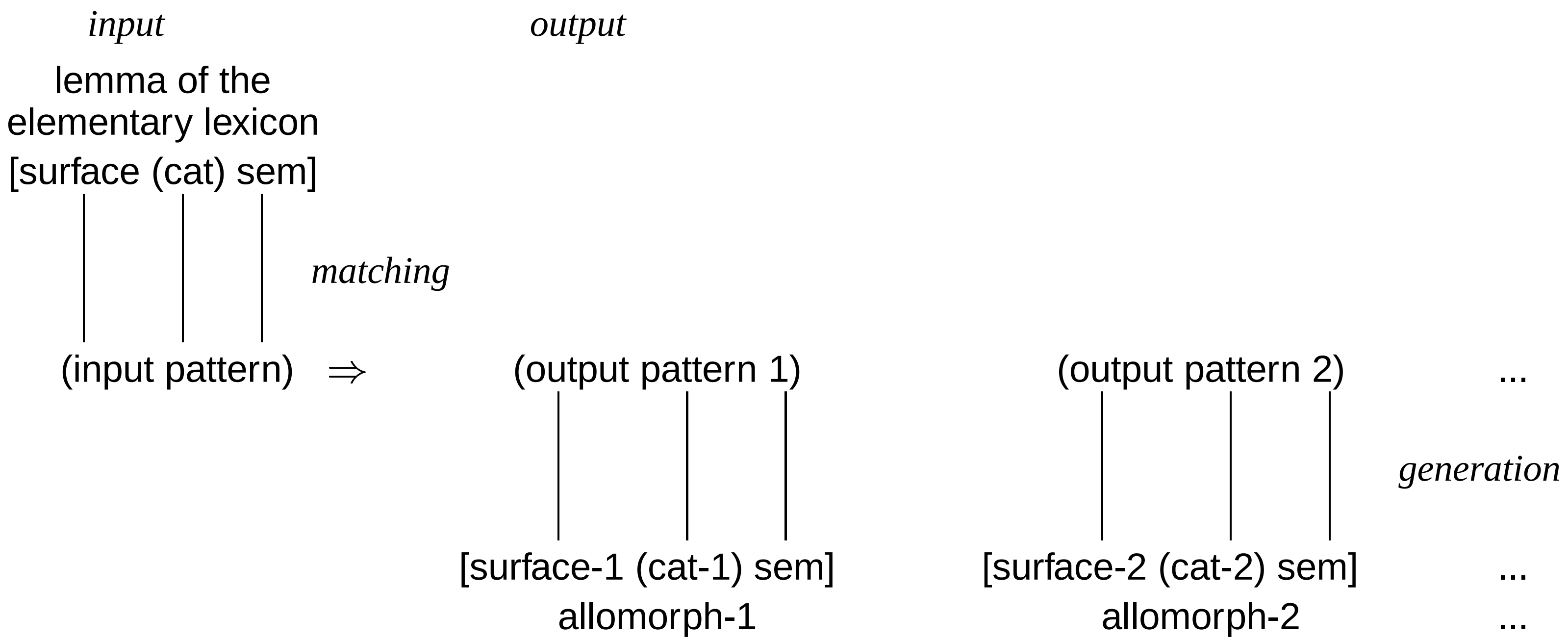
14. Word form recognition in LA-Morph

14.1 Allo-rules

14.1.1 Abstract format of an allo-rule



14.1.2 Example of a base form lemma

```
("derive" (nom a v) derive)
```

14.1.3 Result of applying allo-rules to base form lemma

```
("derive" (sr nom a v) derive)
("deriv" (sr a v) derive)
```

14.1.4 Base form entry of schlafen

```
("schla2fen" (KV VH N GE  {hinueber VS GE } {durch VH A GE }
             {aus VH GE } {ein VS GE }\$ <be VH A GE- >
             <ent VS GE- > <ueber VH A GE- > <ver VH A GE- >)
             schlafen)
```

