6.34, the subordination conjunction attaches to the matrix clause (or the nominal) from the right and becomes the head of the new

¹⁰Embedded clauses that are fronted will be analysed as constituents, but embedded clauses that appear inside the clause will not be analysed as constituents.

¹¹The Norwegian letter *å* was not possible to display with the tree browser distributed with the LKB system, so I used *aa* instead.

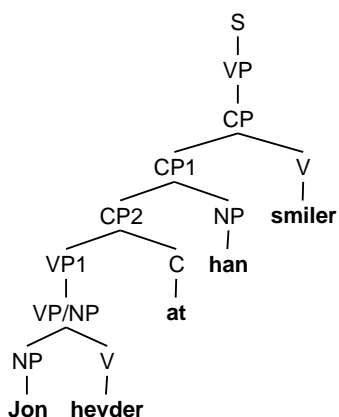


Figure 6.32: Sentence with subordinate clause (BRR: D.17, p. 337)

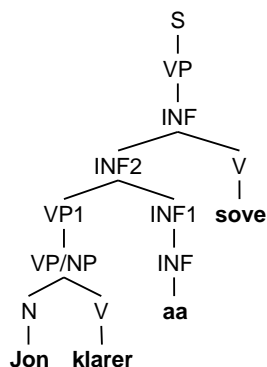


Figure 6.33: Sentence with infinitival clause (BRR: D.18, p. 338)

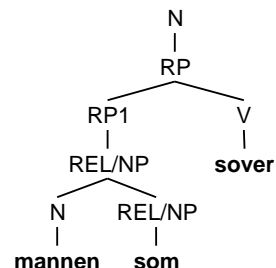


Figure 6.34: NP with relative clause (BRR: D.19, p. 338)

structure. The analyses also show that the matrix projection comes back again higher up the tree.

The rules for subordinate clauses, infinitival clauses and relative clauses are organized in a type hierarchy, as shown in Figure 6.35.

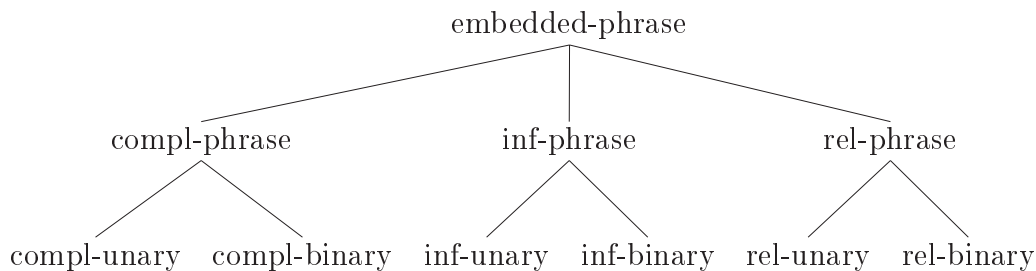
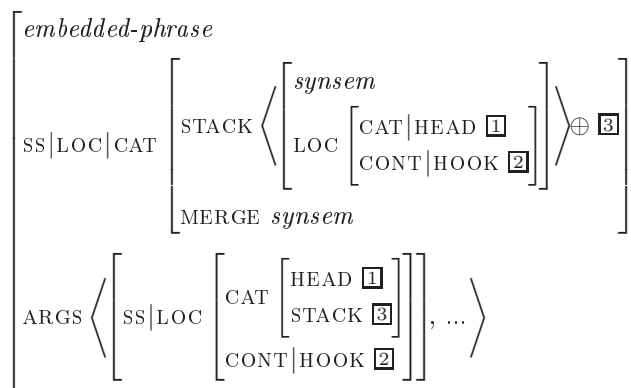


Figure 6.35: Hierarchy of subordination-phrases

The definition of *embedded-phrase* is given in Figure 6.36. It shows that the values of HOOK and HEAD of the first daughter are reentered in the stacked item (see the feature STACK). It also shows that the new constituent has a MERGE requirement (*synsem*), which means that the embedded structure needs to combine with a main verb.

6.6.1 Subordinate clauses

Subordinate clauses are accounted for by means of the *complementizer phrase*. The type for this construction, *compl-phrase*, was introduced in the hierarchy under

Figure 6.36: The type *embedded-phrase*

subconstruction in Section 3.5 as a subtype of *arg2-sign* (see Figure 3.8, p. 85). As shown in Figure 6.35, it also inherits from *embedded-phrase*.

The complementizer construction comes in two versions: one binary version, where the complementizer is expressed, and one unary version, where the complementizer is not expressed. The hierarchy is given in Figure 6.37. Most of the information is given in the supertype *compl-phrase*. It shows that the complementizer phrases take as their first daughter a complementizer, preposition, or verb projection, where the MERGE requirement is fulfilled (*anti-synsem*). It should also be compatible with the first daughter's ARG2 to have *complementizer* as head value. The *compl-phrase* becomes a complementizer projection with an unfulfilled MERGE requirement (*synsem*), and an element on the STACK. Since *compl-phrase* inherits from *arg2-sign*, the ARG2|LINK value is switched from the valence in the first daughter to the valence in the stacked element. This is ensured by unifying IN with CAT of the daughter and OUT with the CAT of the stacked element.

The two subtypes constrain the number of daughters. The binary phrase has a second daughter, the complementizer, and the unary rule has only one daughter.

The subordinate constructions work together with a rule that pops the stacked elements. This rule is presented in Figure 6.38. It is a unary rule that realizes the first element on the stack of its daughter as its own *synsem* value. The negative linking types (*arg1-*, *arg2-*, *arg2-*, and *arg4-*) ensure that all the arguments of the embedded clause are realized.

Given the two complementizer constructions *compl-binary* and *compl-unary*, and the pop-rule, it is possible to analyze sentences with subordinate clauses that either

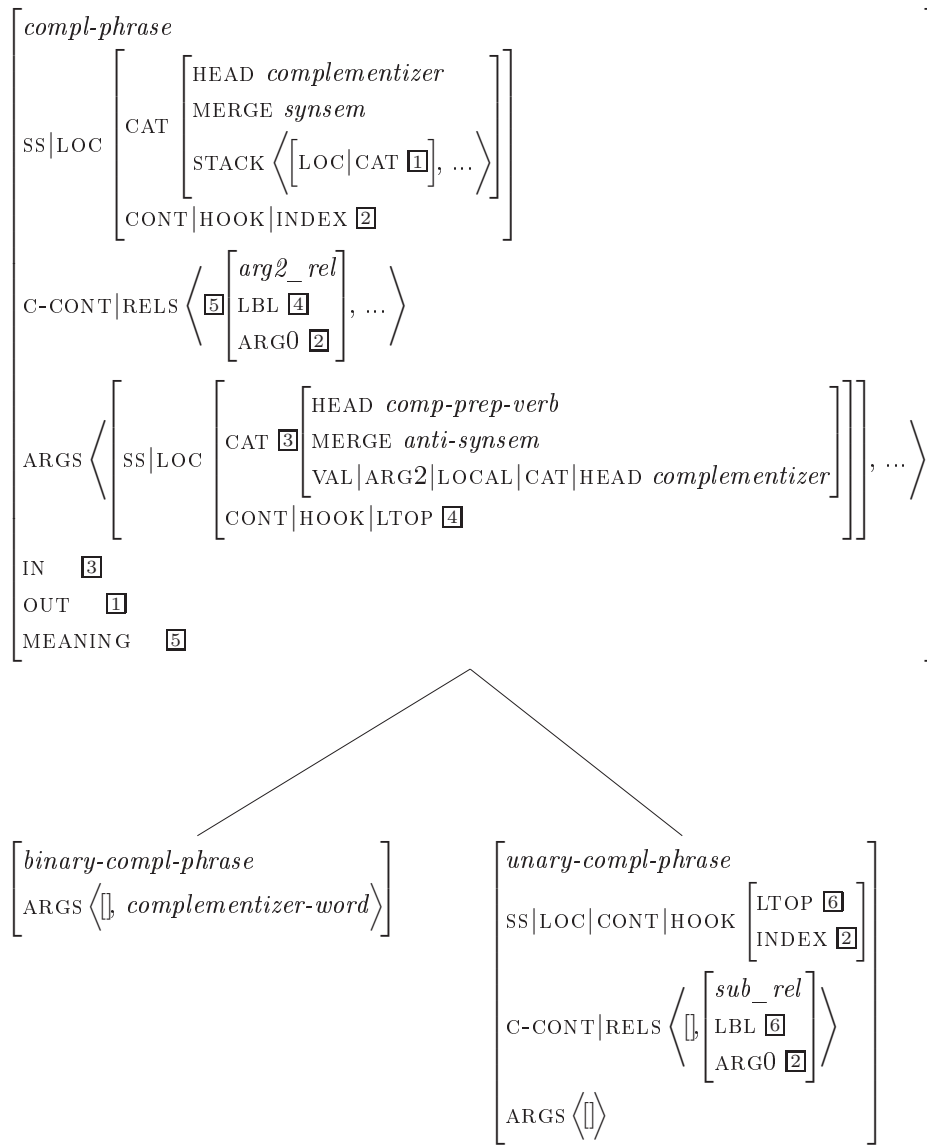


Figure 6.37: Hierarchy of complementizer phrases

have or do not have complementizers. Figure 6.39 shows an analysis of a sentence with a subordinate clause. The node CP2/NP, which is the binary complementizer rule, combines a verb projection and a complementizer. Figure 6.40 shows an analysis of the same sentence without the complementizer. Here the node CP2/NP is the unary complementizer rule.

The trees in Figure 6.39 and 6.40 show that the complementizer (C) (whether it is expressed or not) becomes the head of the structure, and that the following words

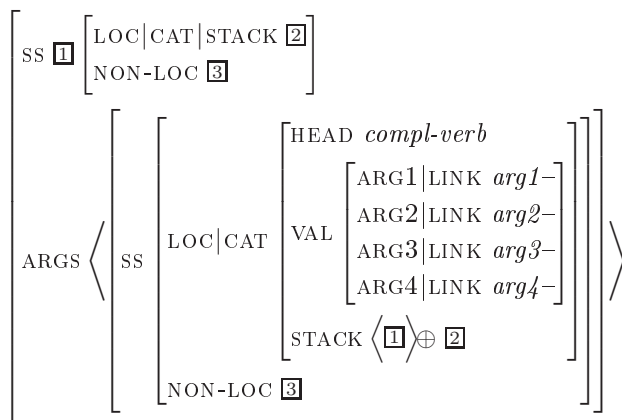


Figure 6.38: Pop rule

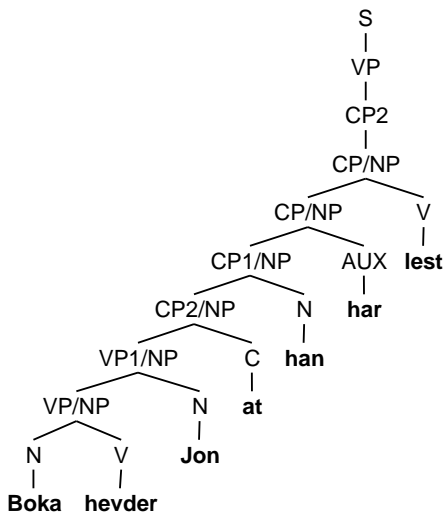


Figure 6.39: Analysis of *Boka hevder Jon at han har lest* ('The book, John claims that he has read') (BRR: D.20, p. 338)

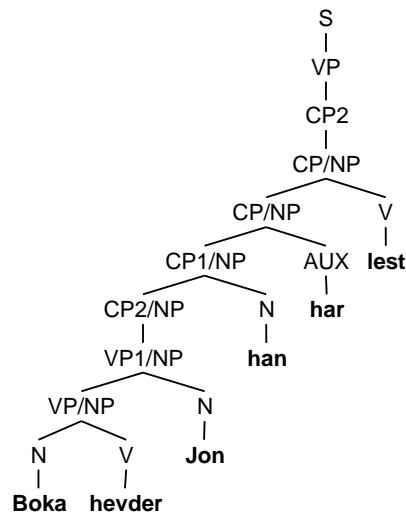


Figure 6.40: Analysis of *Boka hevder Jon han har lest* ('The book, John claims he has read') (BRR: D.21, p. 339)

attach to the C projection. At the top of the trees, the structures are turned back into V projections by means of the pop rule. The trees also illustrate how long distance dependencies work when the extracted constituent is extracted from a subordinate clause. In both trees, the NP *Boka* is extracted by the extraction rule close to the top of the trees (CP2). The SLASH list is then copied down to the filler rule at the bottom of the trees. This is illustrated in Figure 6.41, which is the same tree as 6.39, except that the top node (S) is not displayed.

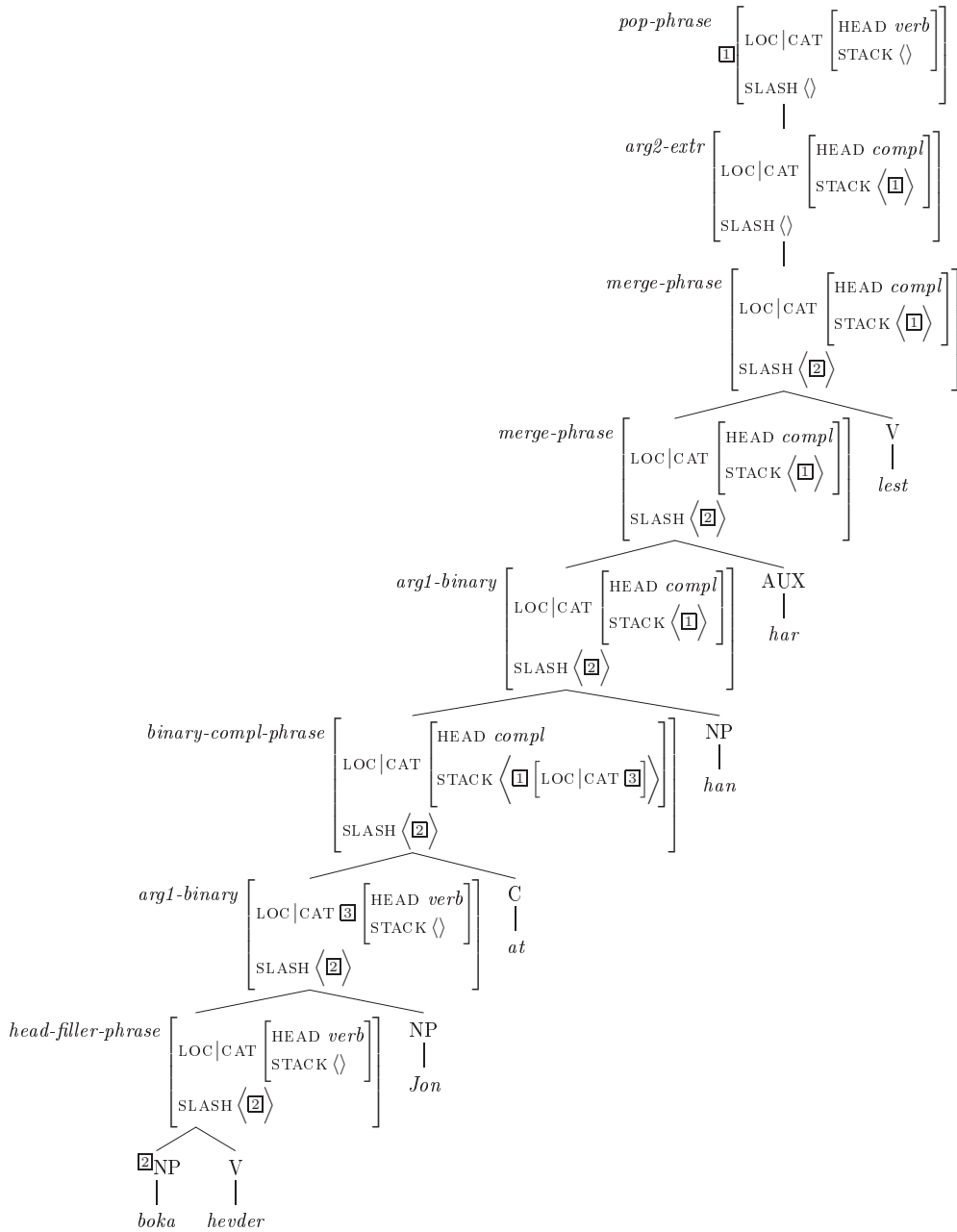


Figure 6.41: Long distance dependencies and stacking in *Boka hevder Jon at han har lest* ('The book, John claims that he has read') (BRR: D.20, p. 338)

The “stacking” and “popping” mechanism allows for several embeddings into subordinate clauses. The function of the pop rule is to arrive at the matrix clause level again after entering a subordinate clause. The pop rule allows for the expected PP attachments, as the trees in Figure 6.42 and 6.43 show. In Figure 6.42, the PP attaches inside the subordinate clause, while in Figure 6.43, the PP attaches at main clause level.

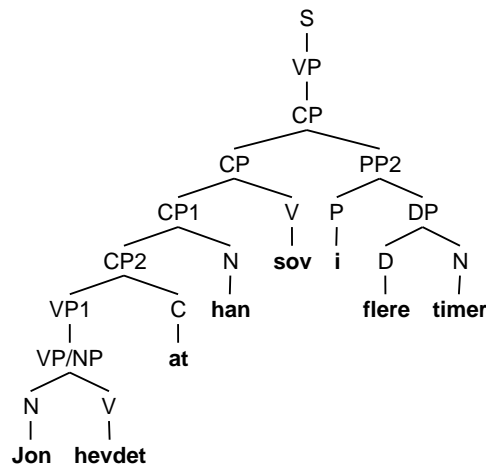


Figure 6.42: Analysis of *Jon hevdet at han sov i flere timer* (‘John claimed that he had slept for several hours’). PP attachment to subordinate clause. (BRR: D.22, p. 339)

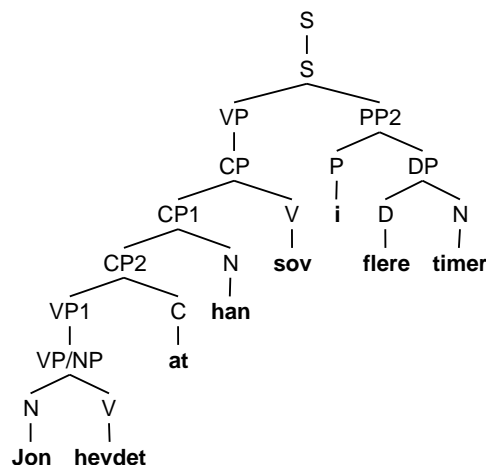


Figure 6.43: Analysis of *Jon hevdet at han sov i flere timer* (‘John claimed that he had slept for several hours’). PP attachment to main clause. (BRR: D.23, p. 340)

6.6.2 Relative clauses

The analysis of relative clauses has much in common with the analysis of subordinate clauses shown in Section 6.6.1. A construction for relative clauses is assumed, where the relative pronoun (if expressed) attaches to the nominal from the right, and stacks the nominal in *STACK*. The construction has two versions, a binary and a unary. The constraints are shown in the type hierarchy of relative clause constructions in Figure 6.44.

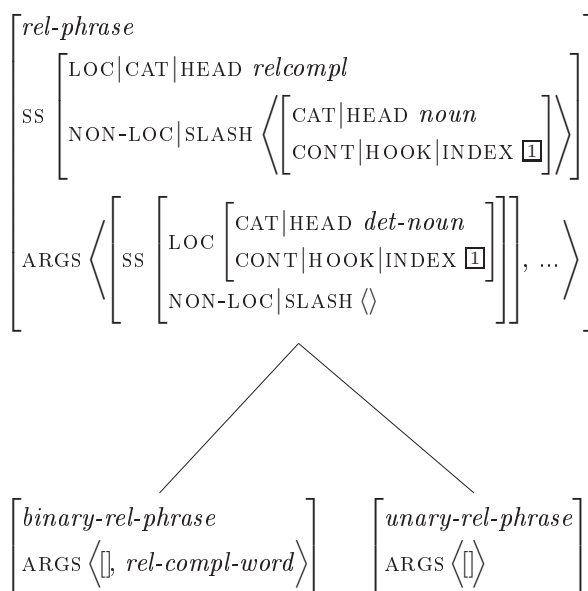


Figure 6.44: Hierarchy of relative clause constructions

The type *rel-phrase* in the hierarchy in Figure 6.44 inherits from *embedded-phrase* (see Figure 6.36). *rel-phrase* shows that relative clause constructions take as their first daughter a structure with *det* or *noun* as head value. They create a structure which has the HEAD value *relcompl*, and which has an element on the SLASH list. This element has the HEAD value *noun*, and it is coindexed with the index of the first daughter of the construction. The type *rel-phrase* has two subtypes. The first subtype is *binary-rel-phrase*, which has a second daughter, the relative pronoun. The second subtype is *unary-rel-phrase*, which is a unary rule.

With the relative clause constructions *rel-binary* and *rel-unary*, and the pop-rule, it is possible to analyze NPs with relative clauses, both with and without the relative pronoun. Figure 6.45 shows an NP with a relative clause where the relative pronoun

combines with the noun and forms the constituent RP/NP. The tree in Figure 6.46 is identical, except from the lack of relative pronoun. The analyses illustrate how the relative clause constructions enter an element on the SLASH list, which has to be extracted higher up in the tree (RP2). The constraints in the relative clause construction ensure that the extracted element is linked to the noun that is modified.

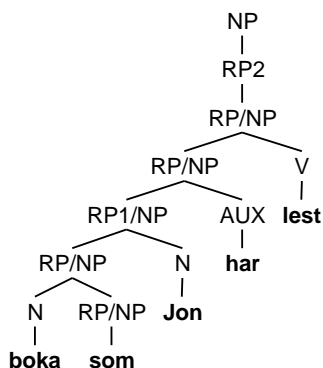


Figure 6.45: Analysis of *boka som Jon har lest* ('the book that Jon has read') (BRR: D.24, p. 340)

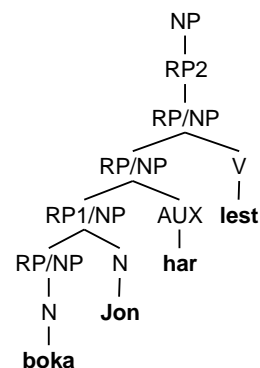


Figure 6.46: Analysis of *boka Jon har lest* ('the book John has read') (BRR: D.25, p. 341)

The long distance dependencies and stacking in the NP *boka som Jon har lest* ('the book John has read') are illustrated in Figure 6.47.

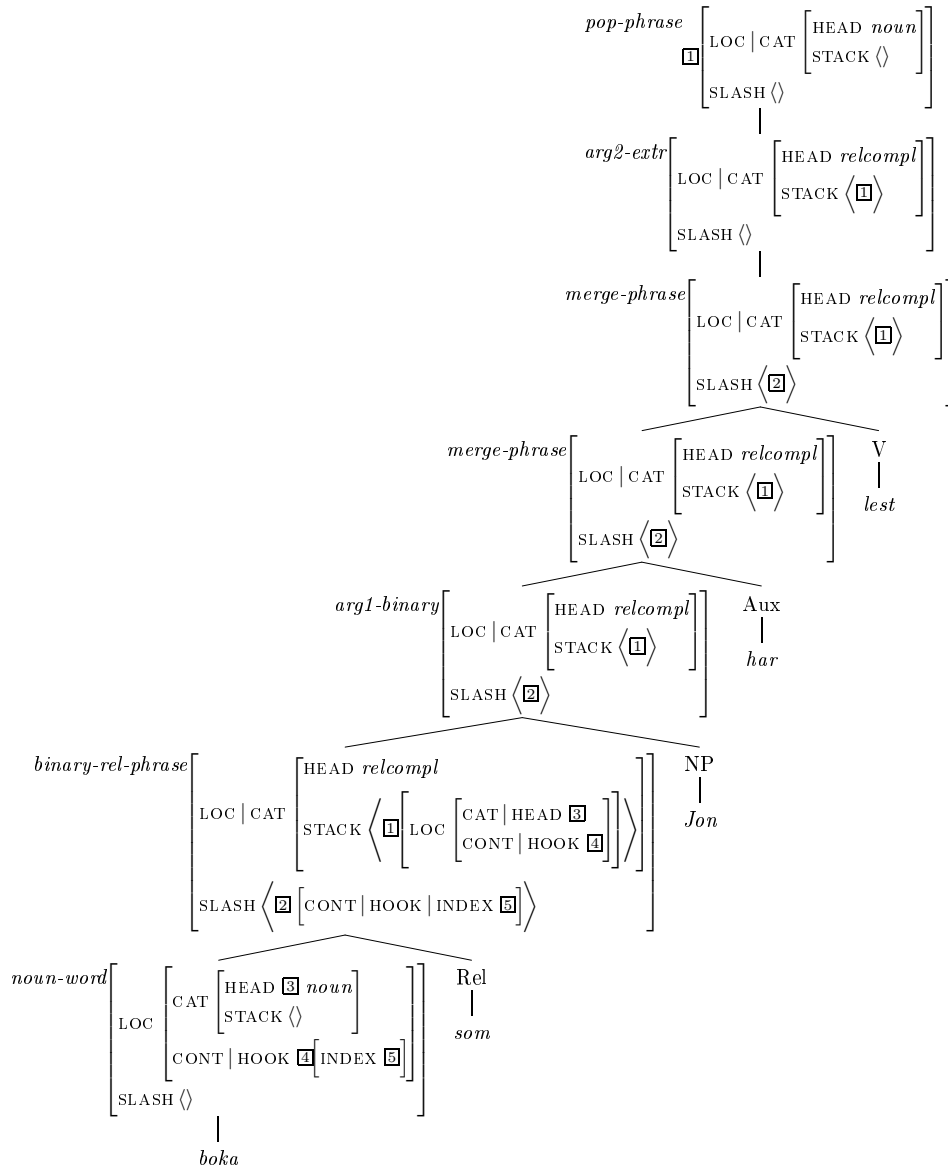


Figure 6.47: Long distance dependencies and stacking in the NP *boka som Jon har lest* ('The book John has read') (BRR: D.24, p. 340)

6.7 Infinitival clauses and small clauses

Before I present how infinitival clauses and small clauses are analyzed, I show how unexpressed subjects are treated.

6.7.1 Unexpressed subjects

Certain functional signs are assumed to realize an unexpressed subject. Examples of such signs are the infinitival marker (148), the small clause construction (149) and the imperative morpheme (150). Common to all these signs is that the subject can have the arg1-role (see (148a), (149a) and (150a)), the arg2-role (see (148b), (149b) and (150b)) and the arg3-role (see (148c), (149c) and (150c)).

- (148) a. John likes to sleep.
 b. John likes to be heard.
 c. John wants to be given a book.
- (149) a. John let her sleep.
 b. John let her be heard.
 c. John let her be given a book.
- (150) a. Sleep!
 b. Be heard!
 c. Be given a book!

In order to account for the linking of the unexpressed subjects, one possibility would be to create one sign for each of the argument roles. This would mean three infinitival marker words, three small clause construction rules and three imperative inflectional rules. In Norsyg, I have generalized over the unexpressed subject constructions by means of three unary linking rules that take the unexpressed subject constructions as input and links the unexpressed subject. The rules make it possible to underspecify the infinitival marker, the small clause construction, and the imperative inflection with regard to what argument role that is linked, and multiple versions of them are avoided. It is however difficult to say which of these options is better. Multiple signs has the advantage that the trees look nicer (there is no unary rule on top of the unexpressed subject construction). Unary linking rules have the advantage that there is only one

infinitival marker in the lexicon, one small clause rule and one imperative inflectional rule.

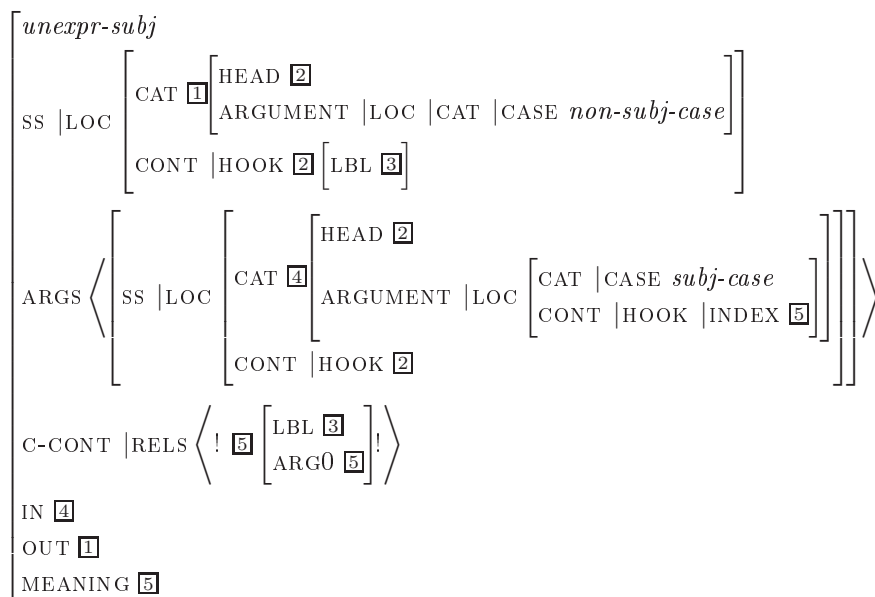


Figure 6.48: The basic unary linking rule

6.7.2 Analyses of infinitival clauses and small clauses

Infinitival clauses and small clauses are analyzed in a similar fashion to subordinate clauses and relative clauses. Also here a rule is assumed that combines the infinitival marker with the projection of the matrix clause. The analysis involves both infinitival clauses as well as small clauses (see Section 6.7.1). A general type *inf-phrase* is assumed that has two subtypes, *inf-binary* and *inf-unary* (see Figure 6.35, p. 174), where *inf-binary* is used in infinitival clauses and *inf-unary* is used in small clauses. *inf-phrase* inherits from *embedded-phrase* (see Figure 6.36).

The type *inf-phrase* in Figure 6.49 shows that the infinitival constructions take a projection where the main verb is realized as its first daughter. (The MERGE value is *anti-synsem*.) The type also shows that infinitival constructions form constituents that need to merge with a verb that has infinite tense.

The type *binary-inf-phrase* inherits from *arg2-sign* in addition to *inf-phrase*. This means that the second argument of the matrix verb is linked to the event of the infinitival projection. The second daughter of the construction is the infinitival marker. *binary-*

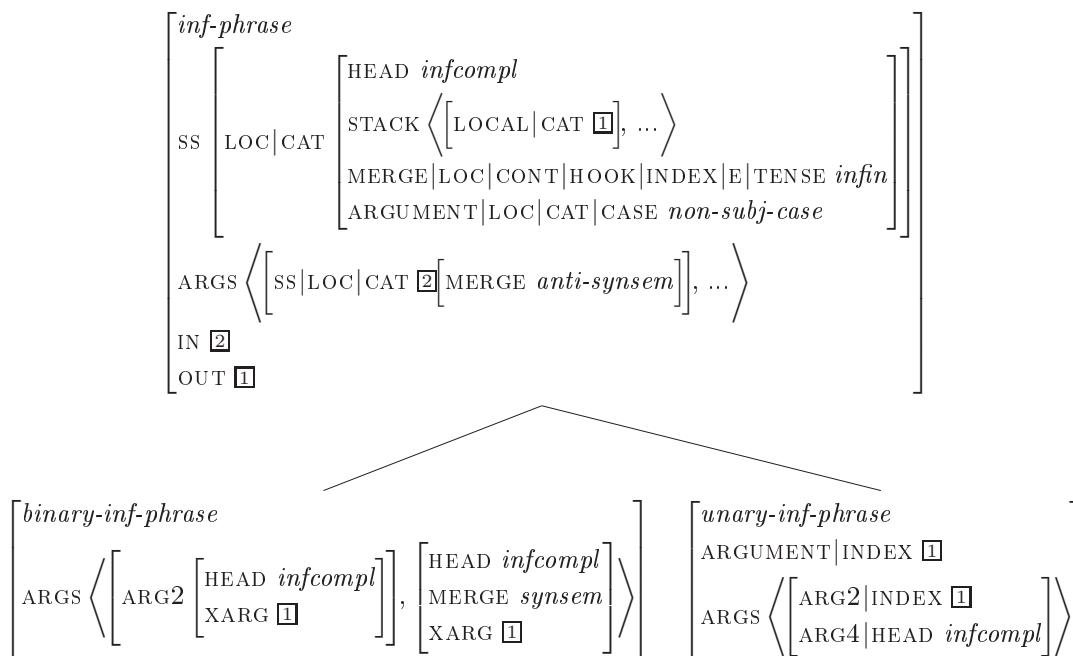


Figure 6.49: Hierarchy of infinitival clause constructions

inf-phrase unifies the XARG value of the first daughter’s ARG2 with the XARG of the second daughter. This ensures that the unexpressed subject of the infinitival marker is linked to the argument that is controlled by the matrix verb (see Section 6.7.3).

The type *unary-inf-phrase* inherits from *arg4-sign* in addition to *inf-phrase*. This implies that the ARG4 of the matrix verb is linked to the event of the infinitival projection. The type also links the index of its ARGUMENT to the index of the ARG2 daughter. This ensures the linking of the unexpressed subject and the ARG2 of the matrix clause.

The lexical type for verbs that take small clauses as complement is given in Figure 6.50.

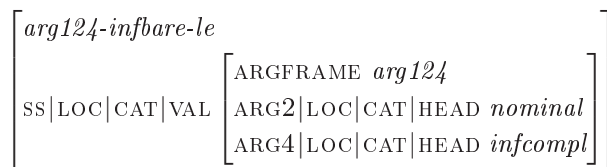


Figure 6.50: Lexical information on a subject control verb

Examples of analyses of the two constructions are given in Figure 6.51 and 6.52.

Figure 6.51 is a sentence with an infinitival clause. The *binary-inf-phrase* combines the infinitival marker with the verb projection, and creates a new infinitival constituent (INF2). Before the infinitival marker combines with the VP, the unary linking rule (INF1) works (see Section 6.7.1) and links the unexpressed subject. What is left for the infinitival projection to realize in INF2 is the main verb, *lese* ('read'), and the non-subject arguments (here: *boka* ('the book'))).

Figure 6.52 is a sentence with a small clause. The small clause construction is initiated by the type *unary-inf-phrase* (SC4). It takes as input the VP2, where the matrix clause has realized its ARG1, *Kari*, and its ARG2, *Jon*. It turns the constituent into an infinitival projection that first undergoes the unexpressed subject linking rule (INF1), and then combines with the main verb *lese* and the non-subject argument *boka*. Before the top of the tree, the matrix projection is popped from the stack (VP).¹²

The long distance dependencies and stacking in the sentence with the small clause *Kari ser Jon lese boka* ('Kari sees John read the book') is illustrated in Figure 6.53.

¹²The analysis I proposed for small clauses cannot account for discontinuous constituents in German. The example (cli) is taken from Müller (2004, 220). The arguments of the verbs *füttern*, *helfen* and *lassen* (*Hans*, *Cecilia*, *John* and *das Nilpferd*) can scramble freely.

(cli) weil Hans Cecilia John das Nilpferd füttern helfen läßt
 because Hans Cecilia John the hippo feed help let
 'because Hans lets Cecilia help John feed the hippo.'

The analysis I have of corresponding data in Norwegian, is that the small clause construction takes a matrix clause as input, stacks it and creates a structure which is the projection of the embedded clause (see *unary-inf-phrase* in Figure 6.49). The analysis presupposes a fixed word order and cannot handle scrambling. In order to analyze discontinuous constituents, I would assume valence rules that were able to look into the arguments of its ARG4 (see *arg2-binary-1embedding* below), and maybe also the arguments of ARG4 of its ARG4 (see *arg2-binary-2embedding* below). It should be noted that data such as (cli) are more difficult to process than their Norwegian and English translations, and there is a limit to how many embeddings that are possible.

$$\left[\begin{array}{l} \textit{arg2-binary-1embedding} \\ \text{SS} \left[\begin{array}{l} \text{VAL|ARG4|VAL|ARG2} \boxed{1} \left[\text{LINK } \textit{arg2-} \right] \\ \text{ARGUMENT} \boxed{1} \end{array} \right] \\ \text{HD-DTR|SS|VAL|ARG4|VAL|ARG2 } \textit{arg2+} \\ \text{NH-DTR|SS} \boxed{1} \end{array} \right]$$

$$\left[\begin{array}{l} \textit{arg2-binary-2embedding} \\ \text{SS} \left[\begin{array}{l} \text{VAL|ARG4|VAL|ARG4|VAL|ARG2} \boxed{1} \left[\text{LINK } \textit{arg2-} \right] \\ \text{ARGUMENT} \boxed{1} \end{array} \right] \\ \text{HD-DTR|SS|VAL|ARG4|VAL|ARG4|VAL|ARG2 } \textit{arg2+} \\ \text{NH-DTR|SS} \boxed{1} \end{array} \right]$$

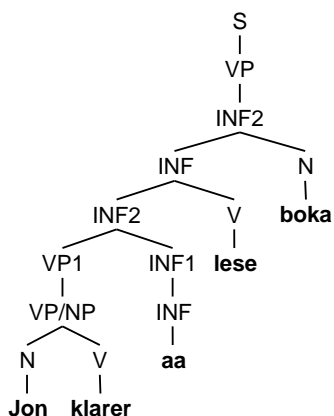


Figure 6.51: Analysis of *Jon klarer å lese boka* ('Jon manages to read the book') (BRR: D.26, p. 341)

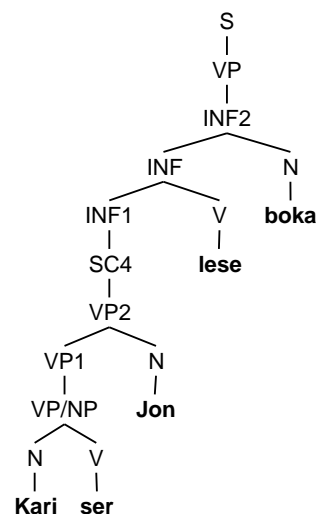


Figure 6.52: Analysis of *Kari ser Jon lese boka* ('Kari sees Jon read the book') (BRR: 6.55, p. 190)

The tree is the same as in Figure 6.52, except that the top node (the force rule) is not shown.

The tree in Figure 6.54 shows the semantic composition of the sentence.

The verb of the matrix clause is *ser* ('sees'). It introduces an underlying event *_se_v_rel* which is linked to the LTOP of the clause. The two lower valence rules *arg1-extr* and *arg2-binary* link the arguments *Kari* and *Jon* to the verb underlying event via the two underlying events *arg1-relation* and *arg2-relation*. The infinitival construction *unary-inf-phrase* realizes an underlying event *arg4-relation*, which shares handle with the underlying events of the matrix clause and takes as argument the handle of the subordinate clause LTOP. The HOOK of the daughter of the construction has the HOOK features of the matrix clause, and the HOOK of the mother has the HOOK features of the subordinate clause. The HOOK value of the daughter is reentered in STACK (see Figure 6.53). The construction also links the index of ARGUMENT to the index of the ARG2 of the matrix clause. This ensures that the unexpressed subject of the small clause is linked to the ARG2 of the matrix clause (*Jon*) since the unexpressed subject is the next argument to be realized. This is done in the *arg1-unexpr* rule (*arg1-unexpr*). It introduces an *arg1* underlying event *arg1-relation* which has as argument the index of ARGUMENT. The handle of the underlying event is linked to the

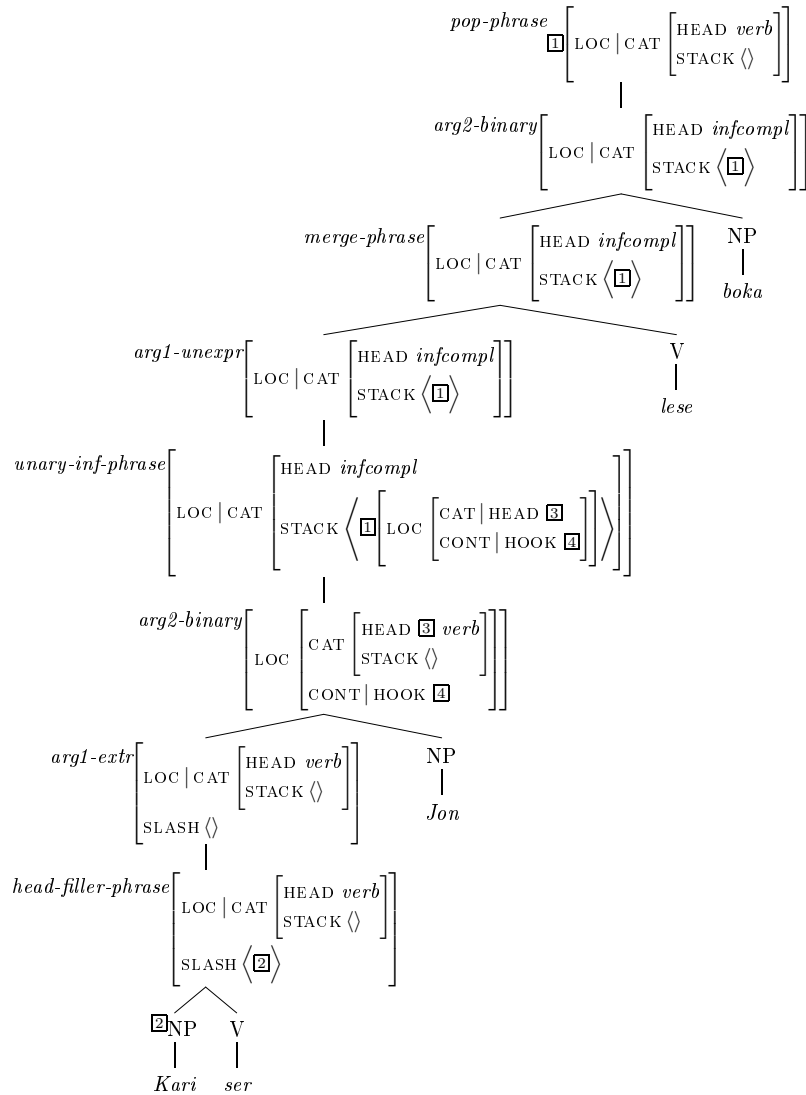


Figure 6.53: Long distance dependencies and stacking in *Kari ser Jon lese boka* ('Kari sees John read the book') (BRR: D.26, p. 341)

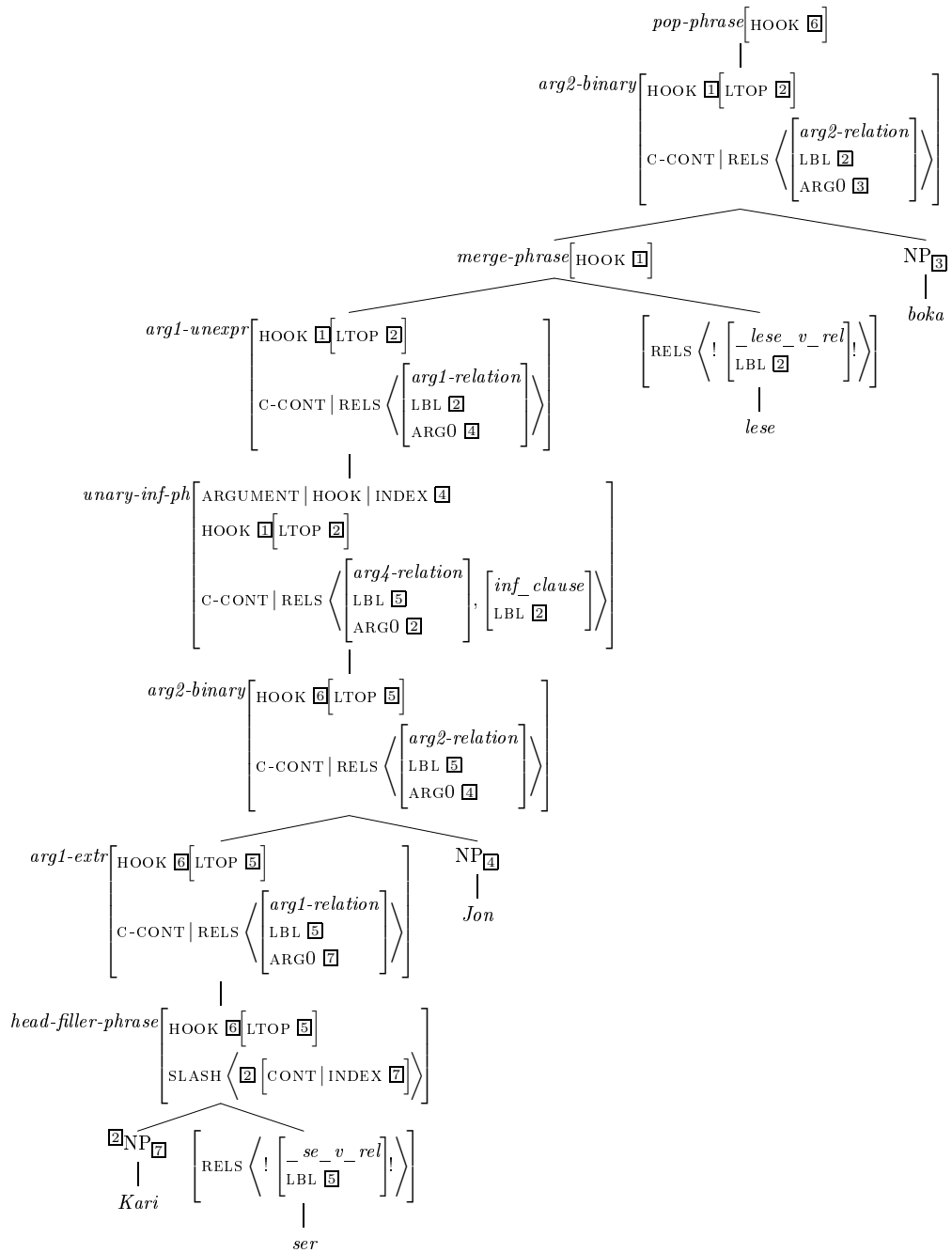


Figure 6.54: Semantic composition in *Kari ser Jon lese boka* ('Kari sees John read the book') (BRR: D.26, p. 341)

LTOP of the subordinate clause. The verb *lese* ('read') introduces an underlying event *_lese_v_rel* which is linked to the LTOP of the subordinate clause by the merge rule. The upper *arg2* binary rule introduces an underlying event *arg2-relation*, which has as argument the index of the NP *boka* ('the book'). The pop rule on top takes the matrix clause projection out of the stack (see Figure 6.53). The BRR is given in Figure 6.55.¹³

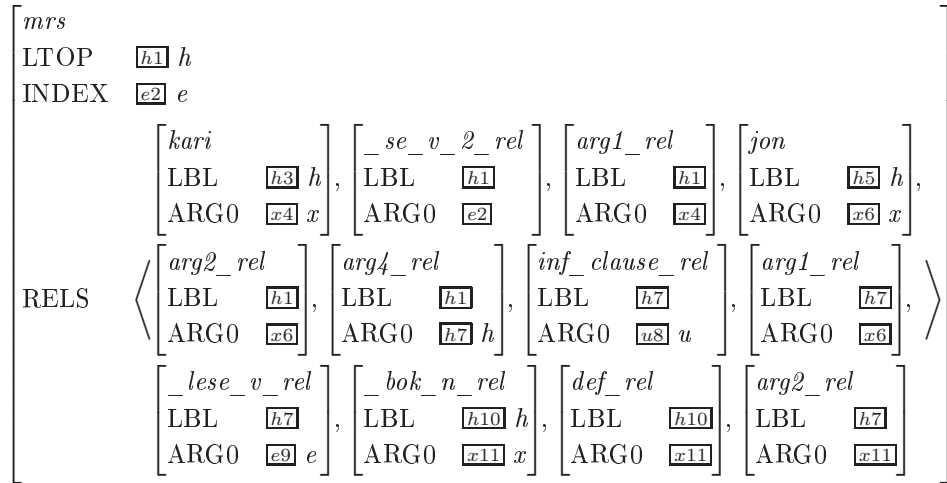


Figure 6.55: BRR of *Kari ser Jon lese boka* ('Kari sees John read the book') (Trees: 6.52, 6.53, and 6.54)

¹³In order to account for sentences like *Vi hørte det regne utenfor* ('We heard it rain outside'), I would assume a subject control verb with the ARGFRAME value *arg124-14*. This would allow the verb to appear both in raising constructions with an *arg2*-relation to the direct object (like *se* in Figure 6.54), and in raising-constructions where the direct object is an expletive. The latter analysis is not implemented in the present version of Norsyng.

6.7.3 Raising and control

In this section I will look at sentences like those in (152), where the subject of the matrix clause is linked to the (unexpressed) subject of the infinitival clause complement.

(152) a. John expects to meet Mary.

b. John seems to smile.

The literature points at differences in behavior between verbs like *expect* in (152a) and verbs like *seem* in (152b), one being that an expletive can be the subject in one group, but not in the other (see e.g. Huddleston (1984, 209-215)). This is illustrated in (153) where (153a) is ungrammatical, whereas (153b) is grammatical.

