Head-Driven Phrase Structure Grammar

The handbook

Edited by

Stefan Müller

Anne Abeillé

Robert D. Borsley

Jean-Pierre Koenig

Empirically Oriented Theoretical Morphology and Syntax

language science press

Empirically Oriented Theoretical Morphology and Syntax

In this series:

1. Lichte, Timm. Syntax und Valenz: Zur Modellierung kohärenter und elliptischer Strukturen mit Baumadjunktionsgrammatiken
2. Bîlbîie, Gabriela. Grammaire des constructions elliptiques: Une étude comparative des phrases sans verbe en roumain et en français
3. Bowern, Claire, Laurence Horn & Raffaella Zanuttini (eds.). On looking into words (and beyond): Structures, Relations, Analyses
4. Bonami, Olivier, Gilles Boyé, Georgette Dal, Hélène Giraudo & Fiammetta Namer. The lexeme in descriptive and theoretical morphology.
5. Guzmán Naranjo, Matías. Analogical classification in formal grammar.

ISSN: 2366-3529

Head-Driven Phrase Structure Grammar

The handbook

Edited by

Stefan Müller

Anne Abeillé

Robert D. Borsley

Jean-Pierre Koenig

language
science
press

Müller, Stefan, Anne Abeillé, Robert D. Borsley & Jean-Pierre Koenig (ed.). 2020. *Head-Driven Phrase Structure Grammar*: *The handbook* (Empirically Oriented Theoretical Morphology and Syntax). Berlin: Language Science Press.

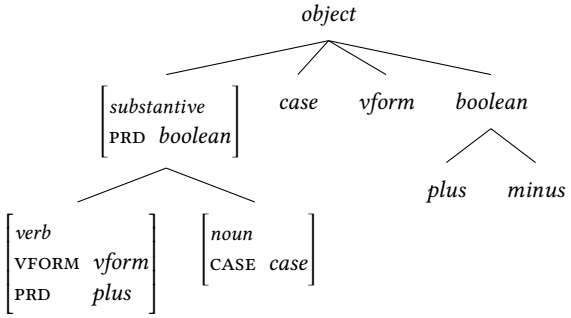
This title can be downloaded at:
http://langsci-press.org/catalog/book/000

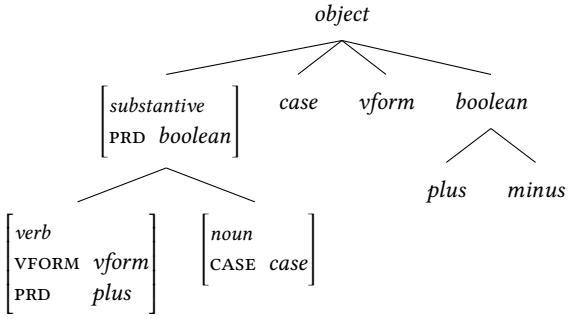
ISBN: no digital ISBN
no print ISBNs!
ISSN: 2366-3529
no DOI
ID not assigned!

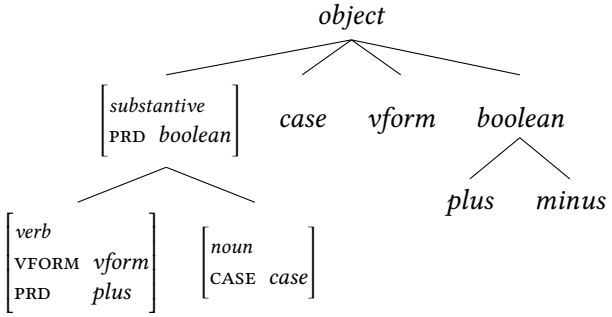
Cover and concept of design: Ulrike Harbort
Typesetting: Stefan Müller, Elizabeth Pankratz
Proofreading: Elizabeth Pankratz
Fonts: Libertinus, Arimo, DejaVu Sans Mono
Typesetting software: XeLaTeX

Language Science Press
Xhain
Grünberger Str. 16
10243 Berlin, Germany
langsci-press.org

Storage and cataloguing done by FU Berlin

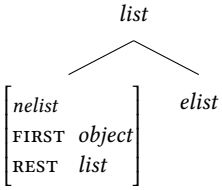


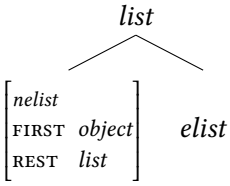




Stefan: The AVMs should come out north and the atomic types (case, vform, ...) too. You did this in the second example. This is nice. Can this be done in the type hierarchy style?

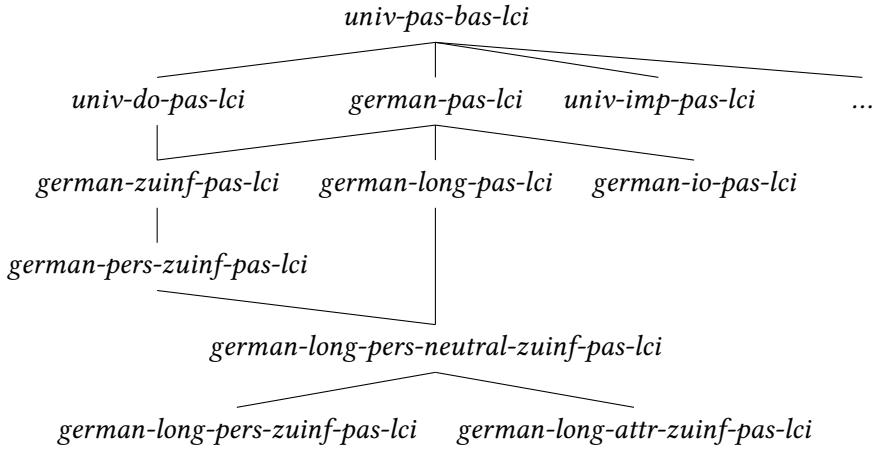
What does “calign children” do?



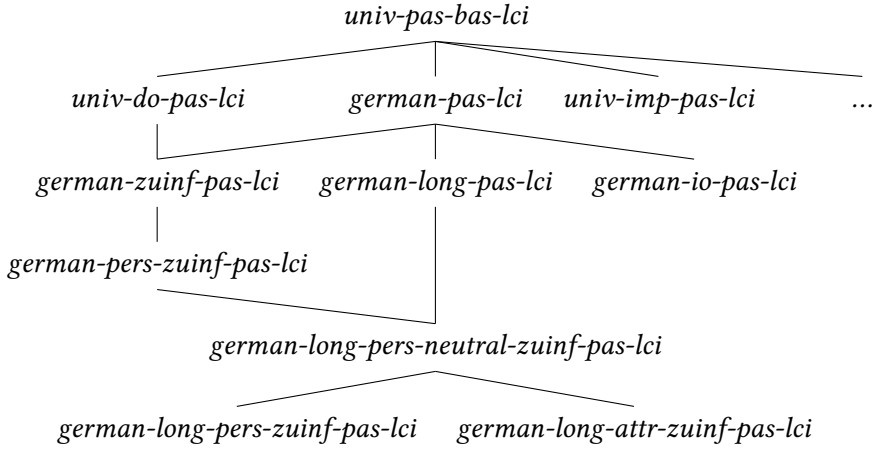


Stefan: Both AVMs and types should come out north.