14.1.5 Output of allo-rules for schlafen

```
("schlaf" (IV V1 VH N GE { hinüber VS GE } { durch VH A GE }
           { aus VH GE } { ein VS GE } $ < be VH A GE- >
           < ent VS GE- > < über VH A GE- > < ver VH A GE- > )
           schlafen)
("schläf" (IV V2 _0 N GE { hinüber VS GE } { durch VH A GE }
           { aus VH GE } { ein VS GE } $ < be VH A GE- >
           < ent VS GE- > < über VH A GE- > < ver VH A GE- > )
           schlafen)
("schlief" (IV V34 _0 N GE { hinüber VS GE } { durch VH A GE }
           { aus VH GE } { ein VS GE } $ < be VH A GE- >
           < ent VS GE- > < über VH A GE- > < ver VH A GE- > )
           schlafen_i)
```

14.1.6 The word forms of schlafen (excerpt)

```
("schlaf/e" (S1 {hinüber}{durch A}{aus}{ein} V) schlafen_p)
("schlaf/e" (S13 {hinüber} {durch A} {aus} {ein} V ) s._k1)
("schlaf/e/n" (P13 {hinüber} {durch A} {aus} {ein} V ) s._pk1)
("schlaf/e/st" (S2 {hinüber} {durch A} {aus} {ein} V ) s._k1)
("schlaf/e/t" (P2 {hinüber} {durch A} {aus} {ein} V ) s._k1)
("schlaf/t" (P2 {hinüber} {durch A} {aus} {ein} V ) s._p)
("schlaf/end" (GER ) schlafen)
("schlaf/end/e" (E ) schlafen)
("schlaf/end/en" (EN ) schlafen)
("schlaf/end/er" (ER ) schlafen)
("schlaf/end/es" (ES ) schlafen)
("schlaf/end/em" (EM ) schlafen)
("schlaf/e/st" (S2 {hinüber} {durch A} {aus} {ein} V ) s._k1)
("schlaf/e/t" (P2 {hinüber} {durch A} {aus} {ein} V ) s._k1)
("schläf/st" (S2 {hinüber} {durch A} {aus} {ein} V ) s._p)
("schläf/t" (S3 {hinüber} {durch A} {aus} {ein} V ) s._p)
("schlief" (S13 {hinüber} {durch A} {aus} {ein} V ) s._i)
("schlief/e" (S13 {hinüber} {durch A} {aus} {ein} V ) s._k2)
("schlief/en" (P13 {hinüber} {durch A} {aus} {ein} V ) s._ik2)
```

```
("schlief/est" (S2 {hinüber} {durch A} {aus} {ein} V ) s._ik2)
("schlief/et" (P2 {hinüber} {durch A} {aus} {ein} V ) s._ik2)
("schlief/st" (S2 {hinüber} {durch A} {aus} {ein} V ) s._ik2)
("schlief/t" (P2 {hinüber} {durch A} {aus} {ein} V ) s._i)
("ge/schlaf/en" (H) schlafen)
("ge/schlaf/en/e" (E) schlafen)
("ge/schlaf/en/en" (EN) schlafen)
("ge/schlaf/en/es" (ES) schlafen)
("ge/schlaf/en/er" (ER) schlafen)
("ge/schlaf/en/em" (EM) schlafen)

("aus/schlaf/e" (S1 V) ausschlafen_pk1)
("aus/schlaf/e" (S13 V ) ausschlafen_k1)
("aus/schlaf/en" (P13 A V ) ausschlafen_pk1)
   ...
("aus/schläf/st" (S2 V) ausschlafen_p)
("aus/schläf/t" (S3 V) ausschlafen_p)
   ...
```