(153) a. * There expects to be a problem with the computer.

b. There seems to be a problem with the computer.

One group of verbs (the *seem* group) is able to share any kind of subject that the infinitival clause wants. These verbs are called *raising verbs*. The subject of the infinitival clause is assumed to be raised from the infinitival clause and realized syntactically by the matrix clause. The subject is assumed to have a semantic relation only to the infinitival clause.

In the other group of verbs (the *expect* group), the matrix verb has both syntactic and semantic requirements to the subject of the unexpressed subject of the infinitival clause. The subject can for instance not be an expletive, as (153a) illustrates. This group of verbs are referred to as *subject control* verbs.

I will also consider a third group of verbs that take infinitival complements, namely the *object control* verbs. These verbs link the unexpressed subject of the infinitival clause to the underlying indirect object (the arg3-role), as illustrated in (154).

(154) Mary expects John to smile.

A subject control verb like *forvente* ('expect') has the specifications in Figure 6.56. By unifying the XARG of the ARG2 with the INDEX of the ARG1 I ensure that the ARG1 of the subject control verb shares index with the unexpressed subject of the infinitival clause.¹⁴

¹⁴Since the values of the valence features are not lists, I can put such constraints on them without requiring the arguments to be realized.

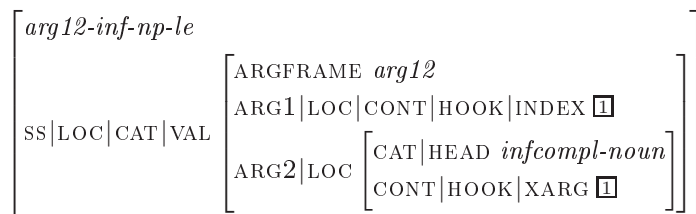


Figure 6.56: Lexical information on a subject control verb

A subject raising verb like *fortsette* ('continue') has the specifications in Figure 6.57. The difference between a subject raising verb and a subject control verb in the account presented here is that the subject raising verb has the ARGFRAME value *arg12-2*, which implies that it may have an expletive subject.

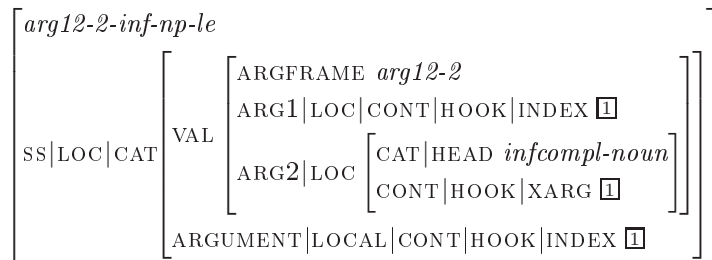


Figure 6.57: Lexical information on a subject raising verb

The verb *continue* can enter the argument frames in (155). In (155a) and (155b) the argument frame is *arg12*. (155a) has an NP object, while (155b) has an infinitival object. (155c) and (155d) have the argument frame *arg2*. (155c) is an unaccusative with an NP subject, while (155d) is a clause with an expletive subject and an infinitival clause object.

- (155) a. John continued the work.
 b. John continued to work.
 c. The work continued.
 d. It continued to rain.

There are two kinds of ditransitive verbs with infinitival clause objects. On the one hand there are verbs like *promise* where the unexpressed subject of the infinitival clause is linked to the subject of the matrix verb, irrespective of whether the matrix

clause is transitive, as in (156a), or ditransitive, as in (156b). In both the transitive and ditransitive version, the infinitival clause can be exchanged with an NP, as (156c) and (156d) illustrate.

- (156) a. John promised to work hard.
 b. John promised her to work hard.
 c. John promised a lot of things.
 d. John promised her a lot of things.

On the other hand, there are verbs like *expect*, where the unexpressed subject of the infinitival clause is linked to the subject if the matrix clause is transitive, as in (157a), and to the indirect object if the matrix clause is ditransitive, as in (157b). The infinitival clause can be exchanged with an NP only if the clause is transitive, as in (157c). If the clause is ditransitive, as in (157d), this is not possible.

- (157) a. John expected to work hard.
 b. John expected her to work hard.
 c. John expected a lot of things.
 d. * John expected her a lot of things.

Verbs like *promise* are treated as subject control verbs, (see Figure 6.56), and they are given the ARGFRAME value *arg1-12-123*. However, for the object control verbs I assume two lexical entries, one where they have the same type as the subject control verbs as in Figure 6.56 and one which inherits from the type *arg123-inf-np-le* shown in Figure 6.58.

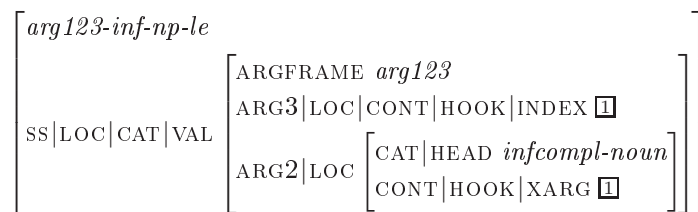


Figure 6.58: Lexical information on an object control verb

6.7.4 Remarks on raising

The fact that raising verbs are assumed to have the ARGFRAME value *arg12-2* (or *arg12-123* in the case of object raising), and thereby allowing for an underlying event *arg1-relation* to relate the predicate of the raising verb to the subject (or *arg3-relation* in the case of object raising), goes against the general assumption made in the literature, namely that the subject (or object) is raised, and therefore is not a semantic argument of the raising verb. The BRR for (152b) is given in Figure 6.59, where *John* is the argument both of the raising verb *seems* and the embedded verb *smile*.

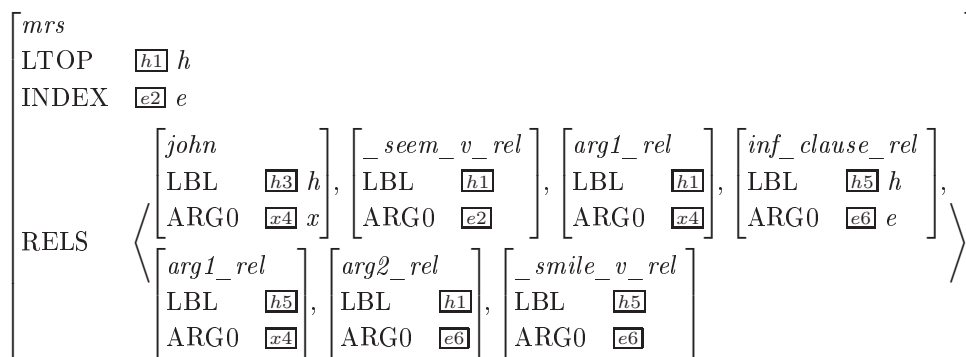


Figure 6.59: BRR of *John seems to smile*

However, this exo-skeletal approach is founded on the assumption that the argument roles of the verbs are *not* specified in the lexicon (as is the case in almost all the literature). They are assigned by the syntax.¹⁵ As argued in Sections 1.2 and 2.4, the argument roles assigned to the verbs by the syntax (as is assumed always to be the case in this approach) can be independent of the lexical meaning of the verbs. This is assumed to be the case when raising verbs “raise” full NPs and the underlying event *arg1-relation* relates the predicate of the raising verb to the subject as in (152b).

The analysis I propose for cases where the relationship between the syntax and the semantics traditionally is represented as ‘skewed’ (raising, small clauses, and resultatives), where an argument belongs semantically to one constituent (see (158)) and syntactically to another (see (159)), is that the argument in question belongs to both categories (see 160, which is an abbreviation of Figure 6.59). This assumption would correspond to a GB analysis with a PRO as subject of the controlled constituent,

¹⁵The fact that lexical entries are constrained via e.g. the ARGFRAME feature is done in order to avoid overgeneration of “odd” sentences (see Section 4.3).

which by the way is *not* how these constructions are analyzed in GB. I will come back to this issue in Section 9.5.2.

(158) seem(e1,e2)
 smile(e2,x3)
 John(x3)

(159) seem(e1,x2,e3)
 smile(e3)
 John(x2)

(160) seem(e1,x2,e3)
 smile(e3,x2)
 John(x2)

The approach does not completely exclude a traditional raising analysis where for example the raised subject is not related to the raising verb by an *arg1-relation* underlying event. A traditional raising analysis can be achieved by introducing valence rules that are not subconstructions as the rules mentioned in Section 6.1, but that rather realize an argument without realizing an underlying event, similar to the presentational rules which I will present in Section 7.2. The raising valence rules (one binary rule and one unary extraction rule) would inherit from the type *basic-rais-val* in Figure 6.60, which takes as argument an NP, and has an empty C-CONT|RELS list. The type for a subject raising verb would have the ARGFRAME value *arg2* rather than *arg12-2* (see Figure 6.57). This would give an BRR as shown in Figure 6.61 where the raised subject is an argument only of the embedded verb. The raising valence rules would be restricted only to apply in raising constructions such as subject raising and object raising and in cases of subcoordination analysed as raising constructions (to be presented in Section 8.4.3).

The analysis involving the suggested raising valence rules could also be used to give a new account of resultatives (see Section 6.4.2) and small clauses (see Section 6.7.2). The sentence *Jon maler veggen rød* ('Jon paints the wall red') is at present given the BRR in Figure 6.20, p. 165, where the object *veggen* is both the ARG2 of the verb *maler* and the ARG1 of the adjective *rød*. The sentence *Kari ser Jon lese boka* ('Kari sees John read the book') is given the BRR in Figure 6.55, p. 190, where the object *Jon* is both the ARG2 of the matrix verb *ser* and the ARG1 of the embedded verb *lese*.

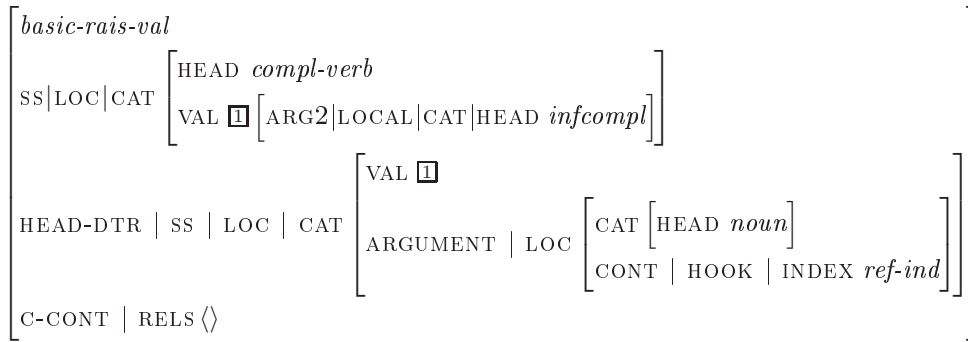


Figure 6.60: Possible type for realization of raised arguments

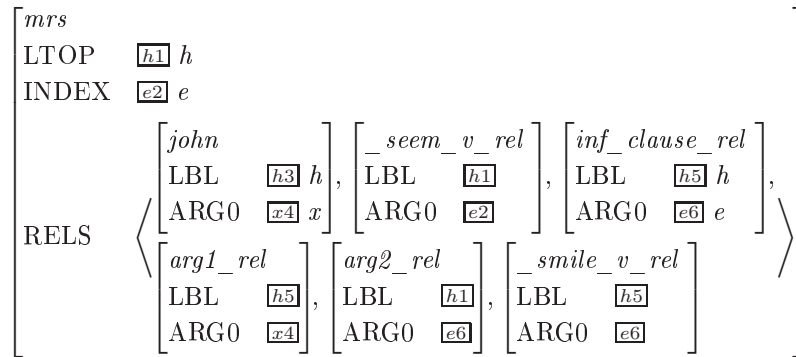


Figure 6.61: Possible BRR of *John seems to smile*

With the raising valence rules, the object in resultative clauses and clauses with a small clause would be an argument of the second predicate only, as shown in Figures 6.62 and 6.63.

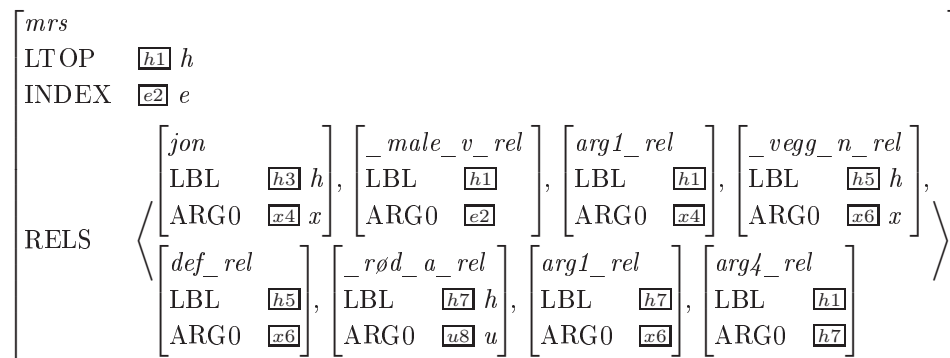


Figure 6.62: Possible BRR of *Jon maler veggen rød* ('Jon paints the wall red')

However, if the valence raising rules are added to the grammar, the BRRs produced by the grammar can no longer be seen as representations of grammatical relations of

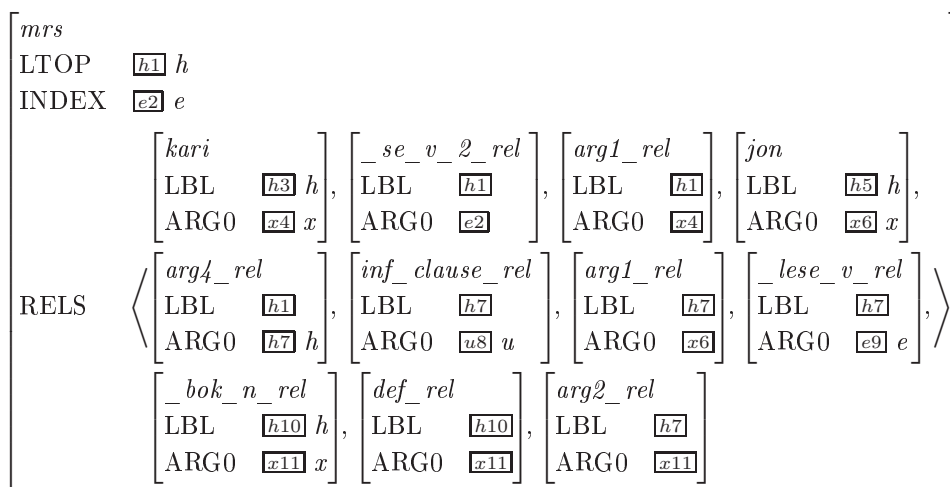


Figure 6.63: Possible BRR of *Kari ser Jon lese boka* (‘Kari sees John read the book’)

a sentence, but rather as semantic representations of a sentence. I believe semantic representations of a sentence is something that is to be inferred from the grammatical relations in a sentence in conjunction with the meaning of the words, and that it is beyond the limits of my grammar formalism. Therefore, using this kind of “empty” valence rules, sensitive to lexical information of control verbs, in an attempt to produce semantics, rather than grammatical relations, is an idea I will not pursue further.

6.8 The modifier rules

The modifier rules in Norsyg have many similarities with the modifier rule types suggested in the Grammar Matrix. Modifiers have a *local* on their MOD list where they constrain the word or phrase that they modify. I assume two kinds of modifier rules in Norsyg, the head modifier rules and the sentence adverb rules.

1. The **head modifier** rules
 - (a) The *head-mod-rule* is a head-initial rule that combines an adjunct like a PP or a relative clause with a noun or verb projection.
 - (b) The *extr-mod-rule* is an extraction rule that applies to a verb projection and extracts a modifier.
2. The **sentence adverb** rules

- (a) The *head-sadv-rule* is a head-initial rule that combines a sentence adverbial with a complementizer (subordinate, relative or infinitival) or verb projection. The CASE value of the projection is *subj-case*.
- (b) The *extr-sadv-rule* is an extraction rule that extracts a sentence adverbial on a complementizer (subordinate, relative or infinitival) or verb projection. The CASE value of the projection is *subj-case*.
- (c) The *sadv-head-rule* is a head-final rule that combines a sentence adverb with a constituent that has the CASE value *non-subj-case*.

The *head-mod-phrase* is illustrated in Figure (6.64) and accounts for modification in sentences like (161a) and (161b). In (161a) a PP is modifying a verb and in (161b) a PP is modifying a noun.

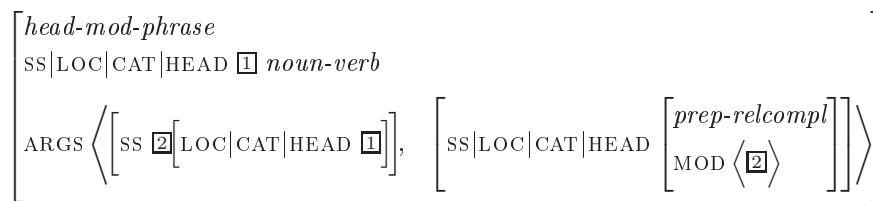


Figure 6.64: Head modifier rule

- (161) a. Jon spaserer i skogen.
 Jon walks in forest-DEF
 ‘Jon walks in the forest.’
- b. Mannen i skogen hogger ved.
 man-DEF in forest-DEF cuts wood
 ‘The man in the forest cuts wood.’

The *extr-mod-phrase* is illustrated in Figure 6.65. It extracts an adjunct that is topicalized. This rule is used in clauses like (162a) and (162b). In (162a) the extracted modifier is a PP, and in (162b) the extracted modifier is a wh-word.

- (162) a. Om ettermiddagen spaserer Jon 5 kilometer.
 in afternoon-DEF walks Jon 5 kilometers
 ‘In the afternoon Jon walks 5 kilometer.’

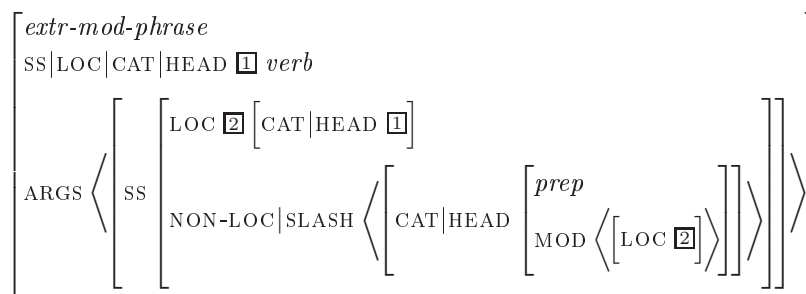


Figure 6.65: Extraction modifier rule

- b. Hvor spaserer Jon om ettermiddagen?
 where walks Jon in afternoon-DEF
 ‘Where does Jon walk in the afternoon?’

The *head-sadv-phrase* is illustrated in Figure 6.66. The modifier is a sentence adverbial, and it modifies a word or phrase with the HEAD value *compl-verb* (which generalizes over all kinds of complementizers (including the relative pronoun and the infinitival marker) + verbs and auxiliaries), and the CASE value of the modified sign is *subj-case*, which means that the projection is the head of the clause and that the merge rule has not worked (yet) in the clause. (When the merge rule applies, the CASE value is constrained to be *non-subj-case*). In (163a) *head-sadv-phrase* combines the sentence adverbial *ikke* with the projection of the verb *hogger*. In (163b) it combines *ikke* with the complementizer projection. As I will show in Chapter 10, the assumption that the head final sentence adverbial rule attaches to projections that have the feature CASE *subj-case* accounts for the position of sentence adverbials in Norwegian. There is also an extraction variant of the *head-sadv-phrase*, *extr-sadv-phrase*.

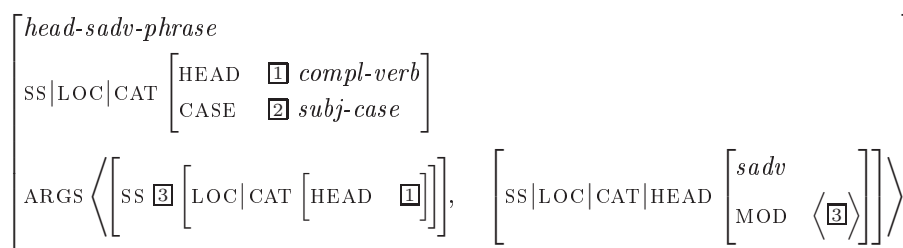


Figure 6.66: The head initial sentence adverb rule

- (163) a. *Mannen hogger ikke ved i skogen.*
 man-DEF cuts not wood in forest-DEF
 ‘The man does not cut wood in the forest.’
- b. *Jon hevder at mannen ikke hogger ved i skogen.*
 Jon claims that man-DEF not cuts wood in forest-DEF
 ‘Jon claims that the man does not cut wood in the forest.’

There is also an extraction variant of the *head-sadv-phrase*, *extr-sadv-phrase*, illustrated in Figure 6.67.

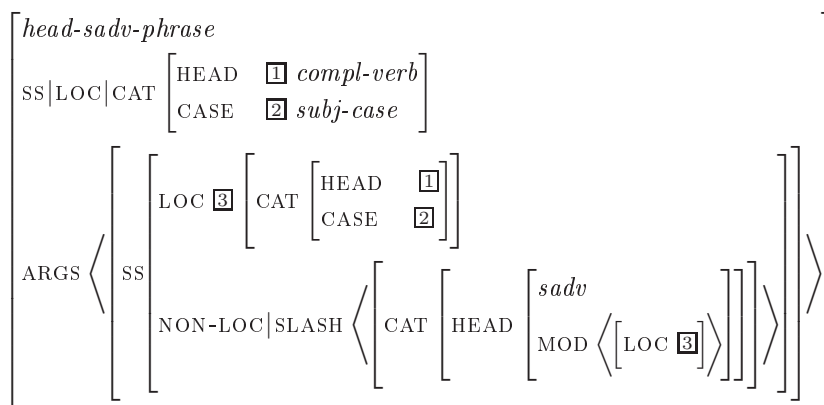


Figure 6.67: The sentence adverb extraction rule

The head final sentence adverb rule, illustrated in Figure 6.68 is used in cases where NPs or imperatives are negated, as illustrated in (164a)-(164c). In (164a) *ikke* is attached to the NP *Marit*, and in (164c) *ikke* is attached to the imperative *le*. However, the grammar does at present not account for cases like (164b) where *ikke* is attached to the infinitival clause *å le*, since the infinitival clause is not assumed to be a constituent (see Section 6.7.2).

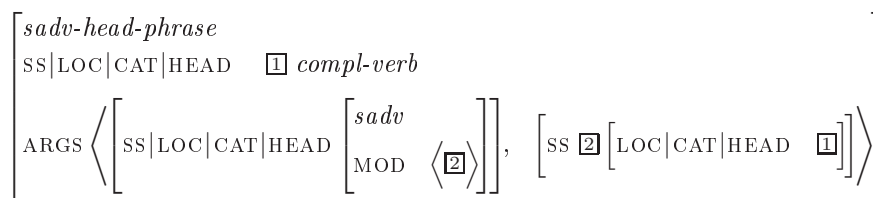


Figure 6.68: The head final sentence adverb rule

- (164) a. Jon hevder at ikke Marit vil vinne. (somebody will win, but not Marit)
 Jon claims that not Marit will win

‘Jon claims that it is not Marit that will win.’

- b. Jon prøver ikke å le. (where Jon is trying not to laugh)
 Jon tries not to laugh

‘Jon tries not to laugh.’

- c. Ikke le!
 not laugh

‘Don’t laugh!’

The example in (165) has two sentence adverbs. The first attaches to the NP *Marit* while the second attaches to the complementizer projection *at ikke Marit*. The analysis is given in Figure 6.69.

- (165) Jon hevder at ikke Marit ikke vil vinne.
 Jon claims that not Marit not will win

‘Jon claims that it is not Marit that will not win.’

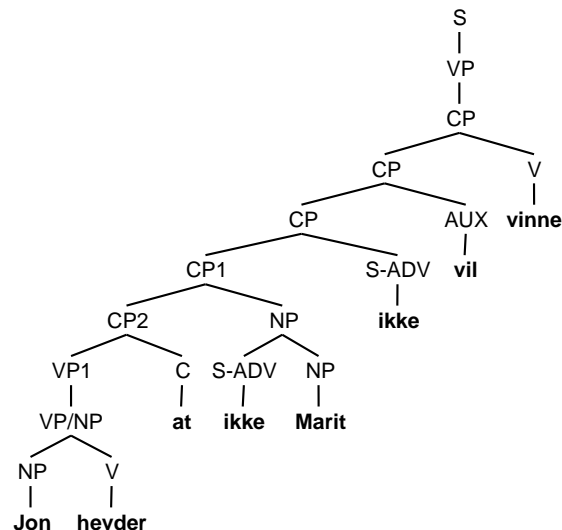


Figure 6.69: Subordinate clause with two sentence adverbs

6.9 Long distance dependencies

According to Levine (2003) there are two main approaches to long distance dependencies in HPSG. One approach stems from Pollard and Sag (1994) and involves traces or unary valence-reducing extraction rules. The other is developed in Bouma *et al.* (2001), and accounts for extraction in the lexicon by means of relational constraints. I will briefly present the two approaches before I present the approach taken in Norsyg. The new approach is necessitated by the account of relative clauses in Section 6.6.2 where the relative pronoun acts as a complementizer and a filler at the same time. The new approach is straightforward to implement, since it does not presuppose the use of relational constraints or sets, only a single list. Still, it can account for challenging data presented in Bouma *et al.* (2001) and Levine (2003) where verbs and complementizers are shown to reflect that they occur on the extraction path.

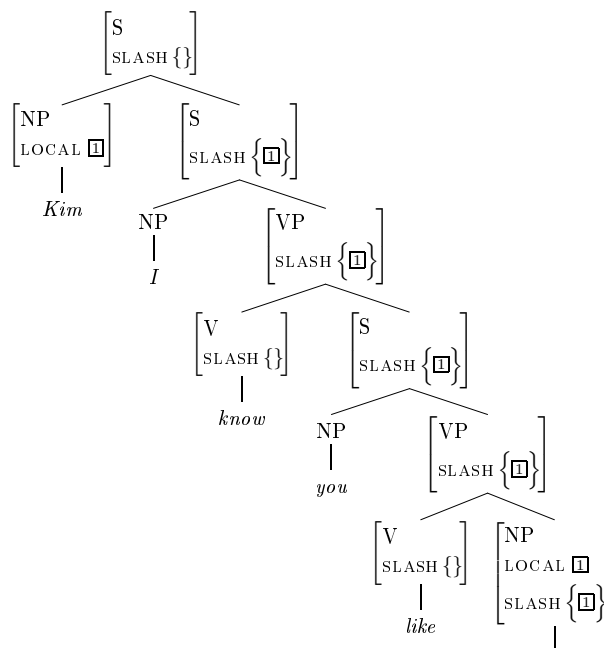
6.9.1 The *trace* approach

In Pollard and Sag (1994) extraction is accounted for with an empty element that unifies its LOCAL value with a slash. A valence rule may take this empty element as its subject or complement. In most rules (except for the head filler rule) the slashes from the daughters are collected in the mother. So the valence rule with the empty element daughter will get the slash, and so will the other rules applying higher up in the tree, until a head filler rule takes the slash and fills it in. This is illustrated in Figure 6.70.

Adjunct extraction is accounted for with a lexical rule that lets a verb with e.g. a subordinate clause on its COMPS list get a slash which is an adjunct that modifies the subordinate clause complement.

6.9.2 Reflection of extraction path

It is later pointed out that in many languages the extraction path is reflected on verbs or complementizers, and that the extracted item can be an argument or an adjunct. Sag (2005) mentions among other languages Chamorro and Irish. So a verb or a complementizer may reflect that the clause it occurs in has an extracted element. The Pollard and Sag (1994) analysis cannot account for this since it is only the empty category and its mothers that have access to the slash as the tree in Figure 6.70 illustrates.

Figure 6.70: The *trace* approach

The Irish data in (166) (originally from McCloskey (1979)) are used by Hukari and Levine (1995) and Sag (2005) among others to illustrate this phenomenon. In Irish, the choice of complementizer reflects whether the complementizer intervenes between an extraction site and the filler or not. The complementizer *goN* is not on the extraction path, while the complementizer *aL* is on the extraction path. In (166a) there is no extraction taking place, so the complementizer *goN* is used. In (166b) there are two complementizers on the extraction path. Both of them *aL*. And in (166c) there are three complementizers, all of them *aL*, on the extraction path. (166d) is an example of an NP with two complementizers, but where only one is on the extraction path. The complementizer on the extraction path is *aL* and the one occurring after the extraction site is *goN*. (166e) has three complementizers. Two on the extraction path (both *aL*), and one after the extraction site (*goN*).

The element that is extracted does not have to be a complement. It can also be an adjunct.

- (166) a. Dúirt mé **gurL** shíl mé **goN** mbeadh sé ann.
 said I goN.PAST thought I COMP would-be he there
 ‘I said that I thought that he would be there.’

- b. an fear **aL** shíl mé **aL** bheadh _ ann
 the man COMP thought I COMP would-be _ there
 ‘the man that I thought would be there’
- c. an fear **aL** dúirt mé **aL** shíl mé **aL** bheadh _ ann
 the man COMP said I COMP thought I COMP would-be there
 ‘the man that I said I thought would be there’
- d. an fear **aL** shíl _ **goN** mbeadh sé ann
 [the man]_j COMP thought _ COMP would-be he_j there
 ‘[the man]_j that thought he_j would be there’
- e. an fear **aL** dúirt sé **aL** shíl _ **goN** mbeadh sé ann
 the man COMP said he COMP thought _ COMP would-be he there
 ‘the man that he said thought he would be there’

Especially adjunct extraction is difficult to account for, since adjuncts normally do not appear in the subcat frame of the verb.

6.9.3 The lexical approach

The extraction path data made Bouma *et al.* (2001) suggest an analysis without a gap or trace (or unary valence-reducing rules). Instead, a lexeme may list all its dependents (including subjects, complements, and adjuncts that modify the KEY of the lexeme) on a DEPS list and collect the slashes from them by means of relational constraints. Then the slash goes up from head-daughter to mother until it reaches the head filler rule. If a verb has a subordinate clause complement with a slash, the relational constraints make sure that the slash of the complement also becomes the slash of the verb. In this way they can account for the registering of extraction paths. This is illustrated in Figure 6.71, where the SLASH \square enters the SLASH set of both the verbs *like* and *know*.

6.9.4 Some problems

The problem with the Pollard and Sag (1994) analysis, as I see it, is what is called the second part of the unbounded dependency analysis, namely the part where phrases collect slashes from their daughters. This part of the analysis implies that slashes go

6.9.5 The approach taken in Norsyg

In an approach where the extraction site dominates the filler, the Irish data can be accounted for without any additional machinery, since the mother (and the sister) of the complementizer will be on the extraction path. This means that the complementizer has local access to the extraction path. As I already have pointed out, there is no such straightforward account of the extraction path facts in the other approaches mentioned, where the filler is on the top of the tree.

The extraction mechanism consists of three parts:

1. The head filler rule
2. The percolation of the SLASH feature
3. The extraction rule

The filler rule (see Section 6.2) works at the bottom of the tree and fills in the extracted element. The mother of the filler rule has a SLASH list with the local information of the extracted element. (The head daughter of the filler rule has an empty SLASH list.) The SLASH list percolates up the tree from (first) daughter to mother. Finally, the SLASH list reaches the extraction site, where an extraction rule empties the SLASH list and links the extracted element to the local predicate. There are seven extraction rules, one for each of the four valence features ARG1-ARG4 (see Section 6.1), one for expletives used in presentational constructions (see Section 7.2), and two for modifiers (see Section 6.8). The general extraction phrase type is illustrated in Figure 6.72. The extraction rules are unary rules that enter a *local* into the SLASH list of the head daughter.¹⁶ This corresponds to a rule that takes a trace as argument in the Pollard and Sag (1994) analysis. The difference is that the extracted element enters the SLASH list of the head daughter and not of the mother. The SLASH list of the mother is empty.

The tree in Figure 6.73 shows how the NP in (166e) can be analyzed. Note that the mothers of the two *aL*-complementizers have a non-empty SLASH list, while the mother of the *goN*-complementizer has an empty SLASH list. That means that the extraction path is locally accessible to the complementizers that reflect that they occur on it.

¹⁶The distinction between *local* and *gap* is not necessary in Norsyg. The type *local* carries syntactic and semantic information about the sign in question.

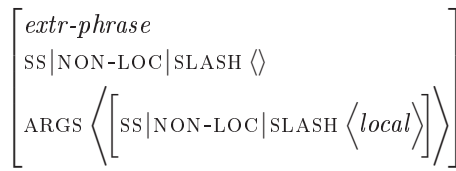


Figure 6.72: The *extr-phrase* type

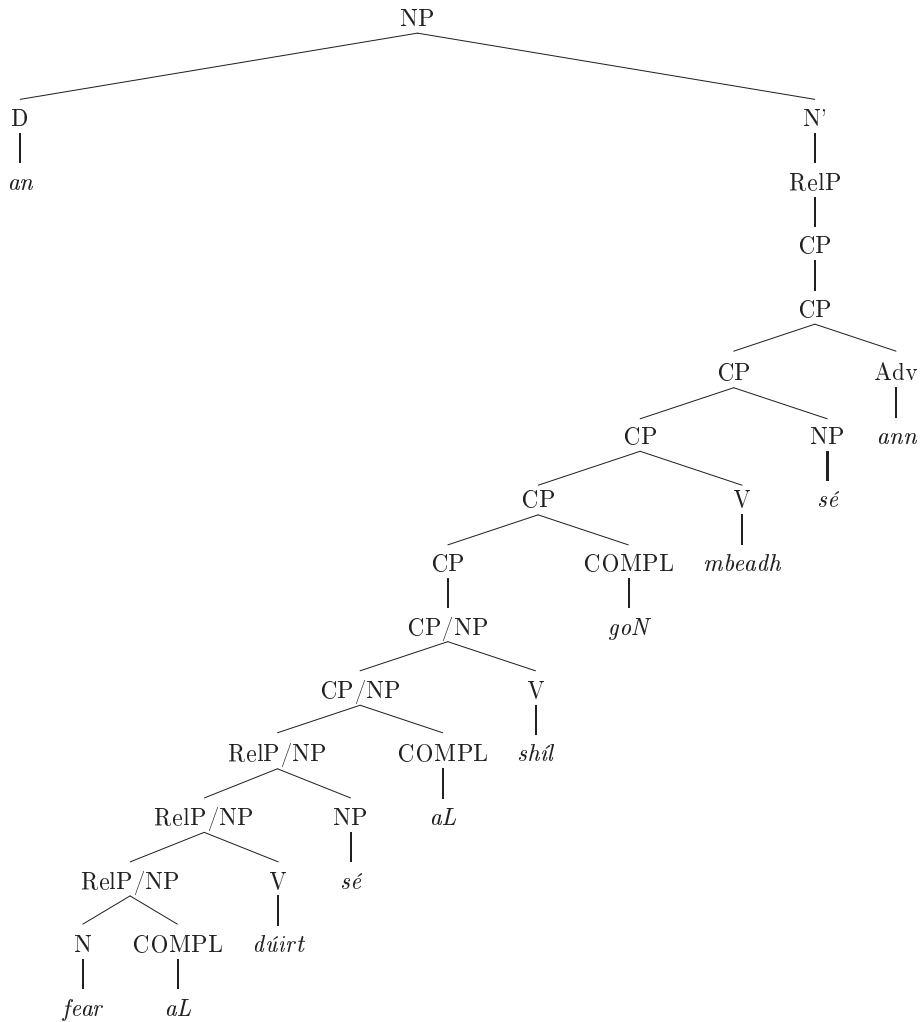


Figure 6.73: Analysis of (166e)

6.10 Summary

I have now presented how basic syntactic structures are treated in Norsyg. There are six main kinds of rules, the valence rules, the filler rule, the merge rule, the subordinating

rules, the clause boundary rules and the modifier rules. A core assumption I have made in this chapter is that the main verb may function as a modifier of a syntactic structure headed by an auxiliary, complementizer, relative pronoun, or infinitival marker. I have also assumed that the subject is realized prior to the objects either by a unary extraction rule, or by a binary valence rule in case of inversion (or by the unexpressed subject rule).

Chapter 7

Passive and Presentation

7.1 Passive

In this section I will show how the *arg1*-role of the *arg1-sign* (see Figure 3.7) also can be expressed as passive voice. I follow the assumption of Jaeggli (1986), Baker (1988) and Åfarli (1992) that there is a syntactic argument PASS, which realizes the external argument role (the role corresponding to the *arg1* subconstruction in the present approach). I also follow the assumption in the exo-skeletal approach to passive in Åfarli (2006), that the passive argument is assigned to the verb by the syntax. (See the short presentation of passive in GB/Minimalism in Section 2.5.1.)

In the *subconstruction* hierarchy in Figure 3.8 the PASS element is called *basic-pass*. I will show how *basic-pass* has two realizations in Norwegian, namely as a passive auxiliary *bli* (*bli-passive*) and as an *s*-morpheme that is attached to the main verb (*s-passive*). First I will present some data.

7.1.1 Data

In Norwegian there are two kinds of passive, periphrastic passive (*bli*-passive) and morphological passive (*s*-passive). The periphrastic passive uses the auxiliary *bli*, (see (167b)), and the morphological passive attaches the suffix *-s* to the finite main verb (see (167c)). There is a slight semantic distinction between the two forms, which I will not go into (see (Hovdhaugen, 1977, 35-39), (Engdahl, 2001) and (Engdahl, 2006)). The data I present here are well known in the literature (see e.g. (Hovdhaugen, 1977) and (Åfarli, 1992)).

- (167) a. En spiller smasher ballen.
 a player smashes ball-DEF
 ‘A player smashes the ball.’
- b. Ballen blir smashet.
 ball-DEF becomes smashed
 ‘The ball is smashed.’
- c. Ballen smashes.
 ball-DEF smash-PASS
 ‘The ball is smashed.’

In the examples (167b) and (167c), the subject (*Ballen*) would have been the direct object if the sentences were active. In (168), the three passive variants of the active clause *Jon gir Marit en is* (Jon gives Marit an ice cream) are given.

- (168) a. Marit blir gitt en is.
 Marit becomes given an ice-cream
 ‘Marit is given an ice cream.’
- b. En is blir gitt Marit.
 an ice-cream becomes given Marit
 ‘Marit is given an ice cream.’
- c. Det blir gitt Marit en is.
 it becomes given Marit an ice-cream
 ‘Marit is given an ice cream.’

In (168a) what would have been the indirect object in an active clause is the subject. In (168b) what would have been the direct object in active is the subject, and in (168c) the expletive *det* is the subject.

It is also possible for a prepositional object to function as a subject in a passive clause. In (169a), the active object functions as subject. In (169b) the expletive *det* functions as subject. In (169c) the prepositional object functions as subject.¹ The fact that this is a subject and not a topicalized NP is illustrated in (169d) which is

¹At present I do not have an analysis of “deep” prepositional objects functioning as subject in Norsyg.

an inverted version of (169c). The NP ‘barna’ has to be subject because it comes in the position after the finite verb. It should also be noted that both the prepositional construction (see (169f)) and the realization of the prepositional object as subject (see (169g)) do not go with an internal argument (arg2-role) that is definite.

- (169) a. Bleier ble byttet på barna.
 nappies were changed on children-DEF
 ‘Nappies were changed on the children.’
- b. Det ble byttet bleier på barna.
 it was changed nappies on children-DEF
 ‘Nappies were changed on the children.’
- c. Barna ble byttet bleier på.
 children-DEF were changed nappies on
 ‘Nappies were changed on the children.’
- d. Ble barna byttet bleier på?
 were children-DEF changed nappies on
 ‘Were nappies changed on the children?’
- e. Bleiene ble byttet på barna.
 nappies-DEF were changed on children-DEF
 ‘The nappies were changed on the children.’
- f. * Det ble byttet bleiene på barna.
 it was changed nappies-DEF on children-DEF
 ‘The nappies were changed on the children.’
- g. * Barna ble byttet bleiene på.
 children-DEF were changed nappies-DEF on
 ‘The nappies were changed on the children.’

7.1.2 The passive types

As I showed in Figure 3.8, I let the type *basic-pass* be a subtype of *arg1-sign*. *basic-pass* unifies the value of OUT with the value of CAT and the value of IN|VAL with the value of VAL-B. This means that *basic-pass* has the constraints in Figure 7.1.

$$\left[\begin{array}{l} \textit{basic-pass} \\ \text{SS|LOC|CAT } \boxed{1} \left[\begin{array}{l} \text{HEAD|VAL-B } \boxed{2} \left[\text{ARG1|LINK } \textit{arg1+} \right] \\ \text{VAL|ARG1|LINK } \textit{arg1-} \end{array} \right] \\ \text{IN | VAL } \boxed{2} \\ \text{OUT } \boxed{1} \\ \text{MEANING } \textit{arg1-relation} \end{array} \right]$$

Figure 7.1: The *basic-pass* type

basic-pass has two subtypes, *pass-aux-lxm* and *s-pass-word*. *pass-aux-lxm* is the type for the passive auxiliary, and it unifies the value of MEANING with a second relation on the RELS list, as Figure 7.2 illustrates. There are two differences between the passive auxiliary *bli* and the other non-modal auxiliary *ha* (‘have’) (see Figure 6.30, p. 173):

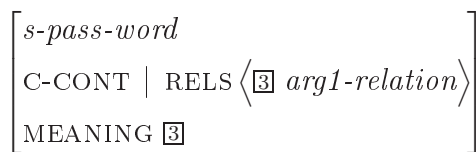
1. *bli* switches the *arg1+* value in VAL-B to *arg1-* in VAL (see Figure 7.1), while *have* unifies the VAL and VAL-B values.
2. *bli* has an additional *arg1-relation* on the RELS list (see Figure 7.2).

$$\left[\begin{array}{l} \textit{pass-aux-lxm} \\ \text{SS | LOC | CONT | RELS } \langle \textit{aux-relation}, \boxed{3} \textit{arg1-relation} \rangle \\ \text{MEANING } \boxed{3} \end{array} \right]$$

Figure 7.2: The *pass-aux-lxm* type

The other subtype of *basic-pass*, namely *s-pass-word*, is an inflectional rule that adds an *s*-morpheme to main verbs. It unifies the value of MEANING with a relation in C-CONT, as Figure 7.3 illustrates. There are two differences between the passive morpheme *-s* and the morpheme for present tense *-r*, that I want to mention here:

1. *-s* switches the *arg1+* value in VAL-B to *arg1-* in VAL (see Figure 7.1), while *-r* unifies the VAL and VAL-B values.
2. *-s* has an *arg1-relation* on the C-CONT|RELS list, while the C-CONT|RELS list of *-r* is empty (see Figure 7.3).

Figure 7.3: The *s-pass-word* type

Since the passive auxiliary and the passive inflection absorb the *arg1*-role of the verb, the subject must be realized by an element that does *not* have the *arg1*-role.²

7.1.3 Analysis

The tree in Figure 7.4 shows in detail how linking is done in a passive transitive clause with the auxiliary *bli*. There are two signs that do linking in the tree. The passive auxiliary adds an *arg1-relation* and shifts the *arg1*- link type in VAL to *arg1+* in VAL-B. This ensures that the *arg1* subconstruction is realized. The *arg2*-extr-phrase adds an *arg2-relation* that it links to the extracted *local* and shifts the *arg2*- link type in the mother to *arg2+* in the daughter. This realizes the *arg2* subconstruction. The tree shows how all the link types *arg1+*, *arg2+*, *arg3-* and *arg4-*, and the ARGFRAME value *arg1-12*, end up in the VAL-B of the auxiliary. The unification of these types (which I have omitted in this illustration) gives the type *arg12*.

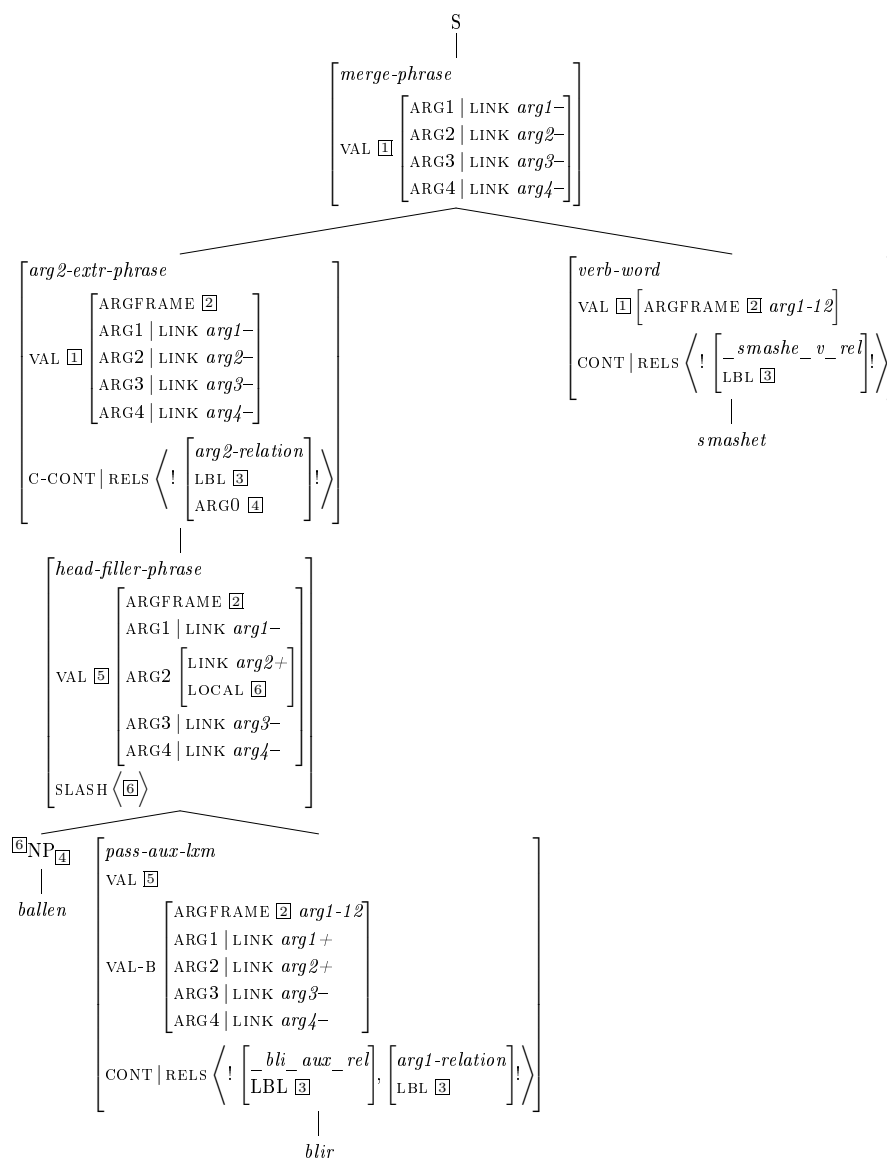
The tree in Figure 7.5 illustrates how linking is done in clauses with *s*-passive. The passive morphology (*s-pass-word*) adds an *arg1-relation* in C-CONT and changes the *arg1*- link in VAL to *arg1+* in VAL-B. This realizes the *arg1* subconstruction. The *arg2*-extr-phrase adds an *arg2-relation* that it links to the extracted subject and changes the *arg2*- link type in the mother to *arg2+* in the daughter. This realizes the *arg2* subconstruction. Now the verb word has the linking types *arg1+*, *arg2+*, *arg3-* and

²It has been brought to my attention that in Yucatec Maya the verb corresponding to *learn* may have the following chain of suffixes: V – PASS – CAUS – PASS, and that the meaning corresponds to *being taught*, as illustrated in (clxx) (Müller, 2006). A possible approach to such examples would be to assume that there are two argument frames, one for the learning predicate and one for the causative morpheme, and that the two passive morphemes each realize an *arg1*-role.

(clxx) k=u ká`an -s -á`al le teòria-o`
 INCOMPL=3.ERG learn.PASS -CAUS -PASS.IMPV Det theory-D1

‘The theory is being taught.’

(Somebody causes that the theory is being learned)

Figure 7.4: Passive ditransitive sentence with the auxiliary *bli* (BRR: D.28, p. 342)

$arg4-$, and the ARGFRAME value $arg1-12$, in VAL-B. When these types are unified, we get the type $arg12$.