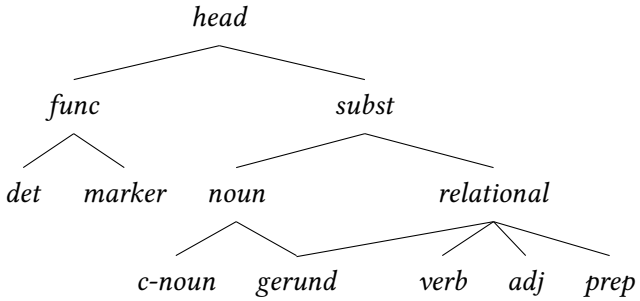
Stefan: Do you know what one can do about the black bars? The figure is scaled to fit the linewidth, so there should not be any black bars.

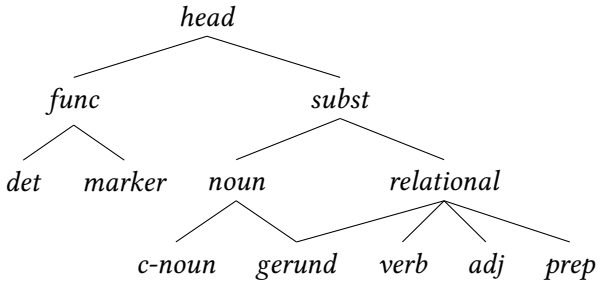


Stefan: Very nice. See my comments in the code. Did I get these right?

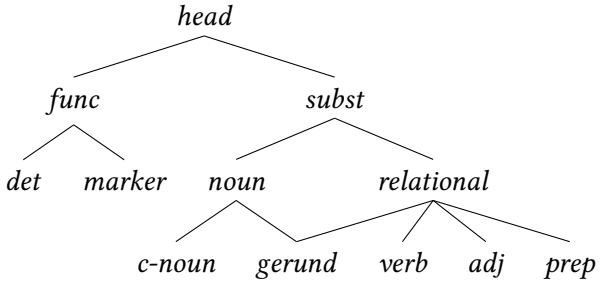


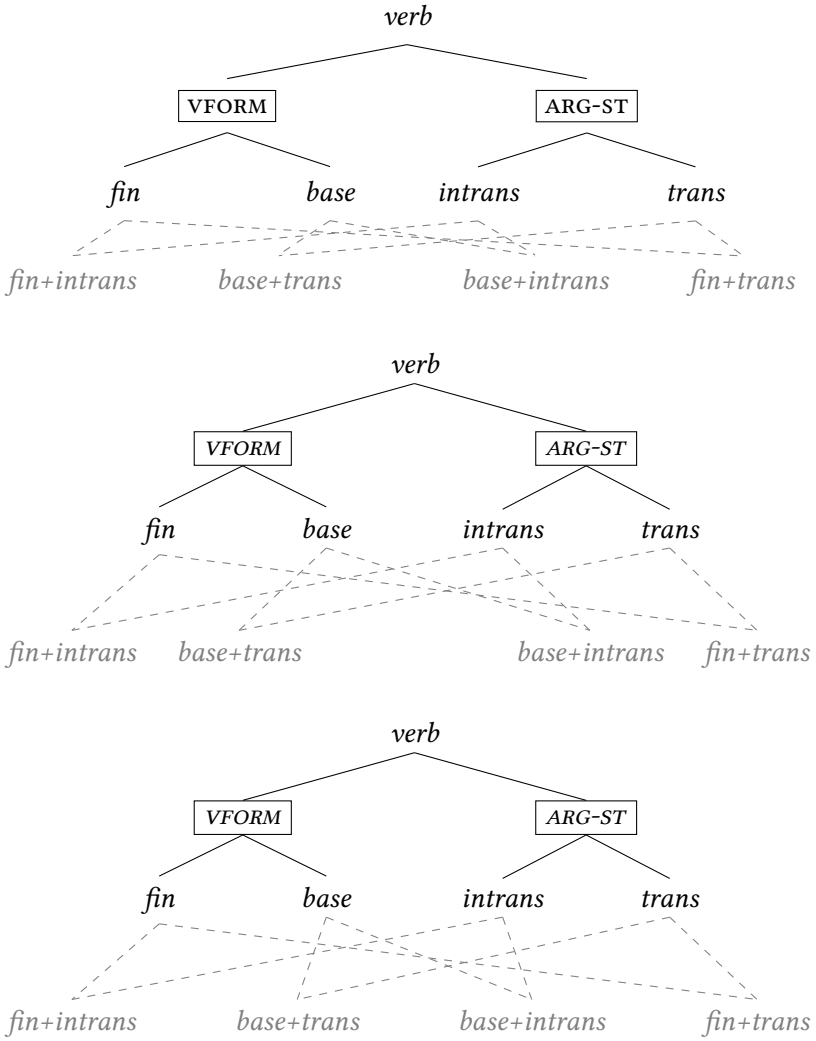
Stefan: It works even without “calign with current”, doesn’t it?





Stefan: What does “calign children” do? “identify” seems to be even better here:

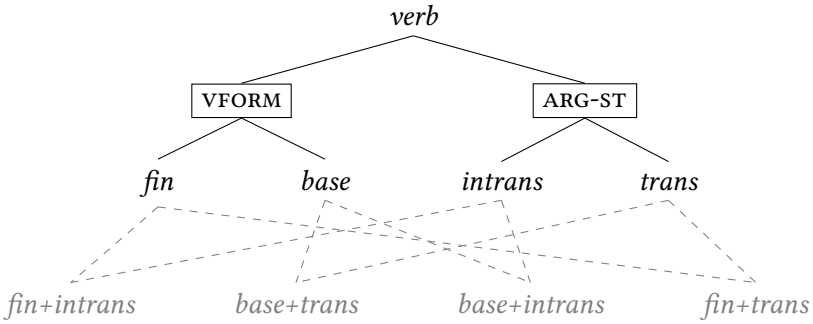


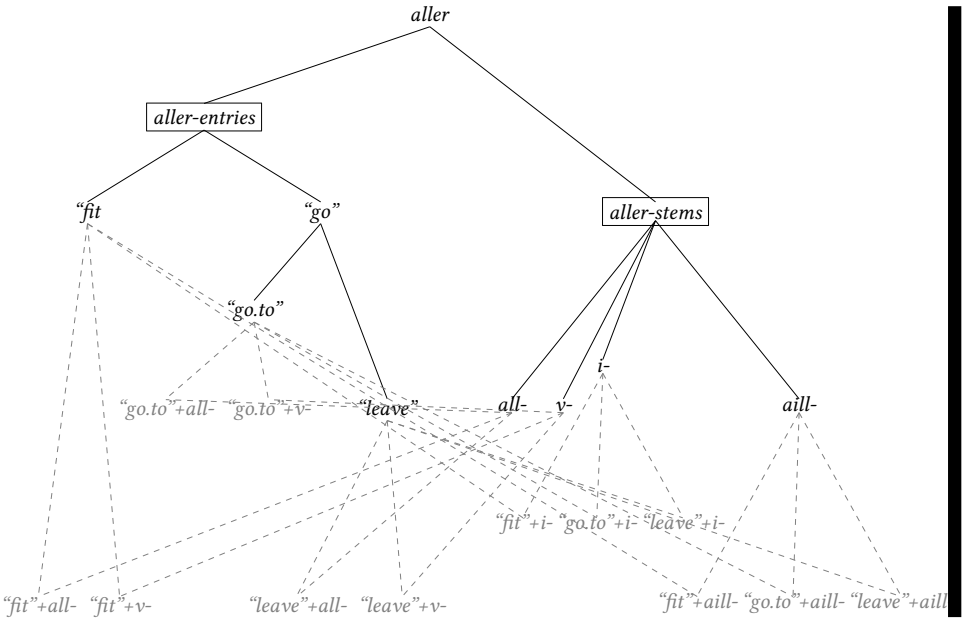


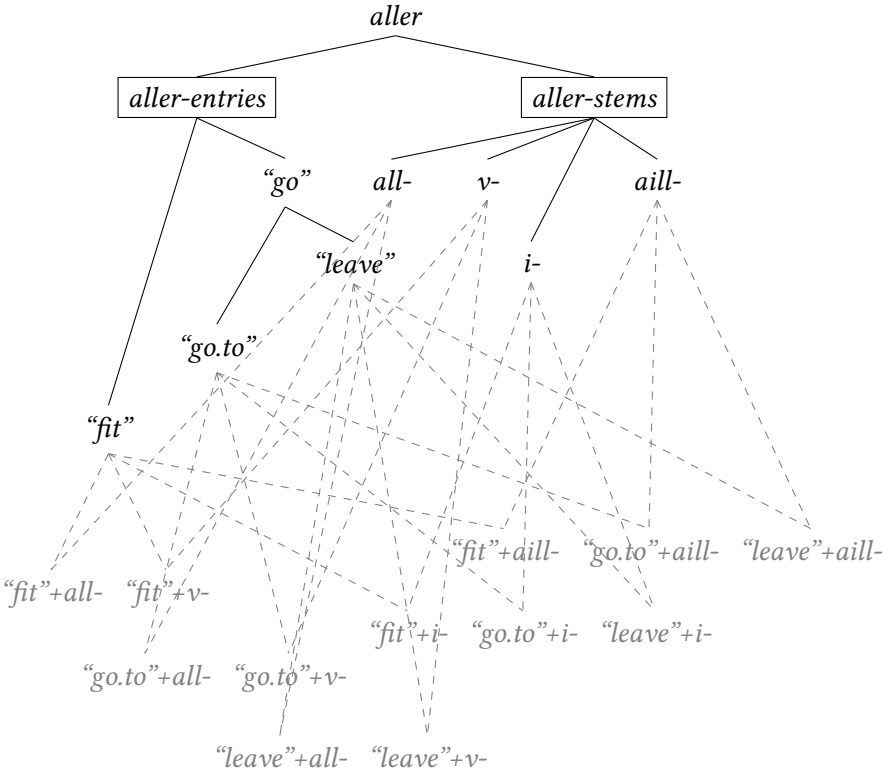
Stefan: Wow! Could you explain why tier=2 is needed? What does the “before drawing tree” magic do?

The lines should not touch the boxes as with the type in other trees.

I introduced the styles “,partition” and “,instance” for the boxed and the grey nodes. This is really cool! Thanks! Here is my version:

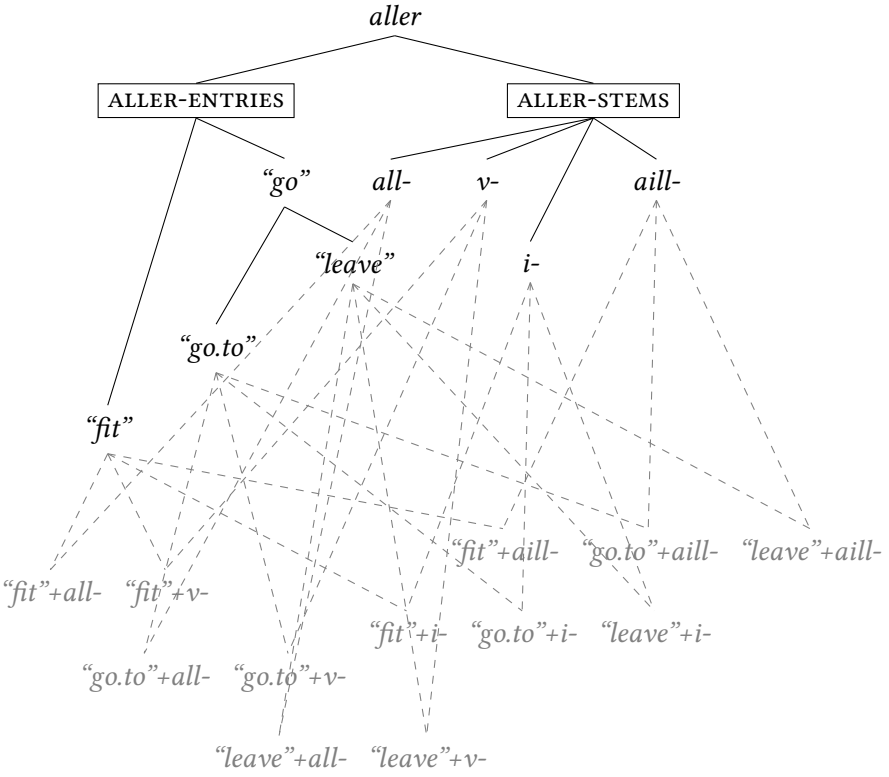

verb
VFORM
ARG-ST
fin
base
intrans
trans
fin+intrans
base+trans
base+intrans
fin+trans