14.1.7 Four degrees of regularity in LA-Morph

- *Regular* inflectional paradigm
  The paradigm is represented by one lemma without any special surface markings, from which one allomorph is derived, e.g. learn ⇒ learn, or book ⇒ book.
- *Semi-regular* inflectional paradigm
  The paradigm is represented by one lemma without any special surface markings, from which more than one allomorph is derived, e.g. derive ⇒ derive, deriv, or wolf ⇒ wolf, wolv.
- *Semi-irregular* inflectional paradigm
  The paradigm is represented by one lemma with a special surface marker, from which more than one allomorph is derived, e.g. swIm ⇒ swim, swimm, swam, swum.
- *Irregular* inflectional paradigm
  The paradigm is represented by several lemmata for suppletive allomorphs which pass through the default rule, e.g. go ⇒ go, went ⇒ went, gone ⇒ gone. The allomorphs serve as input to general combi-rules, as in go/ing.

14.1.8 Tabular presentation of the degrees of regularity

	one lemma per paradigm	lemma without markings	one allomorph per lemma
regular	yes	yes	yes
semi-regular	yes	yes	no
semi-irregular	yes	no	no
irregular	no	no	yes

14.2 Phenomena of allomorphy

14.2.1 Allomorphs of semi-regular nouns

LEX	ALLO1	ALLO2
wolf	wolf	wolv
knife	knife	knive
ability	ability	abiliti
academy	academy	academi
agency	agency	agenci
money	money	moni

14.2.2 Allomorphs of semi-irregular nouns

LEX	ALLO1	ALLO2
analysis	analysis	analyses
larva	larva	larvae
stratum	stratum	strati
matrix	matrix	matrices
thesis	thesis	theses
criterion	criterion	criteria

tempo	tempo	tempi
calculus	calculus	calculi

14.2.3 Allomorphs of semi-regular verbs

LEX	ALLO1	ALLO2
derive	derive	deriv
dangle	dangle	dangl
undulate	undulate	undulat
accompany	accompany	accompani

14.2.4 Allomorphs of semi-irregular verbs

LEX	ALLO1	ALLO2	ALLO3	ALLO4
swIm	swim	swimm	swam	swum
rUN	run	runn	ran	run
bET	bet	bett	bet	bet

14.2.5 Allomorphs of semi-regular adjective-adverbials

LEX	ALLO1	ALLO2
able	able	abl
happy	happy	happi
free	free	fre
true	true	tru

14.2.6 Definition of the allomorph quotient

The allomorph quotient is the percentage of additional allomorphs relative to the number of base form entries.

14.2.7 The allomorph quotient of different languages

Italian: 37%
German: 31%
English: 8,97%

14.2.8 Compounds with ‘pseudo-’ contained in Webster’s New Collegiate Dictionary

pseudoclassic
pseudopregnancy
pseudosalt
pseudoscientific
etc.

14.2.9 Compounds with ‘pseudo-’ not contained in Webster’s New Collegiate Dictionary

pseudogothic
pseudomigrane
pseudoscientist
pseudovegetarian
etc.

14.2.10 Problem for recognition algorithm

In order to recognize the highly productive compositions involving the prefix pseudo, the LA-Morph system must provide a general rule-based analysis. As a consequence, the word forms in 14.2.8, are analyzed as ambiguous whereby the second reading stems from the compositional analysis based on the known forms, e.g. pseudo and classic.

14.2.11 Solution I

Automatic removal of all non-elementary base forms from the on-line lexicon.

14.2.12 Solution II

Leaving the non-elementary base forms like 14.2.8 in the lexicon, but selecting the most likely reading after the word form analysis.

14.2.13 Solution III

Using two lexica. One is an elementary lexicon which does not contain any non-elementary base forms. It is used for the categorization and lemmatization of word forms.

The other is a base form lexicon of content words. It assigns semantic representations to base forms including composita and derivata established in use. During word form analysis the two lexica are related by matching the result of lemmatization onto a corresponding – if present – key word of the base form lexicon (cf. 13.4.7).