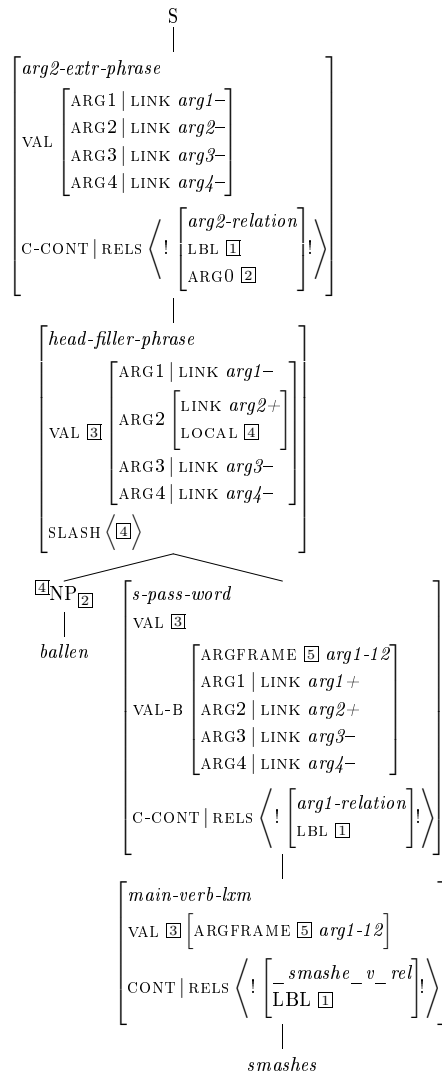


Figure 7.5: Passive sentence with morphological passive (BRR: D.29, p. 343)

7.2 The presentational construction

7.2.1 Some data

Presentational constructions involve an expletive *det* subject, and a direct object that can function as subject in a corresponding non-presentational clause (see Áfarli and Eide (2003, 226–237)). In the examples in (171) the verb is the unaccusative *komme* (‘come’). The examples in (172) have a transitive verb *beundre* (‘admire’) in passive voice. In the a-examples, there is no presentational construction, and the argument of the arg2 subconstruction *mannen* (‘the man’) functions as subject and can be definite. In the b-examples, there is a presentational construction, and the argument of the arg2 subconstruction functions as direct object. As the c-examples show, the direct object in a presentational construction has to be indefinite.³

(171) a. Mannen kommer.

man-DEF comes

‘The man comes.’

b. Det kommer en mann.

it comes a man

‘A man comes.’

c. * Det kommer mannen.

it comes man-DEF

(172) a. Mannen blir beundret.

man-DEF becomes admired

‘The man is admired.’

b. Det blir beundret en mann.

it becomes admired a man

‘A man is admired.’

c. * Det blir beundret mannen.

it becomes admired man-DEF

³See Footnote 7 (p. 65).

7.2.2 The presentational rules

In order to account for presentational constructions, I introduce two presentational rules, one binary head initial rule and one extraction rule. Unlike the other valence rules, these rules do not do any linking. So the VAL value in the daughter is unified with the VAL value of the mother, and the RELS list in C-CONT is empty. On the other hand, they have two constraints that the other valence rules do not have, namely that the ARGUMENT of the head daughter has *subj-case*, and that the cognitive status of the ARG2 is *type-id* (type identifiable).⁴ The basic presentation phrase is represented in Figure 7.6.

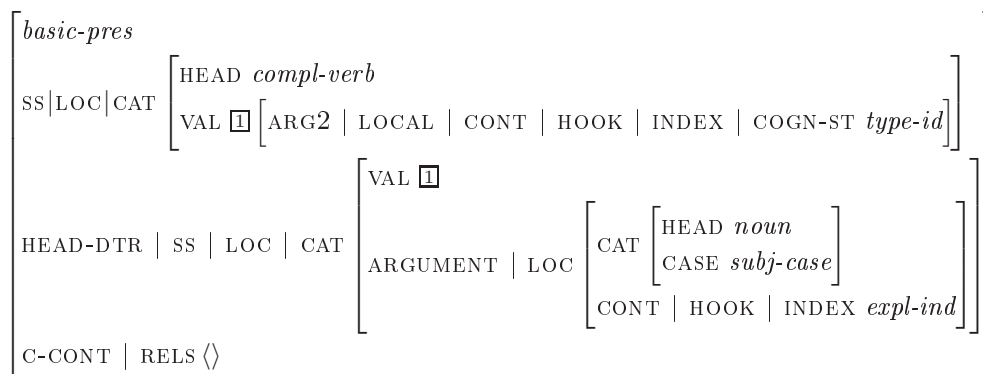


Figure 7.6: The *basic-pres* type

The constraints on ARGUMENT, *noun*, *subj-case* and *expl-ind* (expletive index), ensure that the argument must be an expletive. They also ensure that the function of the expletive is subject. The constraint on ARG2 ensures that the direct object (if it is realized) must have the cognitive status *type identifiable*. A definite noun, which is *uniquely identifiable*, is not compatible with *type identifiable*, and so the data in (171) and (172) are accounted for.

7.3 Summary

In this chapter I have presented an analysis of passive and presentation in Norwegian, accommodating the basic facts about these constructions in the framework proposed. Passive is seen as a syntactic element that realizes the arg1 subconstruction. Norwegian

⁴The use of cognitive statuses to restrict the distribution of nominals is discussed in Borthen and Haugereid (2005).

has two types of passive, periphrastic passive with the auxiliary *bli* and morphological passive with the morpheme *-s*. A small type hierarchy was introduced, with types for the passive auxiliary (*pass-aux-lxm*) and the passive morpheme (*s-pass-word*). Generalizations over the two types was done in the type *basic-pass*.

The Norwegian presentational construction was assumed to be a construction which is not a subconstruction, but which realizes an expletive *det* as the subject. It constrains the argument of the *arg2* subconstruction (if it is realized) to have the cognitive status *type-id* (type identifiable).

Chapter 8

Coordination

In this chapter I will have a look at coordination, and show how the phrasal subconstructions and the Basic Relation Representations they express (see Section 3.4) are suited for cases of coordination with for example no one-to-one correspondence between the number of verbs and the apparent number of argument frames (and argument roles). I will consider four kinds of coordination:

- Coordination of VPs
- Coordination of Vs
- Ellipsis
- Pseudo-coordination (including Sub-coordination and the Empty Object Construction)

The analysis of coordinated VPs, coordinated Vs, and pseudo-coordination is implemented in Norsyg. The analysis of ellipsis is not implemented.

8.1 Coordination of VPs

8.1.1 Data

The example in (173) is usually analysed as a sentence with two coordinated VPs, where the subject is realized after the two VPs have formed a constituent.

- (173) Marit spiser en is og drikker kaffe.
 Marit eats an ice-cream and drinks coffee
 ‘Marit eats an ice cream and drinks coffee.’

Both of the conjuncts can be negated, and the negation only has scope over the conjunct it occurs in. So in (174a) the negator has scope over the eating event, in (174b) the negator has scope over the drinking event, and in (174c) the negators have scope over each their event.

- (174) a. Marit spiser ikke is og drikker kaffe.
 Marit eats not ice-cream and drinks coffee
 ‘Marit doesn’t eat ice cream and drinks coffee.’
- b. Marit spiser is og drikker ikke kaffe.
 Marit eats ice-cream and drinks not coffee
 ‘Marit eats ice cream and doesn’t drink coffee.’
- c. Marit spiser ikke is og drikker ikke kaffe.
 Marit eats not ice-cream and drinks not coffee
 ‘Marit doesn’t eat ice cream and doesn’t drink coffee.’

In examples of coordinated VPs, the coordinated events share one argument (the subject of the sentence). In the constructional approach taken in this thesis, the argument which is shared cannot be not realized by a single subconstruction. In (175) the subject *Marit* has an *arg1*-relation to the predicate in the first conjunct, and an *arg2*-relation to the predicate in the second conjunct. This rules out an analysis where the subject is realized after the two events are conjoined, since the subject then would be realized by only one subconstruction, and have the same relation to the two predicates.

- (175) Marit spiser is og blir servert kaffe.
 Marit eats ice-cream and is served coffee
 ‘Marit eats ice cream and is served coffee.’

8.1.2 Analysis

In my analysis of “coordinated VPs”, I will assume that the two conjuncts are clauses with independent argument frames. The subject is an argument of both of frames.

The tree in Figure 8.2 illustrates the syntactic structures assumed for these cases. The structures are left-branching. The left-branching structures are necessitated by the assumption that the extracted element is realized at the bottom of the tree. In order for the extracted subject of the second conjunct to be accessible for the first conjunct, the second conjunct has to dominate the first. The mother of the coordinator (VP/NP) *coord-vp-rule* is a rule that binds the two conjuncts together. The first conjunct is the first daughter, and the second conjunct is built on top of it.

The type for this rule is illustrated in Figure 8.1. It shows that the rule is a binary head-initial rule which takes a constituent where all the arguments are realized as its first daughter (all the linking types are negative) and a conjunction word as its second daughter. The rule unifies the TOPIC of the first daughter with an element on the SLASH list of the mother. This ensures that the topic of the first conjunct is realized by an extraction rule in the second conjunct (which is built on top of the mother).

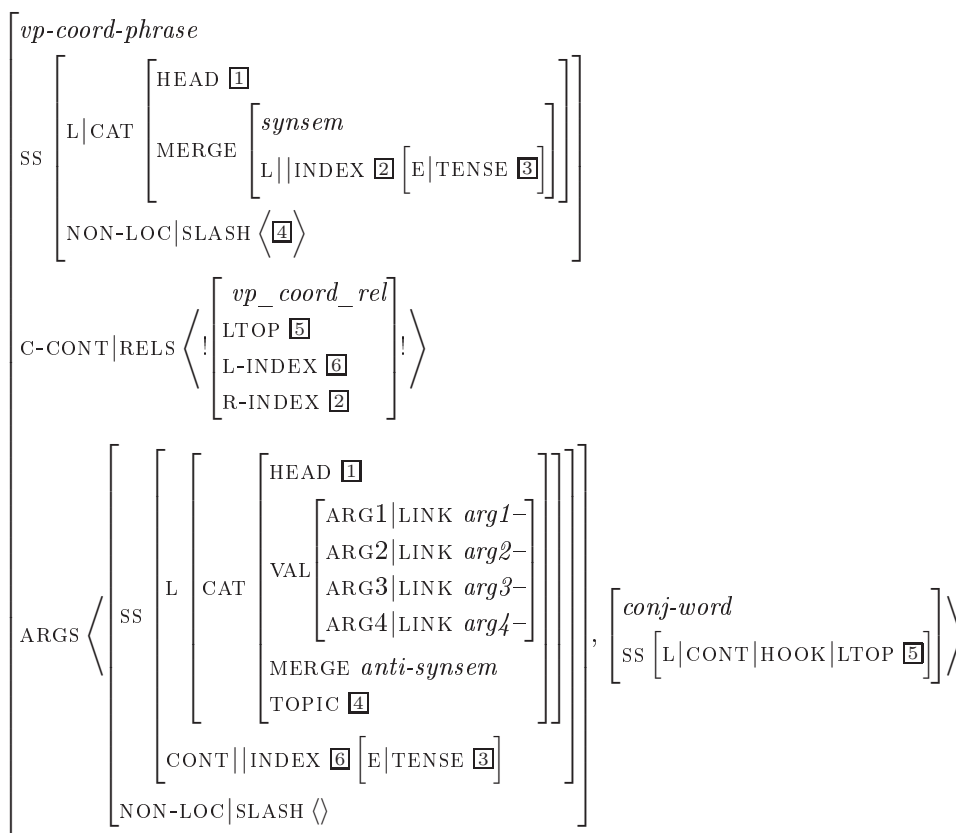


Figure 8.1: Type for coordination of VPs

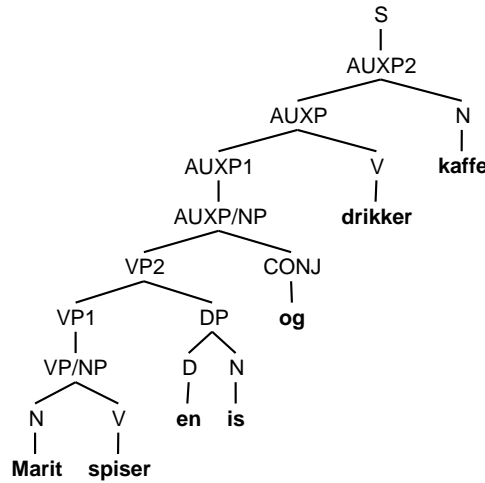


Figure 8.2: Coordination of two VPs (BRR: 8.3, p. 222)

The BRR of example (173) is given in Figure 8.3. It shows that the sentence has two relations. One *arg12*-relation for the eating event (*_spise_v_rel*) linked together by the handle *h5* and one *arg12*-relation for the drinking event (*_drikke_v_rel*) linked together by the handle *h1*. The *arg1*-role of the eating relation is also the *arg1*-role of the drinking relation. The indices of the two events are arguments of the *vp_coord_rel*.

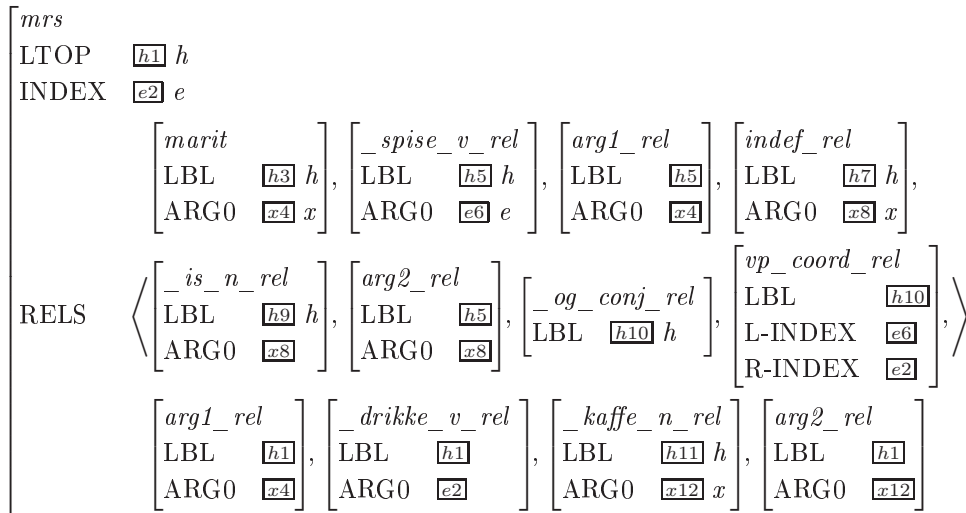


Figure 8.3: BRR of coordination of two VPs (Tree: 8.2, p. 222)

An analysis of coordinated VPs where the subjects of the two clauses are realized by different subconstructions (see (175)) is given in Figure 8.4.

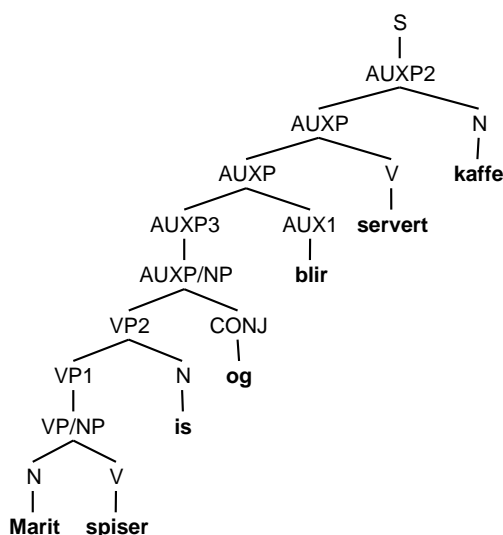


Figure 8.4: Coordination of active and passive VPs (BRR: 8.5, p. 224)

The BRR that results from the analysis of (175) is shown in Figure 8.5. Here, the two events *_spise_v_rel* ('eat') and *_servere_v_rel* ('serve') have different relations to their shared argument *Kari*. *_spise_v_rel* relates to *Kari* via the underlying event *arg1_rel*, while *_servere_v_rel* relates to *Kari* via the underlying event *arg3_rel*.

8.2 Coordination of Vs

8.2.1 Data

The second kind of coordination is illustrated in (176) where the subject *Marit* is catching, frying and eating the fish. The order of the verbs determines the order of the events.

- (176) *Marit fanger, steker og spiser fisken.*
 Marit catches, fries and eats fish-DEF
 'Marit catches, fries and eats the fish.'

It only seems to be possible to have a negator in the position after the last verb as in (178a). The negator then negates the whole series of events. If the negator comes in

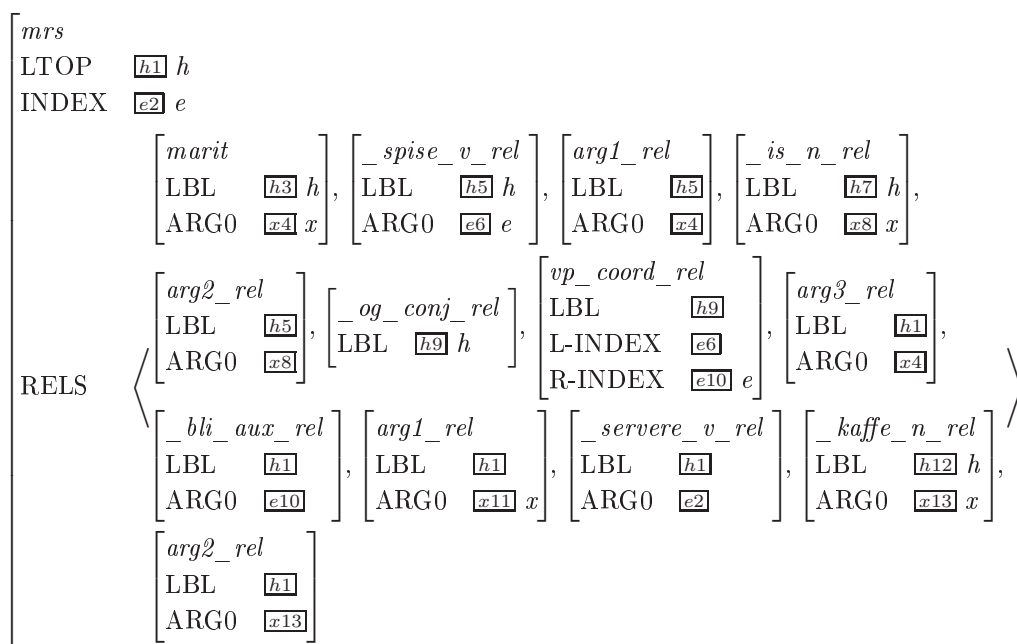


Figure 8.5: BRR of coordination of active and passive VPs (Tree: 8.4, p. 223)

between the verbs as in (178b) and (178c), the sentence is ungrammatical.¹²

- (178) a. ? Marit fanger, steker og spiser ikke fisken.
 Marit catches, fries and eats not fish-DEF

‘Marit doesn’t catch, fry and eat the fish.’

- b. * Marit fanger, steker ikke og spiser fisken.
 Marit catches, fries not and eats fish-DEF

¹I assume that the examples in this section express complex events. If only one of the conjuncts of one of the examples gets modified, then I assume that the object is extraposed, and that the clause is a coordination of VPs. The clause then does not express a single event, but rather one event per verb.

²Example (178a) may sound a bit odd. It is maybe better illustrated with the negative polarity item *noenting* (‘anything’) as in (clxxvii). Here it becomes clearer that the negator modifies the whole cluster of Vs since the negative polarity item only can be the object of verbs that are in the scope of a downward entailing item like the negator.

- (clxxvii) a. Marit fanger, steker og spiser ikke noenting.
 Marit catches, fries and eats not anything

‘Marit doesn’t catch, fry and eat anything.’

- b. * Marit fanger, steker ikke og spiser noenting.
 Marit catches, fries not and eats anything
- c. * Marit fanger ikke, steker og spiser noenting.
 Marit catches not, fries and eats anything

- c. * Marit fanger ikke, steker og spiser fisken.
 Marit catches not, fries and eats fish-DEF

In subordinate clauses the negator comes before the coordinated verbs, as illustrated in (179).

- (179) at Marit ikke fanger, steker og spiser fisken.
 that Marit not catches, fries and eats fish-DEF
 ‘that Marit doesn’t catch, fry and eat the fish.’

Example (180) illustrates how the two coordinated verbs must have the same argument frame. The verb *vente* (‘await’/‘wait for’) may enter both an arg12-frame and an arg23-frame. In (180) the arg-12-frame is the only option since the verb *admire* only can enter the arg12-frame.

- (180) ? En overraskelse venter og beundrer ham.
 a surprise awaits and admires him
 ‘A surprise waits for him and admires him.’

8.2.2 Analysis

The data in Section 8.2.1 indicate that the coordinated verbs should be treated as a complex event with a single argument frame. This is achieved by coordinating the verbs before they combine with any other entities. The coordination rules used for coordination of Vs all inherit from the type *coord-unsat-phrase* illustrated in Figure 8.6.

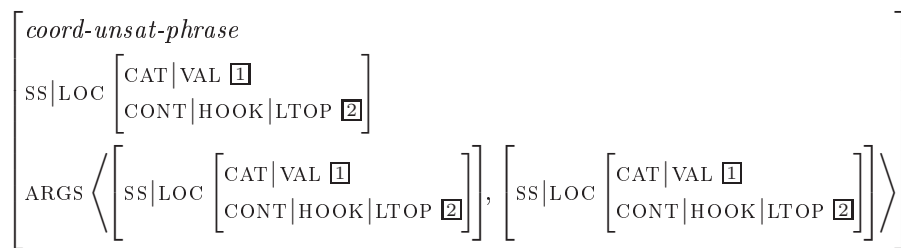


Figure 8.6: The type *coord-unsat-phrase*

The *coord-unsat-phrase* type unifies the VAL values and the LTOP values of the conjuncts. The result is that the coordinated verbs share one argument frame. Since the valence requirements of the verbs are unified, the rules will not coordinate verbs with conflicting valence requirements. The tree in Figure 8.8 shows an analysis of a sentence with three coordinated verbs.

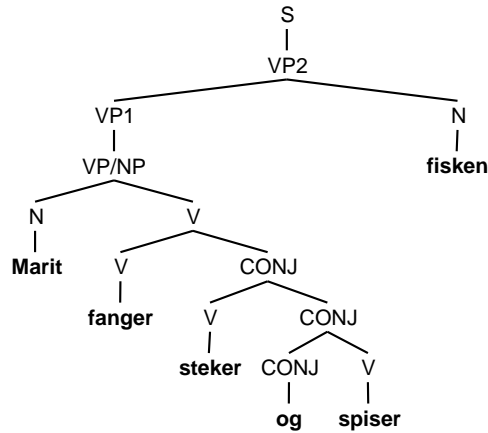


Figure 8.7: Coordination of Vs (BRR: 8.8, p. 226)

As the tree shows, the three verbs *fanger* (‘catches’), *steker* (‘fries’), and *spiser* (‘eats’) form a constituent which acts as a single verb. The semantics of (176) is illustrated in (8.8).

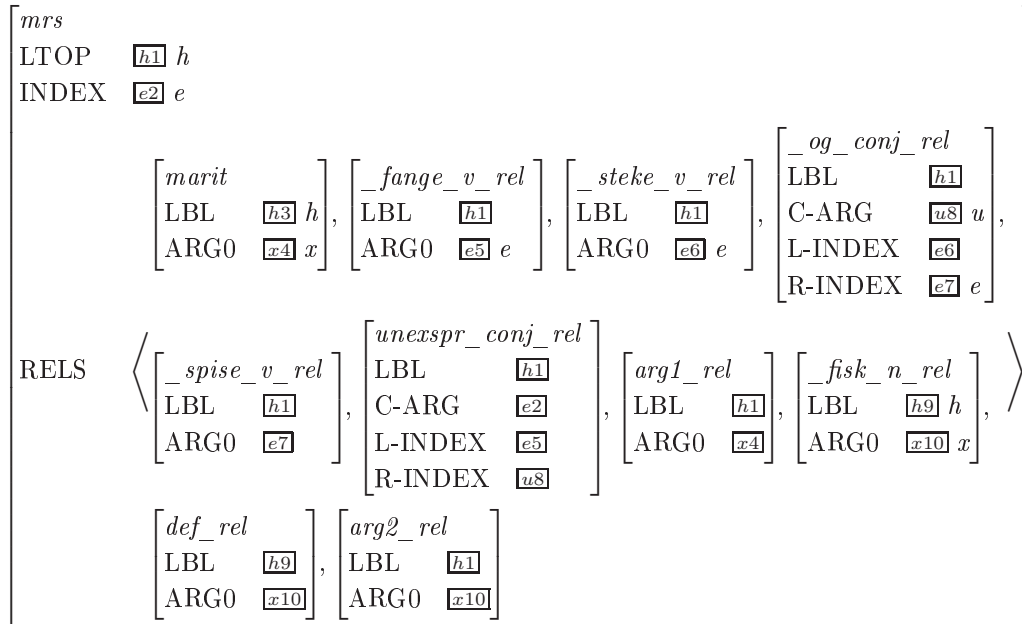


Figure 8.8: BRR of coordination of Vs (Tree: 8.7, p. 226)

The BRR in Figure 8.8 has three sub-events, a catching event, a frying event and an eating event. These events are conjoined with conjunction-relations *_og_conj_rel* and *unexpr_conj_rel*. The events have the same label (*h1*). There is only one *arg1_rel*

and one *arg2_rel*. These two linking relations hold between the three verb predicates on the one side (*h1*) and the subject (*x4*) and the object (*x10*) on the other side.

In a lexicalist approach the three verbs would have had their arguments linked in the lexicon, as illustrated for HPSG in Figure 2.1 (p. 28). That means that there would necessarily have been three argument frames (or relations) in a sentence like (178a), and not just one. With the current approach involving phrasal subconstructions, the linking of the arguments is delayed. This makes it possible to assume just one argument frame and one relation.

8.3 Ellipsis

In the previous section, I gave an analysis of coordinated Vs where it was assumed that several predicates could share one argument frame. In this section, I show that the opposite can also be the case. In sentences like (181), there is only one predicate, but more than one argument frame. In (181a) the subject of the two conjuncts *Marit* is shared, while in (181b) the conjuncts have separate subjects *Marit* and *Kari*.

(181) a. Marit gir Jon en is og Ola en sjokolade.
 Marit gives Jon an ice-cream and Ola a chocolate
 ‘Marit gives Jon an ice cream and Ola a chocolate.’

b. Marit gir Jon en is og Kari Ola en sjokolade.
 Marit gives Jon an ice-cream and Kari Ola a chocolate
 ‘Marit gives Jon an ice cream and Kari gives Ola a chocolate.’

The proposed BRR of (181b) is illustrated in Figure 8.9. The representation has just one *give_rel*, but it has two argument frames. In the first frame, *Marit* has the arg1-role *Jon* has the arg3-role and *ice cream* has the arg2-role. In the other frame *Kari* has the arg1-role *Ola* has the arg3-role and *chocolate* has the arg2-role. The two argument frames are linked to the *give* relation by the *conj_rel*.

8.4 Pseudo-coordination

This section addresses two kinds of coordination called *sub-coordination* and the *Empty Object Construction*.

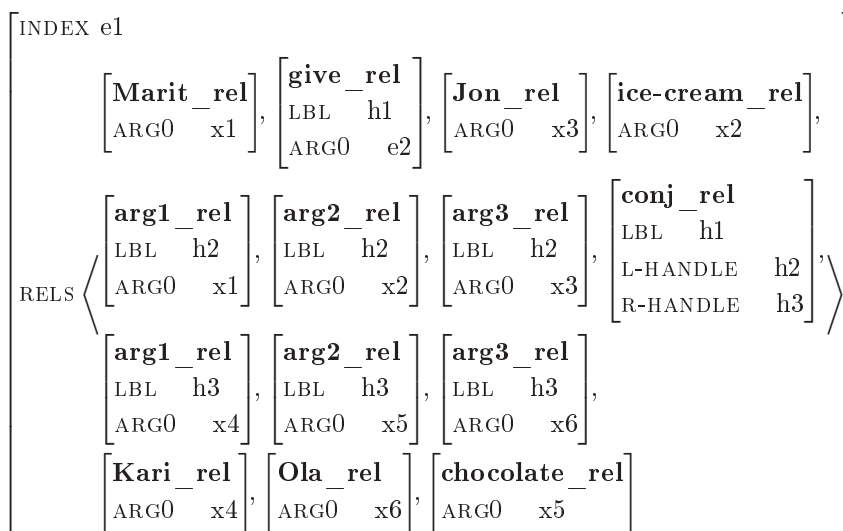


Figure 8.9: Proposed BRR of coordination with ellipsis

8.4.1 Sub-coordination

Lødrup (2002) presents three kinds of sub-coordination in Norwegian, illustrated in (182) (taken from Lødrup (2002, 121)). The first kind of sub-coordination is given in (182a), where the first verb is a positional verb like *sit* ('sit'), *stå* ('stand'), *ligge* ('lay') and *være* ('be'), movement verbs like *komme* ('come'), *gå* ('walk'), verbs of assuming a position like *sette seg* ('sit down') and *legge seg* ('lay down'), and communication verbs like *ringe* ('phone'). Lødrup analyzes these cases of sub-coordination as control constructions. (The first verb governs the unexpressed subject of the second verb.) This implies a biclausal construction.

- (182) a. Han sitter og skriver dikt.
 he sits and writes poems
 'He is writing poetry.'
- b. Han driver og skriver dikt.
 he carries-on and writes poems
 'He is writing poetry.'
- c. Han tok og skrev et dikt.
 he took and wrote a poem
 'He wrote a poem.'

1. A modal in the second conjunct blocks extraction, while a modal in the first conjunct does not.
2. No (modal, time) adverbs can occur in the second conjunct, while they can occur in the first.
3. There can be no subject in the second conjunct.
4. Only the coordinator *og* ('and') can be used. Other coordinators *men* ('but') and *eller* ('or') cannot be used.
5. There are semantic restrictions on what verbs can occur in each of the conjuncts.

8.4.2 The Empty Object Construction

Certain dialects in Norwegian have the Empty Object Construction (EOC), illustrated in (185), (see Creider and Åfarli (1987), Johnsen (1988) and Larson (2005)).

- (185) Han skrev et brev og sendte til England.
 he wrote a letter and sent to England
 'He wrote a letter_i and sent it_i to England.'

In clauses like (185) both conjuncts are understood to have at least an argument corresponding to the realization of an arg1-role and an argument corresponding to the realization of the arg2-role. But neither the arg1-role nor arg2-role can be expressed in the second conjunct. If the arg2-role is expressed like in (186), then the reference of this argument is not bound to be the same as the reference of the arg2-role of the first conjunct. It is then a case of coordinated VPs rather than an EOC.

- (186) Han skrev et brev og sendte det til England.
 he wrote a letter and sent it to England
 'He wrote a letter_i and sent it_{i/j} to England.'

Johnsen (1988) analyzes EOCs as compound verbs that are part of the same VP. Similarly, I will assume that the two conjuncts in an EOC share one argument frame.

8.4.3 Analysis

I have made three rules in order to account for the two kinds of biclausal sub-coordination (see (184a) and (184b)), the monoclausal sub-coordination (see (184c)), and the Empty Object Construction (see (185)).

The supertype for the pseudo-coordination constructions is given in Figure 8.10. It introduces a relation that holds between the event index of the first conjunct and the event index of the second conjunct. The first daughter is the head daughter, and the non-head daughter is the conjunction word *og* ('and'). The tense value of the head daughter is unified with the tense of the second conjunct. (The value of MERGE will be unified with the finite verb of the second conjunct.)

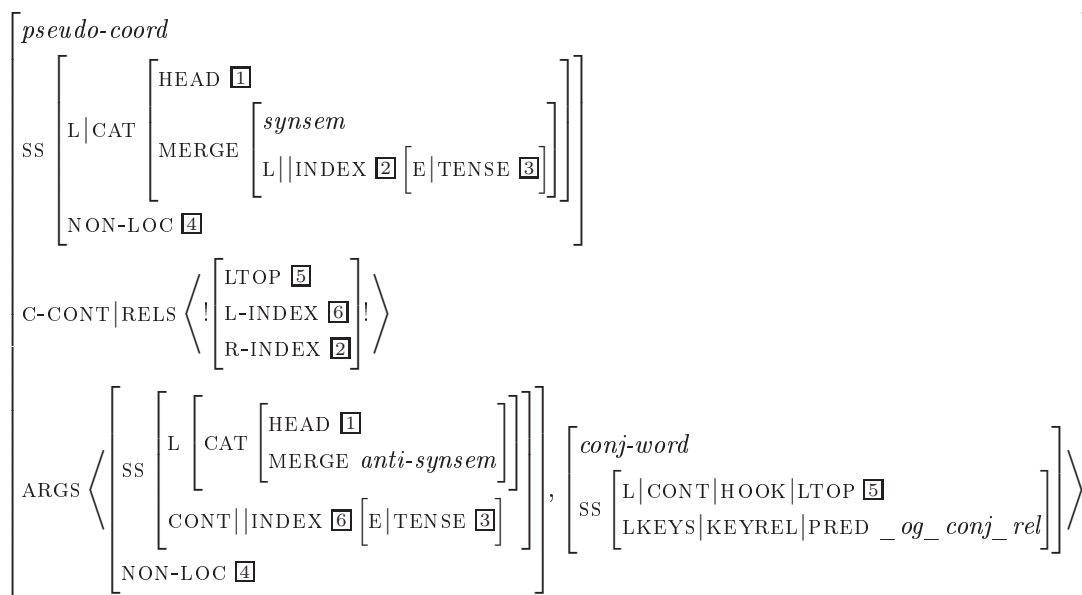


Figure 8.10: Type for pseudo-coordination

The type *pseudo-coord* has three subtypes, *bicl-subcoord* (biclausal sub-coordination), *monocl-subcoord* (monoclausal sub-coordination), and *eoc-coord* (Empty Object Construction).

The type for the biclausal sub-coordination is illustrated in Figure 8.11. It constrains the head daughter to have only negative linking types. This means that all arguments of the first conjunct are realized, and the second conjunct is assigned a separate argument frame.⁴ The index of the first argument of the second conjunct (the unexpressed

⁴The type *bicl-subcoord* inherits from the type *uni-link* which unifies all the link types (see Section

subject) is linked to the index of the ARG2 of the first conjunct.

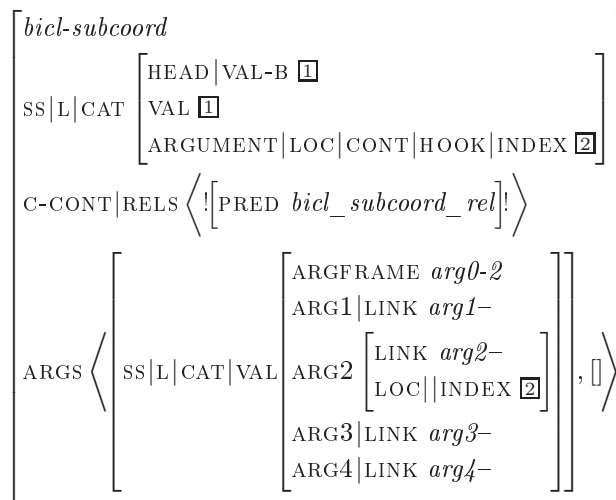


Figure 8.11: Type for biclausal sub-coordination

The trees in Figures 8.12 and 8.13 show analyses where the biclausal subconstruction rule is employed. The tree in Figure 8.12 is an analysis of (184a), and the tree in Figure 8.13 shows an analysis where the object *dikt* ('poems') is fronted. In both trees, the rule for biclausal coordination is the rule that has the conjunct as its right daughter. The rule that takes the biclausal construction rule as input is the (unary) unexpressed subject rule (see Figure 6.48, p. 184). It has the same function in biclausal subcoordination as in small clause constructions (see Section 6.7.2), namely to realize the unexpressed subject and make the index of the unexpressed subject available to the matrix clause via the ARGUMENT feature. The BRR that the analyses in Figures 8.12 and 8.13 produce is given in Figure 8.14.

The construction for biclausal sub-coordination is assumed to hold both for the control type and the raising type of biclausal sub-coordination, pointed out in Lødrup (2002). As with the raising and control sentences (see Section 6.7.3), the difference between these constructions is assumed to be lexical. While the verbs that enter a control construction has a lexical requirement for the controlled argument, the verbs that enter the raising construction have an optional argument. The ARGFRAME value of the first conjunct in *bicl-subcoord* is constrained to be *arg0-2*, which means that the

A.6.1). This is done in a constituent where all the subconstructions of the clause are yet to apply (from a bottom-up perspective).

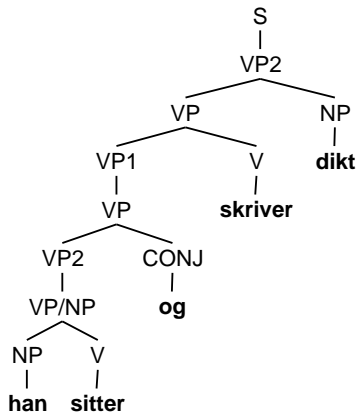


Figure 8.12: Bicausal sub-coordination (BRR: 8.14, p. 233)

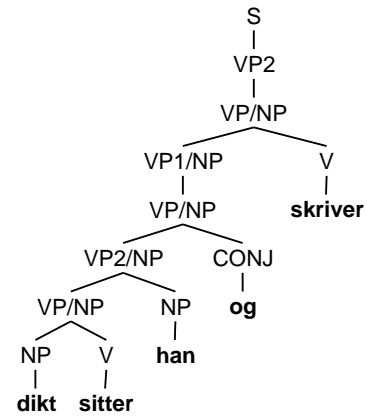


Figure 8.13: Topicalization from second conjunct in a sentence with bicausal sub-coordination (BRR: 8.14, p. 233)

subject is either a full NP (if it is realized by the *arg2* subconstruction), or an expletive (if it is realized by the presentational rule). The control verbs do not have the *arg0* frame as an option, and the controlled argument must be a full NP. Raising verbs on the other hand, have the *arg0* frame as an option (on my account) and may end up with no link to the second conjunct.

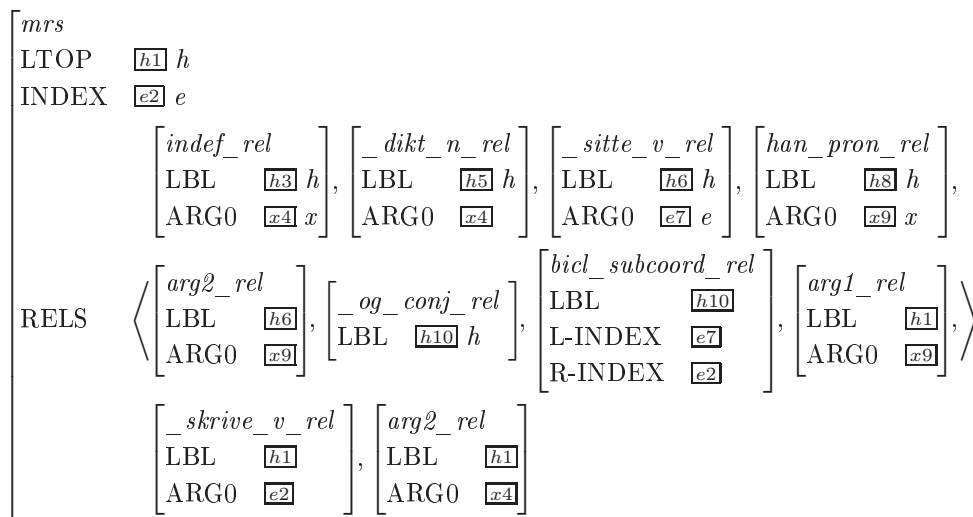


Figure 8.14: BRR of *Han sitter og skriver dikt* ('He is writing poetry') bicausal sub-coordination (Trees: 8.12, p. 233 and 8.13, p. 233)

The BRR in (8.14) has two predicate relations, *_sitte_v_rel* and *_skrive_v_rel*,

that are bound together to a complex predicate by the *bicl_subcoord_rel*. The sub-coordination relation takes the index of the first verb as its first argument (L-INDEX), and the index of the second verb as its second argument (R-INDEX). Each of the predicates are associated with an argument frame. The argument frame of *_sitte_v_rel* is linked together with the handle *h6* and the argument frame of *_skrive_v_rel* is linked together with the handle *h1*.

The type for the monoclausal sub-coordination is illustrated in Figure 8.15. The VAL value is unified with the VAL value of the head daughter. This means that the second conjunct continues to build the valence frame that was started by the first conjunct. The constraints on the LINK values of the head daughter ensure that the arg1 subconstruction and only the arg1 subconstruction has been employed in the first conjunct.⁵ The rest of the arguments are realized in the second conjunct.

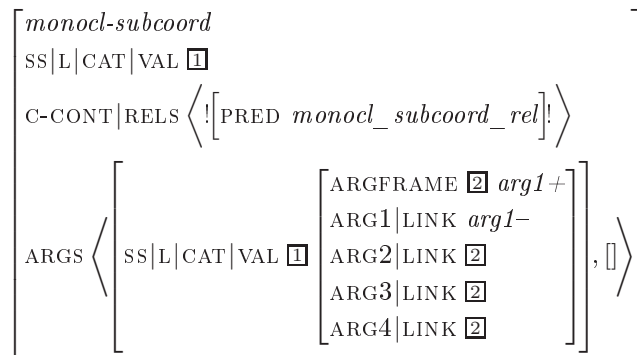


Figure 8.15: Type for monoclausal sub-coordination

The trees in Figures 8.16 and 8.17 show analyses where the monoclausal subconstruction rule is employed. The tree in Figure 8.16 is an analysis of (184c), and the tree in Figure 8.17 shows an analysis where the object *et dikt* ('a poem') is fronted. The BRR that these analyses produce is given in Figure 8.18. In both trees, the rule for monoclausal coordination is the rule that has the conjunct as its right daughter.

The BRR in (8.18) has two predicate relations, *_ta_v_rel* and *_skrive_v_rel*, that are bound together to a complex predicate by the *monocl_subcoord_rel*, which takes the index of the first verb as its first argument, and the index of the second verb as its second argument. The complex predicate has one argument frame that is linked

⁵See Section A.6.1 for an account of the linking mechanism.

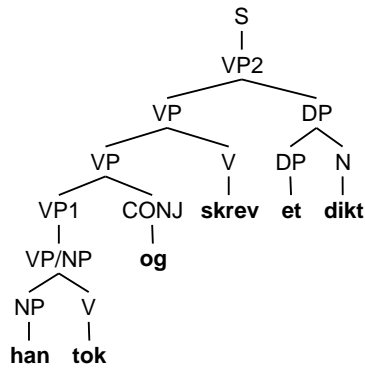


Figure 8.16: Monoclausal sub-coordination (BRR: 8.18, p. 235)

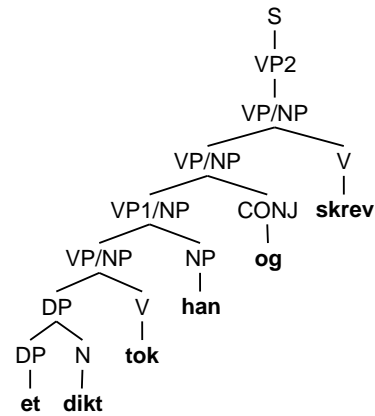


Figure 8.17: Topicalization from second conjunct in a sentence with monoclausal sub-coordination (BRR: 8.18, p. 235)

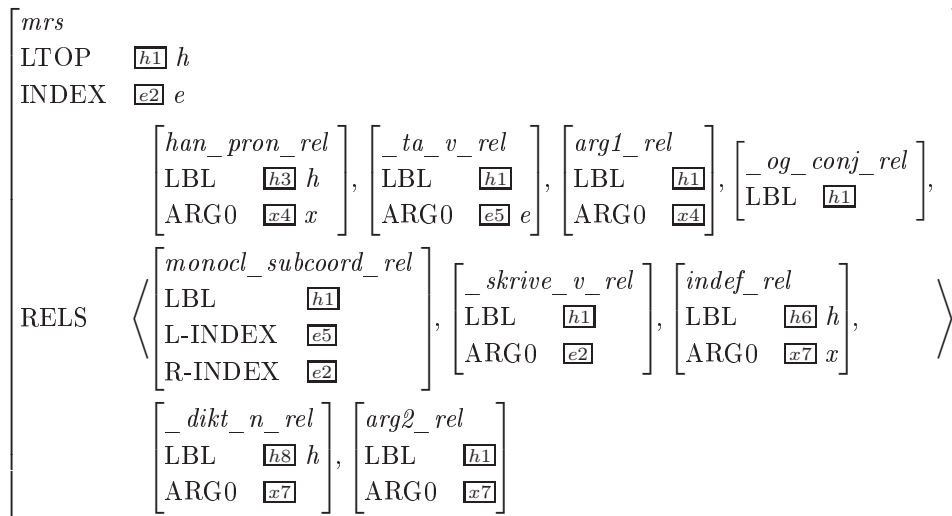


Figure 8.18: BRR of *Han tok og skrev et dikt* ('He wrote a poem'), monoclausal sub-coordination (Trees: 8.16, p. 235, and 8.17, p. 235)

together by the handle *h1*.

The type for the Empty Object Construction is illustrated in Figure 8.19. As in the type for the monoclausal sub-coordination, the VAL value is unified with the VAL value of the head daughter, and also here the second conjunct is assumed to continue to build the valence frame that was started by the first conjunct. The constraints on the LINK

values of the head daughter ensure that both the *arg1* subconstruction and the *arg2* subconstruction have been employed in the first conjunct. The rest of the arguments are realized in the second conjunct.

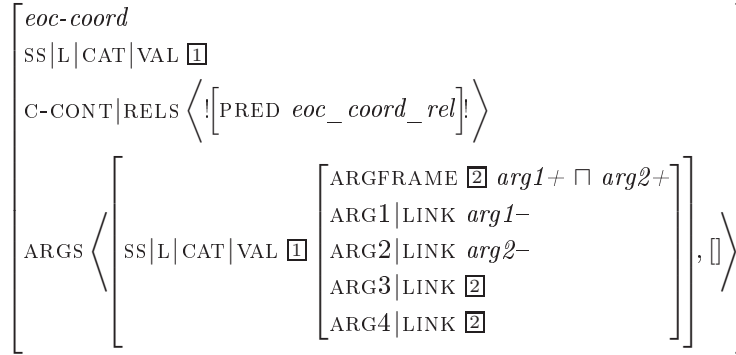


Figure 8.19: Type for the Empty Object Construction

The trees in Figures 8.20 and 8.21 show analyses where the Empty Object Construction rule is employed. The tree in Figure 8.20 is an analysis of (185), and the tree in Figure 8.21 shows an analysis where the object *et brev* (‘a letter’) is fronted. The BRR that these analyses produce is given in Figure 8.22. In both trees, the rule for the Empty Object Construction is the rule that has the conjunct as its right daughter.

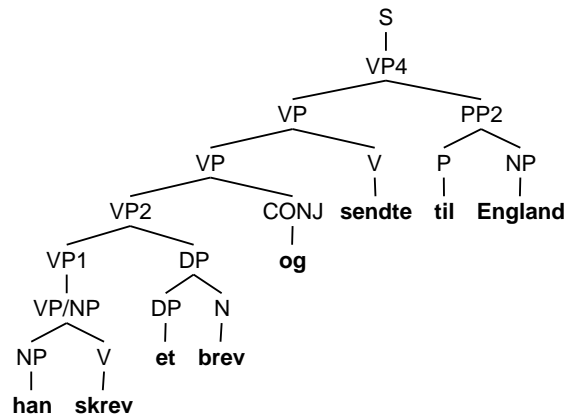


Figure 8.20: The Empty Object Construction (BRR: 8.22, p. 238)

The BRR in (8.22) has two predicate relations, *_skrive_v_rel* and *_sende_v_rel*, that are bound together to a complex predicate by a *eoc_subcoord_rel*, which takes the index of the first verb as its first argument, and the index of the second verb as

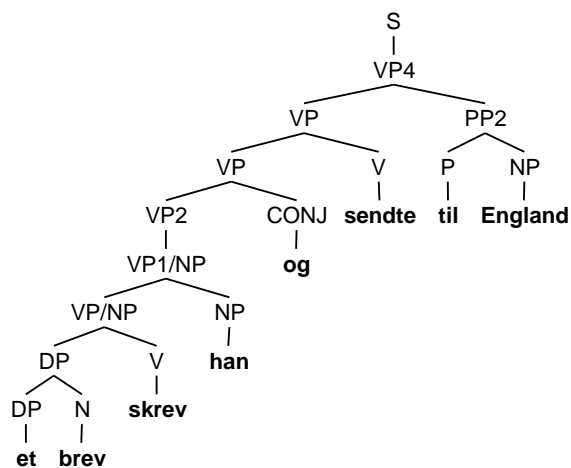


Figure 8.21: Topicalization from first conjunct in a sentence with the Empty Object Construction (BRR: 8.22, p. 238)

its second argument. The complex predicate has one argument frame that is linked together by the handle *h1*.