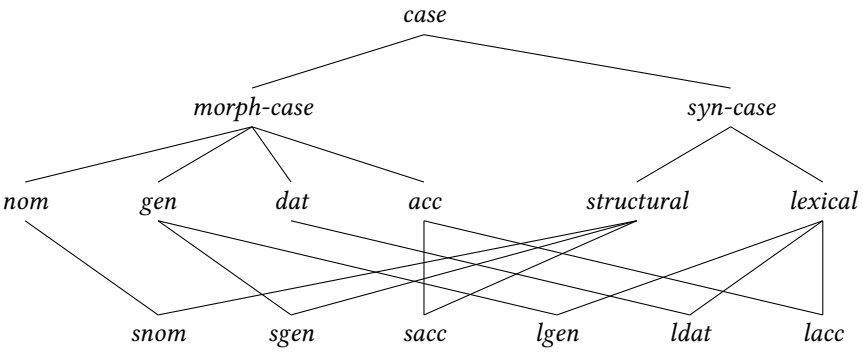

aller
aller-entries
"fit
"go"
aller-stems
"go.to"
i-
"go.to"+all-
"go.to"+v-
"leave"
all-
v-
aill-
"fit"+i-
"go.to"+i-
"leave"+i-
"fit"+all-
"fit"+v-
"leave"+all-
"leave"+v-
"fit"+aill-
"go.to"+aill-
"leave"+aill

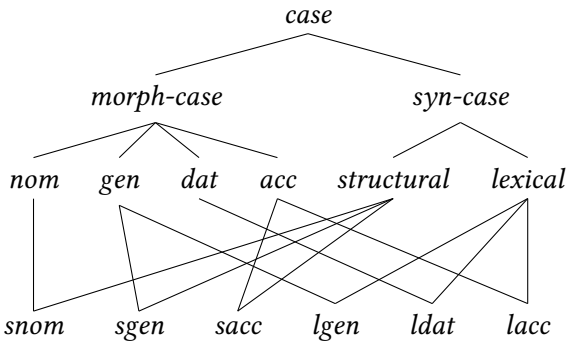


Here I would prefer if the dashed lines would not cross any black nodes. So edges from “all-” should not cross “leave”. I guess this could be reached by shifting “all-” to the right or putting it on the same line as “leave”. But I do not understand the code. Maybe there are better ways.

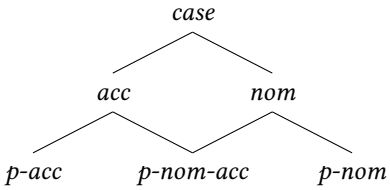
The n children=0 stuff is cool. I turned this into a style. =:-)

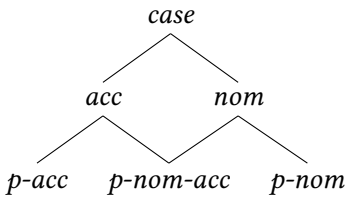

aller
ALLER-ENTRIES
ALLER-STEMS
"go"
all-
v-
aill-
"leave"
i-
"go.to"
"fit"
"fit"+aill-
"go.to"+aill-
"leave"+aill-
"fit"+all-
"fit"+v-
"fit"+i-
"go.to"+i-
"leave"+i-
"go.to"+all-
"go.to"+v-
"leave"+all-
"leave"+v-

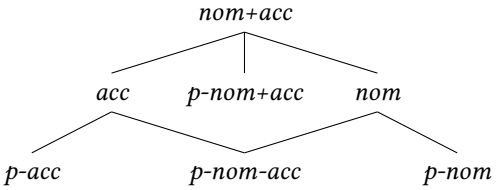


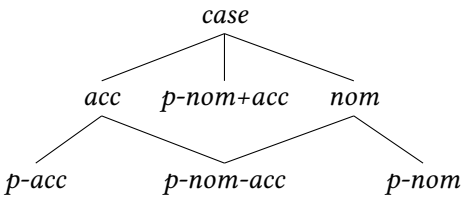


What does the magic code “before drawing tree” do?

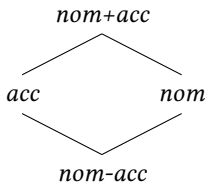

case
acc
nom
p-acc
p-nom-acc
p-nom

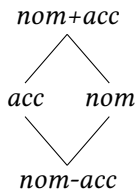

case
acc
nom
p-acc
p-nom-acc
p-nom



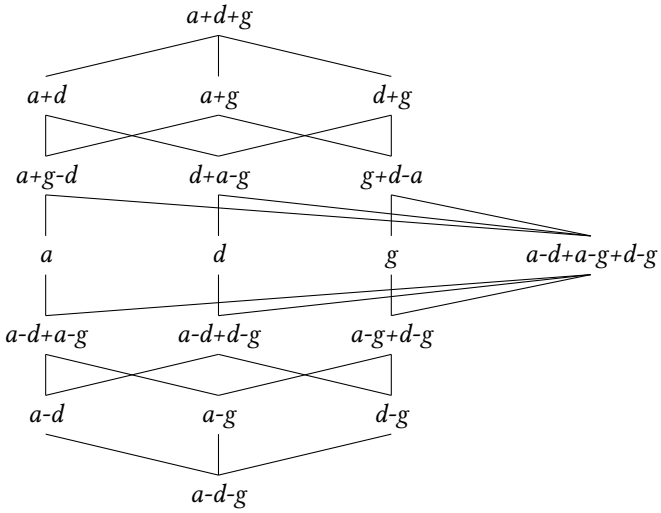


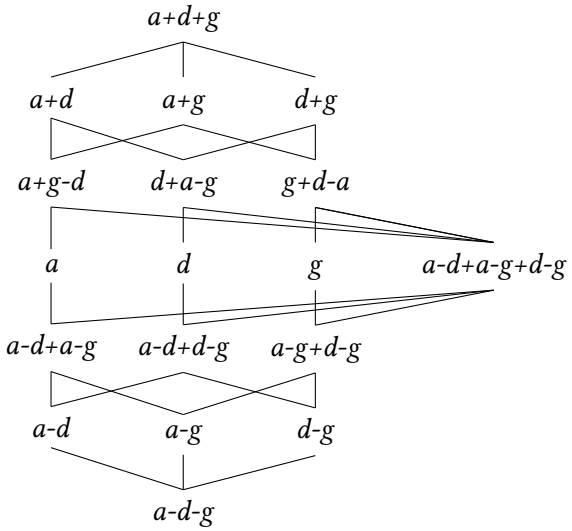
What does the “delay” do? Is it the case that without the “delay”, “identify” would not know that p-nom-acc is a leaf?