14.2.14 Example of solution III

The compositional analysis of kin/ship would be matched onto kinship in the non-elementary base form lexicon, accessing the proper semantic description. In this way, (i) maximal data coverage – including neologisms – is ensured by a rule based analysis, (ii) the possibility of noncompositional meanings is accounted for, and (iii) unnecessary ambiguities are avoided.

14.3 Left-associative segmentation into allomorphs

14.3.1 Left-associative letter by letter matching

attempt 1:	W	O	L	F
	|	|	|	×
surface:	W	O	L	V
	|	|	|	|
attempt 2:	W	O	L	V

b14.3.1.pictex

14.3.2 Hypothetical examples of English allowing alternative segmentations

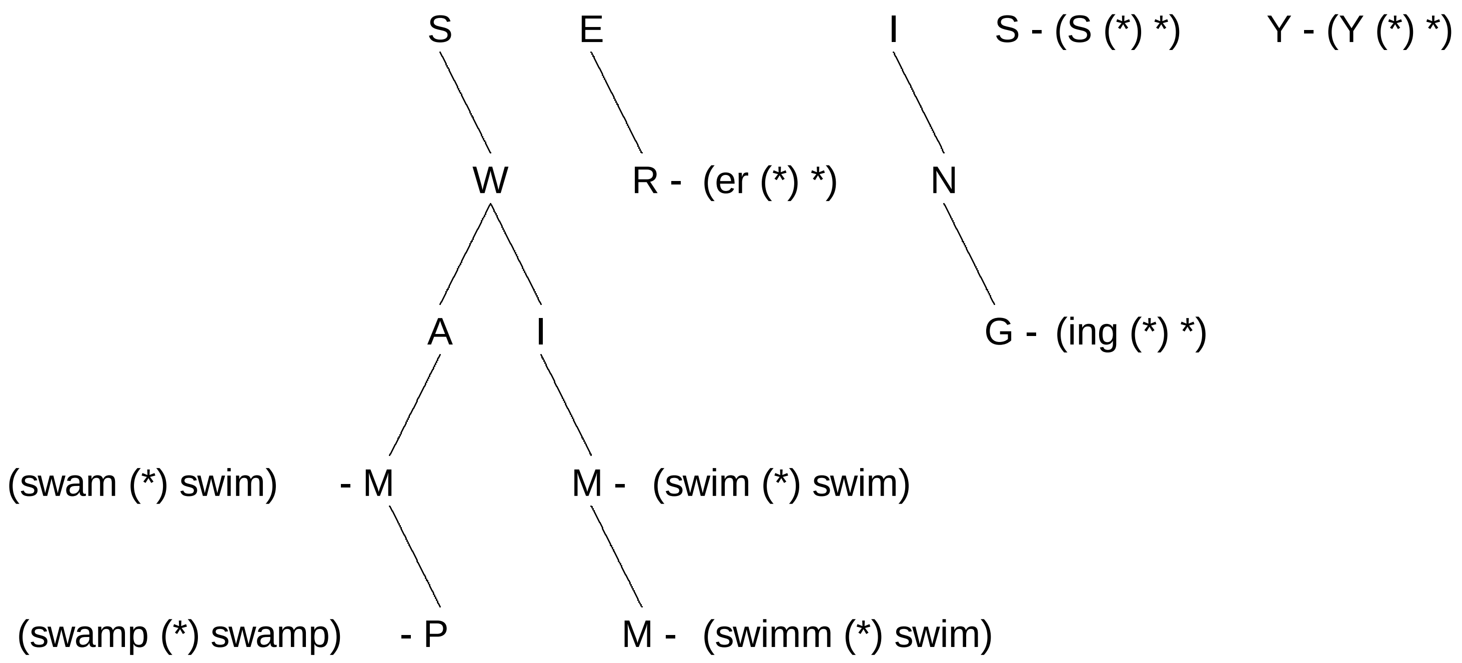
coverage	grandparent	history	lamp/light	land/s/end
cover/age	grandpa/rent	hi/story	lam/plight	land/send
cove/rage		his/tory		

rampage	rampart	scar/face	sing/able	war/plane
ramp/age	ramp/art	scarf/ace	sin/gable	warp/lane
ram/page	ram/part			