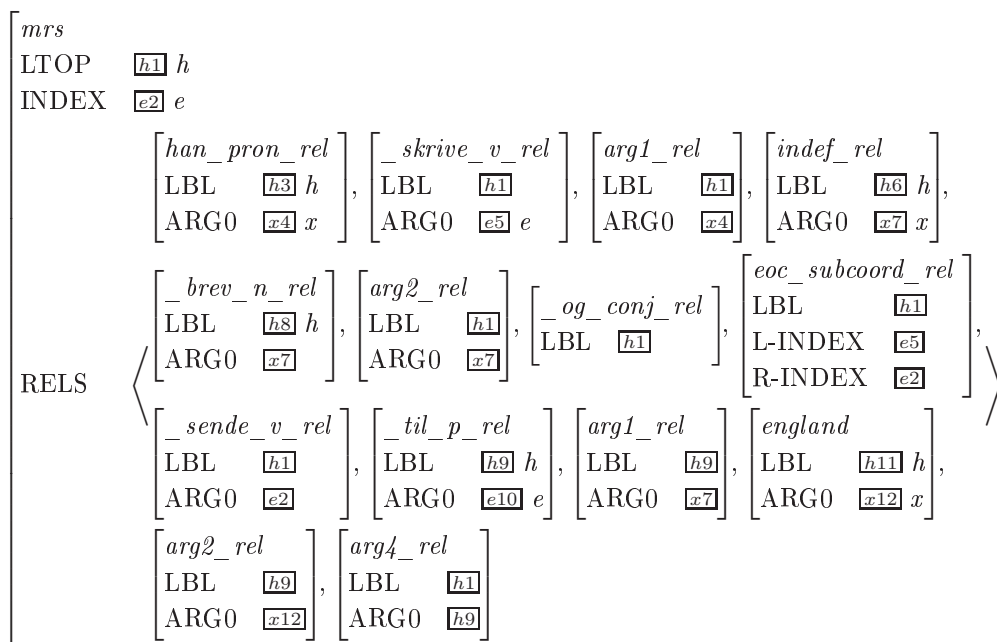


Figure 8.22: BRR of *Han skrev et brev og sendte til England* ('He wrote a letter and sent it to England), Empty Object Construction (Trees: 8.20, p. 236 and 8.21, p. 237)

8.5 Summary

In this chapter I have discussed four kinds of coordination in Norwegian, coordination of VPs, coordination of Vs, ellipsis, and pseudo-coordination. The focus has been on how the semantic representations look. The approach involving phrasal subconstructions has shown to have the flexibility that is needed in order to express that several predicates may be associated with one and the same argument frame, as illustrated in Figure 8.8. I assume that this is the case in coordination of Vs and in cases of pseudo-coordination. The phrasal subconstruction approach also allow several argument frames to be associated to one and the same predicate, as illustrated in Figure 8.9. I assume that this is the case in elliptic constructions, which have not been fully analyzed and implemented.

This chapter has again demonstrated cases which have been successfully analyzed and accommodated by the over-all approach advocated in this thesis.

One further area of Norwegian syntax will be given a demonstration. However, as this is an area where GB-analyses have so far been the more prevalent, I devote the

next chapter to a general comparison between my framework and the GB framework.

Chapter 9

Comparison with the Government and Binding theory

The analysis I have presented in Chapter 6 have certain similarities with a Government and Binding analysis. Both theories are suited for incremental parsing, GB in a left-corner parsing strategy, and Norsyg in a bottom-up, left-to-right parsing strategy. In this chapter I will compare my analyses of basic Norwegian clauses with GB analyses. I will use the comparison to show how the analysis can be extended to English.¹

The GB analysis I will use includes the two clausal categories TP and CP, which have been standard in the GB literature since Chomsky (1986). The clausal categories are shown in Figure 9.1. Here, VP is the projection of the verb, TP is the projection of Tense, and CP is the projection of C (Complementizer or Case). Movement operations to the minimal and maximal projections of TP and CP are in GB used to account for clausal word order. Movement to the minimal projections (T and C) is called head movement. If a minimal projection is free, it is possible for a verb to raise to this position. A verb may raise from V to T to receive Tense. If there is no complementizer in the C position, the verb will continue to C. If the T position is taken by an auxiliary, the verb stays in V. Movement to the maximal projections (that is, to the specifier position of T and C) is called DP movement or Wh movement.

¹Norsyg contains two demo grammars. One for English and one for German. The grammars are meant to illustrate how basic word order in English and German can be accounted for given the approach in this thesis. The grammars can be loaded with the ‘/norsyg/lkb/eng-script’ file and the ‘/norsyg/lkb/ger-script’ file, respectively. Information about what phenomena that are covered and batch tests are given in Appendix B.

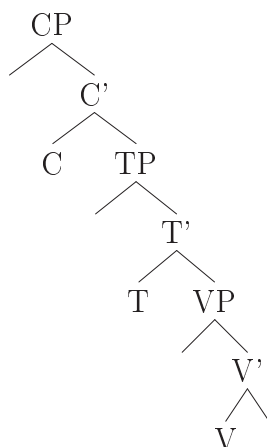


Figure 9.1: Clausal categories in GB

9.1 GB as presented in Carnie 2007

One common assumption in the GB literature is that English has what is referred to as *affix lowering*. Languages with affix lowering has the word order *Subj often V O*, as shown in (187). Languages that do not have affix lowering are assumed to have what is referred to as the $V \rightarrow T$ movement. Languages with this movement have the word order *Subj V often O*.

(187) John often eats ice cream.

The difference is explained by means of the *verb movement parameter*: Verbs raise to T or T lowers to V. In a language like English, the parameter is set to affix lowering, as illustrated in Figure 9.2, where the tense moves down to the main verb.

Main verbs are assumed to be blocked from moving to T in English, but auxiliaries are allowed in this position. This is illustrated for (188) in Figure 9.3 where the auxiliary *has* is positioned in T.

(188) John has often eaten ice cream.

The fact that main verbs are blocked from moving to T in English, is used to explain why subject verb inversion only applies to auxiliaries in English. Subject verb inversion applies when a verb moves from T to C. Since main verbs are blocked from moving to T, they also cannot move to C.

Subject verb inversion is assumed to take place in yes-no-questions. The examples in (189) illustrates that only auxiliary verbs can undergo subject verb inversion in English.

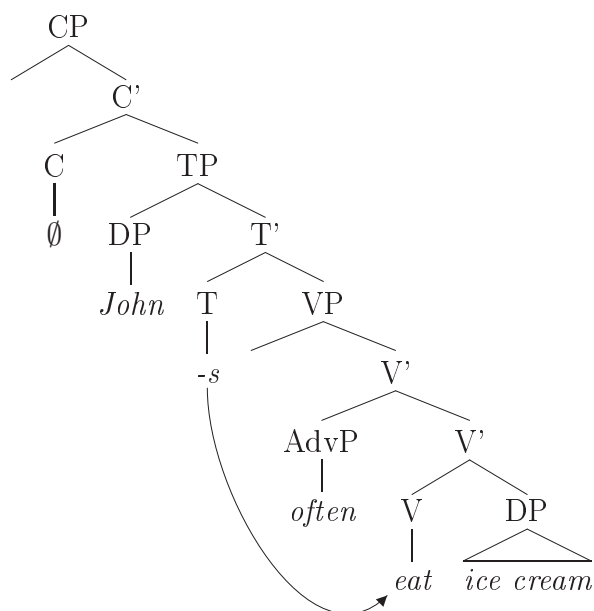


Figure 9.2: Affix lowering in English

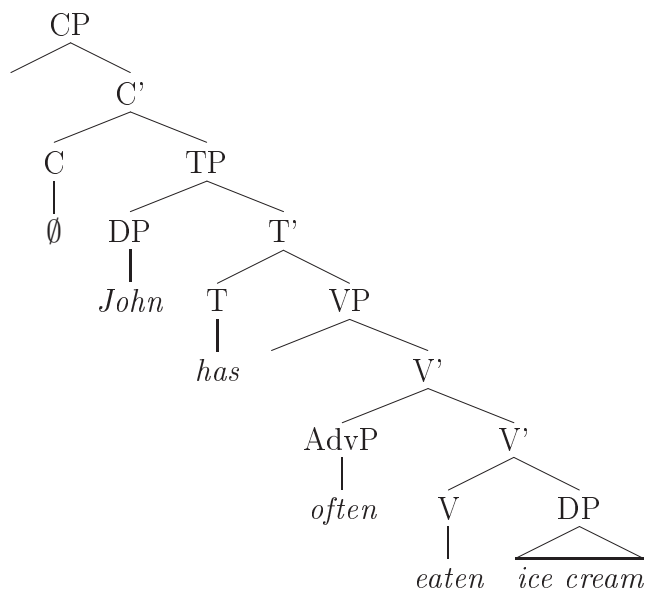


Figure 9.3: Main clause with auxiliary

If the clause does not have an auxiliary, the dummy auxiliary *do* is inserted, as in (189c). In Norwegian, both auxiliaries and main verbs can undergo subject verb inversion, as shown in (190).

(189) a. Has John eaten ice cream?

- b. * Eats John ice cream?
 c. Does John eat ice cream?
- (190) a. Har Jon spist is?
 has Jon eaten ice-cream
 ‘Has Jon eaten ice cream?’
- b. Spiser Jon is?
 eats Jon ice-cream
 ‘Does Jon eat ice cream?’

The analysis of (189a) is given in Figure 9.4, where the auxiliary *has* undergoes movement from T to C.

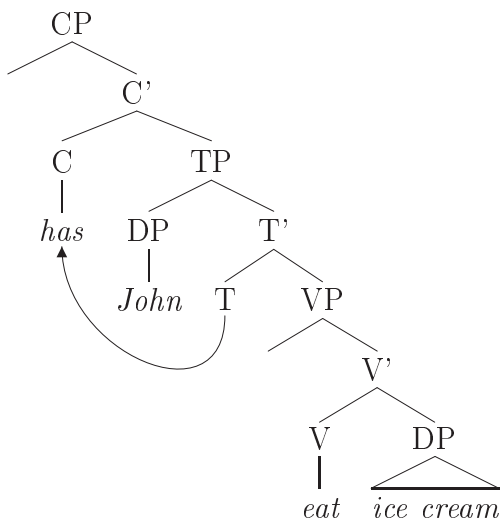
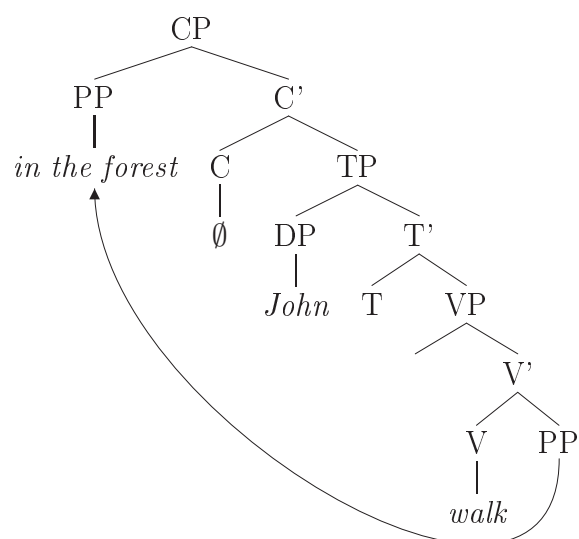


Figure 9.4: GB analysis of *Has John eaten ice cream?*

In a GB analysis of topicalization the topicalized phrase is assumed to be moved to the specifier position of C. This is illustrated in Figure 9.5 where the adverbial *in the forest* is moved out of the VP and into the specifier position of C. If there is a verb in T, it will move to C. Since main verbs are blocked from moving to T, auxiliaries are the only verbs that can occur in C. This accounts for the ungrammaticality of (189b), where a main verb appears before the subject.

Figure 9.5: GB analysis of *In the forest John walks*

9.2 A GB analysis based on Norwegian data

In a GB analysis of Norwegian, given in Åfarli and Eide (2003), sentence adverbials are assumed to attach to the T projection, rather than V'.² This is one way to account for the position of sentence adverbials in subordinate clauses. As (191a) and (191b) show, the sentence adverbial *ofte* comes before the finite verb, while it comes after the finite verb in main clauses (see (191c)). Given the assumption that the finite verb in Norwegian moves to T, the sentence adverbial cannot be adjoined to V', as shown in Figure 9.6.³

- (191) a. at Jon ofte spiser epler
 that Jon often eats apples
 ‘that Jon often eats apples’
- b. at Jon ofte har spist epler
 that Jon often has eaten apples
 ‘that Jon often has eaten apples’
- c. Jon spiser ofte epler.
 Jon eats often apples
 ‘John often eats apples.’

Another difference between Norwegian and English is the fact that Norwegian is a V2 language. In Norwegian, when a phrase is topicalized, the finite verb must come in second position. This is shown in (192) where the word order is Adv V Subj, both when the finite verb is a main verb as in (192a), and when the finite verb is an auxiliary as in (192b). In English, the subject must precede the main verb when a phrase is topicalized, as shown in (193a) and (193b). However, if the sentence has an auxiliary, the auxiliary will appear before the subject as in Norwegian, as shown in (193c).

- (192) a. I skogen spaserer Jon.
 in forest-DEF walks Jon
 ‘In the forest John walks.’

²There are different approaches to the position of sentence adverbials in Scandinavian languages. Lightfoot (1993) and Holmberg and Platzack (1995) assume that sentence adverbials attach to VP and Vikner (1995) assume that sentence adverbials attach to V'. (But Vikner also allows for sentence adverbials to attach to VP.)

³Vikner (1995), on the other hand, argues that the Scandinavian languages have affix lowering. That is, main verbs do *not* raise to T.

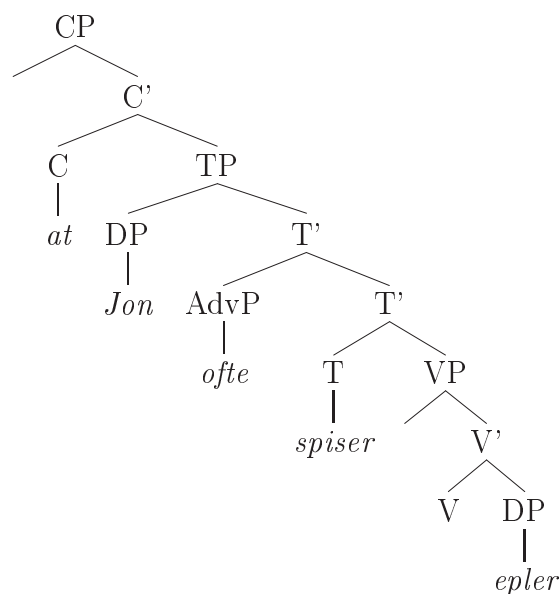


Figure 9.6: Analysis of Norwegian subordinate clause in GB

- b. I skogen har Jon spasert.
 in forest-DEF has Jon walked
 ‘In the forest has John walked.’

- (193) a. In the forest John walks.
 b. * In the forest walks John.
 c. In the forest has John walked.

Given the assumption that sentence adverbials attach to T', the analysis of a main clause presupposes that the preverbal phrase, be it the subject or a topicalized element, has moved to the specifier position of C. This is an established assumption for V2 languages like Dutch, German and the Scandinavian languages in the GB literature. (See Lightfoot (1993), Holmberg and Platzack (1995) and Vikner (1995)⁴). Åfarli and Eide (2003, 87-100) analyse a transitive sentence as shown in Figure 9.7.

In what follows, I will extend the analysis where sentence adverbials attach to T' to English. This analysis will differ from the Carnie (2007) analysis in that main verbs

⁴These authors have in common the assumption that the constituent occurring before the finite verb in main clauses has moved to this position (specifier position of C). They do however not agree on whether the main verb may move from V to I. While Lightfoot and Holmberg & Platzack assume that the main verb moves from V to I (to C), Vikner assumes that the main verb stays in V.

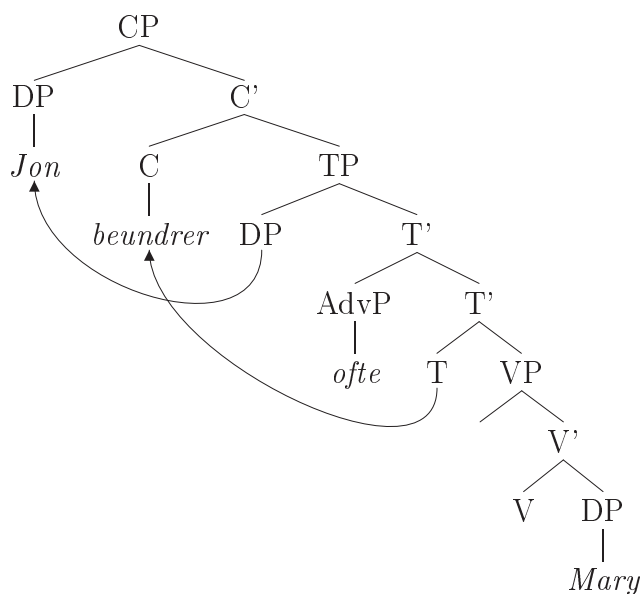


Figure 9.7: GB analysis of *Jon beundrer ofte Mary* ('Jon often admires Mary')

may move to T, but not to C.⁵ An analysis of an English main clause where the sentence adverbial is attached to T' rather than V', and where the verb has moved to T is given in Figure 9.8.

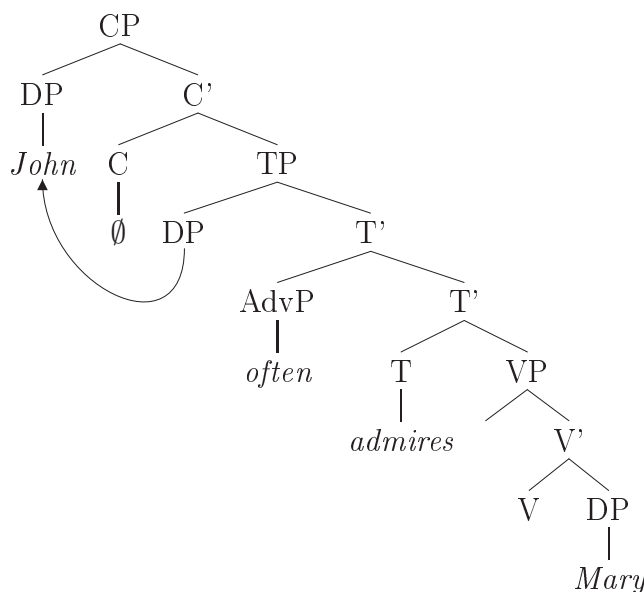


Figure 9.8: Alternative GB analysis of *John often admires Mary*.

⁵An analysis of English where verbs are assumed to move to T (or I), but not to C is given in Holmberg and Platzack (1995, 44–69).

The difference between the English analysis in Figure 9.8 and the Norwegian analysis in Figure 9.7 is that the verb in the Norwegian analysis is allowed to move from T to C. The C position in the English analysis is held by an empty complementizer.

In an English sentence with an auxiliary, the auxiliary will move to C, and the analysis in Figure 9.9 follows.

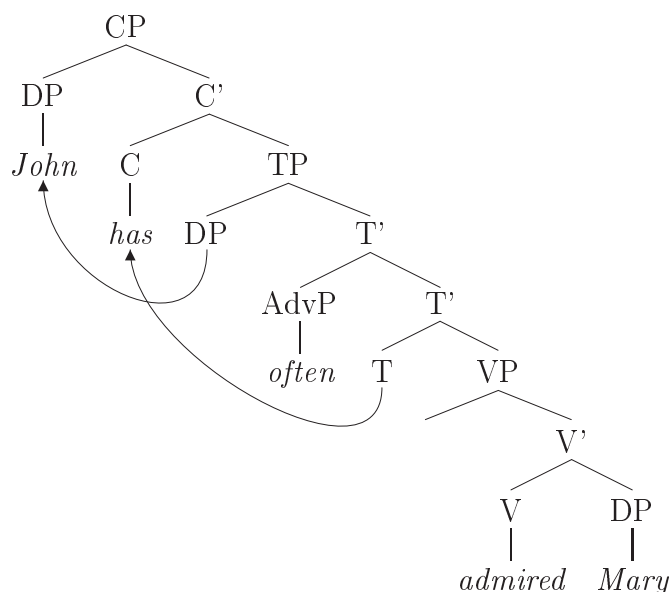


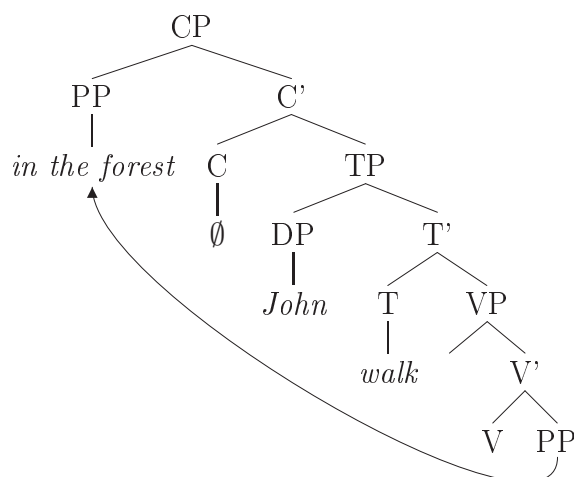
Figure 9.9: Alternative GB analysis of *John has often admired Mary*.

The new GB analysis of a sentence with a topicalized PP is given in Figure 9.10. The difference from the standard analysis (see Figure 9.5) is that the main verb here appears in T, rather than in V.

9.3 Three positions for verbs

In the following I will use the GB analysis shown in the previous section as a means to show how the analysis of basic syntactic structures in Chapter 6 can be compared to GB. The GB analyses are based on Åfarli and Eide (2003), where sentence adverbials are assumed to attach to the T projection.⁶ The movement of the external argument

⁶The reason for assuming that sentence adverbials attach to the T projection rather than just T' in Norwegian is the fact that sentence adverbials may occur both after and before the subject in main clauses with topicalized elements. This is shown in (cxciv). In (cxciva) *ofte* comes after the subject, and in (cxcivb) *ofte* comes before the subject. Given that the subject is realized in the specifier position of T, the sentence adverbial must be allowed to attach to TP when it comes before the subject.

Figure 9.10: Alternative GB analysis of *In the forest John walks*

from the specifier position of V to the specifier position of T will not be taken into consideration, since the analysis implies that the external argument always moves to the specifier position of T. As in GB, I assume that there are three positions in a sentence where verbs can be realized. But unlike GB, there will be no verb movement (or head movement). I will compare GB analyses with Norsyg analyses, and I will show that preterminals are enumerated in the same order in the two approaches.⁷

9.3.1 The position corresponding to C

First, the verb can be the head of a main clause. In GB, the verb will then be in C. In English, only auxiliaries can move to C, while in Norwegian, both main verbs and auxiliaries can occur in C. An analysis where an auxiliary moves to C is shown in Figure 9.9. An analysis of a main verb moving to C is shown in Figure 9.7. The position corresponding to C in my analysis is a position before the subject is realized (from a bottom-up, left-to-right perspective). This may be as the second daughter of

(cxciv) a. På fredager kommer Jon ofte for sent.
On Fridays comes Jon often too late

‘On Fridays Jon often comes too late.’

b. På fredager kommer ofte Jon for sent.
On Fridays comes often Jon too late

‘On Fridays Jon often comes too late.’

⁷By *preterminals* I mean the categories DP, PP, V, I, C and AdvP used in the following analyses, which in the displayed trees are preterminal nodes.

the (binary) head filler rule as illustrated in Figure 9.11, where *beundrer* is the second daughter of the head filler rule, and is realized before the rule that extracts the subject.⁸

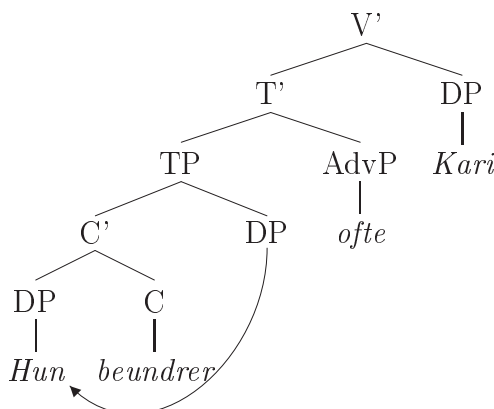


Figure 9.11: Transitive main clause (BRR: D.30, p. 344)

The list in (195a) shows the preterminal nodes of the GB tree in Figure 9.7, page 248, enumerated with a left-corner parsing strategy. The list in (195b) shows the preterminal nodes of the corresponding Norsyg tree in Figure 9.11, enumerated in a bottom-up, left-to-right strategy (as defined in Resnik (1992, 192)).⁹ As the two lists show, the preterminals that the two trees have in common are enumerated in the same order, including the DP trace. The GB tree has a V node and a T node, which are not present in the Norsyg tree. This is due to the fact that the GB analysis has head movement (from V via T to C). Norsyg does not have head movement.

(195) a. [DP_i, C, DP_i, AdvP, T, V, DP]

b. [DP_i, C, DP_i, AdvP, DP]

The position corresponding to C may also be as the head of the valence rule that realizes the subject. This is illustrated in Figure 9.12, where *spiser* is the head of the valence rule that realizes the subject.¹⁰

⁸The tree structure is given GB-like node labels in order to ease the comparison. The force rule on the top of the tree is not displayed. Movement is illustrated by means of a binary rule with a gap (rather than a unary extraction rule) and an arrow between the gap and the filler. The mother of every rule is the second daughter's mother in the GB tree. (The second daughter of V' is DP, and V' is the mother of this DP in the corresponding GB analysis.) An actual analysis of a transitive sentence with Norsyg is given in Figure 6.12 on page 158.

⁹Since I am only enumerating preterminals, I can just as well say "from left to right" since both methods enumerate from left to right, and preterminals cannot dominate each other.

¹⁰Also here, the force rule is not displayed, and the node labels are adapted to the GB analysis. An actual Norsyg tree of a yes-no clause is shown in Figure 6.11, page 158.

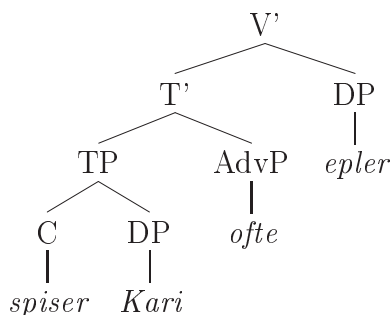


Figure 9.12: Norsyg yes-no clause (BRR: D.31, p. 344)

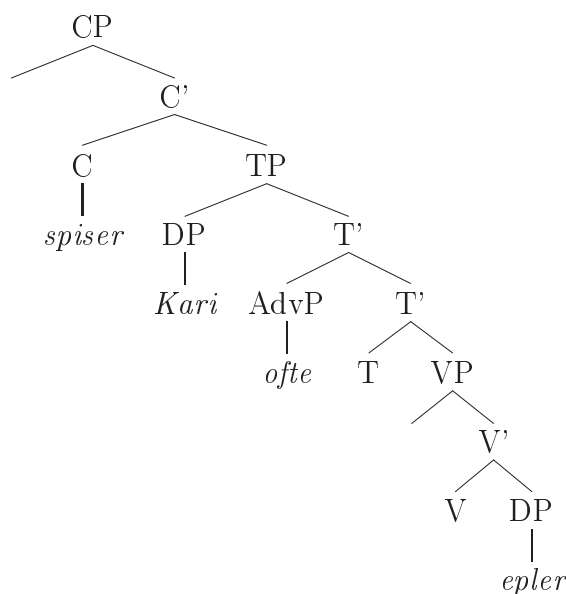


Figure 9.13: GB yes-no clause

The tree in Figure 9.13 shows the GB analysis corresponding to the Norsyg analysis in Figure 9.12. The list in (196a) shows the preterminal nodes of the GB tree in Figure 9.13 enumerated with a left-corner parsing strategy. The list in (196b) shows the preterminal nodes of the Norsyg tree in Figure 9.12, enumerated in a bottom-up, left-to-right strategy. The lists show that the preterminals that the two trees have in common are enumerated in the same order. This is not surprising, since there are no movements or empty categories in the Norsyg analysis. The GB tree has a V node and a T node (due to head movement), which are not present in the Norsyg tree.

(196) a. [C, DP, AdvP, T, V, DP]

b. [C, DP, AdvP, DP]

9.3.2 The position corresponding to T

Second, there is a position for the finite verb in a clause where a complementizer heads the clause. In GB, the verb will then be realized in T. An analysis of a verb occurring in T is shown in Figure 9.6. Here, the complementizer *at* occupies the C position.¹¹ Figure 9.8 shows an analysis of an English main clause where an empty complementizer occurs in C. The main verb, which is blocked from moving to C, appears in T. The

¹¹Here, the English and the Norwegian analyses are identical.

position corresponding to T in my analysis is the position as the second daughter of a merge rule where the TENSE value is *finite* and where the subject is realized. This is illustrated in Figure 9.14, where *beundrer* is the second daughter of the merge rule and is realized after the subject. A complementizer has as value of MERGE an element with the TENSE value *finite* (see Figure 6.31, page 173), and the verb that it merges with *beundrer* ('admires') is in the position that corresponds to T.

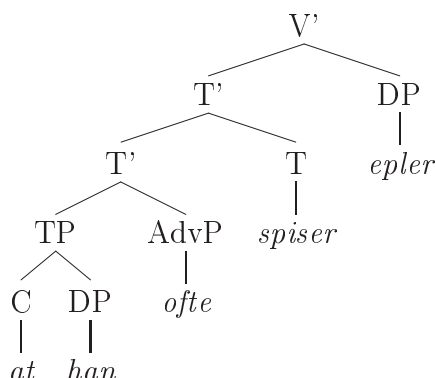


Figure 9.14: Alternative representation of subordinate clause with sentence adverbial (BRR: D.32, p. 345)

The list in (197a) shows the preterminal nodes of the GB tree in Figure 9.6, page 247, enumerated with a left-corner parsing strategy. The list in (197b) shows the preterminal nodes of the corresponding Norsyg tree in Figure 9.14, enumerated in a bottom-up, left-to-right strategy.¹² As the two lists show, the preterminals that the two trees have in common are enumerated in the same order. Due to head movement, the GB tree has a V node which is not present in the Norsyg tree.

(197) a. [C, DP, AdvP, T, V, DP]

b. [C, DP, AdvP, T, DP]

9.3.3 The position corresponding to V

Third, there is a position for non-finite main verbs. In GB, a non-finite main verb is realized in V. This is shown in Figure 9.9, where the non-finite main verb *admired* appears in V. The corresponding position in my analysis is as the second daughter of a merge rule where the TENSE value is *non-finite*. The subject is realized before a verb

¹²The tree is given GB-like node labels.

is realized in this position. This is exemplified in 9.15 where the main verb *beundret* ('admired') is unified with the merge requirement of the auxiliary *har* ('has'). This happens after the subject is extracted.¹³

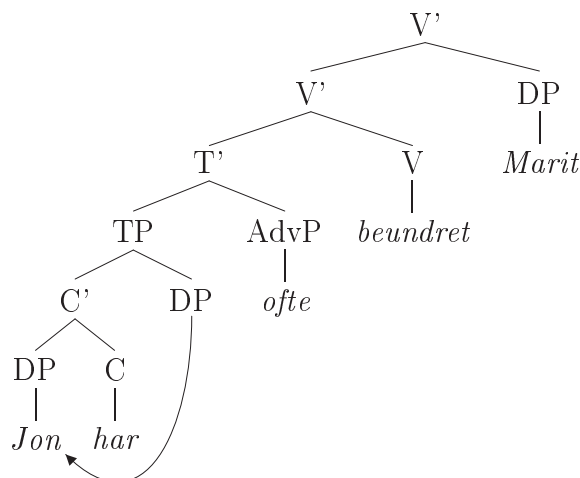


Figure 9.15: Alternative representation of main clause with auxiliary (BRR: D.33, p. 345)

The Norsyng analysis in Figure 9.15 corresponds to the GB analysis in Figure 9.9, page 249. The list in (198a) shows the preterminal nodes of the GB tree, enumerated with a left-corner parsing strategy. The list in (198b) shows the preterminal nodes of the Norsyng tree (Figure 9.15), enumerated in a bottom-up, left-to-right strategy. The two lists show that the preterminals that the two trees have in common are enumerated in the same order. The GB tree has a T node, which is not present in the Norsyng tree.

(198) a. [DP_i , C, DP_i , AdvP, T, V, DP]

b. [DP_i , C, DP_i , AdvP, V, DP]

9.4 An account of basic clause structure in English

The differences mentioned in Section 9.1 and Section 9.2 between English and Norwegian can be accounted for by making two changes to the Norwegian grammar: Blocking main verbs from being realized before the subject, and assuming an empty

¹³Also non-finite auxiliaries will occur in this position. They are however distinct from main verbs in that they require to merge with another verb, while main verbs do not merge with another verb.

complementizer.¹⁴

9.4.1 Blocking main verbs from appearing before the subject

The first change is to block main verbs from being realized before the subject. This is achieved by constraining the CASE value of the ARGUMENT of main verbs to be *non-subj-case* (see Figure 9.16). (This is similar to the blocking of main verbs from moving to C in GB.) This means that the subject must be realized before the main verb is attached, and accounts for the fact that subject verb inversion does not apply for main verbs in English.

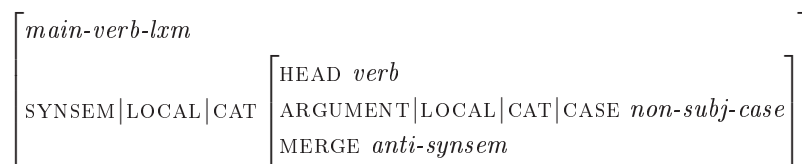


Figure 9.16: The type *main-verb-lxm* in the English grammar

Auxiliaries are not blocked from being realized before the subject, and become necessary in yes-no-questions. The new analysis of (189a) is given in Figure 9.17, where the auxiliary *has* combines with the subject before the main verb is attached. It corresponds to the GB analysis in Figure 9.4. The tree in Figure 9.18 is a modified version of the tree in Figure 9.17, where the node labels are adapted to GB and the force rule is not displayed.

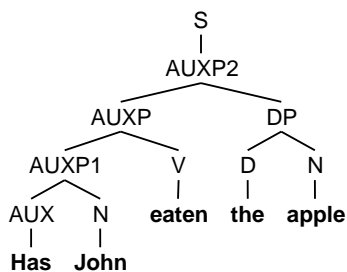


Figure 9.17: New analysis of *Has John eaten the apple?* (1) (BR: D.34, p. 345)

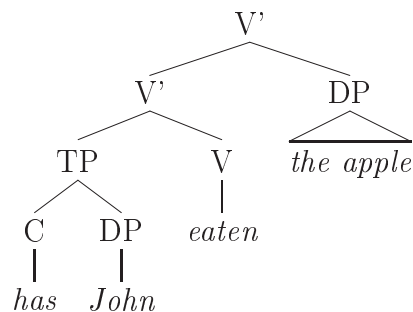


Figure 9.18: New analysis of *Has John eaten the apple?* (2)

¹⁴The changes suggested in this section are implemented in the English demo grammar. (See Appendix B.1.)

The list in (199a) shows the preterminal nodes of the GB tree in Figure 9.4 enumerated with a left-corner parsing strategy. The list in (199b) shows the preterminal nodes of the (adapted) Norsyg tree in Figure 9.18 enumerated in a bottom-up, left-to-right strategy. The preterminals that the two trees have in common are enumerated in the same order. The GB tree has a T node which is not present in the Norsyg tree. It is a result of head movement from T to C.

(199) a. [C, DP, T, V, DP]

b. [C, DP, V, DP]

9.4.2 Assuming an empty complementizer

The second change is to assume an empty complementizer. The empty complementizer is accounted for by means of a unary filler rule in addition to the binary head-filler-rule (see Figure 6.8, page 156). The unary filler rule realizes the slashed element as its daughter. The mother is a complementizer projection with the *local* of the daughter on the SLASH list. The English binary filler rule can only apply in sentences with auxiliaries since main verbs are blocked from applying before the subject is realized. The unary filler rule only applies in sentences with a finite main verb (see the value of MERGE). The rule is given in Figure 9.19.¹⁵

An analysis of a transitive sentence in English is given in Figure 9.20. Like in the analysis for Norwegian, it is assumed that the subject is extracted before it is filled in. The analysis shows how the unary filler rule (AUXP/NP) realizes the slashed element as its daughter (*John*). The subject is extracted by the mother of the unary filler rule (AUXP1). The adverb *often* attaches to the projection that realizes the subject. The analysis corresponds to the alternative GB analysis in Figure 9.8.

The difference from a Norwegian analysis is that it is the unary filler rule, and not the binary filler rule that works. The unary filler rule initiates a complementizer projection that heads the sentence. Since the subject is realized on this projection, the adverbial *often* attaches before the merge rule attaches the main verb *admires*. The use of a unary filler rule in a sentence where the subject comes first is similar to assuming

¹⁵The HEAD value of the mother of the unary filler rule is in the implemented grammar specified as *aux* rather than *complementizer*. This is because the grammar does not recognize a clause headed by a complementizer as a main clause, while it will if it is headed by an auxiliary. Therefore, the projections of the empty complementizer will be labelled as an auxiliary projection rather than a complementizer projection in the LKB trees in Figure 9.20 and 9.22.

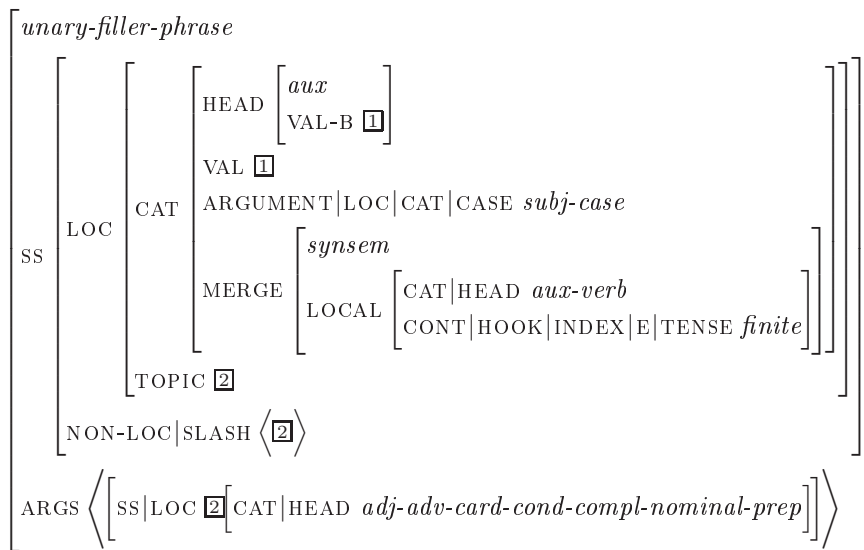


Figure 9.19: Unary filler rule for English

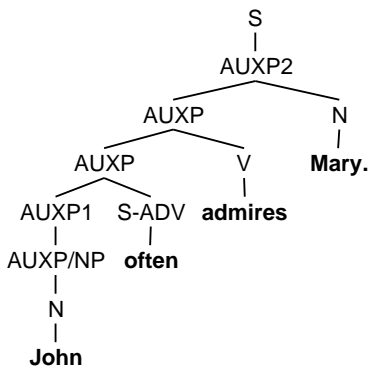


Figure 9.20: New analysis of *John often admires Mary* (1) (BRR: D.35, p. 346)

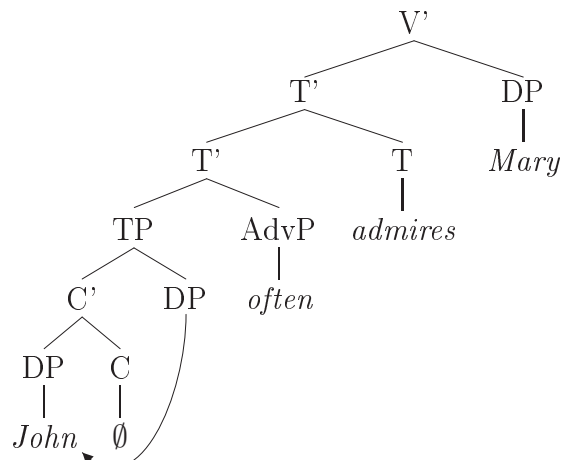


Figure 9.21: New analysis of *John often admires Mary* (2)

that the subject has moved from the specifier position of T to the specifier position of C, and that there is an empty complementizer in C, in a GB analysis.¹⁶

The tree in Figure 9.21 is an alternative representation of the structure shown in Figure 9.20. Here, the empty complementizer is represented as the second daughter of a binary filler rule, and the moved subject is represented by means of an arrow (rather

¹⁶From an engineering point of view, it is potentially risky to introduce a unary filler rule that can apply to every phrase that in a given sentence would be possible to topicalize. It is however possible to restrict the parser in such a way that the rule only applies to phrases that appear first in a sentence.

than a unary extraction rule and slashes). The tree also has GB-like node labels and does not display the force rule.

The list in (200a) shows the preterminal nodes of the GB tree in Figure 9.8, p. 248, enumerated with a left-corner parsing strategy. The list in (200b) shows the preterminal nodes of the (adapted) Norsyg tree in Figure 9.21 enumerated in a bottom-up, left-to-right strategy. The preterminals that the two trees have in common are enumerated in the same order. This includes the DP movement (DP_i) and the empty complementizer. The GB tree has a V node which is not present in the Norsyg tree, as a result from head movement from V to T.

(200) a. [DP_i , C, DP_i , AdvP, T, V, DP]

b. [DP_i , C, DP_i , AdvP, T, DP]

The unary filler rule also accounts for topicalization in English where the main verb is finite. The new analysis of a sentence with a finite main verb and a topicalized PP is given in Figure 9.22. It corresponds to the GB analysis in Figure 9.10, p. 250. The tree in Figure 9.23 is a modified version of the tree in Figure 9.22, where the unary filler rule is represented as a binary rule with an empty complementizer as its second daughter, the long distance dependency is represented by means of an arrow, and the node labels are adapted to GB.

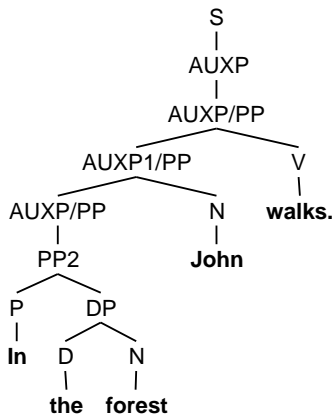


Figure 9.22: New analysis of *In the forest John walks* (1) (BRR: D.36, p. 346)

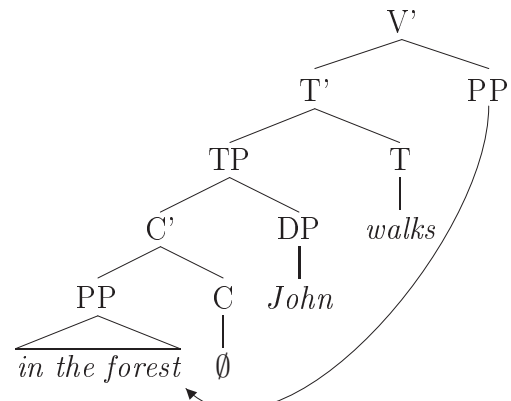


Figure 9.23: New analysis of *In the forest John walks* (2)

The PP *in the forest* is extracted by the adjunct extraction rule (AUXP) (in Figure 9.22) and filled in by the unary extraction rule at the bottom of the tree (AUXP/PP).

As the three shows, the subject attaches to the projection of the complementizer projection that the unary extraction rule initiates. The main verb then is merged with the complementizer projection in the position corresponding to T in GB.

The list in (201a) shows the preterminal nodes of the GB tree in Figure 9.10, p. 250, enumerated with a left-corner parsing strategy. The list in (201b) shows the preterminal nodes of the (adapted) Norsyg tree in Figure 9.23 enumerated in a bottom-up, left-to-right strategy. The preterminals that the two trees have in common are enumerated in the same order. This includes the topicalization of the PP (PP_i) and the empty complementizer (C). The GB tree has a V node which is not present in the Norsyg tree, due to head movement from V to T.

(201) a. [PP_i , C, DP, T, V, PP_i]

b. [PP_i , C, DP, T, PP_i]

Topicalization with an auxiliary (and Wh-movement) is made possible with the binary filler rule. This is illustrated in Figure 9.24.

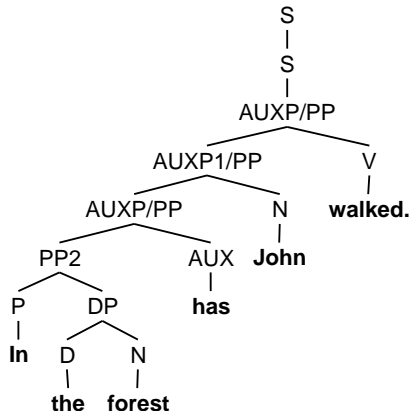


Figure 9.24: Analysis of *In the forest has John walked* (BRR: D.37, p. 346)

9.5 Difference between Norsyg and GB

In this section I will point out a couple of differences between a Norsyg analysis and a GB analysis.

9.5.1 Difference in parsing strategy

The most apparent difference between the two grammar formalisms is the syntactic structures and the parsing strategies associated. (I presuppose that a GB analysis is conducted with a left-corner parsing strategy.)

The difference in parsing strategy associated with the two grammar formalisms has certain implications. As mentioned in Section 5.2.4, when right-branching trees are parsed in a left-corner parsing strategy, constituents are created which are still to realize something. That is, a constituent may consist of everything but the right-corner daughter. In Norsyg, the only constituents that are created are the constituents shown in the tree structures. This is illustrated by the analyses of subordinate clauses. Norsyg does not construct constituents of subordinate clauses when they are not sentence-initial. This was shown in Figure 6.32, repeated here as Figure 9.25. The complementizer *at* attaches to the constituent to its left *Jon hevder* to form the constituent *Jon hevder at* (given a bottom-up parsing strategy). There is no constituent *at han smiler*, as in the GB tree (see (9.26)). However, given a left-corner parsing strategy, no C' constituent *at han smiler* is constructed in the GB analysis either. Rather, the constituent *Jon hevder at* is constructed, just as in the Norsyg tree. (If a top-down or bottom-up parsing strategy had been employed on the GB tree, the constituent *at han smiler* had been constructed.)

Also, Norsyg does not have head movement. This difference is illustrated by the lists of preterminals in (195), (196), (197), (198), (199), (200), and (201), where the lists of preterminals in the GB trees all have the categories C, T, and V, while the lists of preterminals in the Norsyg trees only have one category per complementizer (possibly empty), auxiliary and/or main verb.

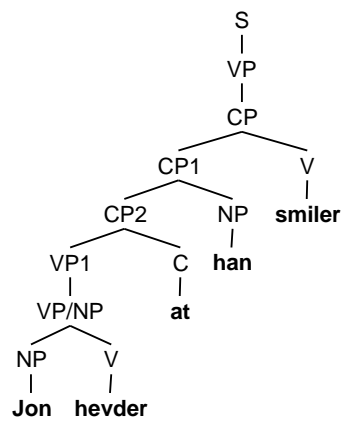


Figure 9.25: Sentence with subordinate clause (BRR: D.17, p. 337)

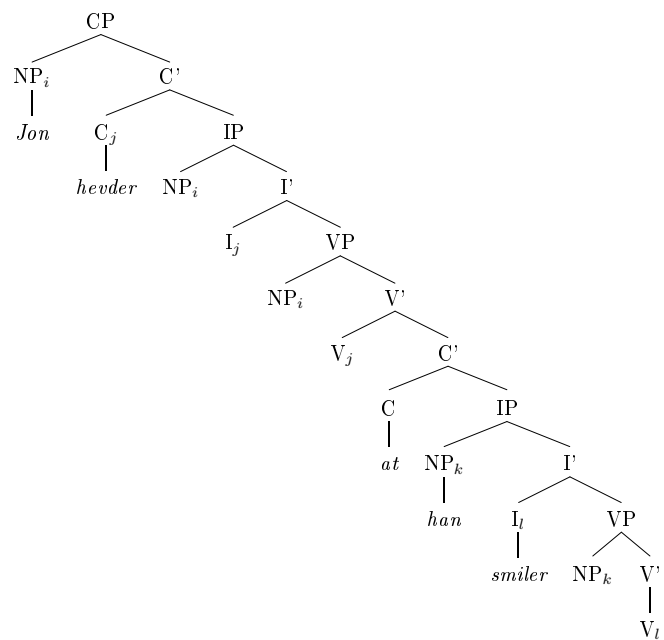


Figure 9.26: GB analysis with subordinate clause

9.5.2 Infinitival clauses and ‘skewed’ syntactic-semantic relations

The tree in Figure 9.27 shows a GB analysis of a sentence with an infinitival clause argument where the infinitival marker *å* appears in C of the infinitival clause, and a PRO (an unexpressed pronominal element) appears in the specifier position of T (in the infinitival clause). The PRO is coindexed with the subject of the matrix verb, *Jon*, but it has not moved to the matrix clause.