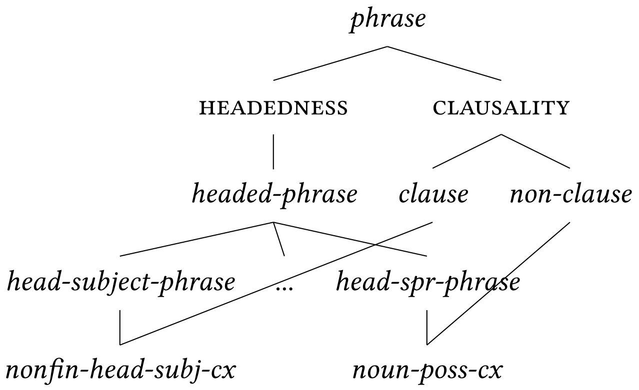


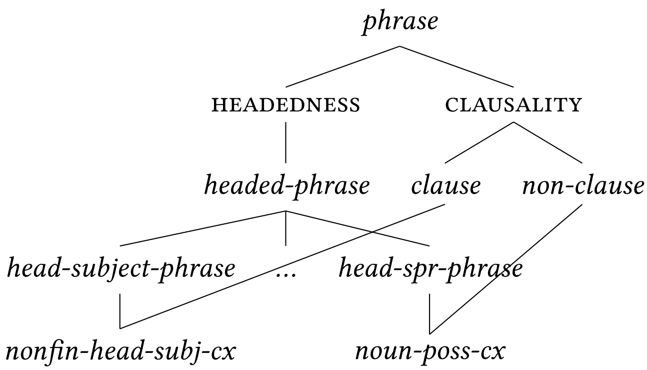
What is “before drawing tree” doing?

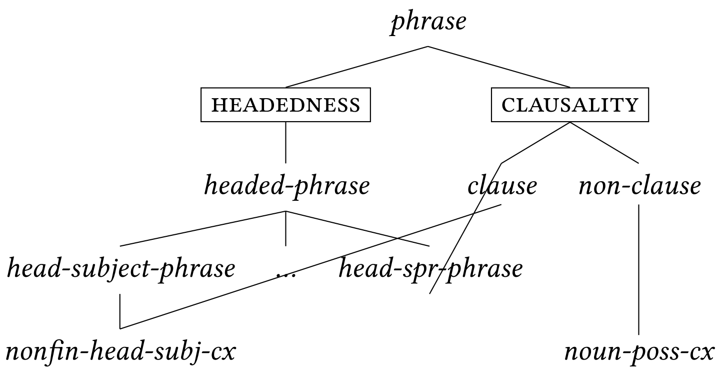




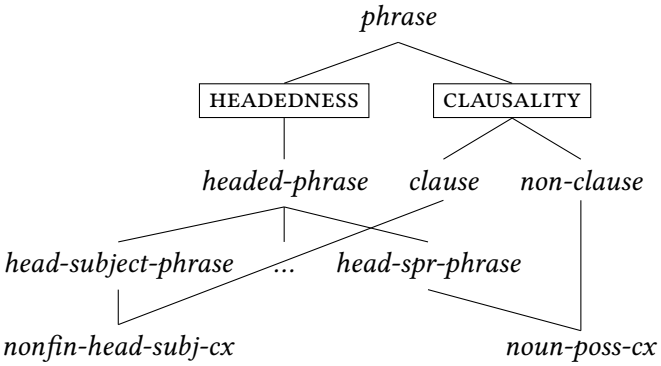
Is “edge to” the counter part of “edge from”?



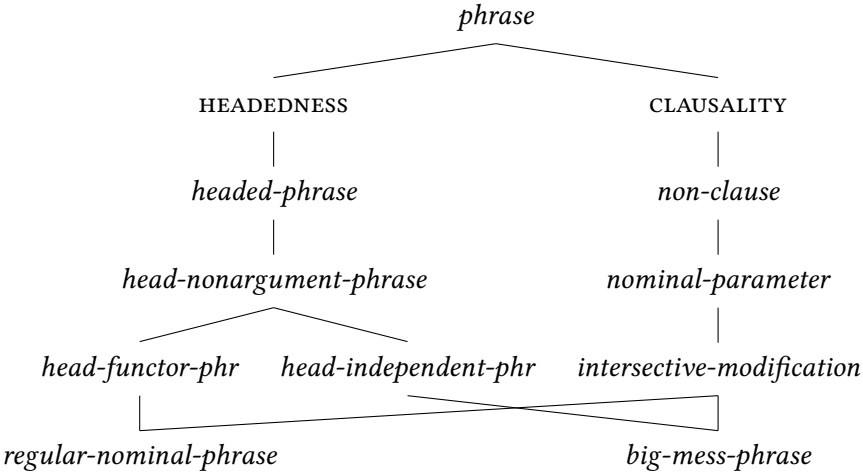


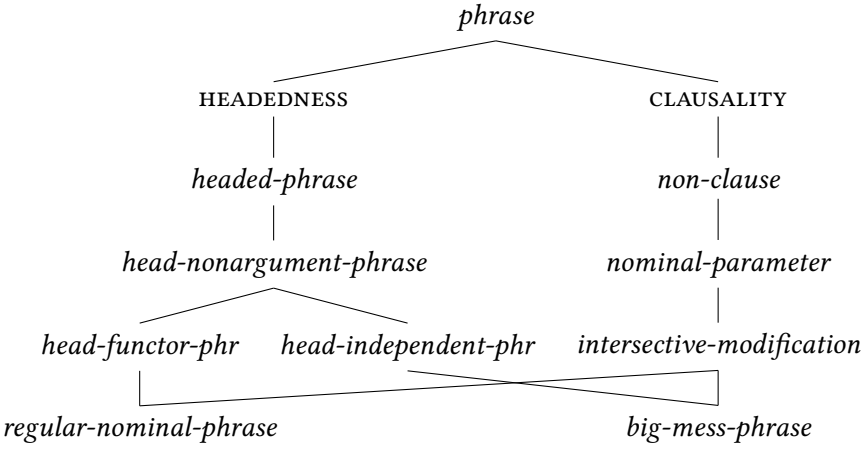


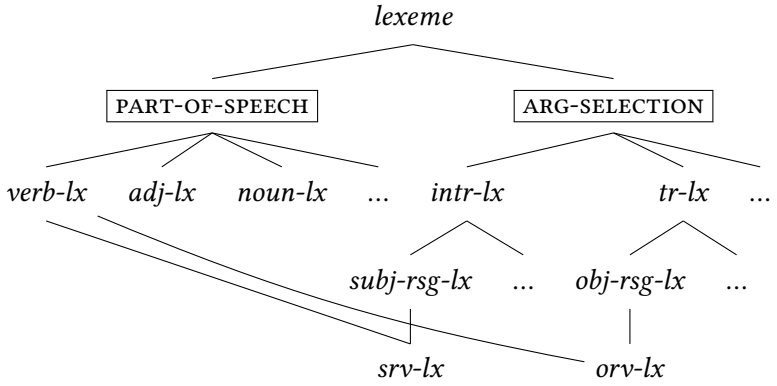
identify=!N doesn't seem to work.

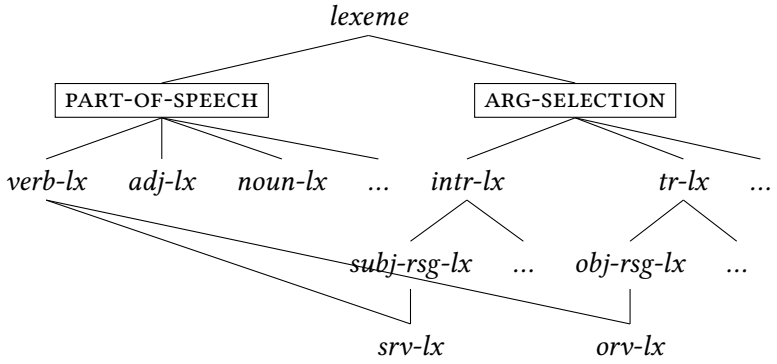


This works without crossing branches, but maybe it would be nicer if noun-poss-cx would be in the midlle of head-spr-phrase and non-clause.