14.3.3 Alternative segmentations of a word form in German

surface:	Staubecken	Staubecken
segmentation:	Stau/becken	Staub/ecke/n
translation:	*reservoir*	*dust corners*

14.3.4 Storing allomorphs in a trie structure



14.3.5 Possibilities after finding an entry in the trie structure

- There are no letters left in the surface of the unknown word form, e.g. SWAM. Then the program simply returns the analysis stored at the last letter, here M.
- There are still letters left in the surface of the unknown word form. Then one of the following alternatives applies:
  - The allomorph found so far *is part* of the word form, as swim in SWIMS. Then the program (i) gives the lexical analysis of swim to the combi-rules of the system and (ii) looks for the next allomorph (here s), starting again from the top level of the trie structure.
  - The allomorph found so far *is not part* of the word form, as swam in SWAMPY. In this case the program continues down the trie structure provided there are continuations. In our example, it will find swamp.

  Because it becomes apparent only at the very end of a word form which of these two possibilities applies – or whether they apply simultaneously in the case of an ambiguity – they are pursued simultaneously by the program.

14.4 Combi-rules

14.4.1 Structure of combi-rules

input *output*

r_n: (pattern of start) (pattern of next) $\Rightarrow$ rp_n (pattern of new start)

14.4.2 Difference between allo- and combi-rules

Combi-rules differ from allo-rules in that they are defined for different domains and different ranges:

An *allo-rule* takes a lexical entry as input and maps it into one or more allomorphs.
A *combi-rule* takes a word form start and a next allomorph as input and maps it into a new word form start.

14.4.3 Tasks of combi-rules

The combi-rules ensure that

1. the allomorphs found in the surface are not combined into ungrammatical word forms,
   e.g. *swam+ing or *swimm+s (input condition),
2. the surfaces of grammatical allomorph combinations are properly concatenated,
   e.g. swim+s ⇒ swims,
3. the categories of the input pair are mapped into the correct result category,
   e.g. (NOM V) + (SX S3) ⇒ (S3 V),
4. the correct result is formed on the level of semantic interpretation, and
5. after a successful rule application the correct rule package for the next combination is activated.

14.4.4 Derivation of unduly in LA-Morph

```
 1 +u [NIL . NIL]
 2  +n [NIL . (un (PX PREF) UN)]
 RP:{V-START N-START A-START P-START}; fired: P-START
 3   +d [(un (PX PREF) UN) . (d (GG) NIL)]
     +d [NIL . NIL]
 4    +u [(un (PX PREF) UN) . (du (SR SN) DUE (SR ADJ-V) DUE)]
 RP:{PX+A UN+V}; fired: PX+A
      +u [NIL . NIL]
 5 L [(un+du (SR ADJ) DUE) . (l (GG) NIL (ABBR) LITER)]
 RP:{A+LY}; fired: none
       +l [(un (PX PREF) UN) . NIL]
       +l [NIL . NIL]
 6      +y [(un+du (SR ADJ) DUE) . (ly (SX ADV) LY)]
 RP:{A+LY}; fired: A+LY
("un/du/ly" (ADV) due)
```