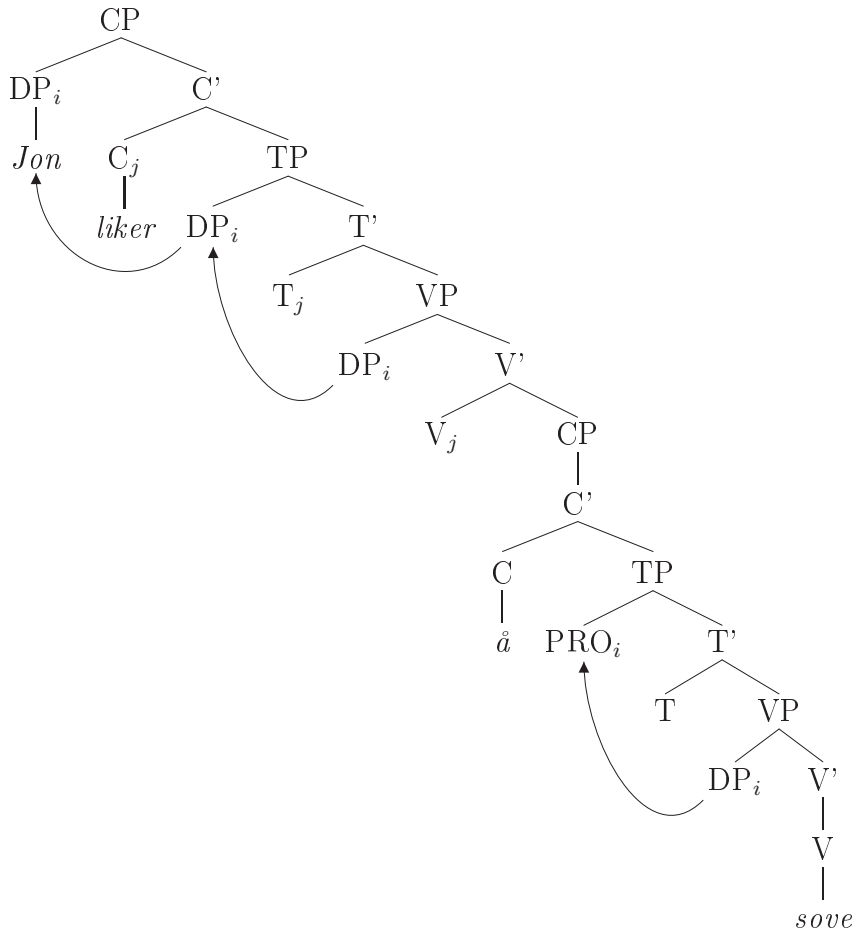


Figure 9.27: GB Analysis of *Jon liker å sove* (‘John likes to sleep’)

This accounts for the fact that *Jon* is an argument both of the matrix verb *liker* (‘likes’) and of the embedded verb *sove* (‘sleep’). Both the verbs assign theta roles to an argument (*Jon* in the case of *liker*, and the coindexed PRO in the case of *sove*).

A GB analysis of a raising construction is given in Figure 9.28. The infinitival clause

of a raising construction is assumed not to have a C projection, and so the argument that receives the thematic role of the infinitival clause has to move to the matrix clause to receive Case. The control verb is assumed to assign Case but no thematic role to one of its arguments (in the tree in Figure 9.28 it is the subject), so the argument moves to that position.

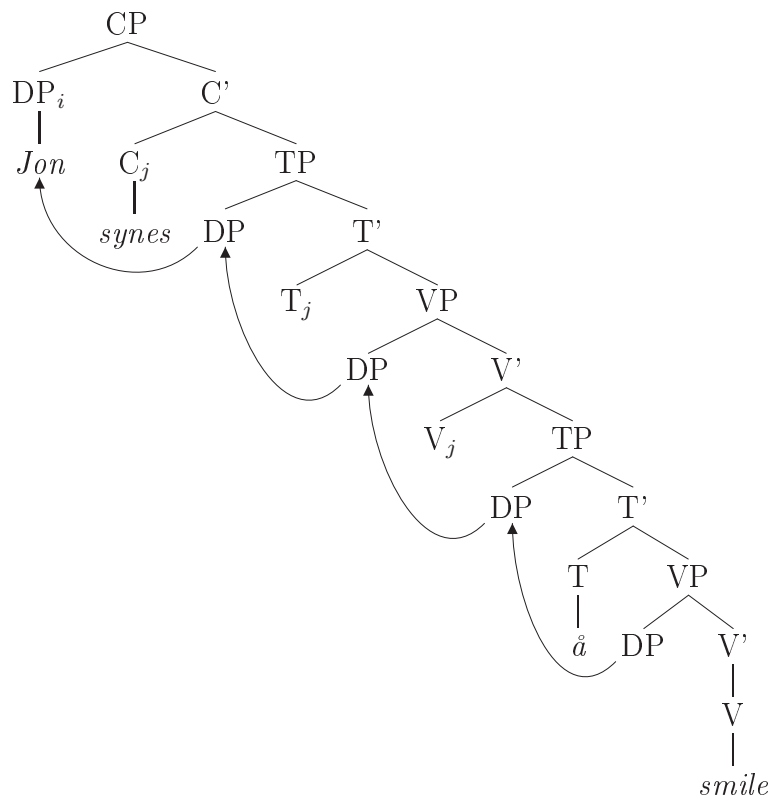


Figure 9.28: GB Analysis of *Jon synes å smile* ('John seems to smile')

As mentioned in Section 6.7.4, my grammar formalism does not represent the kind of skewed relation generally assumed to hold between syntax and semantics in cases of raising, small clauses and resultatives, since what is represented is grammatical relations of a sentence, and not the semantics of a sentence. Therefore, raising constructions are assumed to have the same analysis as sentences with infinitival clauses that are not raising constructions. That is, they correspond to the analysis in Figure 9.27 which has one grammatical relation for the argument in the matrix clause *Jon*, and one for the pronominal element PRO, which is coreferent with *Jon*. A similar line of thinking goes for small clauses and resultatives.

9.6 Summary

In this chapter I have showed how the syntactic structures assumed in this thesis can be compared to syntactic structures in GB. I first presented GB analysis of English as presented in Carnie (2007), where it is assumed that main verbs do not move to T, but rather that tense moves down to V (when the finite verb is a main verb). I showed how main clauses with and without auxiliaries, subordinate clauses, yes-no questions and topicalization are analysed in this tradition.

I then showed how basic syntactic structures in Norwegian are accounted for in Åfarli and Eide (2003), where sentence adverbials are assumed to attach to the T projection, and where main verbs are assumed to move to T. I used this analysis in order to make a link to the syntactic structures proposed in Chapter 6. Three positions for verbs were identified, corresponding to the positions C, I, and V in GB.

Finally, I showed how basic syntactic structures for English can be accounted for by changing a constraint on the type for main verbs, and by adding a unary filler rule, which represents an empty complementizer. I also demonstrated the similarity of the syntactic structures assumed in this thesis with syntactic structures assumed in GB by showing that preterminals in the trees, including empty complementizers and traces, are enumerated in the same order.

This chapter illustrates that even though the framework presented in this thesis and (a selected version of) the GB framework appear to be very different, the two frameworks have certain common assumptions, such as movement to the specifier position of C (in my framework: using the extraction/filler mechanism to “move” an element) and syntactic structures without center-embedding (in GB: right-branching structures; in my framework: left-branching structures). These assumptions make it possible to account for syntactic differences between Norwegian and English by means of two assumptions usually attributed to the GB framework: i) that main verbs are blocked from moving to C in English (in my framework: blocking main verbs from appearing before the subject) and ii) the assumption of an empty complementizer in English. The link to the GB framework will also be used to illustrate the approach to the position of sentence adverbials in the next chapter.

Chapter 10

Sentence adverbials

In this chapter I will show how the position of sentence adverbials in Norwegian are accounted for in Norsyng. Sentence adverbials in Norwegian clauses can occur in different positions with regard to the finite verb and the arguments. In main clauses they come after the finite verb, and occur before, in between, or after the arguments. In subordinate clauses the sentence adverbials precede the finite verb, and occur after the subject.¹ The position of sentence adverbials in Scandinavian languages has been a topic in Scandinavian linguistics for a long time (see Diderichsen (1946); Hellan (1971); Fretheim and Halvorsen (1975); Holmberg (1986); Holmberg and Platzack (1995); Hellan and Platzack (1995); Vikner (1994, 1995)). The data I am presenting in Section 10.1 is a summary of the data from the literature.

I first illustrate the behavior of sentence adverbials in Norwegian with some data. Then I briefly sketch a GB account, which involves verb movement and ‘Object Shift’, before I give an account which does not involve movement, but rather the conception of two ‘fields’. One field where the subject is realized, before the first merge rule (if there are any merge rules), and one field where the other arguments are realized. If the merge rule does not apply, the two fields are the same, and a situation arises where the sentence adverbial may occur before, in between, or after the arguments.

¹As mentioned in Section 6.8, adverbs that precede NP subjects in subordinate clauses, are not assumed to be sentence adverbials, but rather modifiers of the NP. (See (165), p. 201 and its analysis in Figure 6.69, p. 201.)

10.1 Data

According to Faarlund *et al.* (1997), sentence adverbials in Norwegian can be realized 1) as single words (adverbs or adjectives), or 2) as phrases (mostly adjectival phrases or prepositional phrases):²

1. Single words that can function as sentence adverbials:

- (a) Adverbs: *bare* ('only'), *ikke* ('not'), *kanskje* ('maybe'), *aldri* ('never'), *dessverre* ('unfortunately'), *forresten* ('by the way'), *muligens* ('possibly'), *neppe* ('hardly'), *nesten* ('almost'), *også* ('also'), *visstnok* ('apparently'). This group also includes a number of adverbs ending with *-lig*: *antagelig* ('probably') and adverbs ending with *-vis* like *heldigvis* ('luckily'), *muligvis*, ('possibly')
- (b) Some adjectives with neuter gender: *absolutt* ('absolutely'), *sikkert* ('probably'), *åpenbart* ('obviously'), *egentlig* ('really'), *faktisk* ('actually'), *selvfølgelig* ('of course'), *umulig* ('not possibly')

2. Phrases that can function as sentence adverbials:

- (a) Some adjectives in the combination with *nok* ('enough'): *pussig nok* ('peculiarly'), *merkelig nok* ('peculiarly'), *fornuftig nok* ('sensibly')
- (b) Perfect participles of verbs like *si* ('say') and *tale* ('speak') in combination with characterizing adjectives: *kort sagt* ('in brief'), *ærlig talt* ('honestly'), *mellom oss sagt* ('between us')
- (c) Some fixed preposition phrases: *i grunnen* ('really'), *til en viss grad* ('to some degree'), *av den grunn* ('therefore'), *for eksempel* ('for example'), *i realiteten* ('in reality')
- (d) The preposition *for* ('for') in combination with an infinitival construction: *for å si det som det er* ('in truth')
- (e) Subordinate clauses: *hvis jeg ikke tar mye feil* ('if I am not mistaken')
- (f) Prepositions and adjectives conjoined by *og* ('and'): *til og med* ('even'), *først og fremst* ('first and foremost')

²The following list has a selection of the examples given in Faarlund *et al.* (1997), translated from Nynorsk into Bokmål.

- (g) The preposition *som* in combination with a perfect participle or an adjective:
som kjent ('as we know'), *som nevnt* ('as mentioned'), *som sagt* ('as said'),
som vanlig ('as usual')
- (h) Infinitival constructions like *sant å si* ('truthfully'), *vel å merke* ('however')

In this chapter, I will only consider sentence adverbials that are realized as a single word, like *aldri* ('never') and *ikke* ('not').

10.1.1 Sentence adverbials in different clause types

In a Norwegian main clause the sentence adverbial has to come after the finite verb. In (202) the finite verb is the main verb. In (202a) the sentence adverbial *aldri* comes after the finite verb *sover*, and the sentence is grammatical, while in (202b) the sentence adverbial precedes the main verb, and the sentence is ungrammatical. In (203) the finite verb is an auxiliary. If the sentence adverbial occurs in the position after the finite auxiliary and before the main verb, as in (203a), the sentence is grammatical. The sentence adverbial can not occur in the position after the non-finite main verb, as in (203b).

(202) a. Kari sover aldri.
 Kari sleeps never
 'Kari never sleeps.'

b. * Kari aldri sover.
 Kari never sleeps

(203) a. Kari har aldri sovet.
 Kari has never slept
 'Kari has never slept.'

b. * Kari har sovet aldri.
 Kari has slept never

In subordinate clauses the sentence adverbial has to come before the finite verb. In (205) this is illustrated with regard to finite main verbs. If the sentence adverbial comes before the finite verb, as in (205a), the sentence is grammatical, and if the sentence adverbial comes after the finite verb, the sentence is ungrammatical, as in (205b).³

³It is possible to have main clause structure in subordinate clauses if the clause is presupposed. The matrix verb then typically is a verb of "uttering", and the matrix clause cannot be negated (see

- (205) a. at Kari aldri sover.
 that Kari never sleeps
 ‘that Kari never sleeps.’
 b. * at Kari sover aldri.
 that Kari sleeps never

(206) shows that sentence adverbials in subordinate clauses must precede the finite auxiliary if the clause has an auxiliary.⁴ In (206a) the sentence adverbial precedes the finite auxiliary, and the clause is grammatical, and in (206b) the sentence adverbial comes after the finite auxiliary and the sentence is ungrammatical.

- (206) a. at Kari aldri har sovet.
 that Kari never has slept
 ‘that Kari never has slept.’
 b. * at Kari har aldri sovet.
 that Kari has never slept

In yes-no clauses, the finite verb comes first and the sentence adverbial has to follow it, as illustrated in (207).

10.1.2 Sentence adverbials and the arguments

Sentence adverbials can have different positions with regard to the subject, direct object and indirect object. In this section I will suggest that the status of a nominal’s reference

Faarlund *et al.* (1997, 983-984)). This is shown in (cciv) (from Fløgstad (1977), cited in Faarlund *et al.* (1997, 983), in Nynorsk), where a predicate adverbial *nå* (‘now’) is topicalized in the subordinate clause.

- (cciv) Ingen liten berrføtt gutunge kjem springande inn på omnshuset og gir Selmer og
 no small barefoot boy comes running in on oven-house-DEF and gives Selmer and
 dei andre i tappen beskjed om at nå har Nygaardsvold danna regjering.
 they others in tap-DEF message about that now has Nygaardsvold formed government
 ‘No small barefoot boy comes running into the oven house and tells Selmer and the others in the
 tap that Nygaardsvold now has formed government.’

The possibility of having main clause structure in subordinate clauses is argued in Platzack (1986), Holmberg and Platzack (1995) and Vikner (1995, 65–130), and they point out that main clause structure in subordinate clauses which are assertions, is possible in several languages (Danish, Faroese, Norwegian, Swedish, English and Frisian), and that it is less restricted in Icelandic and Yiddish. An embedded clause with main clause structure is given this structure: [CP [C CP [Spec C’ [C TP]]]] in Holmberg and Platzack (1995, 83). Subordinate clauses with main clause structure are not dealt with in the thesis.

⁴See Footnote 3.

- (207) Sover aldri Kari?
 sleeps never Kari
 ‘Does Kari never sleep?’

determines how a nominal is positioned with regard to a sentence adverbial. I will distinguish between nominals whose reference is *in focus* and nominals whose reference is *not in focus*.⁵

Nominals whose reference is in focus

One group of nominals are so-called light (or weak) pronouns.⁶ These are unstressed pronouns whose reference are believed by the speaker to be easily accessible to the hearer. They have the cognitive status *in focus*. Light pronouns typically come immediately to the right of a verb, another NP, or a preposition, as illustrated in (208a). They cannot be in the position after the sentence adverbial of the clause, as shown in (208b). If a pronoun occurs in the position after the sentence adverbial, the intonation of the pronoun has to be marked, as in (208c), in which case it is no longer light (or weak).

- (208) a. Marit ser den aldri.
 Marit sees it-LIGHT never
 ‘Marit doesn’t see it.’
- b. * Marit ser aldri den.
 Marit sees never it-LIGHT
- c. Marit ser aldri DEN.
 Marit sees never it-HEAVY
 ‘Marit doesn’t see that.’

Nominals whose reference is not in focus

Other nominals, that are not light pronouns, will in most cases follow the sentence adverbial, as illustrated in (210a) and (210b). There are however certain exceptions.

⁵The semantic notions I use to refer to the status of the reference of a nominal are taken from Borthen and Haugereid (2005), which builds on Gundel *et al.* (1993).

⁶An overview of pronouns in Scandinavian languages is given in Hellan and Platzack (1995). The description below is the one standardly given for the ‘light’ pronouns.

(211a) shows the unmarked order of a sentence adverbial and a proper noun (the sentence adverbial precedes the proper noun). But if the intonation of the verb is marked, as in (211b), an argument which is not a light pronoun, *Jon*, may precede the sentence adverbial. It is possible that the marked intonation of (211b) implies that the reference of *Jon* has the cognitive status *in focus*, and that this is what makes it acceptable in this position.⁷

- (210) a. Marit ser aldri dyreprogram.
 Marit watches never animal-programs
 ‘Marit never watches animal programs.’
 b. * Marit ser dyreprogram aldri.
 Marit watches animal-programs never

- (211) a. Marit sã aldri Jon.
 Marit saw never Jon
 ‘Marit never saw Jon.’
 b. Marit sã Jon aldri.
 Marit saw Jon never
 ‘Marit never **saw** Jon.’

The same applies in yes-no questions, as illustrated in (212). In (212a), the argument (*Kari*) comes after the sentence adverbial, while in (212b) the argument comes before the sentence adverbial. Also here, it is possible that the reference of the argument in the latter case is *in focus*, and that this is what allows it to appear before the sentence adverbial.

- (212) a. Sover aldri Kari?
 sleeps never Kari
 ‘Does Kari never sleep?’
 b. Sover Kari aldri?
 sleeps Kari never
 ‘Does Kari never sleep?’

⁷Also in (ccix) the sentence adverb *aldri* is preceded by an argument. However, in this case I assume that it attaches to the adverb *igjen* (‘again’), and does not function as a sentence adverbial.

- (ccix) Marit sã Jon aldri igjen.
 Marit saw Jon never again
 ‘Marit never saw Jon again.’

A more serious challenge to the generalization, that nominals that are not in focus, cannot precede a sentence adverbial, is posed by subordinate clauses. As shown in (205a) and (206a), the subject precedes the sentence adverbial in subordinate clauses. This also holds for indefinite nouns, as shown in (213). For the generalization to hold, one would be forced to assume that the reference of the subject of a subordinate clause is in focus. Instead, I will modify the generalization in the following way: Nominals that are not in focus and that are not subjects of subordinate clauses, cannot precede a sentence adverbial. I will not attempt to explain why subjects of subordinate clauses do not follow the initial generalization.

- (213) at dyreprogram aldri blir sett av Marit
 that animal-programs never are seen by Marit
 ‘that animal programs are never seen by Marit’

Yes-no questions and topicalization

In Norwegian yes-no questions and in sentences with a topicalized element, the subject is realized after the finite verb and before the objects. The sentence adverbial may occur in the position right after the finite verb, as in (214a), but there may also be arguments intervening between the finite verb and the sentence adverbial, especially if the arguments are light pronouns. In (214b), the subject intervenes between the verb and the sentence adverbial. In (214c), the subject and the indirect object precede the sentence adverbial, and in (214d), the subject, the indirect object and the direct object come before the sentence adverbial.

- (214) a. Gir aldri Jon Marit isen?
 Gives never Jon Marit ice-cream-DEF
 ‘Doesn’t Jon give Marit the ice cream?’
 b. Gir han aldri Marit isen?
 Gives he never Marit ice-cream-DEF
 ‘Doesn’t he give Marit the ice cream?’
 c. Gir han henne aldri isen?
 Gives he her never ice-cream-DEF
 ‘Doesn’t he give her the ice cream?’
 d. Gir han henne den aldri?
 Gives he her it never

‘Doesn’t he give it to her?’

Clitics

The dialect Trøndersk has clitic pronouns (’a and ’n), and a clitic negator (’itj) that can appear as a sentence adverbial. The clitic negator can occur in any of the positions illustrated in (215).

- (215) Ga (’itj) ’n (’itj) ’a (’itj) ’n (’itj)
 Gave (not) he (not) her (not) it (not)
 ‘Didn’t he give it to her?’

10.2 A GB approach

In GB, the position of sentence adverbials in Norwegian are accounted for by means of verb movement (see Åfarli (2003)). While the position of the sentence adverbial is assumed to be relatively constant (attaching to T’ or TP), verbs can be realized in V, T or C (I discussed this in more detail in Chapter 9). The finite verb is originally positioned after the sentence adverbial position and then, if the sentence is a main clause, the verb moves to a position preceding it (C). Figure 10.1 shows the structure of a main clause where the verb *ser* has moved from V via T to C, and where the subject *Kari* has moved from the specifier position of V via the specifier position of T to the specifier position of C.

As shown in Section 10.1, it is possible for DP objects to appear in the position after a finite main verb and before the sentence adverbial. This is referred to as ‘Object Shift’, and is according to Holmberg (1999), an operation that happens after the other movements. It lets objects move to the position to the right of the next main category element to their left. A ‘main category’ here does not include sentence adverbials. This means that an object is allowed to move past a sentence adverbial and find its position to the right of a verb after the verb has moved. This is shown in the tree in Figure 10.2, where the object attaches to the verb after the verb has moved to C.

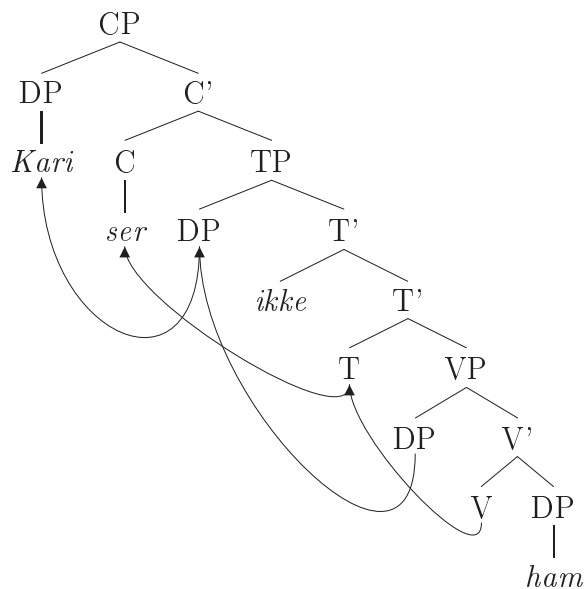


Figure 10.1: Main clause in GB

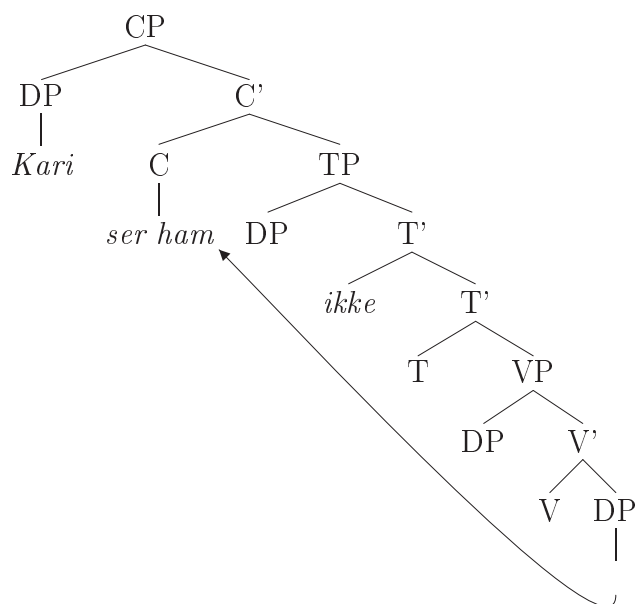


Figure 10.2: Object Shift in GB

10.3 The approach taken in Norsyg

Given the syntactic approach presented in Chapter 6, and the discussion in Section 10.1.2, I can make the following two generalizations about the position of sentence adverbials with regard to the arguments of a clause:

1. Sentence adverbials that are not fronted, occur after the syntactic head of the clause and before non-head verbs that are merged with the head projection.
2. Arguments that have a reference whose cognitive status is *in focus* (mostly light pronouns), and subjects of subordinate clauses cannot occur in the position after a sentence adverbial (on the same projection).

In this section I take up the thread from Section 6.8, where I introduced the rules for adverbs that may function as sentence adverbials. I will here focus on the adverbs that have scope over the event, and which are not fronted. That is, sentence adverbials that are realized by the *head-sadv-rule* (see Figure 6.66, repeated here as Figure 10.3).

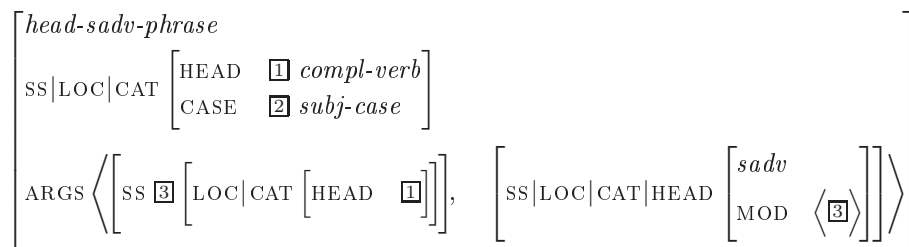


Figure 10.3: The head initial sentence adverb rule

The position of the sentence adverbials is in the account presented in this thesis crucially linked to the realization of the subject. As shown in Figure 10.3, the sentence adverbial attaches to a projection with the CASE value *subj-case*. The CASE value is *subj-case* in a field where the subject is realized, before the (first) merge rule applies, if it applies. (The merge rule is presented in Section 6.5.) And the CASE value is *non-subj-case* in a field where the merge rule has applied. According to the comparison made in Chapter 9, the sentence adverbials attach in a position corresponding to T' or TP in GB. This is illustrated in the analyses that follow.⁸ The left-branching structures imply that no head movement or other kinds of movement like 'Object Shift' is involved.

⁸The same conditions hold for the trees used for comparison to GB (see Figures 10.5, 10.8, 10.10, 10.12, and 10.14 below) as pointed out in Footnote 8, p. 251.

10.3.1 Analysis of sentence adverbials in different clause types

In Figure 10.4 the sentence adverbial attaches to the verb projection. In the GB-adapted version (see Figure 10.5) it attaches to T'.

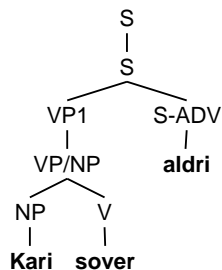


Figure 10.4: Main clause with sentence adverbial (BRR: 10.6)

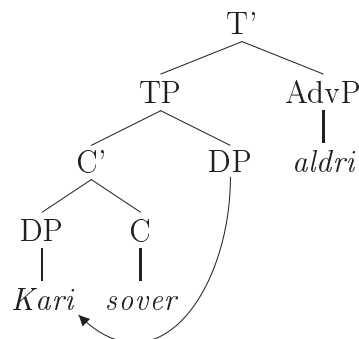


Figure 10.5: Tree in Figure 10.4 adapted to GB.

The BRR (Basic Relation Representation) of the tree in Figure 10.4 is given in Figure 10.6. As far as arg1-4-relations go, there is nothing to represent for sentence adverbials, since they are sentence operators. The BRRs for this chapter say nothing about the semantic scope of the adverbs. The adverb relations are constrained to share LBL values with the verb relations in the clauses they modify.

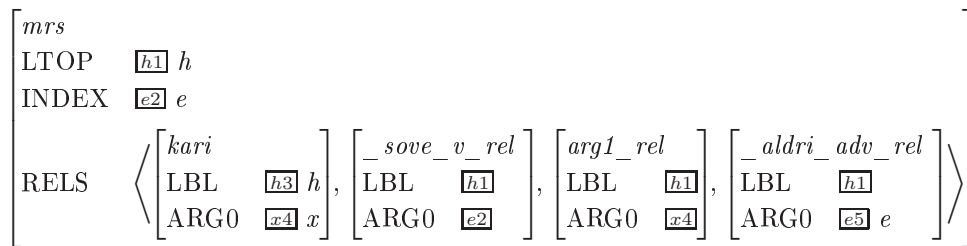


Figure 10.6: BRR of main clause with auxiliary and sentence adverbial (Tree: 10.4)

In Figure 10.7 the sentence adverbial attaches to the auxiliary projection (before the merge rule applies). It cannot attach after the merge rule since the merge constituent is specified as CASE *non-subj-case*. In the GB-adapted version (see Figure 10.8) it attaches to T'.

In Figure 10.9 the sentence adverbial attaches to the complementizer projection (before the merge rule applies). It cannot attach after any of the verbs in the subordinate clause since the MERGE rule, which combines the verbs to the head

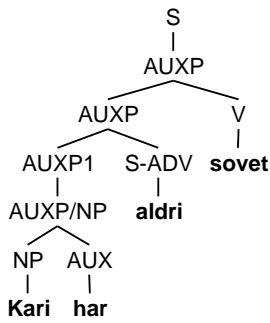


Figure 10.7: Main clause with sentence adverbial (BRR: D.38, p. 347)

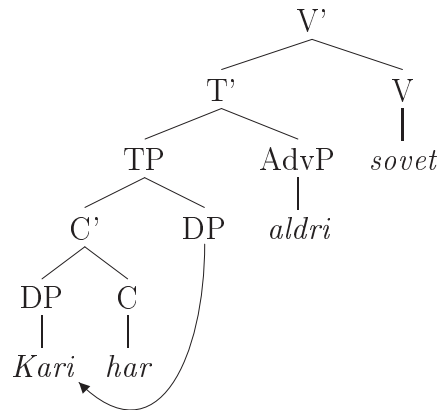


Figure 10.8: Tree in Figure 10.7 adapted to GB.

projection, is constrained to have *CASE* value *non-subj-case*. In the GB-adapted version (see Figure 10.10) the sentence adverbial attaches to T'.

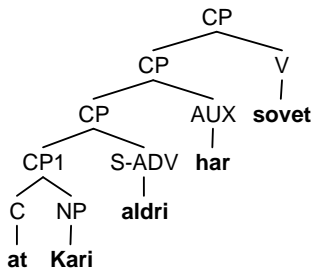


Figure 10.9: Subordinate clause with auxiliary and sentence adverbial (BRR: D.39, p. 347)

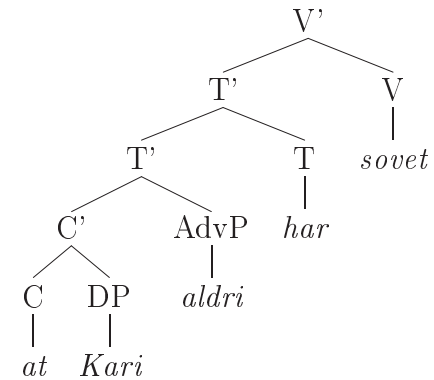


Figure 10.10: Tree in Figure 10.9 adapted to GB.

Figure 10.11 and 10.13 show how yes-no questions are analyzed. The sentence adverbial attaches to the verb projection, where the subject is realized (VP1). The sentence adverbial attaches both before and after the subject. In the GB-adapted versions (see Figure 10.12 and 10.14) the sentence adverbial attaches to TP and T', respectively.

10.3.2 Analysis of sentence adverbials and the arguments

If the merge rule is not applying, that is, if the sentence is a yes-no question or a main clause, and the main verb is finite, the sentence adverbial may come before, in between,

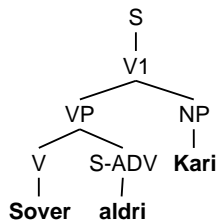


Figure 10.11: Yes-no question with sentence adverbial (BRR: D.40, p. 348)

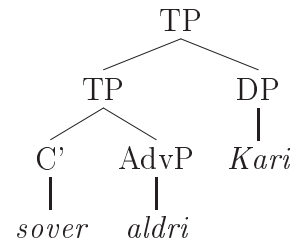


Figure 10.12: Tree in Figure 10.11 adapted to GB.

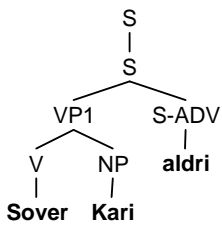


Figure 10.13: Yes-no question with sentence adverbial (BRR: D.41, p. 348)

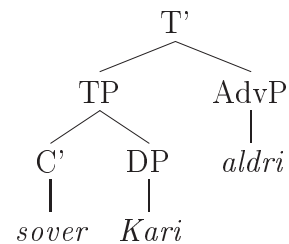


Figure 10.14: Tree in Figure 10.13 adapted to GB.

or after the arguments that follow the verb. This is because the field where the subject is realized (either by a binary valence rule or by an extraction rule), and where the sentence adverbial may attach, is the same as the field where the other arguments are realized. The position of the adverb does contribute some information, namely that all the arguments that precede it have to have a reference whose cognitive status is *in focus* (unless the clause is a subordinate clause). The argument that comes right after it can not have a reference whose cognitive status is *in focus*. (According to the hierarchy of cognitive statuses in Borthen and Haugereid (2005, 11), their cognitive statuses are *activ-or-less* (activated, familiar, uniquely identifiable, or type identifiable).) This information is possible to specify on the indices of the arguments given that indices carry this information (see Borthen (2003, 275)).

The Figures 10.15-10.18 show how the sentence adverbial is positioned between the arguments of a ditransitive yes-no question. In 10.15 it precedes all the arguments, in 10.16 it precedes two of the three arguments, in 10.17 it precedes one argument, and in 10.18, which has three light pronouns, it comes after all the arguments.

Since the analysis is not dependent on any kind of movement, the clitic data in (215)

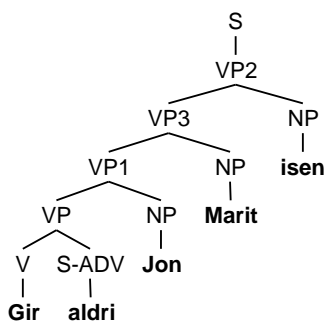


Figure 10.15: No pronouns (BRR: D.42, p. 348)

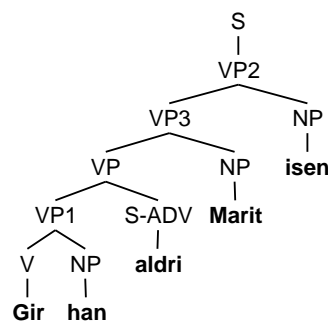


Figure 10.16: 1 pronoun (BRR: D.43, p. 349)

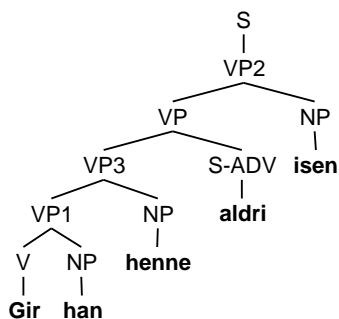


Figure 10.17: 2 pronouns (BRR: D.44, p. 349)

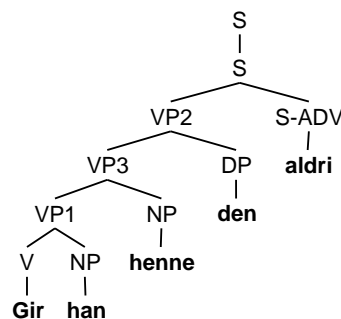


Figure 10.18: 3 pronouns (BRR: D.45, p. 350)

can be accounted for by having a set of binary valence rules turned into inflectional rules that add clitic suffixes. One also needs an inflectional rule for the clitic negator. The ‘tree’ in Figure 10.19 shows how an analysis of *gir'n'a'n'itj* (‘Doesn’t he give him to her?’) looks when parsed with the LKB system. The first unary rule (V) adds present tense (-r). The second unary rule (VP1) adds the masculine/neuter pronoun subject suffix (-n). The third unary rule (VP3) adds the feminine pronoun indirect object suffix (-a). The fourth unary rule (VP2) adds the masculine/neuter pronoun direct object suffix (-n). The fifth unary rule (S) adds the negator suffix (-itj). The last unary rule is the yes-no force rule. The negator rule can also apply before and in between the pronoun rules. The tree in Figure 10.20 is an alternative representation of the LKB “tree” in Figure 10.19.

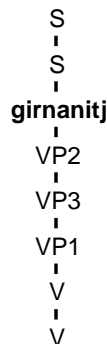


Figure 10.19: Analysis of a verb with four clitics. (BRR: D.46, p. 350)

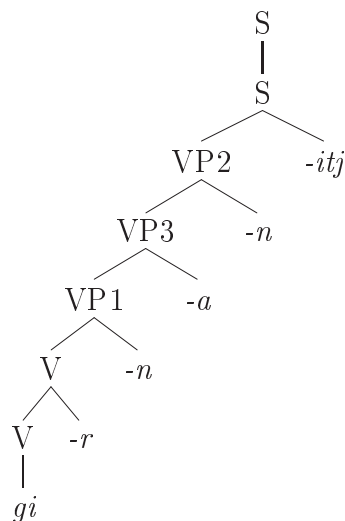


Figure 10.20: Alternative representation of the tree in Figure 10.19.

10.4 Summary

I have shown that by restricting sentence adverbials to attach in a field where the subject is realized, before the first merge rule applies (if it applies), the position of the sentence adverbials in Norwegian is accounted for. The application of a merge rule corresponds to the blocking of a verb from moving to CP in GB. If the merge rule applies, there will not be any ‘Object Shift’, since the field where the sentence adverbials may attach is before the merge rule, and the field where the objects are realized come after the merge rule(s). If there is no merge rule, ‘Object Shift’ may apply, that is, the field where the subject is realized and the field where the objects are realized are the same. The position of the sentence adverbial with regard to the arguments can be captured by saying that the arguments that precede the sentence adverbial have to have a referent whose cognitive status is *in focus* (unless it is a subject in a subordinate clause), and that the argument that comes in the position behind it (on the same projection) can not be a light pronoun (or have a referent whose cognitive status is *in focus*).

The issue of adverb placement does not relate directly to the assignment of arg1-4-relations to verb arguments, but it heavily relates to fine-grained parameters of sentential syntax in Norwegian. I have shown that although unorthodox, the syntactic mechanisms of the present proposal attain the same level of accuracy as any of the more

current approaches.

Chapter 11

Conclusion

In this thesis I have demonstrated that it is possible to implement a grammar where valence alternations are accommodated by means of *phrasal subconstructions*. While other grammar implementations within the HPSG and LFG frameworks rely crucially on fixed argument frame specifications in the lexicon, in order to account for valence alternations, (by means of multiple lexical entries, lexical rules, or disjunctions of lexical templates) I have presented a formalism where the settling of a verb's argument frame is delayed until the syntactic tree is built. This is achieved by letting *functional signs* (inflections, function words, and valence rules) realize phrasal subconstructions, which, when they are put together, constitute constructions or argument frames. In principle, the formalism allows for open lexical items to be listed without any syntactic information; both its category and its argument frame may be underspecified. However, in order to reduce the processing effort of the parsing grammar, I have implemented a *construction-constraining* mechanism (or a *packing* mechanism) where a hierarchy of subconstruction types and construction types makes it possible to specify on a lexical entry what argument frames one can expect it to appear in. This mechanism together with the assumption of phrasal subconstructions gives a grammar implementation which is significantly more efficient than a corresponding implementation where the argument structure is fixed in the lexicon.

I started out by having a look at how HPSG, LFG, Construction Grammar, and Minimalism treat argument structure. I distinguished between three topics in the discussion of argument structure. The first topic was the alternation between different voices (active, passive and middle). This alternation is mostly treated

lexically in frameworks like HPSG and LFG. In the Minimalist frameworks that I have discussed, there is a tendency to treat the active passive alternation syntactically. The second topic was valence alternations. This group of alternations includes alternations in arity, like the intransitive/transitive alternation, and other alternations like the causative/inchoative alternation, the dative alternation, the locative alternation and the resultative construction. These alternations are treated lexically in HPSG and LFG and one of the Minimalist frameworks (Hale and Keyser), and syntactically in the other Minimalist frameworks. The third topic was the alternation between unergative and unaccusative (variable behavior). The verb *drip* is ambiguous between an unaccusative reading and an unergative reading. All frameworks treat this argument structure alternation lexically, except from one Minimalist approach (Borer).¹

I suggested that the different valence alternations and the variable behavior can be accounted for with different constellations of the five argument structure subparts (see Chapter 3). The active/passive alternation is accounted for by assuming that passive is a syntactic object (expressed either as an auxiliary or as a passive morpheme) which realizes the first argument structure subpart (see Section 7.1). Since the argument structure subparts are syntactic objects, I can account for all three kinds of argument structure alternations syntactically.

In my analysis I assume one valence feature for each of the first four argument roles. Each valence feature carries information about whether an argument role is realized or not. When the rule that realizes the subject applies, the valence information from the valence features is unified, and a type hierarchy of “linking” types makes sure that the argument structure produced by the syntax is acceptable (see Figure 4.9 (p. 96)). The mechanism is mainly there to prevent “very odd” sentences like *John smiled his mouth with chocolate* from being parsed.

By assuming that argument structure is assigned to lexical items through their being operated on by syntactic rules, it becomes possible to let one lexical entry enter several argument frames without using lexical rules or multiple lexical entries. The argument frames in TROLL and NorKompLeks are accommodated syntactically.

The decomposition of argument structure into five subparts and the one-to-one relation between syntax and semantics has made several things possible:

- A verb can enter a range of argument frames, as I demonstrated with *drip* in

¹The table that summarizes these findings is given in Figure 2.1 (p. 50).

Figure 3.2 (p. 78) since the argument structure does not have to be fixed in the lexicon.

- Generalizations over syntactic entities that otherwise would be impossible, can be made, as I showed in Figure 3.8 (p. 85).
- Complex predicates like the coordination of Vs and the Empty Object Construction in Norwegian can be accounted for.
- Instances of several argument frames sharing one predicate (ellipsis) can also be given an analysis.

I have shown in detail how a set of six kinds of rules can account for the syntactic structures of Norwegian clauses. These are the valence rules (including binary rules and unary extraction rules), which link arguments to the head projection of the clause, the filler rules, which fill in the extracted argument, the merge rule, which merges the syntactic and semantic information of non-head verbs with the head projection, the subordination rules, which mark the beginning of a subordinate clause, the clause boundary rules (including the force rules for main clauses and the pop rule for embedded clauses), which mark the boundary of clauses, and the modifier rules, which let modifiers attach to the head projection.

The first complementizer or verb of a clause is assumed to be the head, and all other verbs, arguments and/or modifiers are attached to this head by means of the rules mentioned above.

The exo-skeletal nature of the grammar opens for a radically new syntactic analysis, where Diderichsen's "Fundamentet" (the constituent occurring to the left of the finite verb in a main clause) is taken as point of departure, and constituents are attached in a bottom-up, left-to-right fashion. This kind of syntactic structures allows for incremental parsing, and provides a natural account of phenomena such as registering of extraction path in long distance dependencies, binding, and light pronouns.

The flexibility and power of an exo-skeletal approach are also demonstrated by means of a grammar implementation, Norsyg, which tested on a Wikipedia article on concrete (4711 words), gives the intended analysis to 34.2% of the grammatical items.

Abstracting away from the fact that Norsyg is a left-branching grammar and does not have syntactic constituents in the traditional sense, and the fact that GB/Minimalism is a theory that assumes right-branching trees and allows head

movement, the two approaches are quite similar. Both approaches are suited for incremental parsing, Norsyg with a bottom-up parsing strategy, and GB/Minimalism with a left-corner parsing strategy. Pre-terminal constituents of the syntactic trees, including empty complementizers and traces of DP movement are enumerated in the same order. The syntactic structures make it possible to account for phenomena such as long distance dependencies by means of local constraints on trees.

Theories such as HPSG and LFG have mixed left- and right-branching trees. This kind of syntactic structures allow the theories to have constituents in the traditional sense at the same time as they do not allow for head movement. Apparently, it is the best out of two worlds, but it comes with a cost, namely that the phenomena that Norsyg and GB/Minimalism can account for by means of local constraints on trees, such as registering of extraction path, has to be accounted for by other kinds of mechanisms, such as relational constraints on valence lists or argument structure lists as done in HPSG. Also, the mixed left- and right-branching tree structures cannot be parsed incrementally.

Given that one uses a bottom-up parsing strategy, and that one wants to account for phenomena such as registering of extraction path by means of local constraints on trees, the application left-branching tree structures seems to be the most appropriate approach.

Although this thesis has dealt mainly with Norwegian and English, I believe that the main ideas concerning argument structure as a syntactic construct, where valence alternations can be accounted for by means of five subconstructions, and syntactic structures are assumed to be mainly left-branching, should be possible to implement in the grammar of any language. In Appendix B.2, I suggest for example how the formalism can be extended to German.

The work that has been presented in this thesis, describes a procedure for making parsing more efficient. This alone does not make the work unique. The efficiency of the system is a concern to everybody who is implementing a grammar of a certain size. What makes this formalism differ from other formalisms is that it is based on the intuition that unambiguous words should have just one representation. To me, it has always made sense that the argument structure frame of a verb is built incrementally, as the syntactic context is produced. The parse charts in Figures 4.23, page 115, and 4.24, page 116, illustrate my concern for an approach where syntactic flexibility is accounted for in the lexicon. While the first parse chart has only three representations of the

word *presset* ('pressed'/'the pressure'), expressing the ambiguity between a past tensed verb, a past participle, and a definite noun, the second parse chart, which represents a lexicalist approach to valence alternations, has 17 representations of *presset*. This is because the verb *presse* has the potential of entering 8 different argument frames; hence there are 8 versions of the past tensed verb and 8 versions of the past participle. The 8 versions of each of the verb forms are not expressing an ambiguity, only the fact that the verb appears naturally in a range of syntactic contexts. The processing effort of an approach that uses multiple lexical entries to express syntactic flexibility, is significantly higher compared to an approach which only represents *real* lexical ambiguity.

By allowing for syntactic flexibility to be accommodated by the syntax, rather than seeing it as a component of the lexicon, I hope, even though this has not been a main focus of the thesis, to have opened the door to the psychological reality of what happens in sentence processing. In my view, one cannot ignore the psychological reality if one wants to make continued progress in the work on computational grammars.

Appendix A

Norsyg

Norsyg (Norwegian syntax-based grammar) is an implemented grammar for Norwegian. It is a continuation of earlier grammars: NorSource (Jan 2002 - Jan 2004), Saargram (Feb 2004 - Jul 2005) and Phdgram (Aug 2005 - Aug 2006). The initial grammar was based on the Grammar Matrix version 0.6. The implementation platform is the LKB system.

A.1 Download

Download instructions for the Norsyg grammar are given here:

<http://www.hf.ntnu.no/hf/isk/Ansatte/petter.haugereid/norsyg.html>

The version referred to in this thesis (oct-08) is called:

norsyg2.0

Norsyg is distributed with a small handwritten lexicon (1300 entries). It can also run with Norsk Ordbank, which is a computational dictionary for Norwegian. The dictionary can be downloaded from Norsk Ordbank's site at the University of Oslo. Register as a user and download the 'Bokmålsdata' file 'ordbank_bm.zip' into the Norsyg directory. Unzip the file:

```
$ unzip ordbank_bm.zip
```

and run the convlex.py program (in the Norsyg directory):

\$ python convlex.py

This gives four files ‘ordbank.tdl’, ‘oble.tdl’, ‘predicates.tdl’, and ‘irregs_ob.tab’, that together with the rest of the grammar can be loaded with the ‘lkb/bigscript’ file.

A.2 Short description

There are two important assumptions made in Norsyg that distinguishes it from other implemented grammars. First, the linking between the syntax and the basic relations is done in the syntax, rather than in the lexicon. And second, the trees are left-branching, which implies that the topic is realized at the bottom of the tree, and not at the top.

A.2.1 Composing argument structure in the syntax

The term syntax-based means that the grammar has emphasis on the syntax rather than on the lexicon, and linking between for example a verb and its arguments is done by functional signs such as combinatorial rules, inflectional rules (passive morphology) or function words (passive auxiliaries, infinitival markers). Since this linking is assumed not to happen in the lexicon, the grammar becomes much more flexible, and a verb with a large number of argument frames is easily accounted for. So-called valence alternations are more seen as the norm than as the exception. An example of such a verb is *drip*:

- (216) a. The roof drips
 b. The doctor drips into the eyes
 c. The doctor drips with water
 d. The doctor drips into the eyes with water
 e. The roof drips water
 f. The roof drips water into the bucket
 g. The doctor dripped the eyes with water
 h. The doctor dripped into the eyes with water
 i. John dripped himself two drops of water
 j. John dripped himself two drops of water into the eyes

- k. John dripped himself two drops of water into the eyes with a drop counter
- l. Water dripped
- m. Water dripped into the bucket
- n. It drips
- o. It drips into the bucket.

In Norsyg, four argument roles are assumed, corresponding to deep syntactic functions. The four argument roles are:

- Argument 1 role: Corresponds to the external argument role in GB.
- Argument 2 role: Corresponds to the deep direct object role.
- Argument 3 role: Corresponds to the deep indirect object role.
- Argument 4 role: Corresponds to predicatives/resultatives/end-of-paths

With a syntactic approach such as the one in Norsyg, it is possible to account for all the argument frames of drip with only one lexical entry.

Each of the syntactic argument roles are directly mapped to corresponding basic relations, and so the Basic Relation Representation (BRR) is composed as the syntactic structure is built.