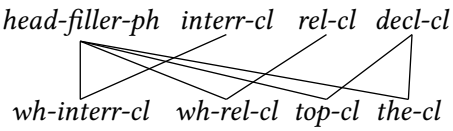

phrase
HEADEDNESS
CLAUSALITY
headed-phrase
non-clause
head-nonargument-phrase
nominal-parameter
head-functor-phr
head-independent-phr
intersective-modification
regular-nominal-phrase
big-mess-phrase

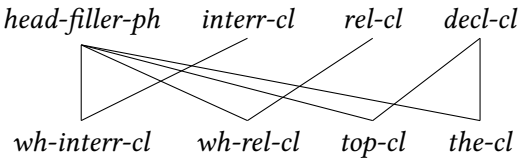

phrase
HEADEDNESS
CLAUSALITY
headed-phrase
non-clause
head-nonargument-phrase
nominal-parameter
head-functor-phr
head-independent-phr
intersective-modification
regular-nominal-phrase
big-mess-phrase



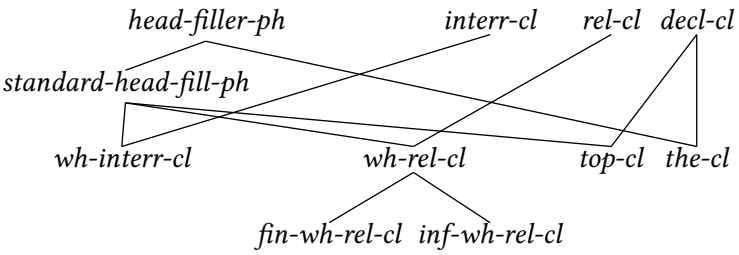


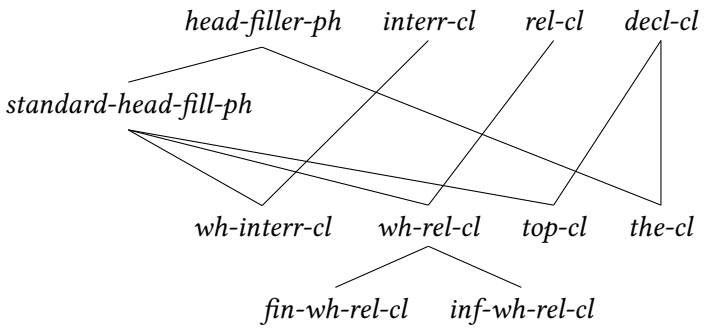
The original did apply a bend to the edge. Would this be possible to avoid the overlap?



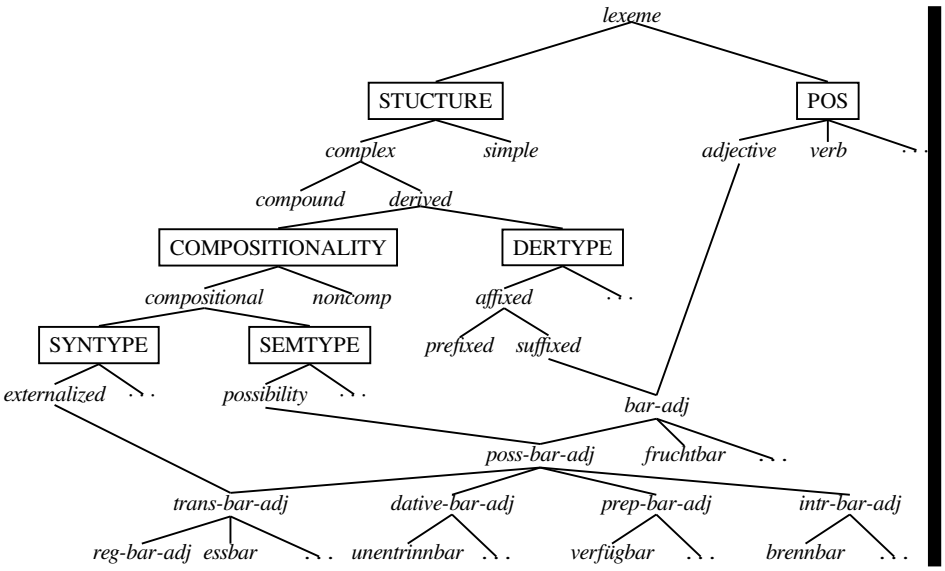


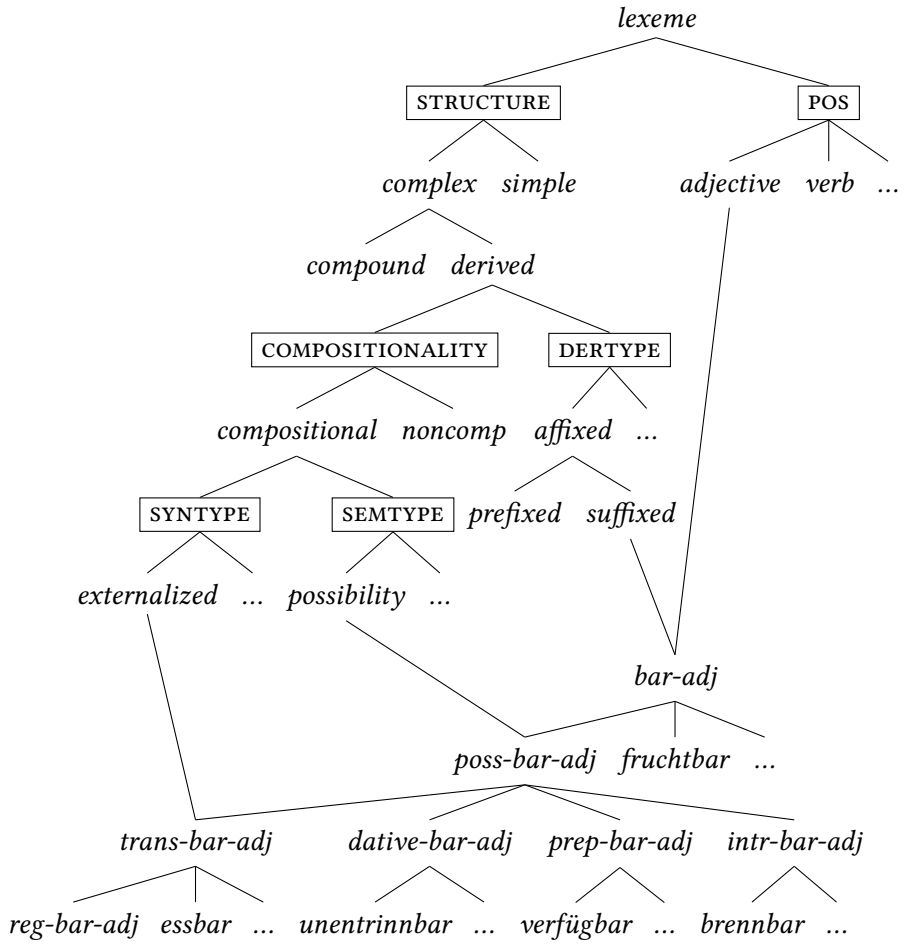
What does “for children” do?