14.4.5 Handling of ungrammatical input in LA-Morph

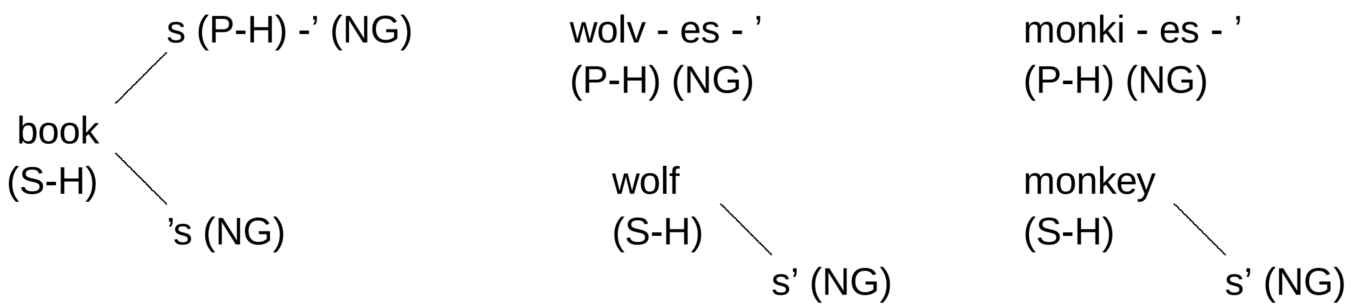
```
 1 +a [NIL . (a (SQ) A)]
 2  +b [NIL . NIL]
 3   +l [NIL . (abl (SR ADJ-A) ABLE)]
 RP:{V-START N-START A-START P-START}; fired: A-START
 4    +e [(abl (SR ADJ) ABLE) . NIL]
      +e [NIL . (able (ADJ) ABLE)]
 RP:{V-START N-START A-START P-START}; fired: none
 5     +l [(abl (SR ADJ) ABLE) . NIL]
ERROR
Unknown word form: "ablely"
NIL
```

14.4.6 Parsing the simplex undulate

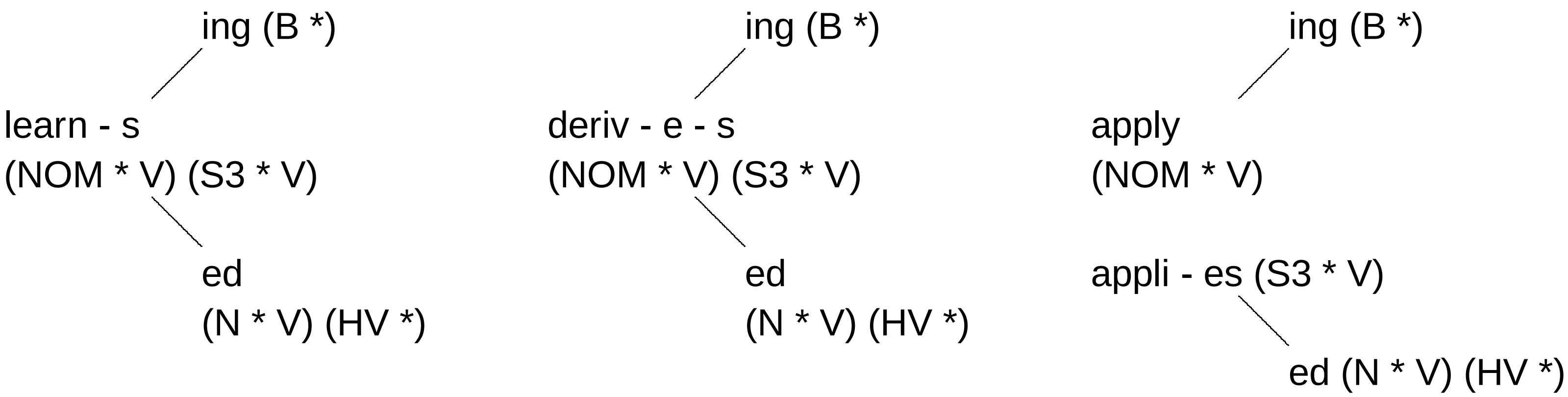
```
 1 +u [NIL . NIL]
 2  +n [NIL . (un (PX PREF) UN)]
 RP:{V-START N-START A-START P-START}; fired: P-START
 3   +d [(un (PX PREF) UN) . (d (GG) NIL)]
     +d [NIL . NIL]
 4    +u [(un (PX PREF) UN) . (du (SR SN) DUE (SR ADJ-V) DUE)]
 RP:{PX+A UN+V}; fired: PX+A
      +u [NIL . NIL]
 5     +l [(un+du (SR ADJ) DUE) . (l (GG) NIL (ABBR) LITER)]
 RP:{A+LY}; fired: none
       +l [(un (PX PREF) UN) . NIL]
       +l [NIL . NIL]
 6      +a [(un+du (SR ADJ) DUE) . NIL]
        +a [NIL . NIL]
 7       +t [(un+du (SR ADJ) DUE) . NIL]
         +t [NIL . (undulat (SR A V) UNDULATE)]
 RP:{V-START N-START A-START P-START}; fired: V-START
 8        +e [(un+du (SR ADJ) DUE) . (late (ADJ-AV) LATE (ADV) LATE)]
 RP:{A+LY}; fired: none
          +e [(undulat (SR A V) UNDULATE) . NIL]
          +e [NIL . (undulate (SR NOM A V) UNDULATE)]
 RP:{V-START N-START A-START P-START}; fired: V-START
("undulate" (NOM A V) UNDULATE)
```