A.2.2 Left-branching tree structures

The second important assumption made in Norsyg is that tree structures are left-branching, which implies that the topic of the sentence is realized at the bottom of the tree. If the topic is topicalized, the extraction site is assumed to dominate the topic. In the analysis of *Kari sover* ('Kari sleeps') in Figure A.1, the VP/NP rule realizes the topic (Kari). The unary rule (VP1) extracts the subject and realizes the arg1-role. (The digit on a node label indicates which argument role that is realized.) The unary S rule marks the clause as a proposition.

An analysis of the ditransitive sentence *Hun gir Kari en is* ('Hun gives Kari an ice-cream') is given in Figure A.2. Here the VP1-rule extracts the subject (which is realized

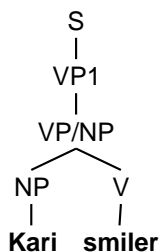


Figure A.1: Intransitive sentence

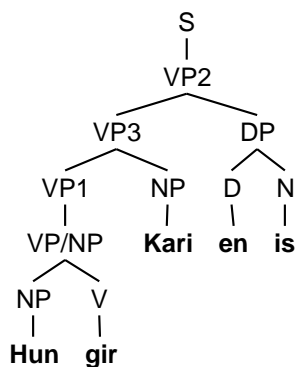


Figure A.2: Ditransitive sentence

by the VP/NP rule), the VP3-rule realizes the indirect object, and the VP2-rule realizes the direct object.

The tree in Figure A.3 gives an analysis of the sentence *Boka hevder Jon at han har lest* ('The book Jon claims that he has read') where the topic *Boka* is extracted from the second subordinate clause. The node VP2 is the rule that extracts the topic, and as the analysis shows, the extraction site dominates the topic.

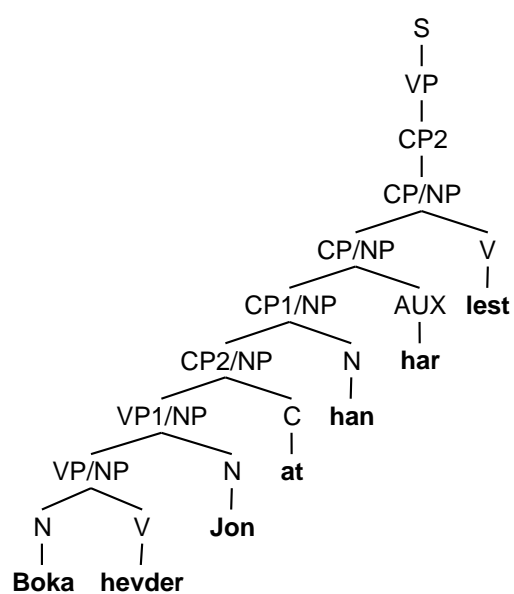


Figure A.3: Extraction from subordinate clause

A.3 Data

Grammar	Norsyg
Author	Petter Haugereid
Start date	2002
Person-years to date	
License	LGPL
Version	oct-08
Number of lexical leaf types	232
Number of lexical rules	0
Number of syntactic rules	52
Total number of types (no GLBs)	1 346
Lexical entries: Hand-built	1 300
Lexical entries: External source	144 156
Lines of TDL (excl lexicon)	5 723
Lines of comments	699
External morphology	No
Preprocessor	Yes
Lexical database	No
Unknown word mechanism	Yes
Idioms	No
Test suites	test.items: general (335) nkl.items: argument frames (107) ex.items: examples from thesis (146) eng-ex.items: English examples (213)
Treebanks	No
Parse-ranking model	No
Generation (trigger rules)	No
Realization-ranking model	No
Paraphrasing rules	No
SEM-I	No
Application(s)	No

Processing engines

LKB

Operating systems

Linux/Windows/MacOS/Solaris

A.4 Coverage

Aggregate	total items #	positive items #	word string \emptyset	lexical items \emptyset	distinct analyses \emptyset	total results #	overall coverage %
i-length in [40 .. 45]	1	1	42.00	0.00	0.00	0	0.0
i-length in [35 .. 40]	3	3	35.33	0.00	0.00	0	0.0
i-length in [30 .. 35]	13	12	31.50	0.00	0.00	0	0.0
i-length in [25 .. 30]	22	22	26.73	99.00	0.00	0	0.0
i-length in [20 .. 25]	38	37	21.46	99.00	625.14	7	18.9
i-length in [15 .. 20]	68	68	17.09	78.97	318.75	16	23.5
i-length in [10 .. 15]	79	71	11.76	52.23	72.54	41	57.7
i-length in [5 .. 10]	72	61	7.18	28.87	10.02	44	72.1
i-length in [0 .. 5]	47	38	2.18	8.24	1.79	34	89.5
Total	343	313	14.14	44.42	91.21	142	45.4

(generated by [incr tsdb()] at 2-oct-08 (10:04))

Figure A.4: Norsyg tested on Wikipedia article on ‘Concrete’

The table in Figure A.4 shows that Norsyg parses 45.4% of the items of an article on concrete. The article, which has 313 grammatical items, was taken from Norwegian Wikipedia articles marked as excellent, and no changes were made to the grammar in order to adapt it to the data. A manual inspection of all the items that parsed, using the [incr tsdb()] treebanking tool (Oepen, 2001), revealed that 107 out of 142 items (75.4%) had the intended analysis. This means that Norsyg has a coverage of 34.2% of the grammatical items in the article. Some of the overgeneration stems from the use of an unknown word mechanism which assigns a underspecified nominal interpretation to all unknown words.

A.5 NorKompLeks test sentences

The following table shows the result of a batch parse of the example sentences for argument frames in NorKompLeks. It contains 107 items and Norsyg parses all of them. For each item, the NorKompLeks code is given in the right column. A few frames like *part5* and *predic11*, *trans2* and *trans18*, *trans3* and *trans19*, *refl12* and *refl18*, *adv2* and *adv13* share one example. *part3* and *refl14* share two examples. *adv15* and *refl10* each correspond to two examples, and *aux1* corresponds to three examples. A text file (nkl.items) containing all the examples below is distributed with Norsyg.

Nr	Example	Parses	Edges	NKL-frame
1	det buldrer	1	28	nullv
2	det rabler for ham	1	41	nullv2
3	det kvakk i henne	1	37	nullv1
4	det løper en hund opp bakken	4	82	present2
5	det sitter en hund på trappen	3	83	present3
6	det kommer en mann	1	46	present1
7	det aner meg at jon smiler	1	69	scomp1
8	de tenker	1	18	intrans1
9	de krangler	1	16	intrans4
10	han fryser	1	16	intrans3
11	brevet ankom	1	19	intrans2
12	han stoler på jon	1	54	trans11
13	jon kakker på døra	1	40	adv4
14	jon truer med at han smiler	2	57	trans20
15	de bytter på å smile	1	73	trans23
16	jon lengter etter kari	2	40	trans15
17	jon tviler på at kari smiler	2	60	trans21
18	jon frastår fra å smile	1	60	trans13
19	jon lurert på hva som skal skje	2	84	hv3
20	resultatet avhenger av at jon kommer	2	53	trans12
21	jon bor i byen	1	36	adv5
22	jon avhenger av å smile	2	58	trans22
23	jon jobber som lærer	2	58	predik13
24	jon later som han er syk	1	85	adv16

25	jon later som om han er syk	2	133	adv17
26	jon framstår som en god lærer	2	74	predik12
27	kaffen lukter is	2	30	adv15
28	kaffen lukter godt	1	32	adv15
29	jon er lærer	2	52	predik1
30	jon er snill	1	37	predik2
31	jon livner til	1	25	part4
32	jon kler på seg	1	39	refl13
33	jon fyrer opp	1	24	part5,predik11
34	jon labber til byen	2	43	adv3
35	bilen slingrer nedover veien	2	44	adv12
36	han gleder naboen	6	54	trans10
37	mannen kjøpte en bil	5	71	trans1
38	han bygger hus	4	53	trans9
39	han sa at han kommer	1	49	trans2,trans18
40	jon prøver å komme	1	45	trans3,trans19
41	de diskuterer hva som skjedde	3	72	hv1
42	han foretrekker opera	2	38	trans8
43	han hater at kari smiler	1	48	trans16
44	han hater å smile	1	56	trans17
45	jon arver en skog	2	39	trans14
46	han vet hva som skjedde	2	64	hv2
47	saken irriterer gutten	6	55	trans7
48	svampen absorberer vann	2	29	trans5
49	saken gjelder gutten	2	30	trans6
50	kari byr jon på is	7	65	ditrans5
51	kari ansporer jon til å smile	3	83	ditrans6
52	harald samlet norge til et rike	7	114	ditrans8
53	kari gir en bok til jon	13	126	ditrans4
54	jon arver en skog fra kari	4	72	ditrans9
55	han opphøyer seg til gud	2	40	refl9
56	han begraver seg i arbeid	4	55	refl15
57	han forlover seg med noen	2	40	refl19
58	stolen avtegnet seg mot taket	2	53	refl11

59	jon setter koppen på bordet	10	139	adv6
60	jon anser kari for å være snill	3	110	predik10
61	kari ser jon komme	2	89	trans4
62	kari lar noe være usagt	12	320	kaus1
63	jon maler stolen grønn	3	43	predik7
64	jon gasjerte kari høyt	2	55	adv14
65	jon får tilbake pengene	1	51	part6
66	jon kler klærne av seg	4	64	part3,refl14
67	jon kler på seg klær	2	50	part3,refl14
68	jon later etter seg noe	3	67	refl20
69	jon dresser seg opp	1	25	part2
70	jon kreker seg fram	1	25	adv8
71	jon får pengene igjen	2	62	part6
72	jon kaller ham en tosk	2	45	predik3
73	jon verdsetter stolen til en krone	7	102	part7
74	jon anfører stolen som bevis	6	170	predik8
75	jon kaller ham for en tosk	5	71	predik4
76	jon kaller seg direktør	1	26	predik5
77	jon kaller seg snill	1	25	predik6
78	jon kasserer inn pengene	1	35	part1
79	jon klamrer seg til pengene	1	37	adv9
80	jon kanalisierer vannet til skogen	7	65	adv7
81	jon skrubber henne på ryggen	3	44	adv11
82	kari gir ham en bok	1	42	ditrans1
83	kari bemektiget seg skogen	1	34	refl6
84	per lovet jon at han skulle komme	3	108	ditrans2
85	per tenker seg at noe skjer	2	65	refl7
86	per lovet jon å komme	2	84	ditrans3
87	per pålegger jon å komme	2	58	ditrans7
88	per lot dem komme	1	43	trans4
89	jon kan tenke seg å komme	1	53	refl8
90	jon kylte henne en snøball i nakken	1	71	adv10
91	jon ombestemmer seg	1	19	refl4
92	de samrår seg	1	16	refl2

93	de skammer seg	1	20	refl1
94	blodet kaker seg	1	29	refl5
95	jon bemøyer seg med isen	1	35	refl9
96	jon nedlater seg til å smile	1	56	refl3
97	kari gleder seg over isen	4	50	refl12,refl18
98	kari gleder seg over at per kommer	4	67	refl16
99	kari gleder seg over å smile	4	70	refl17
100	oppskriften baserer seg på frukt	1	41	refl10
101	summen beløper seg til en krone	1	53	refl10
102	jon åpenbarte seg som en god lærer	3	88	predik9
103	saken arter seg merkelig	2	32	adv2,adv13
104	jon lar seg lure	1	31	kaus2
105	jon blir beundret	1	33	aux1
106	jon har beundret kari	1	47	aux1
107	jon kan smile	1	26	aux1

Total CPU time: 7020 msecs

Mean edges: 56.90

Mean parses: 2.41

A.6 Technical details about case and linking

The way information about which subconstructions that have applied in a clause is gathered, is theoretically not very interesting, since it can be implemented in different ways. In this section I give a presentation of how it is implemented in Norsyng.

A.6.1 The linking mechanism

In this section, I will explain in more detail how the argument structure information provided by the functional signs is matched with the argument structure constraints specified on the main verb.

As argued in Sections 4.3, 5.1, and 6.1 there are four kinds of valence rules. In the linking rules I assume that the linking type of the argument that the linking rules realize, is switched from minus in the mother to plus in the daughter. The other valence

features are kept the same. So the *arg1-val* has the constraints in Figure A.5, where ARG1|LINK *arg1-* in the mother is switched to ARG1|LINK *arg1+* in the daughter.

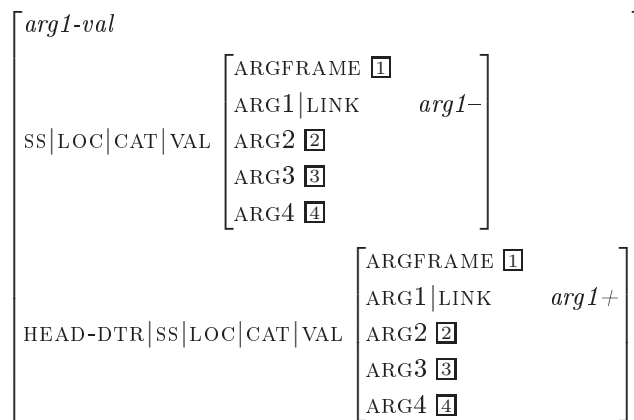


Figure A.5: Valence constraints on the *arg1-val*

As mentioned in Section 6.3 the force-rules constrain their head daughters to have only negative values of the LINK features (see *force-phrase* in Figure 6.9, p. 157). Each valence rule switches one negative value in the mother to a positive value in the daughter. After the valence rules have worked, the relevant linking information of the clause is ready to be gathered as positive and/or negative linking types in the first constituent of the clause, or in the rule that realizes the first constituent of the clause. This was shown in Sections 4.3.4 and 6.1. The unification of linking types is done in the type *uni-link* (see Figure A.6).

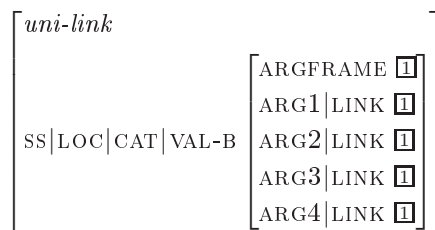


Figure A.6: Unification of linking types

The feature *val-b* used in Figure A.6 was introduced in Section 5.1, and is used to account for subconstructions that are not realized as phrase structure rules (see Section 7.1). Words that do not realize subconstructions (all words except passive verbs, passive auxiliaries, imperative verbs, and infinitival markers) unify the value of

VAL-B with the value of VAL. Phrases that do not have a second daughter that realizes a subconstruction (e.g. unary phrases), unify the value of VAL-B with the value of VAL. In phrases where the second daughter may realize a subconstruction (the filler rule, if the second daughter is a passive verb or a passive auxiliary, or the binary infinitival rule, where the second daughter is the infinitival marker), the value of VAL-B is the output of this subconstruction, and the unification of linking types only applies here. The words and phrases that inherit from *uni-link* are the following:

1. The words that introduce embedded structures:
 - Complementizers
 - The relative pronoun
 - The infinitival marker
2. The unary rules that introduce embedded structures:
 - *unary-compl-phrase*
 - *unary-rel-phrase*
 - *unary-inf-phrase*
3. The *head-filler-phrase*
4. The first constituent

By unifying linking types in the first two kinds of constituents, I account for the unification of linking in subordinate structures. By unifying linking types in the last two kinds of constituents, I account for the unification of linking in the main clauses.

Letting the first constituent unify the linking types is necessary in main clauses where the head filler rule is not employed (yes-no questions and imperatives). An example of linking types in a yes-no question is given in Figure A.7.¹

¹In Figure A.7, the subconstructions *arg1-sign* and *arg2-sign* switch LINK values from *minus* in the mother to *plus* in the daughter. The result of all the switches ends up in the first constituent of the clause (or the rule that realizes the first constituent). The value of ARGFRAME specified on the verb *smiled* (*arg1-12*) is unified with the ARGFRAME of the projection of the auxiliary, and is therefore also present in the first constituent.

The first constituent of the tree in Figure A.7 (the auxiliary *has*) unifies the LINK values and the ARGFRAME value, as shown in Figure 4.11. This unification is left out in Figure A.7 in order to show how the different LINK values end up in the first word. The unification of the types *arg1+*, *arg2+*, *arg3-*, *arg4-*, and *arg1-12* gives the type *arg12* (see Figure 4.9, page 96).

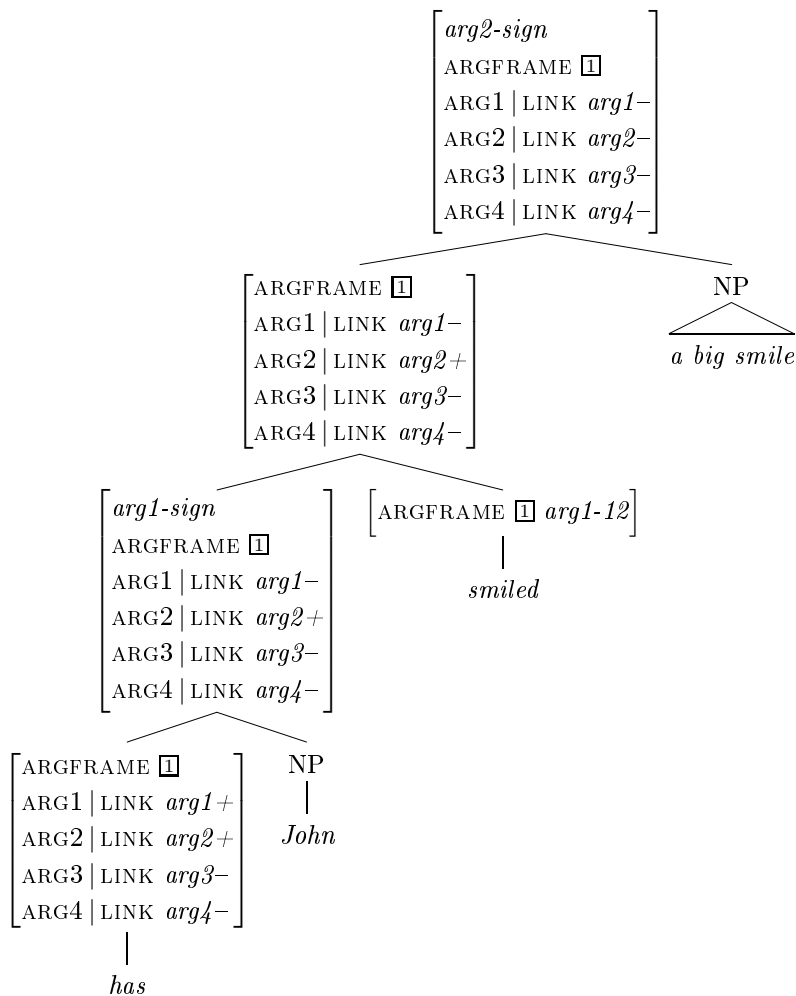


Figure A.7: Linking types in a transitive clause

I use two strategies to find the first word. Neither of them are satisfactory, since they attempt to do something that should rather be a part of the LKB system than the grammar. The first strategy is to let the first word in a sentence start with the letter ‘q’, like in *qJon sover*. The prefix *q* is realized by an inflectional word-to-word rule that inherits from the type *uni-link*. The second strategy is to let the word-to-word rule be a non-inflectional rule. Then one can parse sentences without using the prefix *q*, but instead I use a mechanism that involves a feature `FIRST-WORD` *bool*. The force rules constrain their daughter to have the `FIRST-WORD` value *plus*. All rules unifies the `FIRST-WORD` value in the mother with that of the first daughter. Non-first daughters are constrained to have the `FIRST-WORD` value *minus*. In this way, the first word, and

only the first word, will be constrained to have the FIRST-WORD value *plus* when the whole sentence is parsed. This is not an optimal procedure since the settling of the FIRST-WORD value is delayed until the whole sentence is parsed, and the word-to-word inflectional rule is allowed to apply to all words and be part of several subtrees that lead to no parse.

A comparison of the two strategies tested on the Norwegian example data used in this thesis, show that the strategy that involves the *q* suffix is far more efficient than the strategy that employs the non-inflecting word-to-word rule. The comparison is shown in Figure A.8, where the test with the non-inflected rule is marked as ‘(g)old’, and the test with the ‘q’ prefix is marked as ‘new’. The tabular reports a 42.7% reduction in tasks, a 38.6% reduction in time, and a 15.4% reduction in space on average with the strategy that involves the *q* suffix compared to the strategy that employs the non-inflecting word-to-word rule. The two strategies have the same coverage on the data. In the batch tests of Norwegian, English, and German data in Appendixes A, B, and C, I report how the grammar performs with the most efficient strategy.

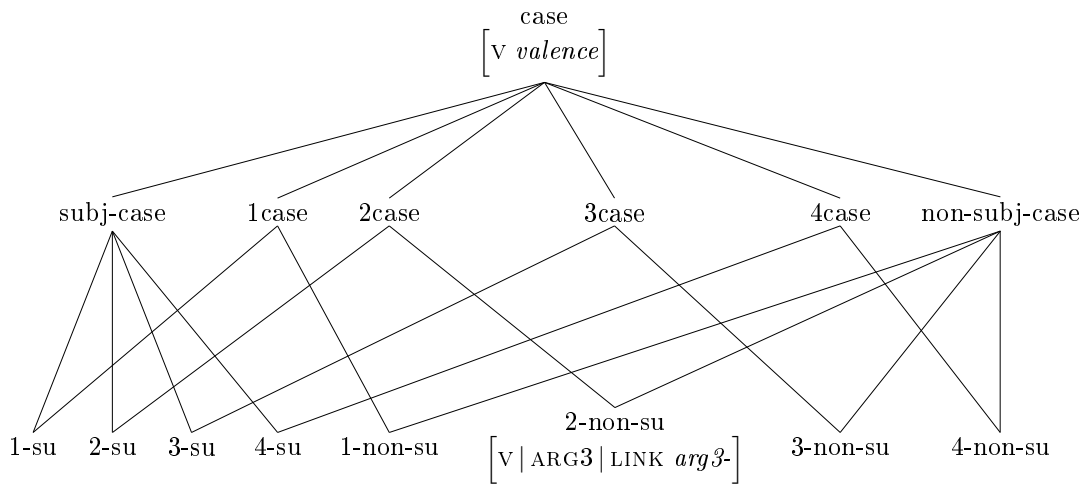
Aggregate	(g)old			new			reduction		
	tasks ∅	time ∅	space ∅	tasks ∅	time ∅	space ∅	tasks %	time %	space %
i-length in [10 .. 15]	2807	1.01	39385	1456	0.58	24968	48.1	43.1	36.6
i-length in [5 .. 10]	590	0.21	15012	331	0.12	12369	43.9	42.2	17.6
i-length in [0 .. 5]	214	0.08	10448	143	0.06	9831	33.2	19.8	5.9
Total	494	0.18	13793	283	0.11	11672	42.7	38.6	15.4

(generated by [incr tsdb()] at 2-oct-08 (17:35))

Figure A.8: Comparison of two strategies for settling the first word.

A.6.2 Case

Norwegian has two cases, *subj-case* case and *non-subj-case* case. These two cases I cross-classify with information about what kind of role the argument is, as illustrated in Figure A.9. This gives me eight case types: *arg1-su-case*, *arg2-su-case*, *arg3-su-case*, *arg4-su-case*, *arg1-non-su-case*, *arg2-non-su-case*, *arg3-non-su-case* and *arg4-non-su-case*.

Figure A.9: Type hierarchy below the type *case*

The type *case* in Figure A.9 introduces a feature *v* with the value *valence*.² The constraint on *2-non-su* in Figure A.9 makes sure that no *arg3*-role is realized after the non-subjective *arg2*-role is realized. There is a constraint in *arg2-sign* that unifies its *ARG3|LINK* value with the *ARG3|LINK* value of the *case* type of the non-head daughter. This means that if the non-head daughter is non-subjective, then the *ARG3|LINK* value of the phrase is *arg3-* and the *arg3-binary* rule cannot apply later in the projection. It must have applied earlier in the projection or not at all. This is illustrated in Figure A.10.³

²The only function of the feature *v* is to introduce the type *valence*.

³It should be noted that these constraints are language-specific. In languages with variable word order, there would be no such constraints on the case types. This is exemplified for German in Appendix B.2.

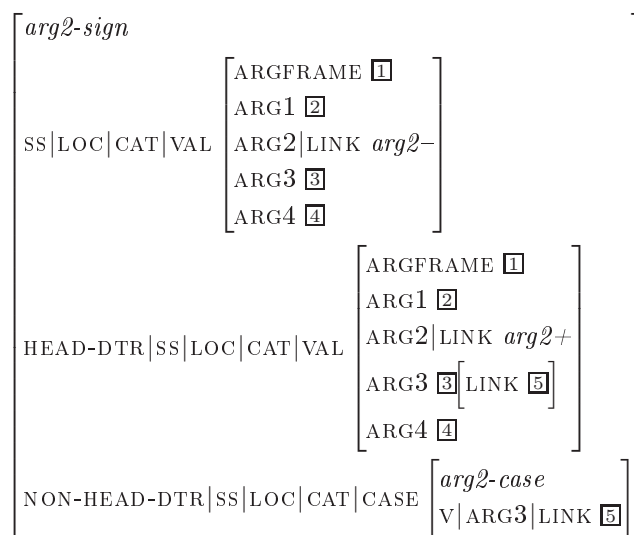


Figure A.10: Valence constraints on the arg2-phrase

Appendix B

Demo grammars for English and German

In order to demonstrate how the analysis in Norsyg can be extended to English and German, I have made demo grammars for the two languages. The demo grammars can be loaded with the ‘/norsyg/lkb/eng-script’ file and the ‘/norsyg/lkb/ger-script’ file, respectively.¹ In addition to the type files of Norsyg, they have a separate language specific type file that overwrite/add types. The grammars are equipped with tiny lexicons and test suites (‘eng.items’ and ‘ger.items’). The results of batch tests of the test files are given below, where phenomena such as valency, word order in main clauses and subordinate clauses, yes-no questions, passive, long distance dependencies, and position of sentence adverbials are tested. Both grammars generate.

B.1 English demo grammar

The demo grammar for English has almost all of its types in common with Norsyg. The difference between the two grammars is given in the file ‘eng.tdl’ where types from Norsyg are either overwritten or given additional subtypes. (9 types are changed and 20 types are added.)

¹See Appendix A for download instructions.

Nr	Example	Parses	Edges
1	John sleeps.	1	25
2	John admires Mary.	1	26
3	*John admires.	0	18
4	*John sleeps Mary.	0	30
5	John gives Mary Bill.	1	30
6	John gives Bill.	1	25
7	John is admired.	2	37
8	*John is slept.	0	30
9	John likes Mary.	1	26
10	John says that Mary smiles.	1	47
11	John says Mary smiles.	1	43
12	John likes to smile.	1	54
13	John lets Mary sleep.	1	48
14	John does admire Mary.	2	44
15	John has admired Mary.	2	46
16	Mary, John admires.	1	26
17	Who does John admire?	1	38
18	Mary, John lets sleep.	1	57
19	Mary, John lets Bill admire.	1	57
20	That Mary smiles, John says.	1	59
21	*That Mary smiles, has said John.	0	41
22	John never sleeps.	1	28
23	*Never John sleeps.	0	24
24	*John sleeps never.	0	30
25	Bill, John never admires.	1	32
26	*Bill, never John admires.	0	24
27	*Bill, John admires never.	0	29
28	Who does John never admire?	1	45
29	*Who never does John admire?	0	32
30	*Who does never John admire?	0	34
31	John has been admired.	2	44
32	John has never been admired.	1	45
33	John never has been admired.	1	41

34	John says that Mary never sleeps.	1	51
35	*John says that never Mary sleeps.	0	39
36	*John says that Mary sleeps never.	0	46
37	John says Mary never sleeps.	1	47
38	*John says never Mary sleeps.	0	35
39	*John says Mary sleeps never.	0	43
40	John likes to never sleep.	1	49
41	John likes never to sleep.	0	44
42	*John likes to sleep never.	0	56
43	John says that Bill likes to admire Mary.	1	87
44	Mary, John says that Bill likes to admire.	1	80
45	To admire Mary, John says that Bill likes.	1	76
46	That Bill likes to admire Mary, John says.	1	89
47	*Bill, John says sleeps.	0	41
48	Who is John given?	1	42
49	Who is John never given?	1	48
50	Bill, John says that Mary is given.	1	68
51	John lets Mary let Bill admire John.	1	143
52	Does John dine?	1	18
53	*Dines John?	0	10
54	Does John never dine?	1	24
55	*Does never John dine?	0	16
56	*Does John dine never?	0	21
57	Has John slept?	1	18
58	Has John been admired?	1	29
59	*Does John have dined?	0	28
60	John dines in Trondheim.	1	30
61	In Trondheim, John dines.	1	34
62	*In Trondheim, dines John.	0	39
63	Where does John dine?	1	25

Total CPU time: 3270 msecs

Mean edges: 41.13

Mean parses: 0.71

The coverage of the grammar on the test suite is shown in Table B.2. One item did not parse, namely *John likes never to sleep*, where *never* modifies the infinitival clause. (Also the Norsyg fails to give this analysis to the corresponding Norwegian sentence *Jon liker aldri å sove.*) There was no overgeneration. The test suite was also batch parsed with the ERG grammar (version 17-Mar-07), which had 100% coverage and no overgeneration.

Aggregate	total items #	positive items #	word string Ø	lexical items Ø	distinct analyses Ø	total results #	overall coverage %
i-length in [5 .. 10]	25	17	6.00	13.41	1.00	16	94.1
i-length in [0 .. 5]	38	25	3.60	7.80	1.16	25	100.0
Total	63	42	4.57	10.07	1.10	41	97.6

(generated by [incr tsdb()] at 2-oct-08 (15:04))

Table B.2: Coverage of the English demo grammar on the English test sentences

B.2 German demo grammar

The German demo grammar is more different from Norsyg than the English demo grammar, mainly due to the fact that German allows for scrambling, has a more developed case system, and that non-head verbs (that is, non-finite verbs and verbs in subordinate clauses) tend to be realized at the end of the clause. The file ‘ger.tdl’, where types from Norsyg are either overwritten or given additional subtypes, has 45 types. The grammar does not handle infinitival clauses and raising/control, and test items involving these phenomena are not included in the test suite.

Scrambling is accounted for by removing the constraints on the *case* types that account for the order of the syntactic arguments in the Norwegian grammar (see Appendix A.6) and by adding 4 extra valence rules. The valence rules are added in order to account for the fact that German does not have a fixed subject position. New case types are added in order to account for (a fraction of) the case system. Analyses of the subordinate clauses in (55), p. 54 are given in Figures B.1–B.4.

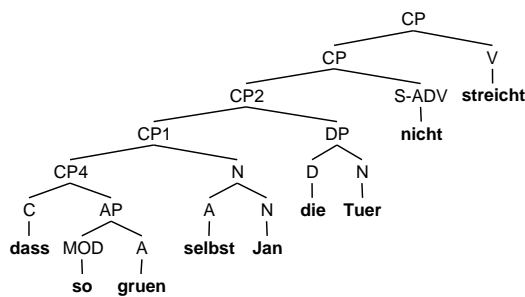


Figure B.1: Analysis of *daß so grün selbst Jan die Tür nicht streicht* ('that not even Jan would paint the door that green') (BRR: D.47, p. 351)

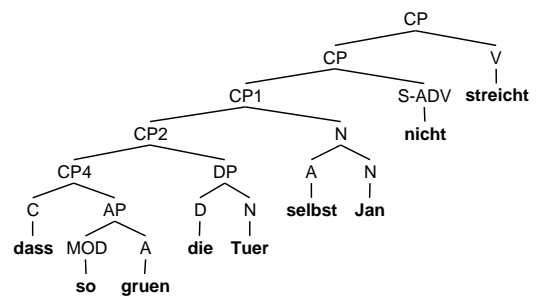


Figure B.2: Analysis of *daß so grün die Tür selbst Jan nicht streicht* ('that not even Jan would paint the door that green') (BRR: D.47, p. 351)

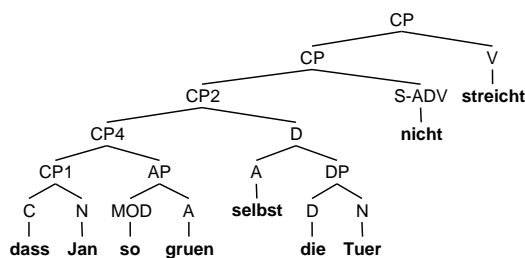


Figure B.3: Analysis of *daß Jan so grün selbst die Tür nicht streicht* ('that not even Jan would paint the door that green') (BRR: D.48, p. 352)

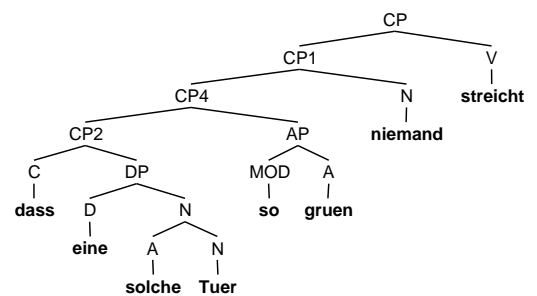


Figure B.4: Analysis of *daß eine solche Tür so grün niemand streicht* ('that nobody paints such a door that green') (BRR: D.49, p. 352)

The order of the German verbs is accounted for by means of an auxiliary feature MERGE2 which allows the merge requirement go to the end of the clause, and then come as a top-down constraint. As the rule in Figure B.5 illustrates, the value of MERGE is unified with the MERGE value of the first daughter. This means that the merge requirement of a constituent is not unified directly with the first verb that merges with it. The clause boundary rules unifies the MERGE value with the MERGE2 value. The MERGE2 feature imposes a top-down constraint, and constrains the last verb in the clause. The merge constraints of the verbs that merge with the head projection are unified with the MERGE2 value of the first daughter of the merge rule. In this way, verbs that merge with the head projection constrain verbs that precede them.

The tabular below demonstrates some of the phenomena that the German demo grammar covers. These test items are in the file 'ger.items' in the norsyg directory.

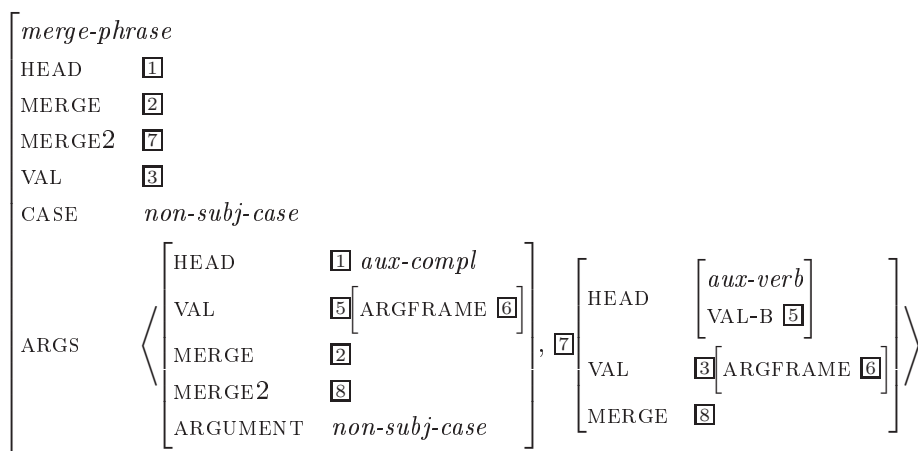


Figure B.5: The German merge rule

Nr	Example	Parses	Edges
1	John schläft.	1	12
2	Er bewundert Mary.	1	25
3	*Er bewundert.	0	16
4	*Er schläft Mary.	0	16
5	Er gibt ihm ihn.	1	26
6	Er gibt ihn.	1	19
7	*Er gibt ihm.	0	19
8	Er mag dass Mary schläft.	1	44
9	Er mag dass Mary ihn bewundert.	1	61
10	*Er mag dass Mary bewundert ihn.	0	55
11	*Er mag er dass schläft.	0	26
12	Er hat Mary bewundert.	1	45
13	Mary soll er bewundern.	1	40
14	Mary hat er bewundern sollen.	1	60
15	*Mary hat er sollen bewundern.	0	48
16	Mary mag er.	1	21
17	Mary hat er bewundert.	1	44
18	*Mary hat bewundert er.	0	31
19	Dass Mary schläft mag John.	1	35
20	*Dass Mary schläft John mag.	0	26

21	John schläft nie.	1	19
22	Nie schläft John.	1	17
23	*John nie schläft.	0	12
24	Er bewundert nie Mary.	1	32
25	Er bewundert Mary nie.	1	35
26	Mary bewundert er nie.	1	32
27	Mary bewundert nie er.	1	32
28	*Mary nie bewundert er.	0	19
29	*Nie er bewundert Mary.	0	20
30	*Nie Mary bewundert er.	0	19
31	Er hat nie Mary bewundert.	1	52
32	Er hat Mary nie bewundert.	1	61
33	*Er hat Mary bewundert nie.	0	48
34	*Er nie hat Mary bewundert.	0	27
35	Er sagt dass Mary nie schläft.	1	61
36	Er sagt dass nie Mary schläft.	1	50
37	Er sagt dass Mary nie geschlafen hat.	1	67
38	*Er sagt dass Mary geschlafen hat nie.	0	55
39	Er gibt ihm ihn.	1	26
40	Er gibt ihn ihm.	1	26
41	Ihm gibt ihn er.	1	25
42	Ihn gibt ihm er.	1	25
43	*Er gibt ihn er.	0	23
44	Er wird bewundert.	1	22
45	*Er wird geschlafen.	0	15
46	Er ist bewundert worden.	1	29
47	Er ist nie bewundert worden.	1	36
48	*Er nie ist bewundert worden.	0	23
49	*Er ist bewundert nie worden.	0	31
50	Er soll bewundert worden sein.	1	39
51	Ihm wird er gegeben.	1	26
52	Ihm wird er nie gegeben.	1	32
53	Schläft John?	1	11
54	Schläft John nie?	1	17

55	Schläft nie John?	1	15
56	Gibt er ihm ihn?	1	21
57	Gibt nie er ihm ihn?	1	25
58	Gibt er nie ihm ihn?	1	26
59	Gibt er ihm nie ihn?	1	26
60	Gibt er ihm ihn nie?	1	27
61	Gibt ihm er ihn?	1	21
62	Gibt ihm ihn er?	1	21
63	Gibt er ihn ihm?	1	22
64	Gibt ihn er ihm?	1	21
65	Gibt ihn ihm er?	1	20
66	Ihn sagt er dass er sah.	1	42
67	*Er sagt er dass ihn sah.	0	31
68	John sagt dass er ihm gegeben worden ist.	1	59
69	Ihm sagt John dass er gegeben worden ist.	2	74
70	*Er sagt John dass ihm gegeben worden ist.	0	56

Total CPU time: 3160 msecs

Mean edges: 32.00

Mean parses: 0.71

The grammar parses all of the items in the test suite, as Table B.4 shows. There was no overgeneration. The test suite was also tested with the German Grammar (<http://gg.dfki.de/demo/gg>), where all positive items parsed, and there was no overgeneration.

Aggregate	total items #	positive items #	word string \emptyset	lexical items \emptyset	distinct analyses \emptyset	total results #	overall coverage %
i-length in [5 .. 10]	30	19	5.63	8.68	1.05	19	100.0
i-length in [0 .. 5]	40	30	3.60	5.47	1.00	30	100.0
Total	70	49	4.39	6.71	1.02	49	100.0

(generated by [incr tsdb()] at 2-oct-08 (16:25))

Table B.4: Coverage of the German demo grammar on the German test sentences

Appendix C

Example sentences of the thesis

In this appendix batch parses of the example sentence of the thesis are shown. The Norwegian data are shown in Section C.1, and the English data are shown in Section C.2. Text files ('ex.items' and 'eng-ex.items') containing all the examples are distributed with Norsyg.¹

C.1 Norwegian example sentences

This section shows the result of a batch parse of all the Norwegian examples in the thesis. The examples are listed in the order they occur in the thesis. Norsyg parses 99.2% of the grammatical items. Some phenomena, like extraction from PP complements and ellipsis, are yet to be covered by Norsyg. The coverage of the Norsyg grammar on the Norwegian example sentences is illustrated in Table C.1.

¹The tests can be replicated by adding a *q*-prefix to the first word of each item in the test files and activating the *first-word-prefix* inflectional rule that inherits from the type *infl-first-prefix*.

Aggregate	total items #	positive items #	word string ∅	lexical items ∅	distinct analyses ∅	total results #	overall coverage %
i-length in [10 .. 15]	3	3	10.33	27.33	7.00	3	100.0
i-length in [5 .. 10]	88	78	5.94	16.86	4.32	77	98.7
i-length in [0 .. 5]	55	43	3.44	10.23	2.02	43	100.0
Total	146	124	5.18	14.81	3.59	123	99.2

(generated by [incr tsdb()] at 2-oct-08 (17:41))

Table C.1: Coverage of the Norsyg grammar on the Norwegian example sentences

Nr	Example	Parses	Edges
(xxvii)	Studiet lar seg lett kombinere med en jobb.	8	126
(41a)	Jon overrekker Kari to bananer.	8	114
(41b)	Kari blir overrakt to bananer.	5	69
(41c)	To bananer blir overrakt Kari.	2	54
(41d)	Det blir overrakt Kari to bananer.	4	87
(45a)	Det forsvant en mynt i gresset.	2	82
(45b)	Det lekte noen barn i gresset.	2	89
(59a)	Det kommer en mann.	1	47
(59b)	Det blir sendt en pakke.	4	67
(60a)	En mann arbeider på åkeren.	2	60
(60b)	Det blir arbeidet på åkeren.	7	103
(60c)	Det arbeider en mann på åkeren.	2	100
(lxi)	Den broen ble det funnet et lik under.	2	162
(62a)	Marit snakker Jon med.	3	40
(62b)	*Mandag kommer Jon på.	4	48
(lxiii a)	*Det utstråler en sol varme.	0	63
(lxiii b)	Det utstråler varme fra sola.	7	82
(66a)	En spiller smashet.	1	32
(66b)	*Det smashet en spiller.	2	61
(67a)	En spiller smashet en ball.	2	54
(67b)	En ball ble smashet.	1	42
(67c)	Det ble smashet en ball.	1	66
(69a)	En avis brenner.	1	23
(69b)	Det brenner en avis.	3	59
(lxxa)	Læreren forberedte seg godt.	3	54
(lxxb)	*Læreren forberedte godt.	0	50
(lxxiii)	Det fins ikke matbiten i huset.	12	200
(lxxiv)	Det står navnet ditt på døra.	8	124
(75a)	Jon gir Kari en bok.	3	62
(75b)	Kari blir gitt en bok.	2	45
(75c)	Det blir gitt Kari en bok.	1	65
(75d)	*Det blir gitt Kari boka.	0	59
(75e)	En bok blir gitt Kari.	2	41

(83a)	Bilen kjører inn i garasjen.	11	74
(83b)	Marit kjører bilen inn i garasjen.	14	103
(83c)	Det kjører en bil inn i garasjen.	22	162
(83d)	Det kjøres inn i garasjen.	11	81
(85a)	Vann drypper fra taket.	5	65
(85b)	Taket drypper vann.	2	40
(85c)	Det drypper vann fra taket.	14	152
(85d)	*Det drypper et tak vann.	3	120
(91a)	Jon spiser.	1	17
(91b)	Jon spiser en kake.	2	44
(91c)	Det spises.	2	27
(91d)	Kaker spises.	1	23
(91e)	*Det spiser en mann.	2	54
(107a)	En bil skramlet.	1	32
(107b)	*Det skramlet en bil.	3	66
(107c)	En bil skramlet inn oppkjørselen.	3	60
(107d)	Det skramlet en bil inn oppkjørselen.	13	133
(109a)	Vi ventet en overraskelse.	3	97
(109b)	Det ble ventet en overraskelse.	1	73
(109c)	En overraskelse ventet oss.	3	44
(109d)	Det ventet oss en overraskelse.	1	80
(120)	Jon ombestemmer seg.	1	20
(141a)	Jon ser ikke Kari.	4	123
(141b)	Jon har ikke sett Kari.	1	96
(141c)	at Jon ikke har kommet	1	39
(142a)	I går leste Kari en bok.	1	78
(142b)	Leste Kari en bok?	1	47
(147)	at han beundrer Marit	1	30
(161a)	Jon spaserer i skogen.	1	34
(161b)	Mannen i skogen hogger ved.	2	54
(162a)	Om ettermiddagen spaserer Jon 5 kilometer.	2	80
(162b)	Hvor spaserer Jon om ettermiddagen?	2	50
(163a)	Mannen hogger ikke ved i skogen.	14	109
(163b)	Jon hevder at mannen ikke hogger ved i skogen.	4	92

(164a)	Jon hevder at ikke Marit vil vinne.	2	66
(164b)	Jon prøver ikke å le.	1	65
(164c)	Ikke le!	1	18
(165)	Jon hevder at ikke Marit ikke vil vinne.	1	65
(167a)	En spiller smasher ballen.	2	35
(167b)	Ballen blir smashet.	1	38
(167c)	Ballen smashes.	1	14
(168a)	Marit blir gitt en is.	2	47
(168b)	En is blir gitt Marit.	2	43
(168c)	Det blir gitt Marit en is.	1	67
(169a)	Bleier ble byttet på barna.	1	71
(169b)	Det ble byttet bleier på barna.	2	93
(169c)	Barna ble byttet bleier på.	0	59
(169e)	Bleiene ble byttet på barna.	1	69
(169f)	*Det ble byttet bleiene på barna.	0	85
(169g)	*Barna ble byttet bleiene på.	0	57
(171a)	Mannen kommer.	1	18
(171b)	Det kommer en mann.	1	47
(171c)	*Det kommer mannen.	0	37
(172a)	Mannen blir beundret.	1	38
(172b)	Det blir beundret en mann.	1	72
(172c)	*Det blir beundret mannen.	0	60
(173)	Marit spiser en is og drikker kaffe.	2	101
(174a)	Marit spiser ikke is og drikker kaffe.	4	131
(174b)	Marit spiser is og drikker ikke kaffe.	2	101
(174c)	Marit spiser ikke is og drikker ikke kaffe.	4	143
(175)	Marit spiser is og blir servert kaffe.	5	110
(176)	Marit fanger, steker og spiser fisken.	2	88
(clxxviiia)	Marit fanger, steker og spiser ikke noenting.	2	94
(clxxviiib)	*Marit fanger, steker ikke og spiser noenting.	0	57
(clxxviiic)	*Marit fanger ikke, steker og spiser noenting.	0	68
(178a)	?Marit fanger, steker og spiser ikke fisken.	4	118
(178b)	*Marit fanger, steker ikke og spiser fisken.	0	60
(178c)	*Marit fanger ikke, steker og spiser fisken.	0	76

(179)	at Marit ikke fanger, steker og spiser fisken.	1	104
(180)	?En overraskelse venter og beundrer ham.	3	87
(181a)	Marit gir Jon en is og Ola en sjokolade.	3	155
(181b)	Marit gir Jon en is og Kari Ola en sjokolade.	3	216
(182a)	Han sitter og skriver dikt.	4	121
(182b)	Han driver og skriver dikt.	14	226
(182c)	Han tok og skrev et dikt.	6	172
(184a)	Han satt i stuen og skrev et brev.	11	217
(184b)	Det var et brev han satt i stuen og skrev .	7	310
(184c)	Det var stuen han satt i og skrev et brev.	11	312
(185)	Han skrev et brev og sendte til England.	4	209
(186)	Han skrev et brev og sendte det til England.	9	266
(190a)	Har Jon spist is?	2	41
(190b)	Spiser Jon is?	2	33
(191a)	at Jon ofte spiser epler	1	37
(191b)	at Jon ofte har spist epler	1	45
(191c)	Jon spiser ofte epler.	2	41
(192a)	I skogen spaserer Jon.	1	39
(192b)	I skogen har Jon spasert.	1	54
(cxciva)	På fredager kommer Jon ofte for sent.	7	101
(cxcivb)	På fredager kommer ofte Jon for sent.	5	92
(202a)	Kari sover aldri.	1	27
(202b)	*Kari aldri sover.	0	20
(203a)	Kari har aldri sovet.	1	46
(203b)	*Kari har sovet aldri.	0	37
(205a)	at Kari aldri sover.	1	30
(205b)	*at Kari sover aldri.	0	28
(206a)	at Kari aldri har sovet.	1	43
(206b)	*at Kari har aldri sovet.	0	41
(207)	Sover aldri Kari?	2	27
(208a)	Marit ser den aldri.	2	130
(208b)	*Marit ser aldri den.	6	162
(208c)	Marit ser aldri DEN.	6	162
(213)	Marit så Jon aldri igjen.	5	165

(210a)	Marit ser aldri dyreprogram.	8	193
(210b)	*Marit ser dyreprogram aldri.	4	179
(211a)	Marit så aldri Jon.	5	122
(211b)	Marit så Jon aldri.	2	106
(212a)	Sover aldri Kari?	2	27
(212b)	Sover Kari aldri?	1	25
(213)	at dyreprogram aldri blir sett av Marit	6	147
(214a)	Gir aldri Jon Marit isen?	4	67
(214b)	Gir han aldri Marit isen?	2	54
(214c)	Gir han henne aldri isen?	2	52
(214d)	Gir han henne den aldri?	1	53
	Mean	3.18	81.92

C.2 English example sentences

This section shows a batch parse of the English sentences in the thesis by the English demo grammar. The coverage of the English demo grammar on the example sentences is illustrated in Table C.3. The grammar parses 94.9% of the examples. (Some examples, like (xiia) and (xiib), are not parsed since they are “odd”.)