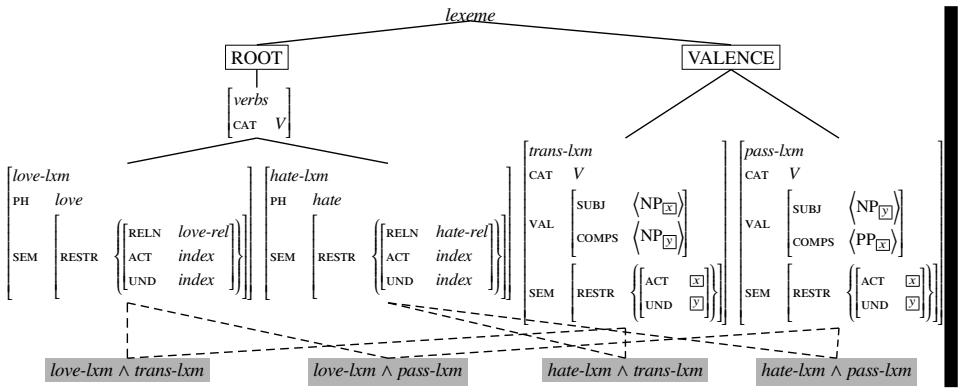


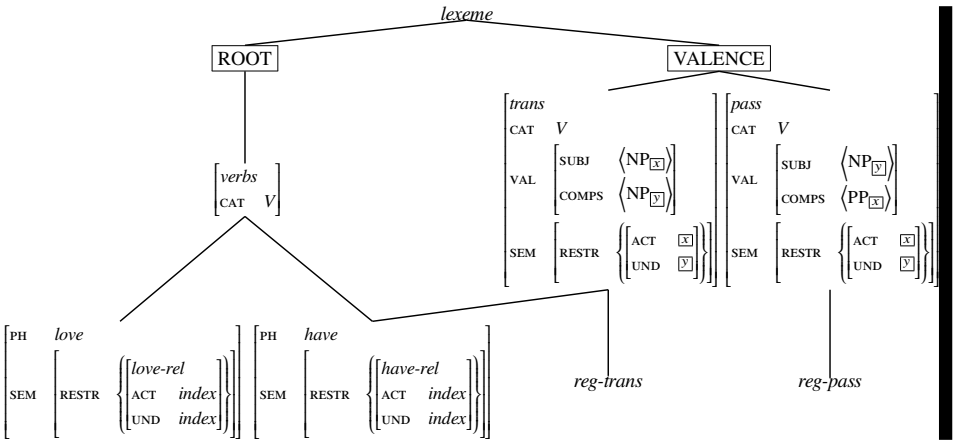
What does “for siblings” do? What does “anchor” do?

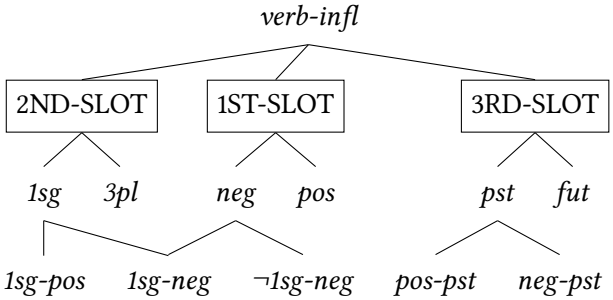

lexeme
STUCTURE
POS
complex
simple
adjective
verb
compound
derived
COMPOSITIONALITY
DERTYPE
compositional
noncomp
affixed
SYNTYPE
SEMTYPE
prefixed
suffixed
externalized
possibility
bar-adj
poss-bar-adj
fruchtbar
trans-bar-adj
dative-bar-adj
prep-bar-adj
intr-bar-adj
reg-bar-adj
essbar
unentrinnbar
verfügbar
brennbar

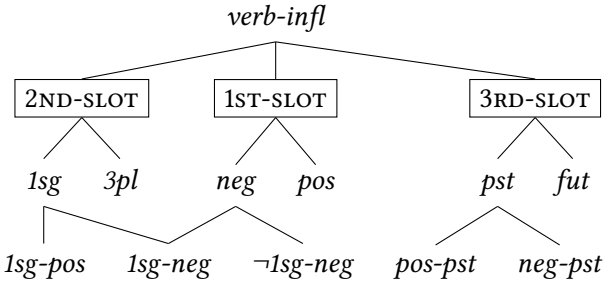

lexeme
STRUCTURE
POS
complex
simple
adjective
verb
...
compound
derived
COMPOSITIONALITY
DERTYPE
compositional
noncomp
affixed
...
SYNTYPE
SEMTYPE
prefixed
suffixed
externalized
...
possibility
...
bar-adj
poss-bar-adj
fruchtbar
...
trans-bar-adj
dative-bar-adj
prep-bar-adj
intr-bar-adj
reg-bar-adj
essbar
...
unentrinnbar
...
verfügbar
...
brennbar
...

Wow!