14.5 Concatenation patterns

14.5.1 Concatenation patterns of English nouns



14.5.2 Concatenation patterns of English verbs



14.5.3 Concatenation patterns of adjective-adverbs

```
              ly (ADV)                              ly (ADV)                              ly (ADV)
             /                  able (ADJ)         /              steady (ADJ)           /
quick - er (CAD)                abl - er (CAD)                    steadi - er (CAD)
(ADJ)        \                                     \                                     \
              est (SAD)                             est (SAD)                             est (SAD)
```

14.5.4 Concatenation patterns of German nouns

schmerz (M-G): es (-FG) / -e (-FD) / en (P)

tag (M-G): es (-FG) / e (MDP-D) -n (PD)

leib (M-G): es (-FG) / -e (-FD) / er (P-D) -n (PD)

gipfel (M-GP-D): s (-FG) / n (PD)

stachel (M-G): s (-FG) / n (P)

thema (N-G): s (-FG) / themen (P)

vAter (M-G): s (-FG) / (P) -n (PD)

auge (N-G): s (-FG) / n (P)

uhu (M-G): -s (MGP)

braten (M-GP)	-s (-FG)	hAnd (F)	-e (P-D) -n (PD)	frau (F)	-en (P)
drangsal (F)	-e (P-D) -n (PD)	kenntnis (F)	-se (P-D) -n (PD)	mUtter (F)	- (P-D) -n (PD)

14.5.5 Category segments of German noun forms

MN	= Masculinum Nominativ	(Bote)
M-G	= Masculinum no Genitiv	(Tag)
-FG	= no Femininum Genitiv	(Tages, Kindes)
-FD	= no Femininum Dativ	(Schmerze, Kinde)
M-NP	= Masculinum no Nominativ or Plural	(Boten)
M-GP	= Masculinum no Genitiv or Plural	(Braten)
MGP	= Masculinum Genitiv or Plural	(Uhus)
M-GP-D	= Masculinum no Genitiv or Plural no Dativ	(Gipfel)
F	= Femininum	(Frau)
N-G	= Neutrum no Genitiv	(Kind)
NG	= Neutrum Genitiv	(Kindes)
ND	= Neutrum Dativ	(Kinde)
N-GP	= Neutrum no Genitiv or Plural	(Leben)
N-GP-D	= Neutrum no Genitiv or Plural no Dativ	(Wasser)
NDP-D	= Neutrum Dativ or Plural no Dativ	(Schafe)
P	= Plural	(Themen)
P-D	= Plural no Dativ	(Leiber)
PD	= Plural Dativ	(Leibern)