Aggregate	total items #	positive items #	word string \emptyset	lexical items \emptyset	distinct analyses \emptyset	total results #	overall coverage %
i-length in [10 .. 15]	2	2	10.00	13.00	0.00	0	0.0
i-length in [5 .. 10]	116	109	6.09	11.06	2.62	107	98.2
i-length in [0 .. 5]	95	86	3.37	7.05	1.29	80	93.0
Total	213	197	4.94	9.32	2.05	187	94.9

(generated by [incr tsdb()] at 2-oct-08 (20:54))

Table C.3: Coverage of the English demo grammar on the English example sentences

Nr	Example	Parses	Edges
(3a)	Brutus stabbed Caesar.	1	23
(4a)	John smashed the ball.	1	30
(4b)	The ball was smashed.	2	39
(4c)	John tried to smash the ball.	1	66
(5a)	John smashed the ball.	1	30
(5b)	The boat arrived.	1	26
(5c)	The ball was smashed.	2	39
(5d)	The car needed to be washed.	1	72
(6a)	John gave Mary a book.	1	34
(6b)	Mary was given the book.	2	44
(6c)	Mary wanted to be given a book.	2	91
(8)	John punctured the balloon with a needle.	2	54
(9a)	John smiles.	1	26
(9b)	John smashed the ball.	1	30
(9c)	The boat arrived.	1	26
(9d)	John gave Mary a book.	1	34
(9e)	John gave a book to Mary.	3	56
(10a)	John eats.	1	18
(10b)	John eats an apple.	1	29
(xia)	*John tries to slept.	0	35
(xib)	*A men smiles.	0	24
(xiia)	John slept the car.	0	23
(xiib)	John admires.	0	18
(xiiia)	John filled the mouth with chocolate.	2	49
(xiiib)	John smiled the mouth with chocolate.	2	52
(14)	*John eats an apple Mary that he smiles.	0	58
(16a)	The man smiled.	1	28
(16b)	*Mary smiled the man.	1	32
(17a)	The glass broke.	1	22
(17b)	Mary broke the glass.	1	27
(18a)	Mary smiled a big smile.	1	45
(18b)	*The glass broke a crack.	1	33
(19a)	The roof drips.	2	26

(19b)	Water drips from the roof.	5	57
(20a)	John hammered the metal.	1	30
(20b)	John hammered the metal flat.	1	37
(21a)	The river froze.	1	22
(21b)	The river froze solid.	1	29
(22a)	The man smiles.	1	35
(22b)	*The man smiles happy.	0	39
(23a)	John ate the apple.	1	27
(23b)	John ate.	1	16
(24a)	John gave Mary an apple.	1	34
(24b)	John gave an apple to Mary.	3	56
(25a)	John loaded hay onto the wagon.	3	50
(25b)	John loaded the wagon with hay.	3	51
(26a)	The butcher cuts the meat.	1	35
(26b)	The meat was cut by the butcher.	2	67
(26c)	The meat cuts easily.	2	31
(31a)	John pounds the metal.	1	29
(31b)	The metal was pounded.	2	39
(46)	Sally baked her sister a cake.	1	64
(47)	They laughed the poor guy out of the room.	2	66
(48)	He talked himself blue in the face.	1	50
(49)	Frank dug his way out of the prison.	2	60
(51a)	The liquid froze solid.	1	29
(51b)	John froze the liquid solid.	1	34
(52a)	It drips.	1	15
(52b)	The roof drips.	2	26
(52c)	The roof drips water.	1	35
(52d)	John drips medicine in the glass.	4	65
(52e)	John drips himself medicine.	1	34
(52f)	John drips himself medicine in the glass.	3	61
(52g)	Water drips.	2	22
(52h)	Water drips into the bucket.	5	57
(53a)	It smiles.	0	21
(53b)	The roof smiles.	1	35

(53c)	The roof smiles water.	1	44
(53d)	John smiles medicine in the glass.	2	64
(53e)	John smiles himself medicine.	0	35
(53f)	John smiles himself medicine in the glass.	0	54
(53g)	Water smiles.	1	31
(53h)	Water smiles into the bucket.	1	63
(58a)	The roof drips.	2	26
(58b)	Water drips from the roof.	5	57
(64a)	John smashed the ball.	1	30
(64b)	The ball was smashed.	2	39
(65)	The roof drips water.	1	35
(68a)	The man likes ice cream.	1	51
(68b)	The man likes to compete.	1	46
(68c)	The man says that it rains.	1	51
(71a)	a punctured ball	0	15
(71b)	*a shouted man	0	15
(71c)	*a come man	0	16
(71d)	an arrived message	0	15
(72a)	Mary gave John a book.	1	34
(72b)	John was given a book.	2	44
(76a)	John put the glass on the table.	2	85
(76b)	John kicked the ball flat.	1	37
(76c)	He sprayed his new car a brilliant shade of green.	0	81
(77)	John kicked the ball flat out of the room.	0	59
(78a)	Mary talks about flowers.	1	35
(78b)	Mary talks to Sandy.	1	35
(78c)	Mary talks to Sandy about flowers.	2	62
(78d)	Mary talks John to sleep about flowers.	5	86
(79a)	Jack sprayed paint on the wall.	2	49
(79b)	Jack sprayed the wall with paint.	2	50
(80a)	Jack sprayed the paint wet on the wall.	1	57
(80b)	Jack sprayed the wall wet with paint.	1	53
(81a)	The glass broke.	1	22
(81b)	John broke the glass.	1	27

(82a)	The horse jumped over the fence.	5	57
(82b)	Sylvia jumped the horse over the fence.	3	59
(84a)	Water drips from the roof.	5	57
(84b)	The roof drips water.	1	35
(86)	Water drips from the roof into the bucket.	14	123
(lxxxvii)	Water is dripped by the roof.	6	79
(88a)	John eats.	1	18
(88b)	John eats a cake.	1	29
(89a)	Sarah smiled.	1	22
(89b)	Sarah smiled a charming smile.	1	45
(90a)	She mumbled.	1	22
(90b)	She mumbled her adoration.	1	47
(92a)	John cut the meat.	2	39
(92b)	John cut in the meat.	6	61
(93a)	Martha climbed the mountain.	1	30
(93b)	Martha climbed up the mountain.	1	38
(94a)	John gave Mary the book.	1	34
(94b)	John gave the book to Mary.	3	56
(95a)	Martha carved the baby a toy.	1	46
(95b)	Martha carved a toy for the baby.	3	61
(96a)	Jack sprayed paint on the wall.	2	49
(96b)	Jack sprayed the wall with paint.	2	50
(97a)	Martha carved the piece of wood into a toy.	9	114
(97b)	Martha carved a toy out of the piece of wood.	0	79
(98a)	Brian hit the stick against the fence.	6	69
(98b)	Brian hit the fence with the stick.	6	69
(99a)	Alison pierced the needle through the cloth.	2	53
(99b)	Alison pierced the cloth with a needle.	2	53
(100a)	Mira blamed the accident on Terry.	3	49
(100b)	Mira blamed Terry for the accident.	3	47
(101a)	The judge presented a prize to the winner.	3	67
(101b)	The judge presented the winner with a prize.	3	63
(102a)	The jeweler inscribed the name on the ring.	2	59
(102b)	The jeweler inscribed the ring with the name.	2	59

(103a)	The guests drank.	1	24
(103b)	The guests drank the teapot dry.	1	42
(103c)	*The guests drank dry.	0	30
(104a)	Pauline hammered the metal.	1	30
(104b)	Pauline hammered the metal flat.	1	37
(105a)	The river froze.	1	22
(105b)	The river froze solid.	1	29
(106a)	The car rumbled.	1	25
(106b)	The car rumbled into the driveway.	2	47
(108a)	We awaited their report.	2	53
(108b)	Their report awaited us.	2	55
(111)	John drips himself water into the eyes.	3	63
(114)	I talked to her yesterday about John.	2	70
(115a)	John gave a book.	1	29
(115b)	John gave Mary a book.	1	34
(115c)	John gave a book to Mary.	3	56
(116a)	John broke the cup.	1	27
(116b)	John broke the cup to pieces.	2	53
(116c)	The cup broke.	1	22
(116d)	The cup broke to pieces.	1	44
(117a)	John smiles.	1	26
(117b)	John smiles a big smile.	1	63
(118a)	It rains.	1	24
(118b)	It rains money.	1	37
(119a)	We awaited their report.	2	53
(119b)	Their report awaited us.	2	55
(121)	The rain let up.	2	72
(122a)	John throws.	1	19
(122b)	John throws the ball.	1	32
(122c)	John throws Mary the ball.	1	37
(122d)	John throws to Mary.	3	47
(122e)	John throws the ball to Mary.	4	65
(122f)	John throws out.	1	27
(122g)	John throws out the ball.	3	49

(122h)	John throws out to Mary.	2	51
(122i)	John throws out the ball to Mary.	9	99
(139a)	I showed Mary herself.	1	37
(139b)	*I showed herself Mary.	1	37
(140a)	I showed every worker her paycheck.	1	74
(140b)	*I showed its owner every paycheck.	0	42
(143)	Sheila gave the toys to the children.	3	64
(144)	Sheila gave the children the toys.	1	45
(145)	Sheila donated the toys to the children.	3	61
(146)	*Sheila donated the children the toys.	0	43
(148a)	John likes to sleep.	1	51
(148b)	John likes to be heard.	1	66
(148c)	John wants to be given a book.	2	90
(149a)	John let her sleep.	4	116
(149b)	John let her be heard.	4	133
(149c)	John let her be given a book.	8	204
(150a)	Sleep!	1	18
(150b)	Be heard!	1	32
(150c)	Be given a book!	2	42
(152a)	John expects to meet Mary.	1	60
(152b)	John seems to smile.	2	57
(153a)	*There expects to be a problem with the computer.	0	75
(153b)	There seems to be a problem with the computer.	3	106
(154)	Mary expects John to smile.	3	68
(155a)	John continued the work.	1	35
(155b)	John continued to work.	2	52
(155c)	The work continued.	1	29
(155d)	It continued to rain.	1	46
(156a)	John promised to work hard.	3	76
(156b)	John promised her to work hard.	8	157
(156c)	John promised a lot of things.	2	64
(156d)	John promised her a lot of things.	2	87
(157a)	John expected to work hard.	2	74
(157b)	John expected her to work hard.	8	123

(157c)	John expected a lot of things.	1	52
(157d)	*John expected her a lot of things.	0	63
(187)	John often eats ice cream.	1	45
(188)	John has often eaten ice cream.	1	54
(189a)	Has John eaten ice cream?	1	38
(189b)	*Eats John ice cream?	0	28
(189c)	Does John eat ice cream?	1	39
(193a)	In the forest John walks.	2	46
(193b)	*In the forest walks John.	0	45
(193c)	In the forest has John walked.	2	58
Mean		1.81	47.97

Appendix D

Basic Relation Representations (BRRs) of example trees

This appendix shows BRRs (based on RMRSs (Copestake, 2003)) produced by the syntactic analyses displayed in the thesis (see Section 3.4.2). They are all derived automatically with the help of the LKB system (Copestake, 2002). I have used the Norsyg grammar (Appendix A) and the English and the German demo grammars (Appendix B) to do the analyses.

D.1 BRRs of example trees in Chapter 4

$$\left[\begin{array}{l}
 mrs \\
 LTOP \quad \boxed{h1} \ h \\
 INDEX \quad \boxed{e2} \ e \\
 \\
 RELS \quad \left\langle \begin{array}{l}
 \left[\begin{array}{l} john \\ LBL \quad \boxed{h3} \ h \\ ARG0 \quad \boxed{x4} \ x \end{array} \right], \left[\begin{array}{l} arg1_rel \\ LBL \quad \boxed{h1} \\ ARG0 \quad \boxed{x4} \end{array} \right], \left[\begin{array}{l} _smash_v_rel \\ LBL \quad \boxed{h1} \\ ARG0 \quad \boxed{e2} \end{array} \right], \left[\begin{array}{l} def_rel \\ LBL \quad \boxed{h5} \ h \\ ARG0 \quad \boxed{x6} \ x \end{array} \right], \\
 \left[\begin{array}{l} _ball_n_rel \\ LBL \quad \boxed{h7} \ h \\ ARG0 \quad \boxed{x6} \end{array} \right], \left[\begin{array}{l} arg2_rel \\ LBL \quad \boxed{h1} \\ ARG0 \quad \boxed{x6} \end{array} \right]
 \end{array} \right\rangle
 \end{array} \right]$$

Figure D.1: BRR of *John smashed the ball* (Tree: 5.1, p. 132 and 4.10, p. 99)

D.2 BRRs of example trees in Chapter 5

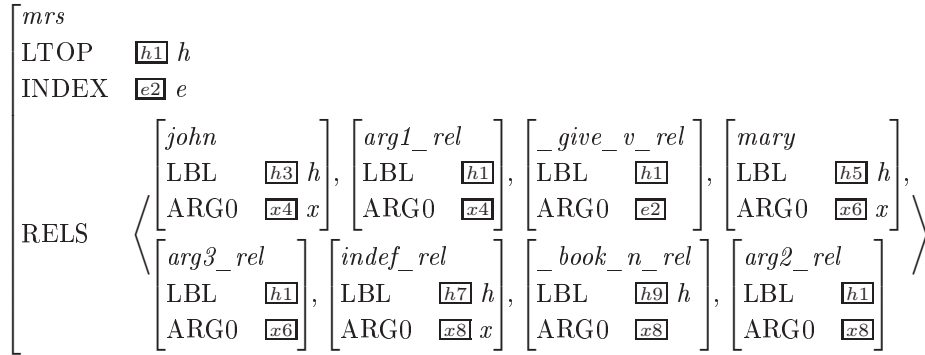


Figure D.2: BRR of *John gave Mary a book* (Tree: 5.2, p. 132 and 5.14, p. 139)

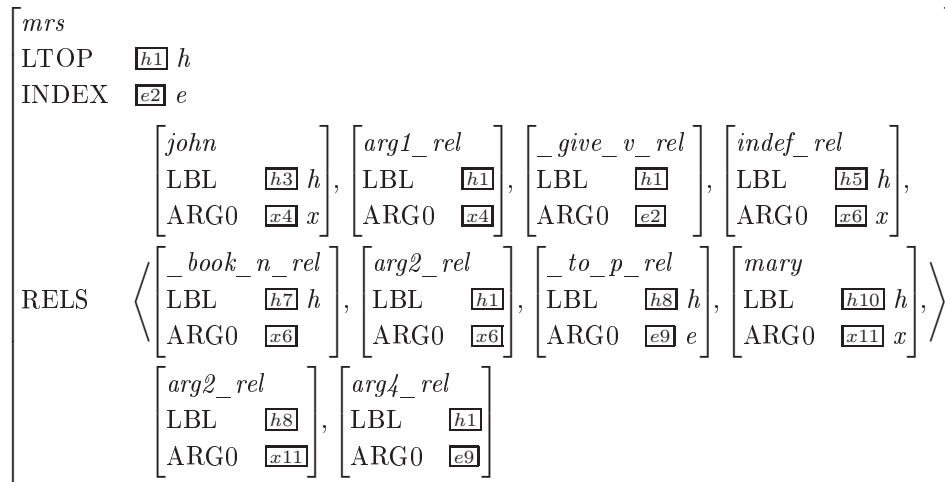


Figure D.3: BRR of *John gave a book to Mary* (Tree: 5.3, p. 133)

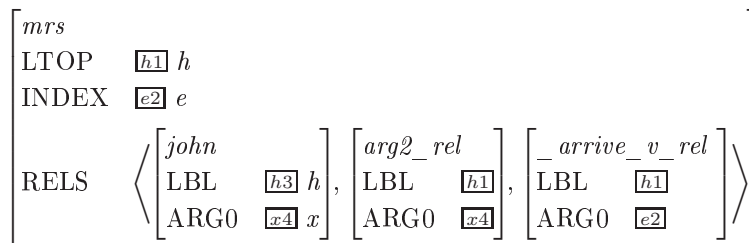


Figure D.4: BRR of *John arrived* (Tree: 5.4, p. 133)

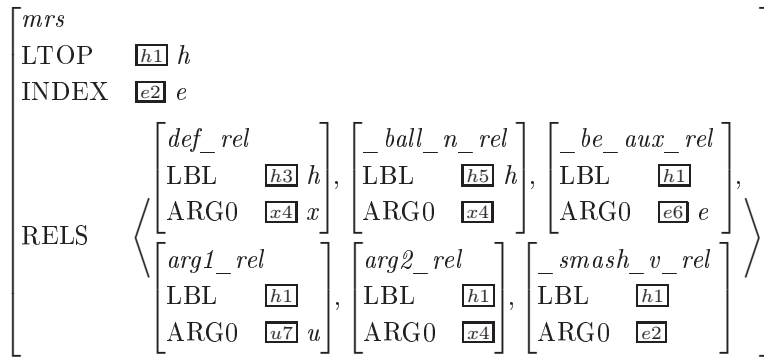


Figure D.5: BRR of *The ball was smashed* (Tree: 5.5, p. 133)

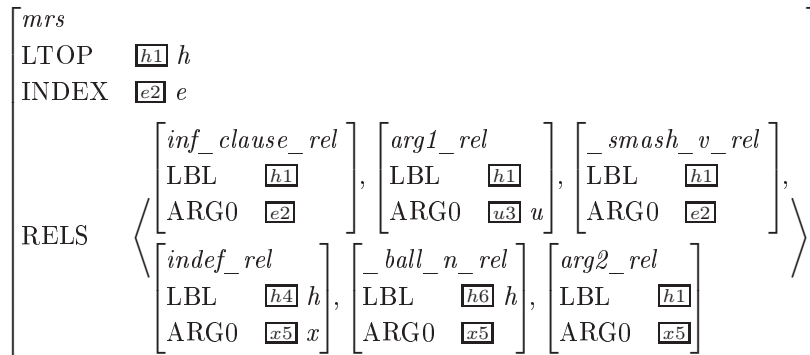


Figure D.6: BRR of *to smash a ball* (Tree: 5.6, p. 134)

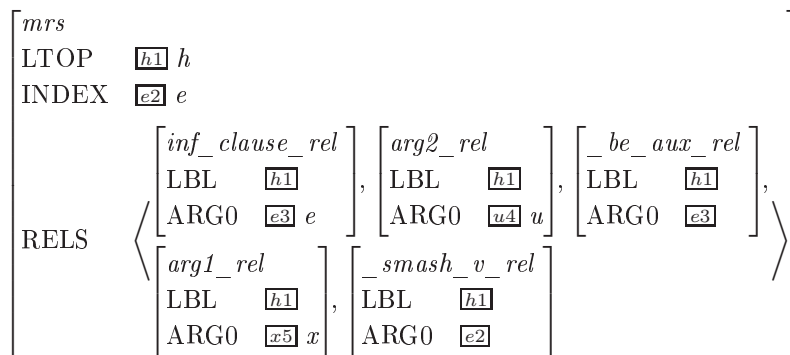
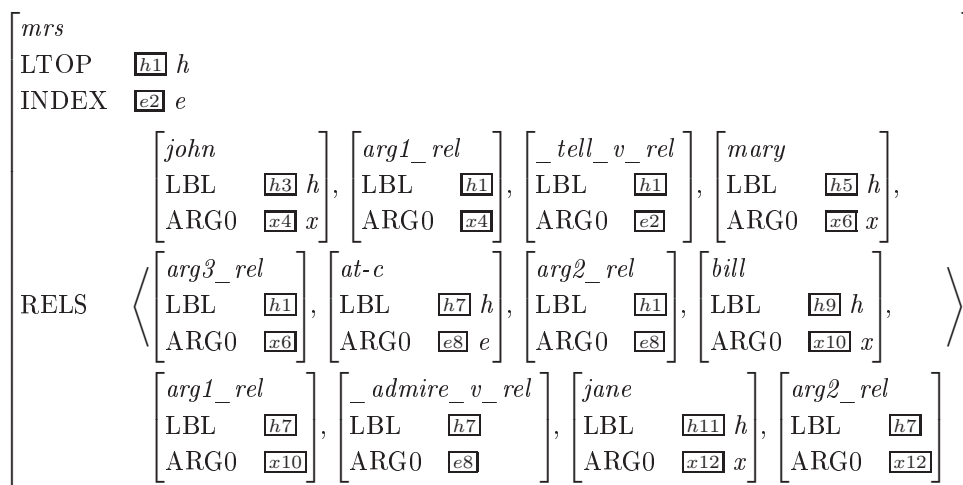


Figure D.7: BRR of *to be smashed* (Tree: 5.7, p. 134)

Figure D.8: BRR of *John told Mary that Bill admires Jane* (Tree: 5.15, p. 140)

D.3 BRRs of example trees in Chapter 6

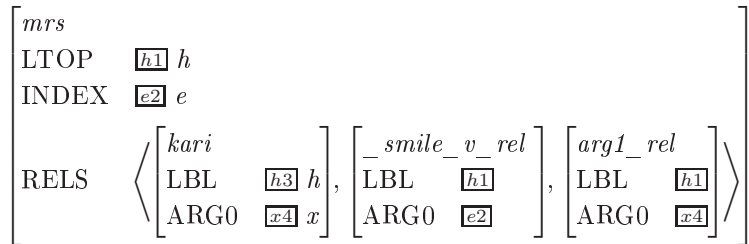


Figure D.9: BRR of intransitive clause (Tree: 6.10, p. 158)

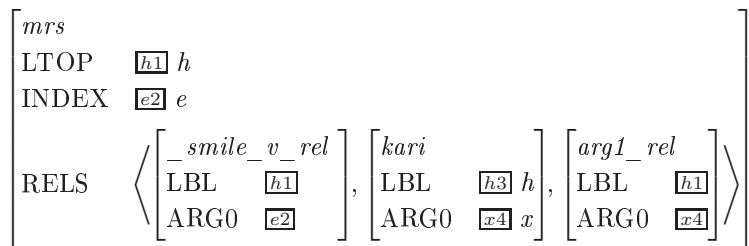


Figure D.10: BRR of yes-no clause (Tree: 6.11, p. 158)

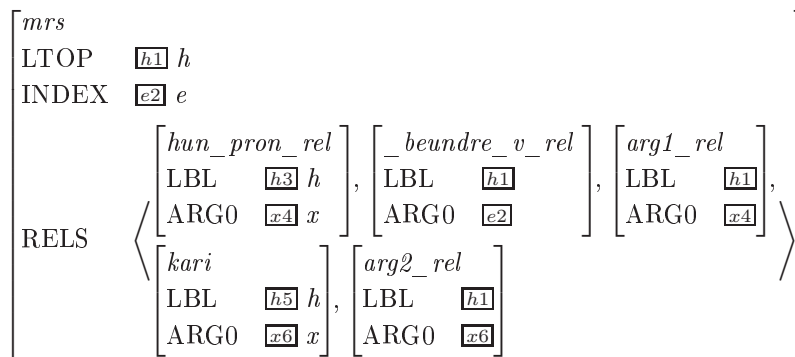


Figure D.11: BRR of transitive sentence (Tree: 6.12, p. 158)

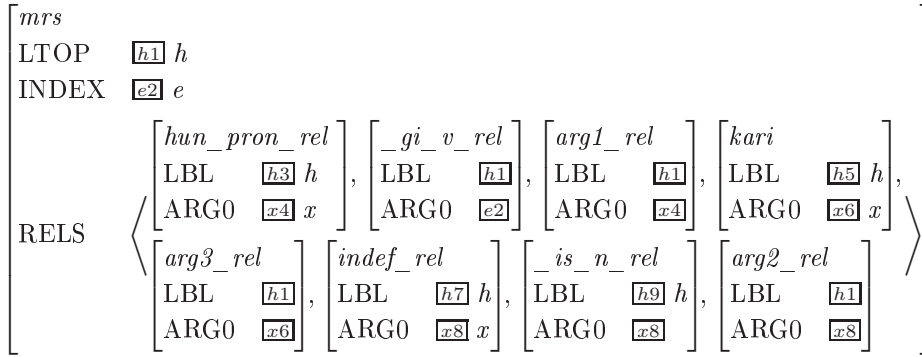


Figure D.12: BRR of ditransitive sentence (Tree: 6.13, p. 158)

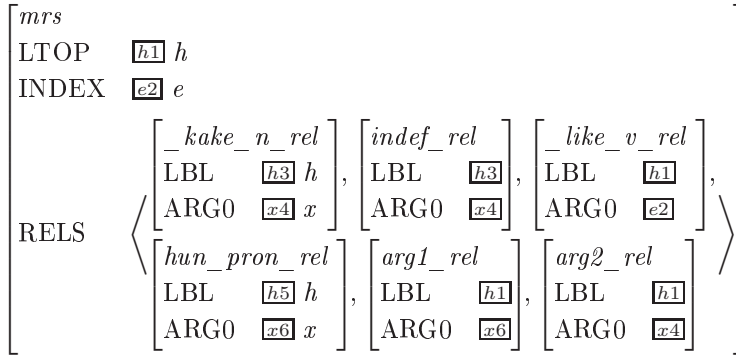
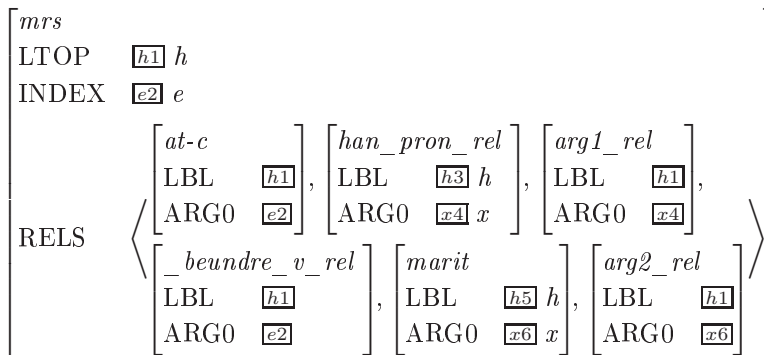
Figure D.13: BRR of transitive main clause with topicalized object with the verb *liker* ('likes') (Tree: 6.14, p. 160)

Figure D.14: BRR of subordinate clause (Tree: 6.23, p. 168)

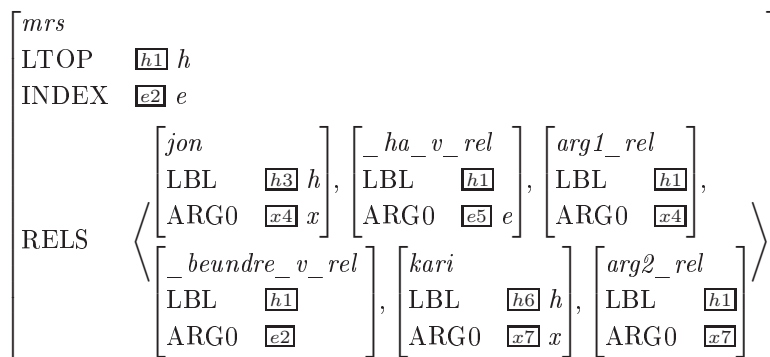


Figure D.15: BRR of sentence with auxiliary (Tree: 6.28, p. 172)

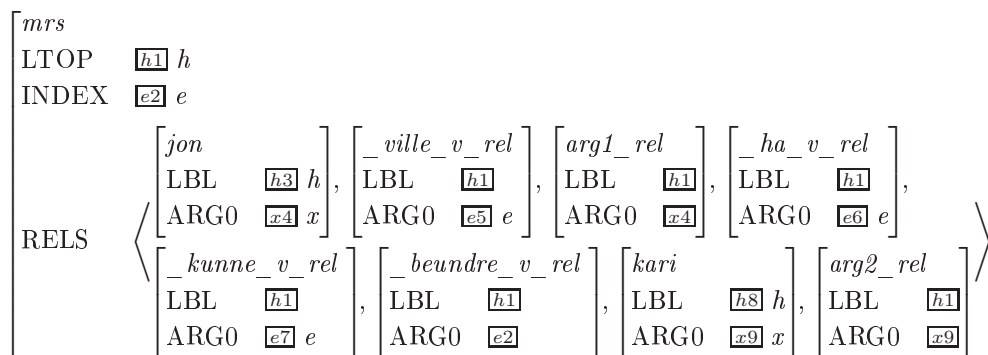


Figure D.16: BRR of sentence with three auxiliaries (Tree: 6.29, p. 172)

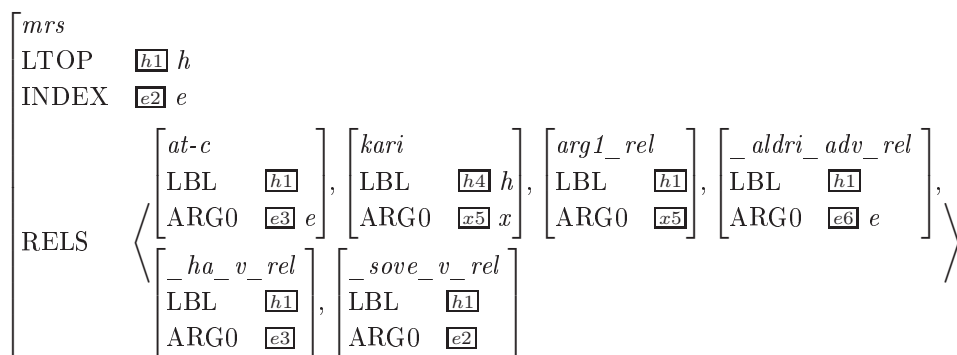


Figure D.17: BRR of subordinate clause (Tree: 6.32, p. 174)

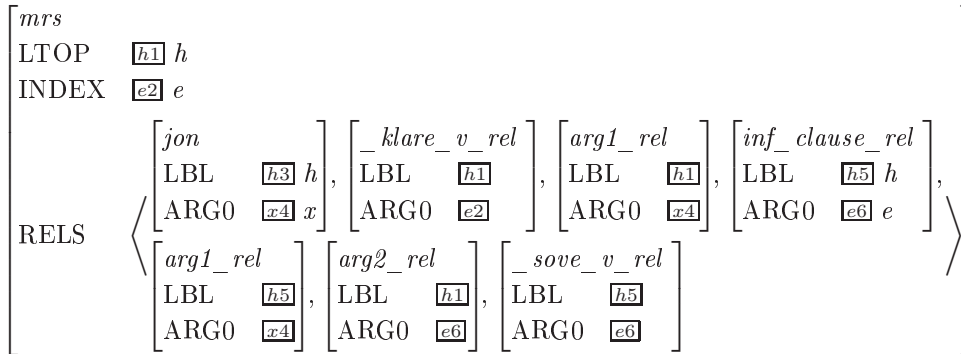


Figure D.18: BRR of sentence with infinitival clause (Tree: 6.33, p. 174)

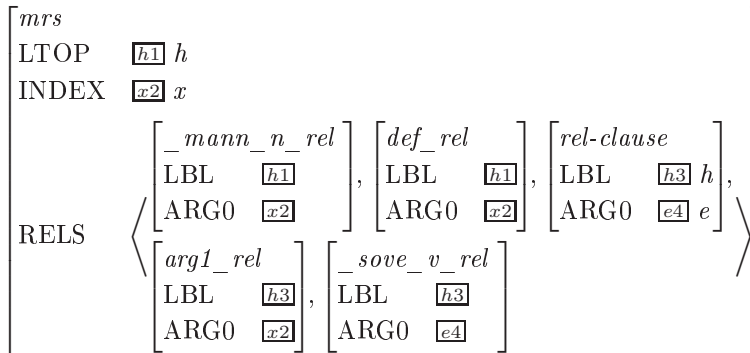
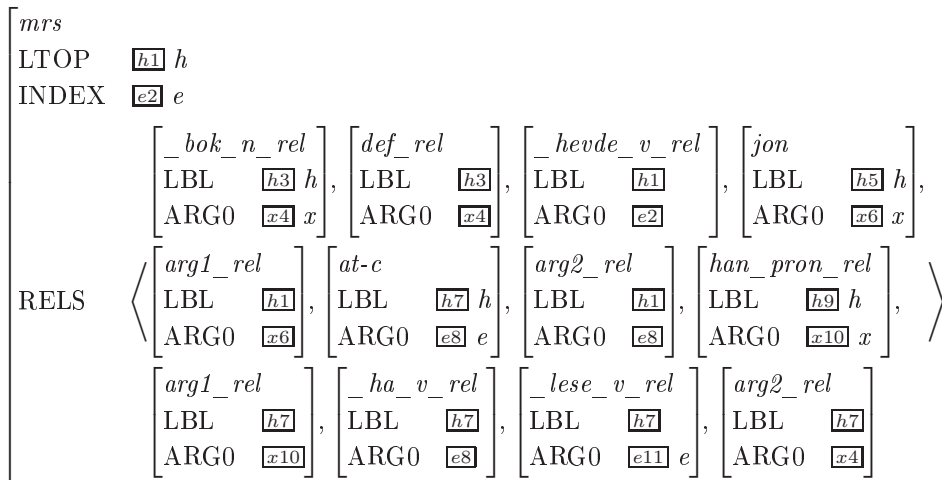


Figure D.19: BRR of NP with relative clause (Tree: 6.34, p. 174)

Figure D.20: BRR of *Boka hevder Jon at han har lest* ('The book, John claims that he has read') (Tree: 6.39, p. 177)

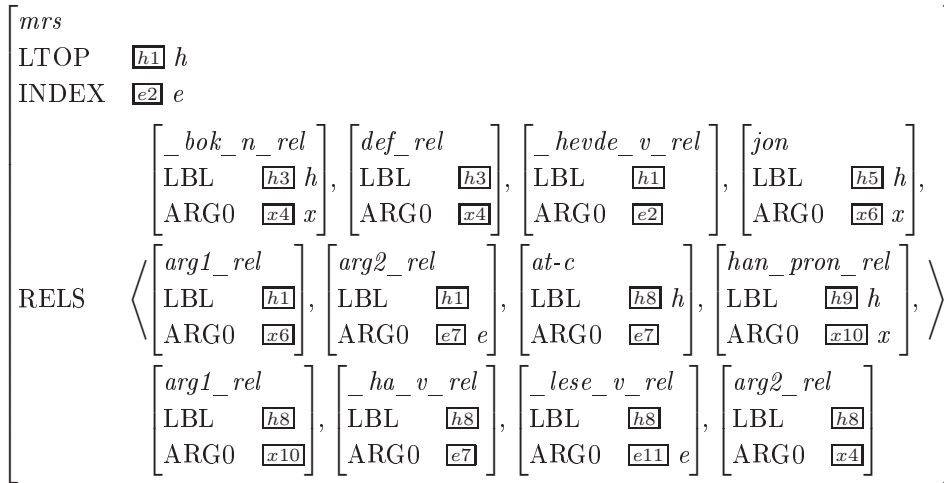


Figure D.21: BRR of *Boka hevder Jon han har lest* ('The book, John claims he has read') (Tree: 6.40, p. 177)

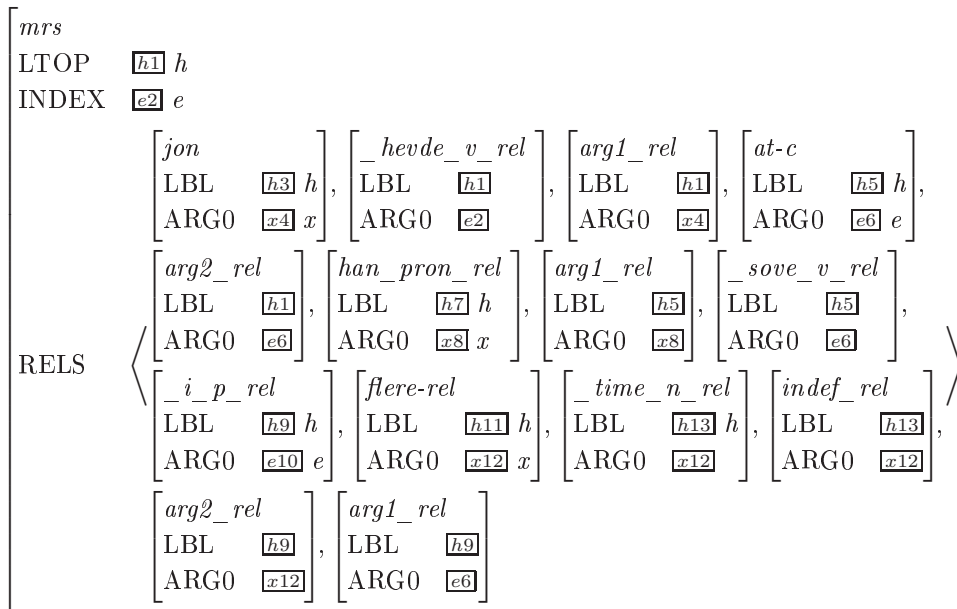


Figure D.22: BRR of *Jon hevdet at han sov i flere timer* ('John claimed that he had slept for several hours'). PP attachment to subordinate clause (Tree: 6.42, p. 179)

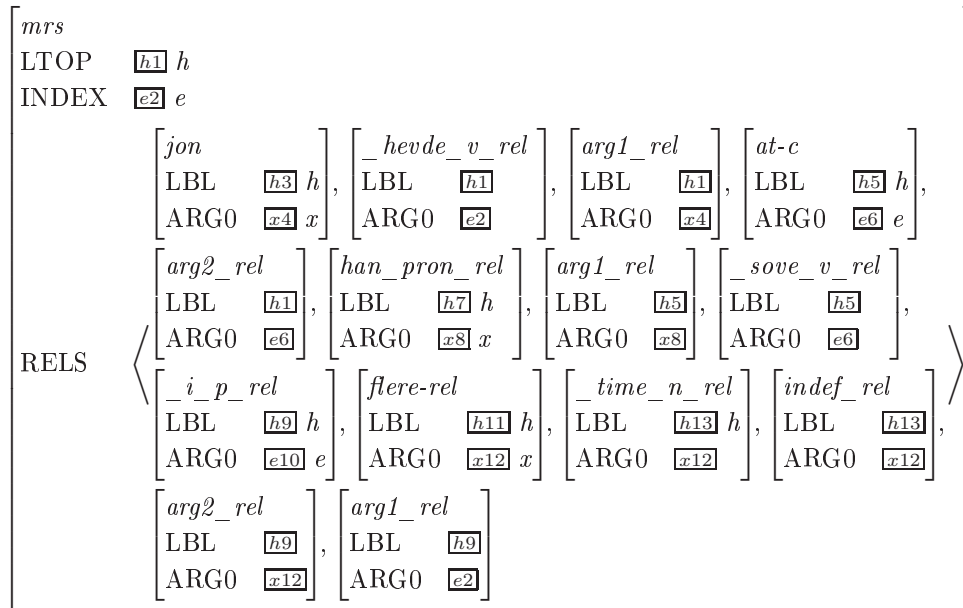


Figure D.23: BRR of *Jon hevdet at han sov i flere timer* ('John claimed that he had slept for several hours'). PP attachment to main clause. (Tree: 6.43, p. 179)

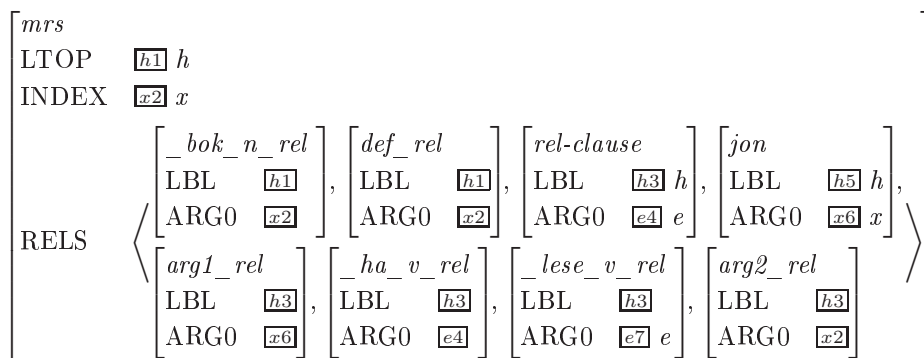


Figure D.24: BRR of *boka som Jon har lest* ('The book that Jon has read') (Tree: 6.45, p. 181)

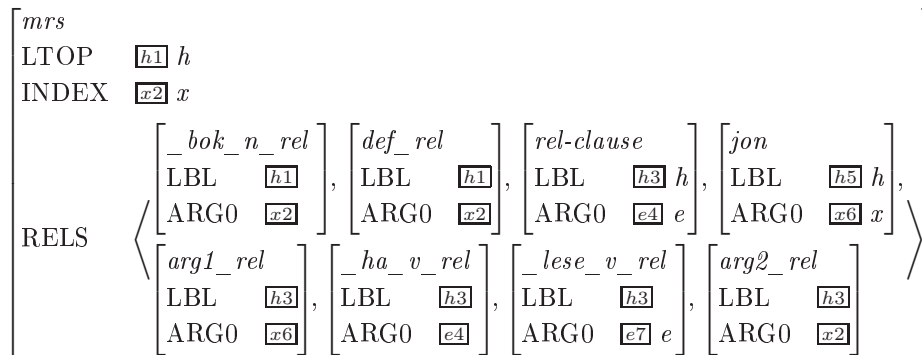


Figure D.25: BRR of *Boka Jon har lest* ('The book John has read') (Tree: 6.46, p. 181)

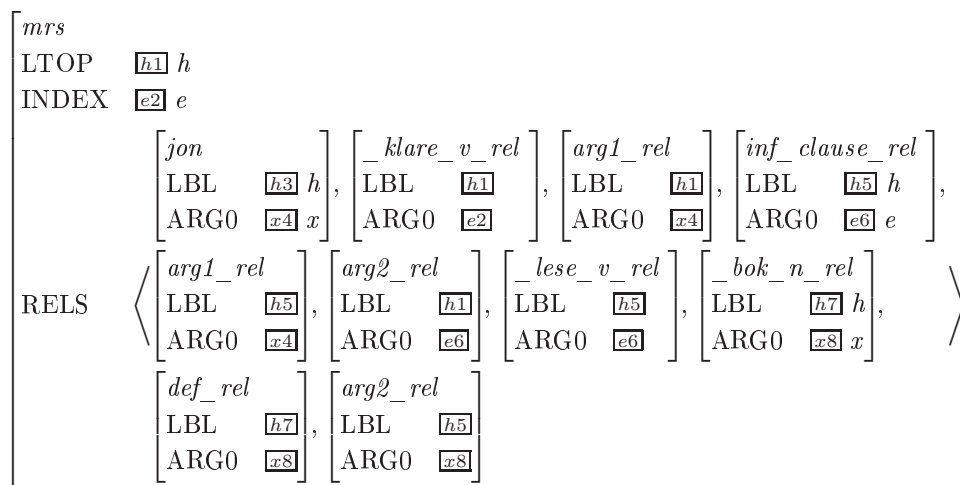


Figure D.26: BRR of *Jon klarer å lese boka* ('Jon manages to read the book') (Tree: 6.51, p. 187)

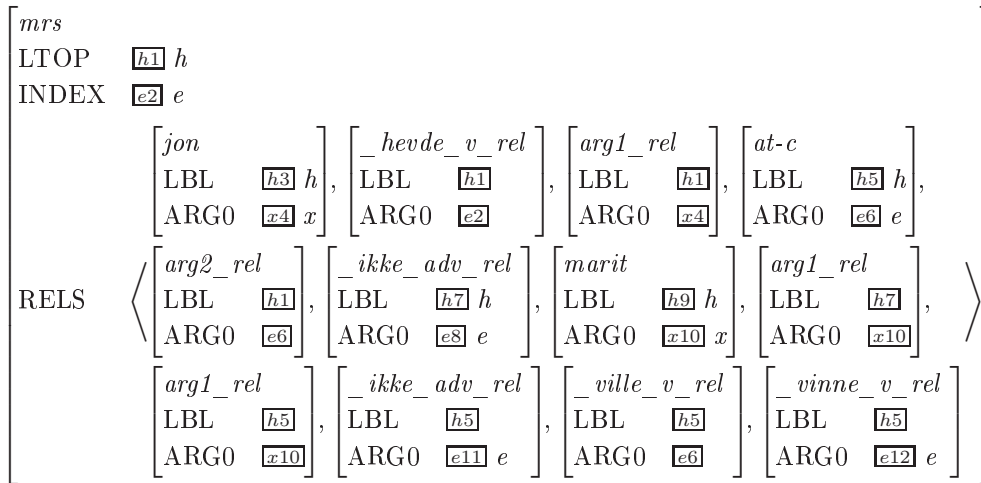


Figure D.27: BRR of subordinate clause with two sentence adverbials (Tree: 6.69, p. 201)

D.4 BRRs of example trees in Chapter 7

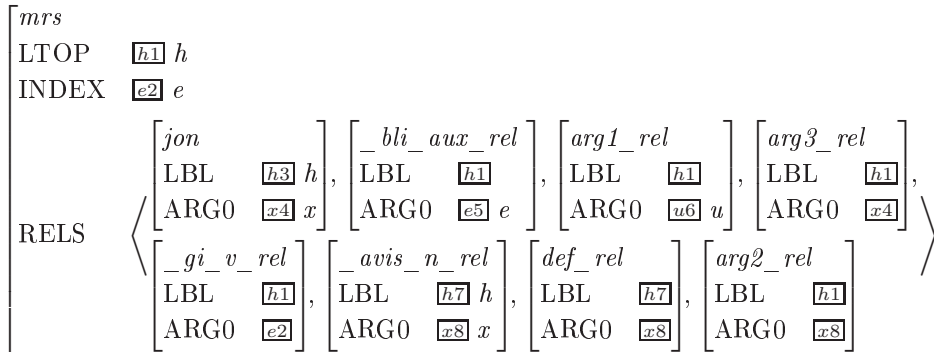


Figure D.28: BRR of passive ditransitive sentence with the auxiliary *bli* (Tree: 7.4, p. 214)

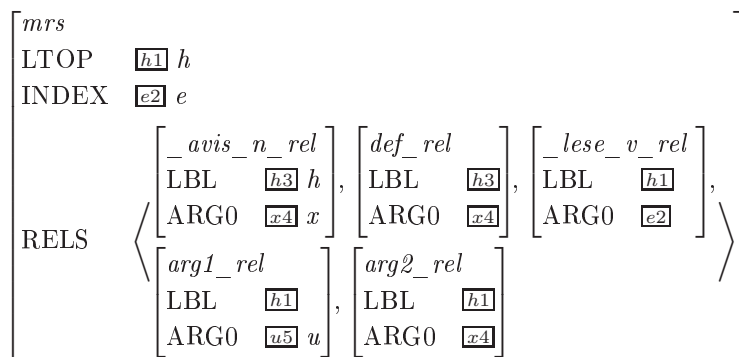


Figure D.29: BRR of passive sentence with morphological passive (Tree: 7.5, p. 215)

D.5 BRRs of example trees in Chapter 9

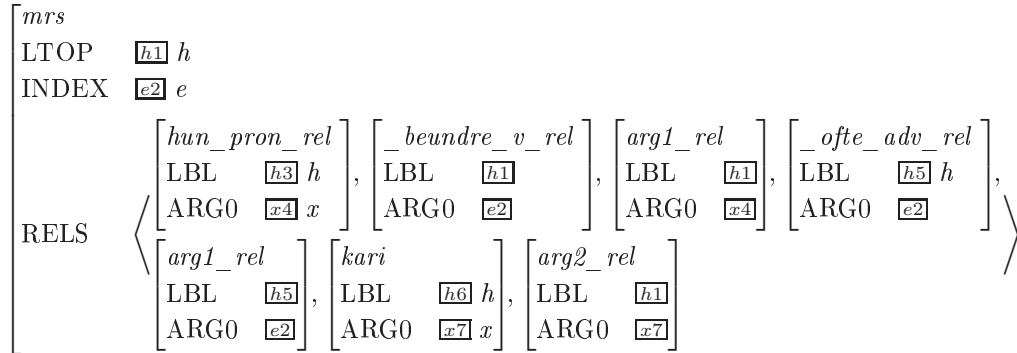


Figure D.30: BRR of transitive main clause (Tree: 9.11, p. 251)