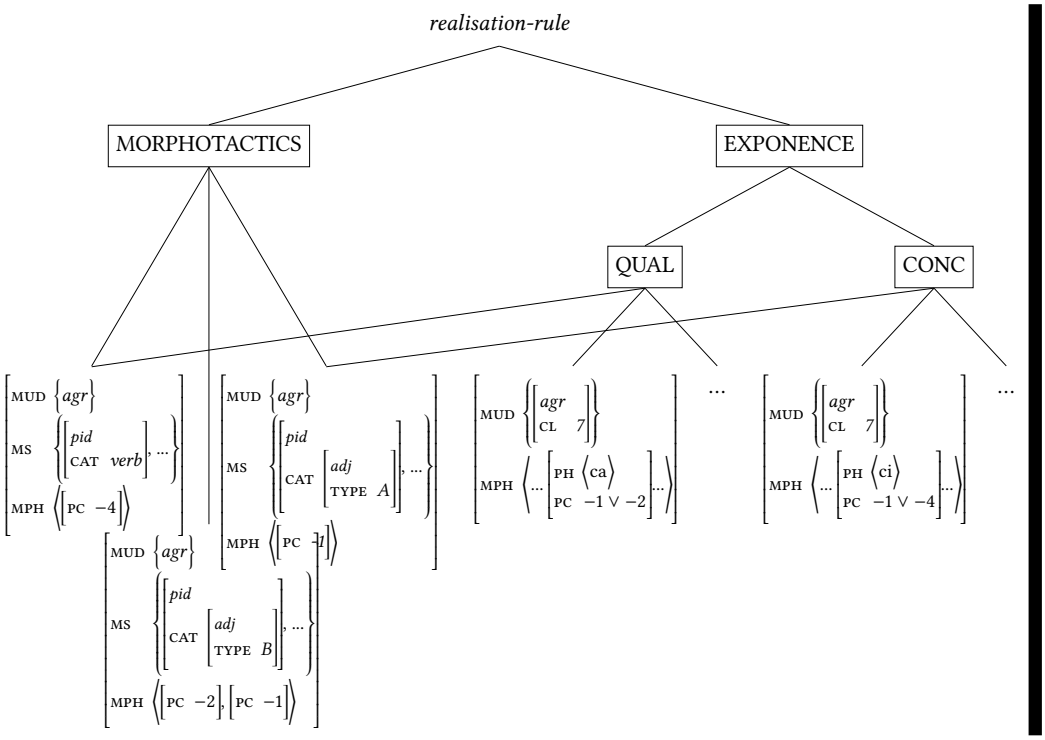

lexeme
ROOT
VALENCE
verbs
CAT V
love-lxm
PH love
SEM RESTR RELN love-rel ACT index UND index
hate-lxm
PH hate
SEM RESTR RELN hate-rel ACT index UND index
trans-lxm
CAT V
VAL SUBJ ⟨NP x⟩ COMPS ⟨NP y⟩
SEM RESTR ACT x UND y
pass-lxm
CAT V
VAL SUBJ ⟨NP y⟩ COMPS ⟨PP x⟩
SEM RESTR ACT x UND y
love-lxm ∧ trans-lxm
love-lxm ∧ pass-lxm
hate-lxm ∧ trans-lxm
hate-lxm ∧ pass-lxm

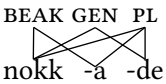

lexeme
ROOT
VALENCE
verbs
CAT V
trans
CAT V
VAL SUBJ ⟨NP[x]⟩
COMPS ⟨NP[y]⟩
SEM RESTR {[ACT x] [UND y]}
pass
CAT V
VAL SUBJ ⟨NP[y]⟩
COMPS ⟨PP[x]⟩
SEM RESTR {[ACT x] [UND y]}
PH love
SEM RESTR {[love-rel] [ACT index] [UND index]}
PH have
SEM RESTR {[have-rel] [ACT index] [UND index]}
reg-trans
reg-pass

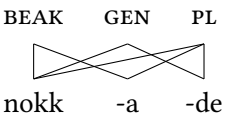


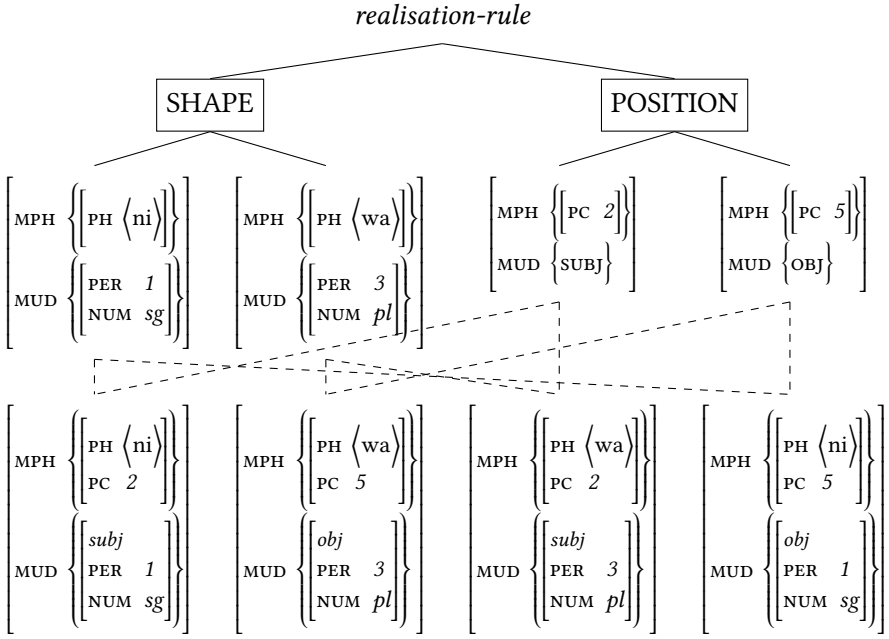


What does “Pp” mean? Isn’t P the previous leaf and p the previous sibling. But the previous leaf is 1sg-pos and it does not have a previous sibling.


realisation-rule
MORPHOTACTICS
EXPONENCE
QUAL
CONC
MUD {agr}
MS {[pid, CAT verb], ...}
MPH ⟨[PC −4]⟩
MUD {agr}
MS {[pid, CAT [adj, TYPE A]], ...}
MPH ⟨[PC −3]⟩
MUD {agr}
MS {[pid, CAT [adj, TYPE B]], ...}
MPH ⟨[PC −2], [PC −1]⟩
MUD {[agr, CL 7]}
MPH ⟨... [PH ⟨ca⟩, PC −1 ∨ −2] ...⟩
MUD {[agr, CL 7]}
MPH ⟨... [PH ⟨ci⟩, PC −1 ∨ −4] ...⟩


BEAK GEN PL
nokk -a -de


BEAK GEN PL
nokk -a -de


realisation-rule
SHAPE
POSITION
MPH
PH
ni
MUD
PER
1
NUM
sg
MPH
PH
wa
MUD
PER
3
NUM
pl
MPH
PC
2
MUD
SUBJ
MPH
PC
5
MUD
OBJ
MPH
PH
ni
PC
2
MUD
subj
PER
1
NUM
sg
MPH
PH
wa
PC
5
MUD
obj
PER
3
NUM
pl
MPH
PH
wa
PC
2
MUD
subj
PER
3
NUM
pl
MPH
PH
ni
PC
5
MUD
obj
PER
1
NUM
sg

Head-Driven Phrase Structure Grammar

Head-Driven Phrase Structure Grammar (HPSG) is a linguistic framework that models linguistic knowledge on all descriptive levels (phonology, morphology, syntax, semantics, pragmatics) by using feature value pairs, structure sharing, and relational constraints. This volume summarizes work that has been done since the mid 80s. Various chapters discus formal foundations and basic assumptions, describe the evolution of the framework and go into the details of various syntactic phenomena. Separate chapters are devoted to non-syntactic levels of description. The book also handles related fields and research areas (gesture, sign languages, computational linguistics) and has a part in which HPSG is compared to other frameworks (Lexical Functional Grammar, Categorial Grammar, Construction Grammar, Dependency Grammar and Minimalim).