Butt, Frank, Kuhn:
Development of large scale LFG grammars
ESSLLI 1999, Utrecht

The Interface to Morphological Analyzers

Jonas Kuhn
jonas@ims.uni-stuttgart.de

Overview

- What the morphological analyzers do
- The interface in XLE
- Building interface rules in the gramar

Lexical Resources

- For wide-coverage grammars, extensive lexical resources are essential
- Hand-coded full-form lists are impractical, especially for languages with rich inflection
- Wide-coverage morphological analyzers are available, so use these
  - (note: they do not provide the full amount of lexical information required for LFG parsing – subcategorization of verbs etc. is missing)

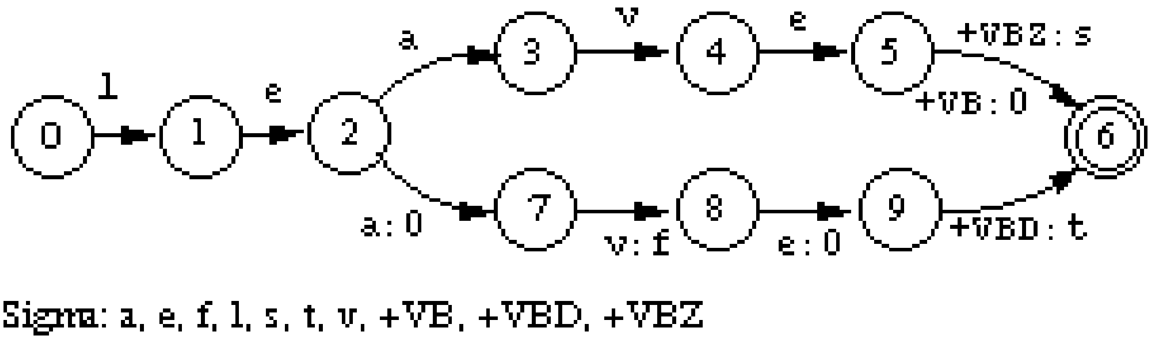
Technology used in morphological analyzers

- Finite-state technology (Koskenniemi 1983; Karttunen, Kaplan, and Zaenen 1992; Kaplan and Kay 1994)
  - very efficient processing (complexity is linear in the length of the input string)
  - Xerox finite-state morphologies `http://www.rxrc.xerox.com/research/mltt/fst/`

What the morphological analyzers do

- (Inflectional) morphology modelled by a **finite-state transducer** (an automaton, where the arcs are labeled with a pair of symbols – from the input and the output language)

```
[ [l e a v e %+VBZ .x. l e a v e s] |
  [l e a v e %+VB  .x. l e a v e] |
  [l e a v e %+VBG .x. l e a v i n g] |
  [l e a v e %+VBD .x. l e f t] |
  [l e a v e %+NN  .x. l e a v e] |
  [l e a v e %+NNS .x. l e a v e s] |
  [l e a f %+NNS   .x. l e a v e s ] |
  [l e f t %+JJ    .x. l e f t] ]
```



Application:

```
APPLY DOWN> leave+VBD
left

APPLY UP> leaves
leave+NNS
leave+VBZ
leaf+NNS
```

What the morphological analyzers do

The linguistic specification of morphological generalizations

- Lexical transducers are typically modelled by composing a network of lexical forms with rule transducers that encode the regular morphological alternations of the language

Morphological analyzers in the context of XLE

- XLE runs the morphological analyzers during lexical lookup (after a *Tokenization* step)
  - a MORPHOLOGY section has to be mentioned in the CONFIG section
  - one can see the effect of tokenization and morphological analysis by using the command `analyze-string`
  - these strings can now be used as the input for an LFG grammar