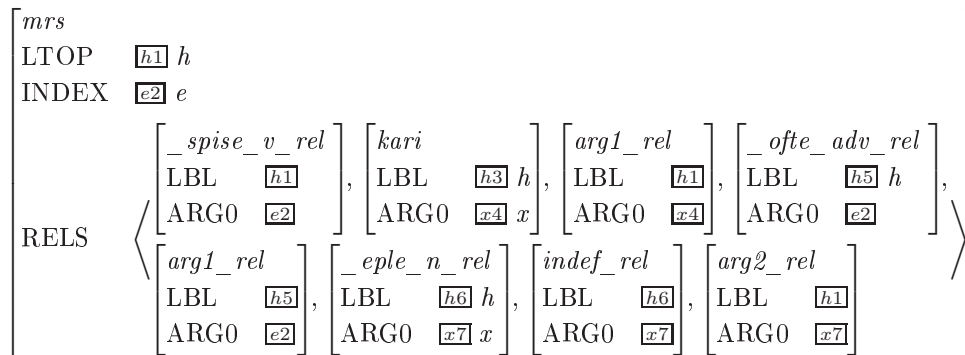


Figure D.31: BRR of Norsyg yes-no clause (Tree: 9.12, p. 252)

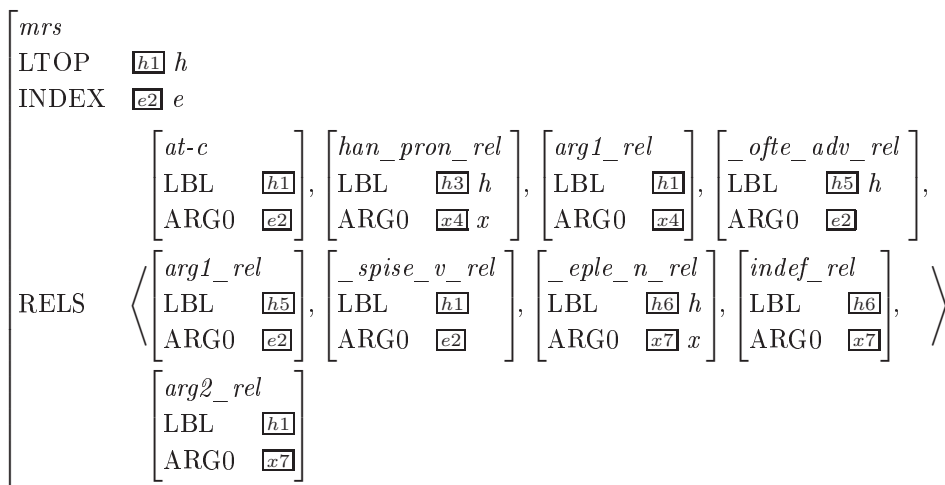


Figure D.32: BRR of alternative representation of subordinate clause with sentence adverbial (Tree: 9.14, p. 253)

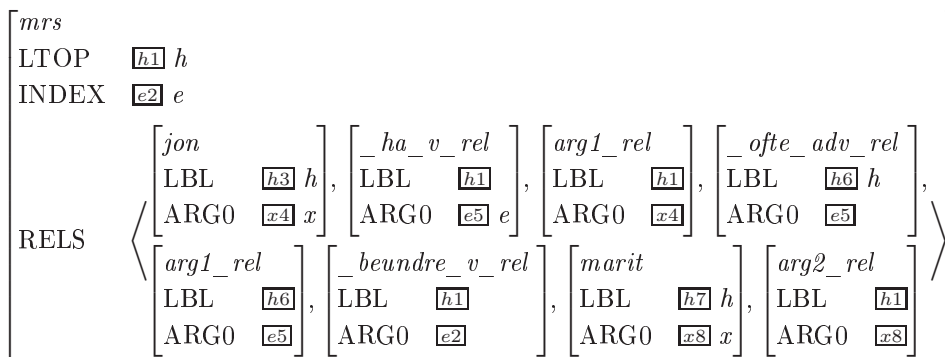


Figure D.33: BRR of alternative representation of main clause with auxiliary (Tree: 9.15, p. 254)

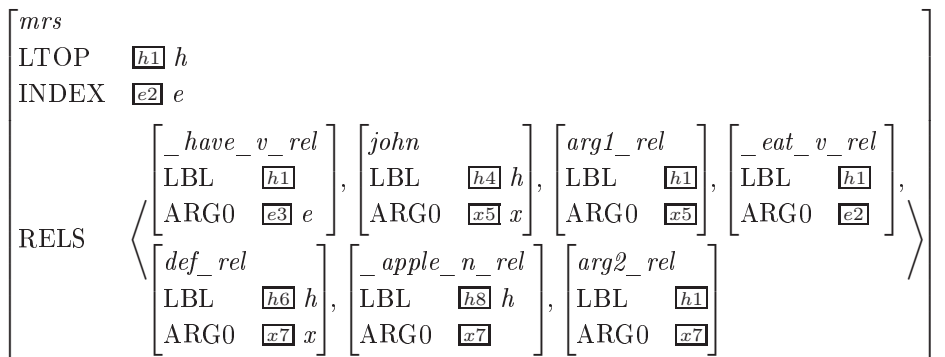
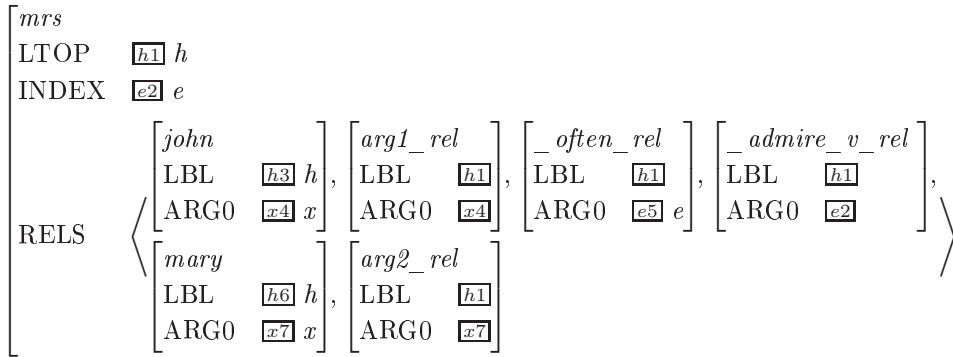
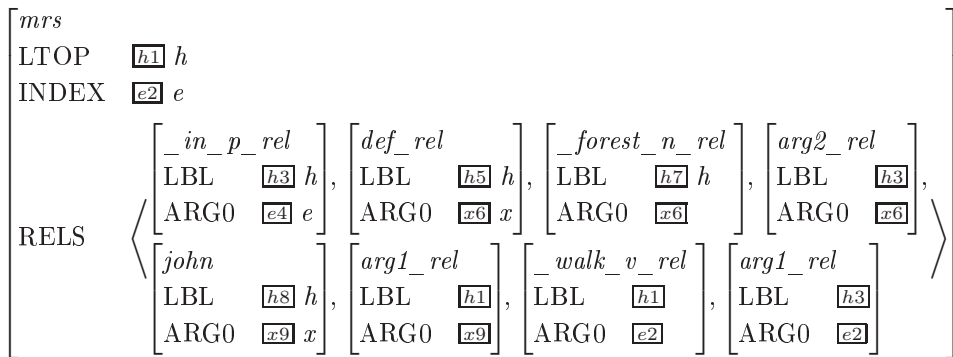
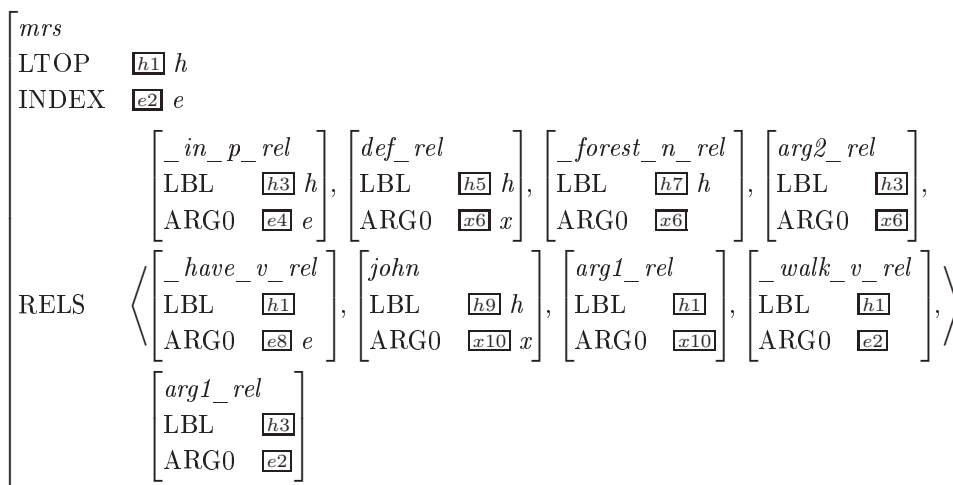


Figure D.34: BRR of new analysis of *Has John eaten the apple?* (Tree: 9.17, p. 255)

Figure D.35: BRR of new analysis of *John often admires Mary* (Tree: 9.20, p. 257)Figure D.36: BRR of new analysis of *In the forest John walks* (Tree: 9.22, p. 258)Figure D.37: BRR of *In the forest has John walked* (Tree: 9.24, p. 259)

D.6 BRRs of example trees in Chapter 10

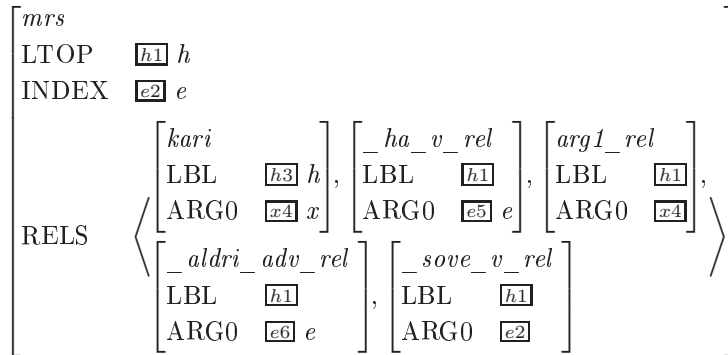


Figure D.38: BRR of main clause with auxiliary and sentence adverbial (Tree: 10.7, p. 276)

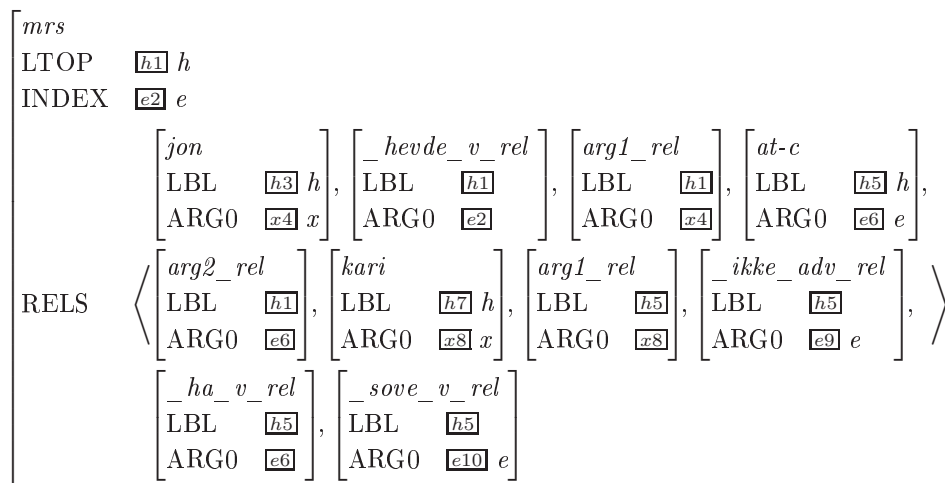


Figure D.39: BRR of subordinate clause with auxiliary and sentence adverbial (Tree: 10.9, p. 276)

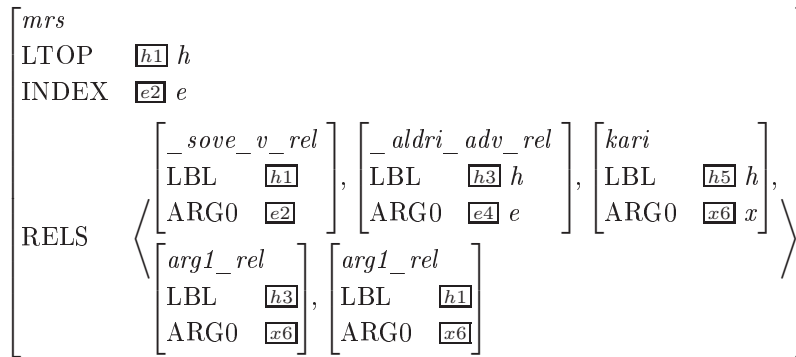


Figure D.40: BRR of yes-no question with sentence adverbial (Tree: 10.11, p. 277)

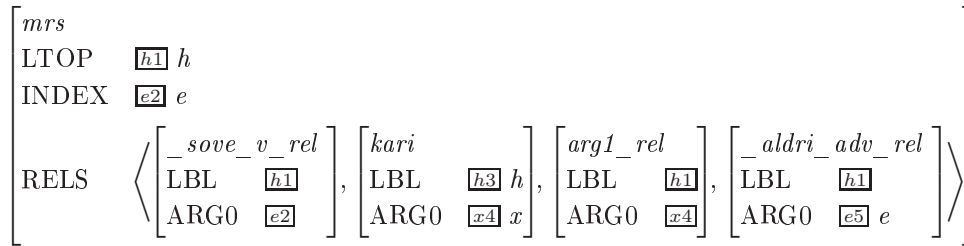


Figure D.41: BRR of yes-no question with sentence adverbial (Tree: 10.13, p. 277)

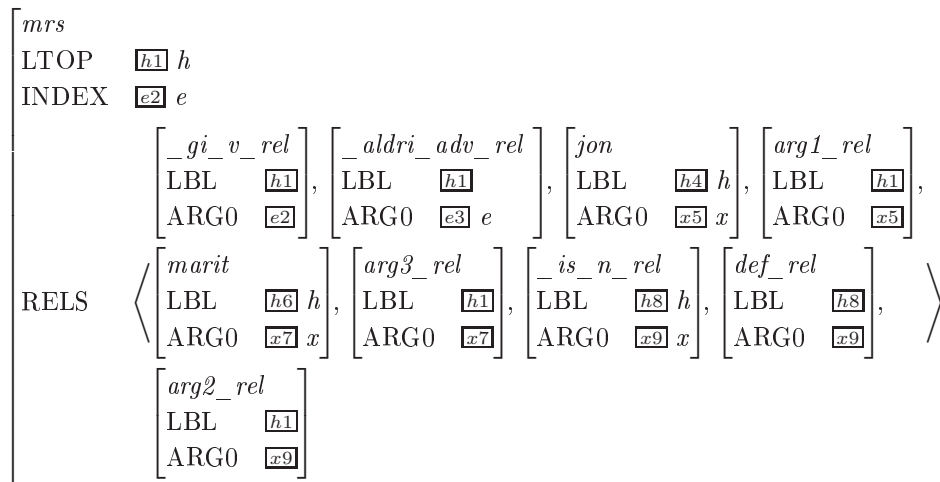


Figure D.42: BRR of ditransitive yes-no question with no pronouns (Tree: 10.15, p. 278)

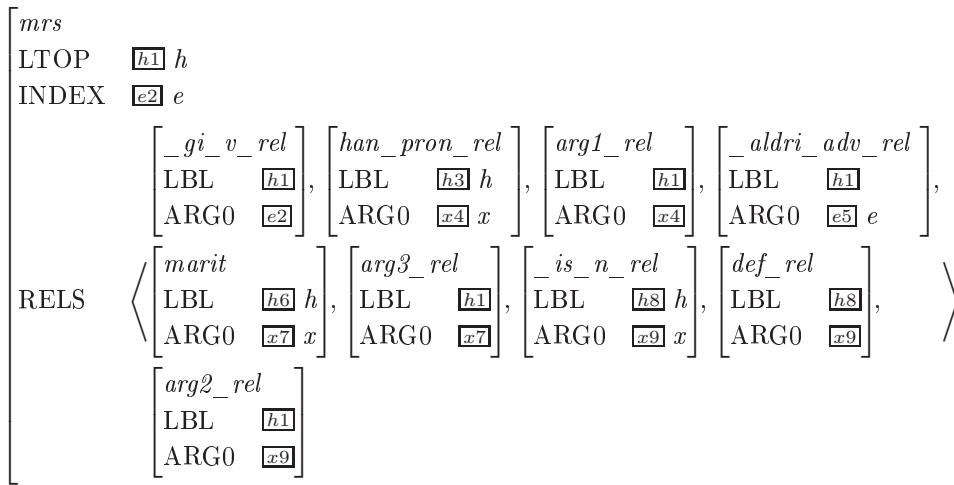


Figure D.43: BRR of ditransitive yes-no question with one pronoun (Tree: 10.16, p. 278)

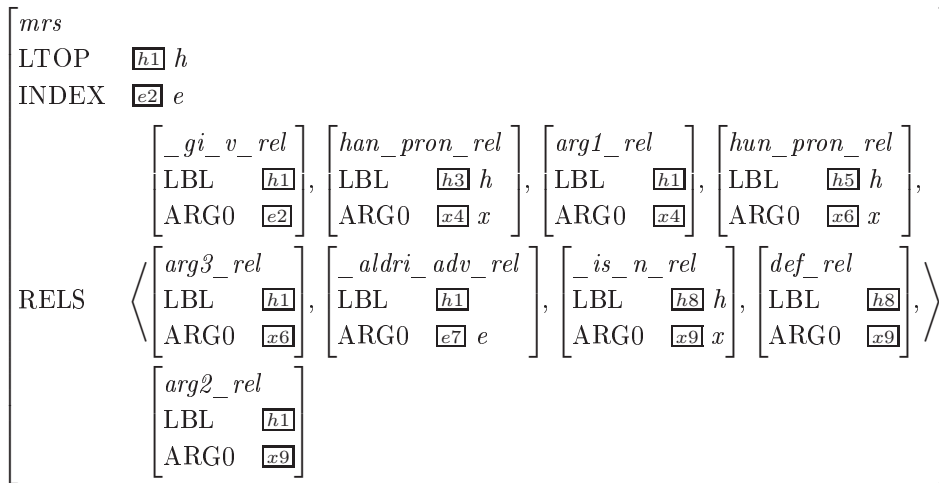


Figure D.44: BRR of ditransitive yes-no question with two pronouns (Tree: 10.17, p. 278)

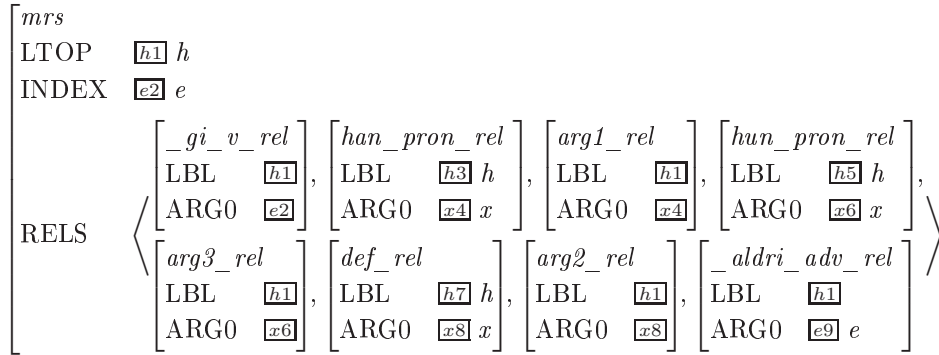


Figure D.45: BRR of ditransitive yes-no question with three pronouns (Tree: 10.18, p. 278)

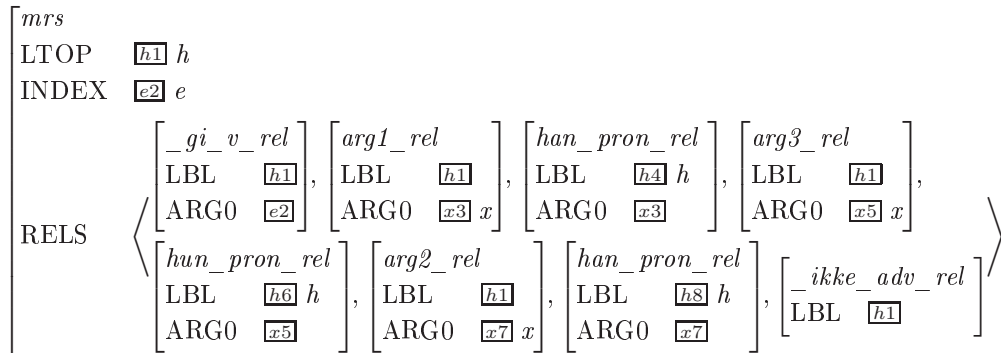


Figure D.46: BRR of (Tree: 10.19, p. 279)

D.7 BRRs of example trees in Appendix B

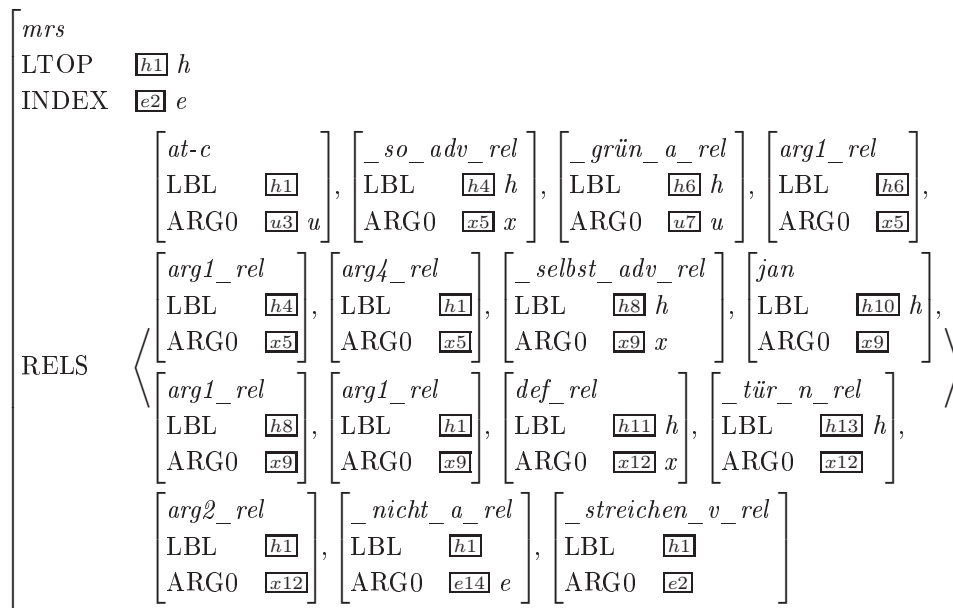


Figure D.47: BRR of *daß so grün selbst Jan die Tür nicht streicht* ('that not even Jan would paint the door that green') (Tree: B.1, p. 309) and *daß so grün die Tür selbst Jan nicht streicht* ('that not even Jan would paint the door that green') (Tree: B.2, p. 309)

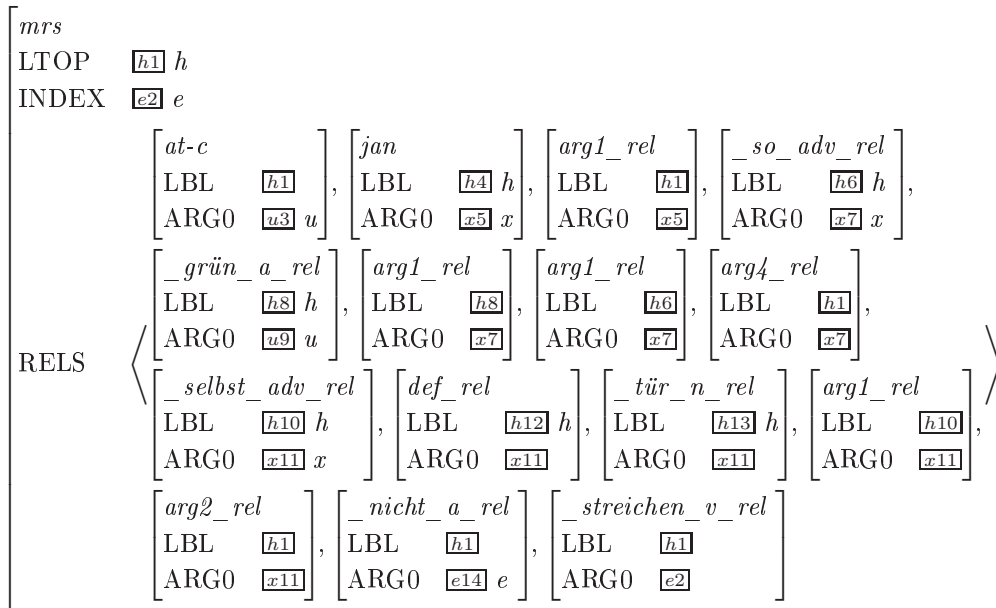


Figure D.48: BRR of *daß Jan so grün selbst die Tür nicht streicht* ('that not even Jan would paint the door that green') (Tree: B.3, p. 309)

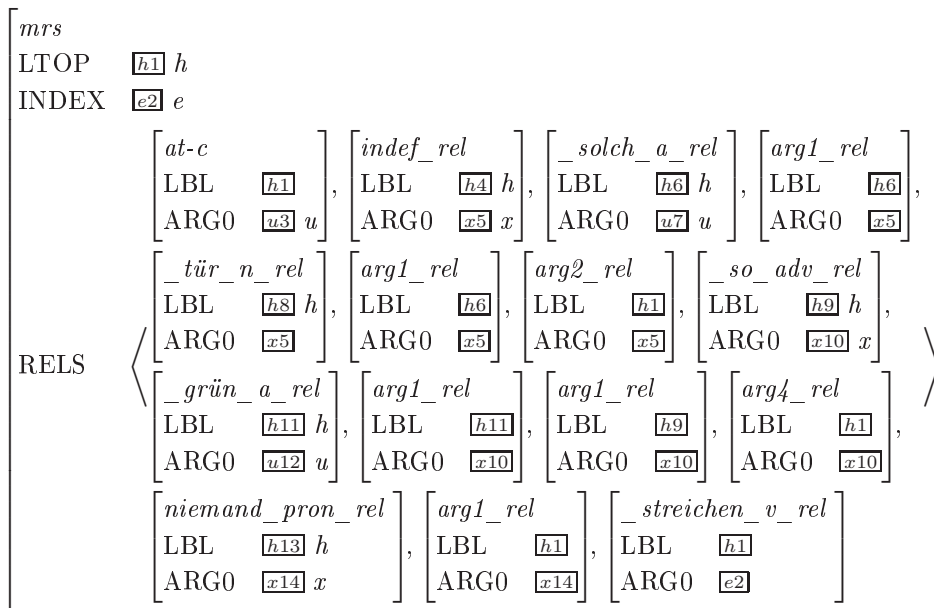


Figure D.49: BRR of *daß eine solche Tür so grün niemand streicht* ('that nobody paints such a door that green') (Tree: B.4, p. 309)

Bibliography

- Abney, S. P. (1989). A Computational Model of Human Parsing. *Journal of Psycholinguistic Research*, **18**, 129–144.
- Baker, M. C. (1988). *Incorporation. A Theory of Grammatical Function Changing*. The University of Chicago Press.
- Bangalore, S. and Joshi, A. K. (1999). Supertagging: An Approach to Almost Parsing. *Computational Linguistics*, **25**(2), 237–265.
- Barg, P. and Walther, M. (1998). Processing unknown words in HPSG. In *Proceedings of COLING-ACL '98*, pages 91–95, Morristown, NJ, USA.
- Beermann, D. and Hellan, L. (2004). Semantic Decomposition in a Computational HPSG Grammar: A Treatment of Aspect and Context-Dependent Directionals. In S. Müller, editor, *Proceedings of the HPSG-2004 Conference, Center for Computational Linguistics, Katholieke Universiteit Leuven*, pages 357–377. CSLI Publications, Stanford.
- Bender, E. M., Flickinger, D. P., and Oepen, S. (2002). The Grammar Matrix: An open-source starter-kit for the rapid development of cross-linguistically consistent broad-coverage precision grammars. In J. Carroll, N. Oostdijk, and R. Sutcliffe, editors, *Proceedings of the Workshop on Grammar Engineering and Evaluation at the 19th International Conference on Computational Linguistics*, pages 8–14, Taipei, Taiwan.
- Blake, B. (1990). *Relational Grammar*. London and New York: Routledge.
- Boguraev, B. and Briscoe, T. (1989). Utilizing the LDOCE grammar codes. In B. Boguraev and T. Briscoe, editors, *Computational Lexicography for Natural Language Processing*. London and New York: LONGMAN.

- Borer, H. (2005a). *Structuring Sense. An Exo-Skeletal Triology. Volume I. In Name Only*. Oxford University Press.
- Borer, H. (2005b). *Structuring Sense. An Exo-Skeletal Triology. Volume II. The Normal Course of Events*. Oxford University Press.
- Borsley, R. D. (1996). *Modern Phrase Structure Grammar*. Number 11 in Blackwell textbooks in linguistics. Blackwell Publishers.
- Borsley, R. D. (1999). *Syntactic Theory. A Unified Approach. Second Edition*. London: Arnold.
- Borthen, K. (2003). *Norwegian Bare Singulars*. Ph.D. thesis, NTNU, Norwegian University of Science and Technology.
- Borthen, K. and Haugereid, P. (2005). A grammar component for referential properties of nominals. *Readings in Language and Computation, Special Issue on Shared Representation in Multilingual Grammar Engineering*, **3**, 221–246.
- Bouma, G., Malouf, R., and Sag, I. A. (2001). Satisfying constraints on extraction and adjunction. *Natural Language and Linguistic Theory*, **1**(19), 1–65.
- Bresnan, J. (2001). *Lexical-Functional Syntax*. Blackwell Publishers.
- Briscoe, T. (2006). An introduction to tag sequence grammars and the RASP system parser. Technical report, University of Cambridge, Computer Laboratory.
- Briscoe, T., Carroll, J., and Watson, R. (2006). The second release of the RASP system. In *Proceedings of the COLING/ACL on Interactive presentation sessions*, pages 77–80, Morristown, NJ, USA. Association for Computational Linguistics.
- Carnie, A. (2007). *Syntax. A Generative Introduction. Second edition*. Blackwell Publishing.
- Carpenter, B. (1992). *The Logic of Typed Feature Structures with Applications to Unification-based Grammars, Logic Programming and Constraint Resolution*, volume 32 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, New York.

- Carroll, J. and Briscoe, T. (2001). High precision extraction of grammatical relations. In *IWPT*.
- Chomsky, N. (1957). *Syntactic Structures*. The Hague and Paris: Mouton.
- Chomsky, N. (1981). *Lectures on Government and Binding. The Pisa Lectures*. Dordrecht, Holland and Cinnaminson, USA: Foris Publications.
- Chomsky, N. (1986). *Barriers*. Cambridge, MA and London: The MIT Press, Linguistic Inquiry Monographs, Volume 13.
- Copestake, A. (2002). *Implementing Typed Feature Structure Grammars*. CSLI publications.
- Copestake, A. (2003). Report on the design of RMRS (preliminary version). Technical report, University of Cambridge, Computer Laboratory.
- Copestake, A. and Flickinger, D. P. (2000). An open-source grammar development environment and broad-coverage English grammar using HPSG. In *Proceedings of the Second Linguistic Resources and Evaluation Conference*, pages 591–600, Athens, Greece.
- Copestake, A., Flickinger, D. P., Pollard, C. J., and Sag, I. A. (2005). Minimal Recursion Semantics: an introduction. *Research on Language and Computation*, **3**(4), 281–332.
- Creider, C. A. and Åfarli, T. A. (1987). Non subject pro-drop in Norwegian. *Linguistic Inquiry*, **18**, 339–345.
- Crocker, M. W. (1996). Mechanisms for sentence processing.
- Croft, W. (1991). *Syntactic Categories and Grammatical Relations*. Chicago.
- Crouch, D., Dalrymple, M., Kaplan, R., King, T. H., Maxwell, J., and Newman, P. (2007). *XLE Documentation*.
- Crysmann, B. (2003). On the efficient implementation of German verb placement in HPSG. In *Proceedings of RANLP 2003*, pages 112–116, Borovets, Bulgaria.
- Culicover, P. W. (1997). *Principles and Parameters. An Introduction to Syntactic Theory*. Oxford University Press.

- Dalrymple, M. (2001). *Lexical Functional Grammar*. New York: Academic Press. Syntax and Semantics, Volume 34.
- Davis, A. R. (2001). *Linking by Types in the Hierarchical Lexicon*. CSLI Publications.
- Diderichsen, P. (1946). *Elementær Dansk Grammatik*. København: Gyldendal.
- Ellingsen, L. (2003). *Norwegian Word Order in Head-Driven Phrase Structure Grammar. Phenomena, Analysis, and Implementation*. Master's thesis, Institutt for Lingvistiske Fag, Universitetet i Oslo.
- Engdahl, E. (2001). Scandinavian passives in HPSG. In A. Holmer, J.-O. Svantesson, and Å. Viberg, editors, *Proceedings of the 18th Scandinavian Conference of Linguistics*, volume 39:2, pages 23–36, Travaux de l'institut de linguistique de Lund. Lund: Universitetsstryckeriet.
- Engdahl, E. (2006). Semantic and syntactic patterns in Swedish passives. *Demoting the Agent. Passive, middle and other voice phenomena*, pages 21–45.
- Erbach, G. (1990). Syntactic processing of unknown words. In P. Jorrand and V. Sgurev, editors, *Artificial Intelligence IV - Methodology, Systems, Applications (AIMSA)*, pages 371–381, North-Holland, Amsterdam.
- Faarlund, J. T., Lie, S., and Vannebo, K. I. (1997). *Norsk Referansegrammatikk*. Oslo: Universitetsforlaget.
- Ferreira, V. S. (1996). Is it better to give than to donate? syntactic flexibility in language production. *Journal of Memory and Language*, **35**, 724–755.
- Fillmore, C. J. (1968). The case for case. In E. Bach and R. T. Harms, editors, *Universals in Linguistic Theory*, pages 1–88. Holt, Rinehart and Winston, Inc., Cambridge, MA and London.
- Fillmore, C. J., Kay, P., and O'Connor, M. C. (1988). Regularity and idiomaticity in grammatical constructions. *Language*, **64/3**, 501–538.
- Flickinger, D. P. (1987). *Lexical Rules in the Hierarchical Lexicon*. Ph.D. thesis, Stanford University.

- Flickinger, D. P. (2000). On building a more efficient grammar by exploiting types. *Natural Language Engineering*, **6**(1), 15–28.
- Flickinger, D. P., Bender, E., and Oepen, S. (2003). MRS in the LINGO Grammar Matrix: A practical user's guide.
- Fløgstad, K. (1977). *Dalen Portland*. Oslo: Samlaget.
- Fouvry, F. (2003). Lexicon acquisition with a large-coverage unification-based grammar. In *EACL '03: Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics*, pages 87–90, Morristown, NJ, USA. Association for Computational Linguistics.
- Fretheim, T. and Halvorsen, P.-K. (1975). Norwegian cliticization. In K.-H. Dalhstedt, editor, *The Nordic Languages and Modern Linguistics*, volume 2, pages 446–465. Almqvist & Wiksell, Stockholm.
- Goldberg, A. (1995). *Constructions: A Construction Grammar Approach to Argument Structure*. The University of Chicago Press.
- Goldberg, A. (2006). *Constructions at Work. The Nature of Generalization in Language*. Oxford University Press.
- Goldberg, A. and Jackendoff, R. (2004). The English resultative as a family of constructions. *Language*, **80**, 532–568.
- Gundel, J. K., Hedberg, N., and Zacharski, R. (1993). Cognitive status and the form of referring expressions in discourse. *Language*, **69**, 274–307.
- Hale, K. and Keyser, S. J. (1993). On argument structure and the lexical expression of syntactic relations. In K. Hale and S. J. Keyser, editors, *View from Building 20, Essays in Linguistics in Honour of Sylvain Bromberger*, volume 24 of *Current Studies in Linguistics*, pages 53–109. MIT Press, Cambridge, MA and London.
- Hale, K. and Keyser, S. J. (2002). *Prolegomenon to a Theory of Argument Structure*. Cambridge, MA and London: The MIT Press, Linguistic Inquiry Monographs, Volume 39.
- Hellan, L. (1971). *Seg, selv og syntaks*. Magisteravhandling, University of Oslo.

- Hellan, L. (1988). *Anaphora in Norwegian and the Theory of Grammar*, volume 32 of *Studies in Generative Grammar*. Dordrecht, The Netherlands: Foris Publications.
- Hellan, L. (1991). The phrasal nature of the double object clusters. In A. Werner, E. J. Reuland, and W. Kosmeijer, editors, *Issues in Germanic syntax*. Berlin: de Gruyter.
- Hellan, L. (2002). NorSource types, correlated with NKL types. Technical report, Dept. of Language and Communication Studies, NTNU.
- Hellan, L. (2005). Implementing Norwegian reflexives in an HPSG grammar. In S. Müller, editor, *The Proceedings of the 12th International Conference on Head-Driven Phrase Structure Grammar, Department of Informatics, University of Lisbon*, pages 519–539, Stanford. CSLI Publications.
- Hellan, L. and Beermann, D. (2005). The linguistic dimensions of prepositions and their use in computational linguistics formalisms and applications. In V. Kordoni and A. Vilavicencio, editors, *Classification of Prepositional Senses for Deep Grammar Applications*. ACL Sigsem.
- Hellan, L. and Haugereid, P. (2004). NorSource - an exercise in the Matrix Grammar building design. In E. Bender, D. P. Flickinger, F. Fouvry, and M. Siegel, editors, *A Workshop on Ideas and Strategies for Multilingual Grammar Engineering, ESSLLI 2003*, Vienna.
- Hellan, L. and Platzack, C. (1995). Pronouns in Scandinavian languages: an overview. *Working Papers in Scandinavian Syntax*, **56**, 47–69.
- Holmberg, A. (1986). *Word order and syntactic features in the Scandinavian languages and English*. Edsbruk: Akademitrykk.
- Holmberg, A. (1999). Remarks on Holmberg’s generalization. *Studia Linguistica*, **53**, 1–39.
- Holmberg, A. and Platzack, C. (1995). *The Role of Inflection in Scandinavian Syntax*. New York and London: Oxford University Press.
- Horiguchi, K., Torisawa, K., and Tsujii, J. (1995). Automatic acquisition of content words using an HPSG-based parser. In *Proceedings of the Natural Language Processing Pacific Rim Symposium*, pages 320–325. Seoul, Korea.

- Hovdhaugen, E. (1977). Om og omkring passiv i norsk. In T. Fretheim, editor, *Sentrale problemer i norsk syntaks*, pages 15–46. Universitetsforlaget.
- Huddleston, R. (1984). *Introduction to the Grammar of English*. Cambridge University Press.
- Hukari, T. E. and Levine, R. (1995). Adjunct extraction. *Journal of Linguistics*, **31(2)**, 195–226.
- Jackendoff, R. (1990). *Semantic structures*. Cambridge, Mass: MIT Press.
- Jaeggli, O. (1986). Passive. *Linguistic Inquiry*, **17**, 587–622.
- Jespersen, O. (1924). *The Philosophy of Grammar*. The University of Chicago Press.
- Johnsen, L. G. (1988). A note on subcoordination. *University of Trondheim Working Papers in Linguistics*, **6**, 195–201.
- Johnsen, L. G., Pitz, A., and Hellan, L. (1989). TROLL (The Trondheim Linguistic Lexicon Project). Technical report, Linguistics Department, University of Trondheim.
- Johnson-Laird, P. N. (1983). *Mental Models: Towards a Cognitive Science of Language, Inference, and Consciousness*. Cambridge: Cambridge University Press.
- Kathol, A. (1994). Passives without lexical rules. In J. Nerbonne, K. Netter, and C. J. Pollard, editors, *German in Head-Driven Phrase Structure Grammar*, number 46 in CSLI Lecture Notes, pages 237–272. CSLI Publications, Stanford University.
- Kay, M. (1986). Algorithm schemata and data structures in syntactic processing. In *Readings in natural language processing*, pages 35–70. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
- Kay, P. and Fillmore, C. J. (1999). Grammatical constructions and linguistic generalizations: The What's X doing Y? *Language*, **64/3**, 501–538.
- Kayne, R. S. (1984). *Connectedness and Binary Branching*. Dordrecht, The Netherlands: Foris Publications.
- Kempen, G. and Hoenkamp, E. (1987). An incremental procedural grammar for sentence formulation. *Cognitive Science*, **11**, 201–258.

- Kim, J.-B. and Yang, J. (2003). Korean phrase structure grammar and its implementations into the LKB system. In D. H. Ji and K. T. Lua, editors, *Proceedings of the 17th Asia Pacific Conference*, pages 88–97. COLIPS Publications.
- Kordoni, V. and Neu, J. (2003). Deep grammar development for Modern Greek. In *Proceedings of the ESSLLI Workshop on Ideas and Strategies for Multilingual Grammar Development*, pages 65–72, Vienna, Austria.
- Larson, M. A. (2005). *The Empty Object Construction and Related Phenomena*. Ph.D. thesis, Cornell University.
- Larson, R. K. (1988). On the double object construction. *Linguistic Inquiry*, **19**, 335–391.
- Levelt, W. J. M. (1989). *Speaking: From Intention to Articulation*. Cambridge, MA and London: The MIT Press.
- Levin, B. (1993). *English Verb Classes and Alternations: A Preliminary Investigation*. University of Chicago Press.
- Levin, B. and Hovav, M. R. (1995). *Unaccusativity*. Cambridge, MA and London: The MIT Press, Linguistic Inquiry Monographs, Volume 26.
- Levine, R. D. (2003). Adjunct valents: cumulative scoping adverbial constructions and impossible descriptions. In J. Kim and S. Wechsler, editors, *The Proceedings of the 9th International Conference on Head-Driven Phrase Structure Grammar*, pages 209–232, Stanford. CSLI Publications.
- Lightfoot, D. (1993). Why UG needs a learning theory: triggering verb movement. In C. Jones, editor, *Historical Linguistics: Problems and Perspectives*, pages 190–214. London and New York: Longman.
- Lødrup, H. (2002). The syntactic structures of Norwegian pseudocoordinations. *Studia Linguistica*, **56(2)**, 121–143.
- Lødrup, H. (1995). The realization of benefactives in Norwegian. In A. Dainora, R. Hemphill, B. Luka, B. Need, and S. Pargman, editors, *CLS 31*. Chicago: Chicago Linguistic Society.

- Lødrup, H. (2000). Underspecification in Lexical Mapping Theory. In M. Butt and T. H. King, editors, *Argument Realization*. Stanford: CSLI Publications.
- Lødrup, H. (2004). Functional structure. Preliminary version. In R. D. Borsley and K. Börjars, editors, *Non-transformational syntax. A guide to current models*. Oxford: Blackwell.
- Manning, C. D. (1996). *Ergativity: Argument Structure and Grammatical Relations*. Stanford: CSLI publications.
- Manning, C. D. (2003). Probabilistic syntax. In R. Bod, J. Hay, and S. Jannedy, editors, *Probabilistic Linguistics*. Cambridge, Massachusetts and London, England: The MIT Press.
- McCloskey, J. (1979). *Transformational Syntax and Model-Theoretic Semantics*. Dordrecht: Reidel.
- Michaelis, L. A. (2005). Entity and event coercion. In J.-O. Östman and M. Fried, editors, *Construction Grammars. Cognitive grounding and theoretical extensions*. John Benjamins Publishing Company.
- Müller, S. (2002). *Complex Predicates: Verbal Complexes, Resultative Constructions, and Particle Verbs in German*. Number 13 in Studies in Constraint-Based Lexicalism. CSLI Publications, Stanford.
- Müller, S. (2004). Continuous or discontinuous constituents? A comparison between syntactic analyses for constituent order and their processing systems. *Research on Language and Computation*, **2**(2), 209–257. Special Issue on Linguistic Theory and Grammar Implementation.
- Müller, S. (2006). Phrasal or lexical constructions? *Language*, **82**(4), 850–883.
- Müller, S. (2007). *Head-Driven Phrase Structure Grammar: Eine Einführung*. Number 17 in Stauffenburg Einführungen. Stauffenburg Verlag, Tübingen.
- Oepen, S. (2001). [incr tsdb()] — competence and performance laboratory. User manual. Technical report, Computational Linguistics, Saarland University, Saarbrücken, Germany. in preparation.

- Oepen, S., Flickinger, D., Toutanova, K., and Manning, C. D. (2004a). LinGO Redwoods. A rich and dynamic treebank for HPSG. *Research on Language and Computation*, **2**(4), 575–596.
- Oepen, S., Dyvik, H., Lønning, J. T., Velldal, E., Beermann, D., Carroll, J., Flickinger, D. P., Hellan, L., Johannessen, J. B., Meurer, P., Nordgård, T., and Rosén, V. (2004b). Som å kapp-ete med trollet? Towards MRS-based Norwegian-English Machine Translation. In *Proceedings of the 10th International Conference on Theoretical and Methodological Issues in Machine Translation*, Baltimore, MD.
- Osam, E. K. (1994). *Aspects of Akan grammar: A functional perspective*. Ph.D. thesis, University of Oregon.
- Parsons, T. (1990). *Events in the Semantics of English. A Study of Subatomic Semantics*, volume 19 of *Current Studies in Linguistics Series*. Cambridge, MA: MIT Press.
- Penn, G. and Richter, F. (2004). Lexical Resource Semantics: From theory to implementation. In S. Müller, editor, *Proceedings of the HPSG-2004 Conference, Center for Computational Linguistics, Katholieke Universiteit Leuven*, pages 423–443. CSLI Publications, Stanford.
- Perlmutter, D. M. (1978). Impersonal passives and the Unaccusative Hypothesis. *Proceedings of the Fourth Annual Meeting of the Berkeley Linguistics Society*, pages 157–189.
- Phillips, C. (2003). Linear order and constituency. *Linguistic Inquiry*, **34**, 37–90.
- Pinker, S. (1989). *Learnability and Cognition*. MIT Press, Cambridge, Mass.
- Platzack, C. (1986). COMP, INFL and Germanic word order. In L. Hellan and K. K. Christensen, editors, *Topics in Scandinavian Syntax*. Dordrecht: Reidel.
- Pollard, C. J. (1994). Toward a unified account of passive in German. In J. Nerbonne, K. Netter, and C. J. Pollard, editors, *German in Head-Driven Phrase Structure Grammar*, number 46 in CSLI Lecture Notes, pages 273–296. CSLI Publications, Stanford University.

- Pollard, C. J. and Sag, I. A. (1994). *Head-Driven Phrase Structure Grammar*. University of Chicago Press, Chicago.
- Ramchand, G. C. (2008). *Verb Meaning and the Lexicon. A First Phase Syntax*. New York: Cambridge University Press.
- Resnik, P. (1992). Left-corner parsing and psychological plausibility. In *COLING:92*, Nantes.
- Riehemann, S. Z. (2001). *A Constructional Approach to Idioms and Word Formation*. Dissertation, Stanford University.
- Ritchie, A. (2004). Compatible RMRS representations from RASP and the ERG. Technical report, University of Cambridge, Computer Laboratory.
- Ross, J. R. (1967). *Constraints on Variables in Syntax*. Ph.D. thesis, MIT.
- Rothstein, S. D. (1985). *The syntactic forms of predication*. Ph.D. thesis, Bar-Ilan University.
- Sag, I. A. (1997). English relative clause constructions. *Journal of Linguistics*, **33**(2), 431–484.
- Sag, I. A. (2005). Adverb extraction and coordination: a reply to Levine. In S. Müller, editor, *The Proceedings of the 12th International Conference on Head-Driven Phrase Structure Grammar, Department of Informatics, University of Lisbon*, pages 322–342, Stanford. CSLI Publications.
- Sag, I. A., Wasow, T., and Bender, E. M. (2003). *Syntactic Theory: A Formal Introduction*. CSLI Publications, Stanford, 2 edition.
- Siegel, M. and Bender, E. M. (2002). Efficient deep processing of Japanese. In *COLING:02*, Taipei, Taiwan.
- Simpson, J. (2006). Resultatives. In M. Butt and T. H. King, editors, *Lexical Semantics in LFG*, pages 149–162. Stanford: CSLI Publications.
- Steedman, M. (1990). Gapping as constituent coordination. *Linguistics and Philosophy*, **13**, 207–263.

- Steedman, M. (2000). *The Syntactic Process*. Cambridge, MA and London: The MIT Press.
- Vikner, S. (1994). Scandinavian object shift and West Germanic scrambling. In H. v. R. Jan Koster, editor, *Studies on Scrambling*, pages 487–517. Mouton de Gruyter, Berlin, New York.
- Vikner, S. (1995). *Verb Movement and Expletive Subjects in the Germanic Languages*. New York and Oxford: Oxford University Press.
- Winkler, S. (1997). *Focus and Secondary Predication*. Berlin and New York: Mouton de Gruyter.
- Åfarli, T. A. (1992). *The Syntax of Norwegian Passive Constructions*. Amsterdam: Benjamins.
- Åfarli, T. A. (2003). Har verbet argumentstruktur? *Motskrift*, pages 87–99.
- Åfarli, T. A. (2006). Passive and argument structure. *Passivization and Typology, Form and Function*, pages 373–382.
- Åfarli, T. A. and Eide, K. M. (2003). *Norsk Generativ Syntax*. Oslo: Novus Forlag.