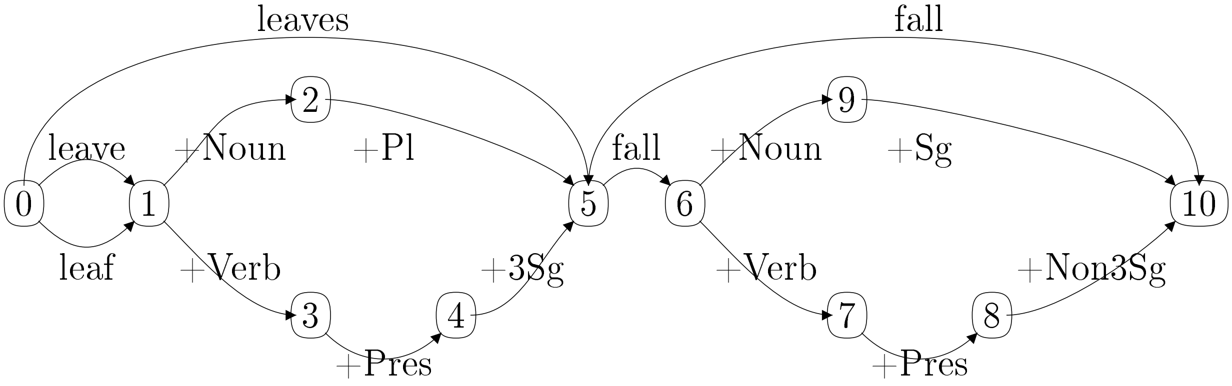
```
analyze-string leaves
analyzing {leaves}

leave +Verb +Pres +3sg
leave +Noun +Pl
leaf +Verb +Pres +3sg
leaf +Noun +Pl
leaves
```

What the morphological analyzers do

Ambiguity

- In most cases, a given word form has several possible analyses
  - enumerating them all as alternative inputs to the actual LFG parser would result in a combinatorial explosion
  - however, being the output of a finite-state transducer, the alternative analyses can be represented very compactly as a finite-state automaton (i.e., a transition graph)
  - XLE does not operate on an enumeration of the possible analyses, but directly on the transition graph:
    * each reading corresponds to a path through the transition graph
    * the chart that XLE works with is initialized with the graph produced by the tokenizer/morphology

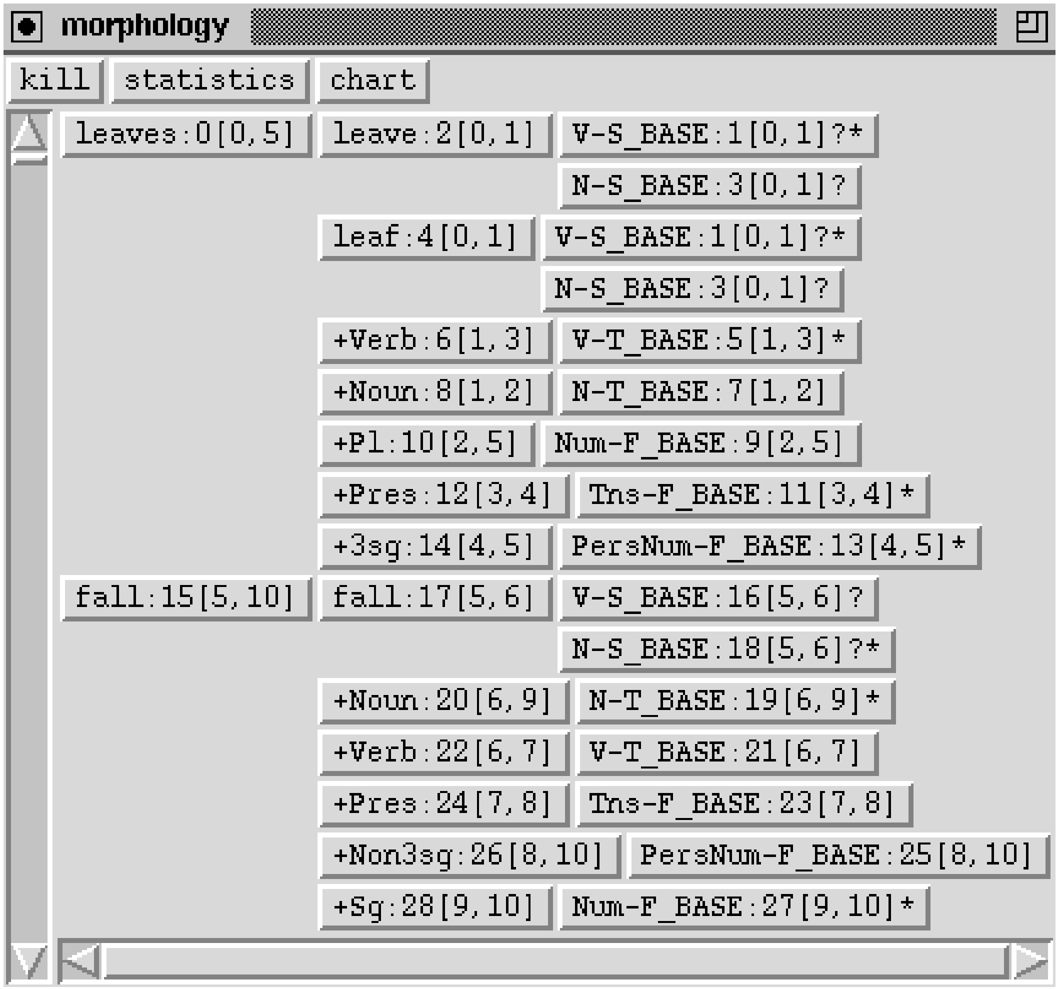
sentence	# analyses	# vertices in graph
leaves fall	15	11
he always leaves early	60	16
die Frau kennt ihn	208	35
das kleine Kind lachte	1,120	36
das kleine Kind kaufte ein	6,720	48
das Kind kaufte einen Ball	11,760	54

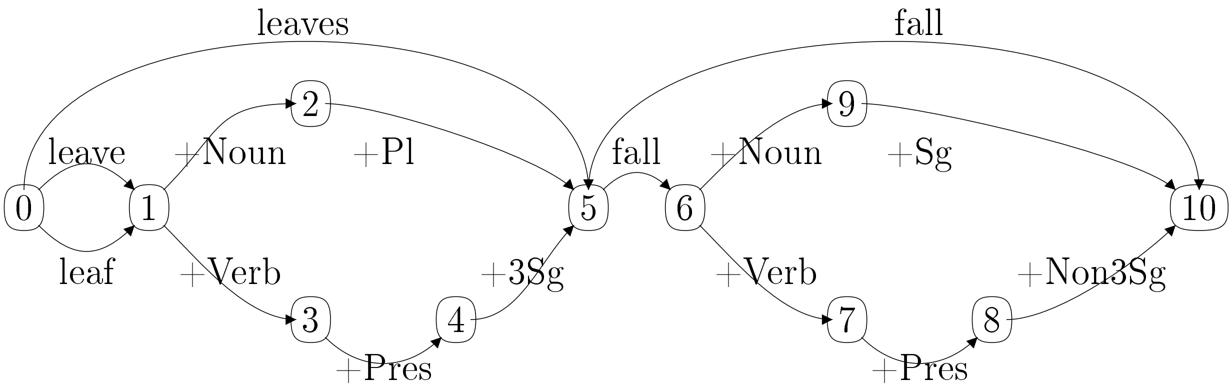



```
% analyze-string {leaves fall}
```

```
leave +Verb +Pres +3sg fall +Noun +Sg
leave +Verb +Pres +3sg fall +Verb +Pres +Non3sg
leave +Verb +Pres +3sg fall
leave +Noun +Pl fall +Noun +Sg
leave +Noun +Pl fall +Verb +Pres +Non3sg
leave +Noun +Pl fall
leaf +Verb +Pres +3sg fall +Noun +Sg
leaf +Verb +Pres +3sg fall +Verb +Pres +Non3sg
leaf +Verb +Pres +3sg fall
leaf +Noun +Pl fall +Noun +Sg
leaf +Noun +Pl fall +Verb +Pres +Non3sg
leaf +Noun +Pl fall
leaves fall +Noun +Sg
leaves fall +Verb +Pres +Non3sg
leaves fall
```

Transition graph in the XLE morphology display





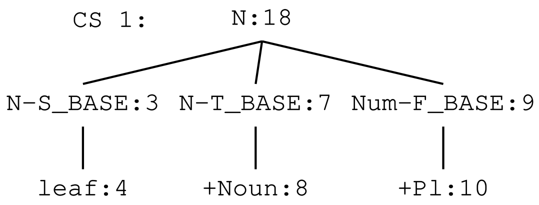
Building interface rules in the gramar

- With an underlying morphological analysis, the input for the LFG grammar is
  - no longer a *string of word forms*,
  - but a *transition graph of lemmata and morphological tags* (the output of the morphological analyzer)
- In the specification of the interface rules, we can ignore the graph character of the input:
  - every syntactic analysis has to be based on a single path through the transition graph
    `leave +Verb +Pres +3sg leaf +Noun +Pl`

Building interface rules in the gramar

- For the terminal symbols of a “normal” syntactic grammar (e.g., `N, V, DET`), so-called “sublexical” rules are specified

```
N  --> N-S_BASE
       N-T_BASE
       Num-F_BASE.
```



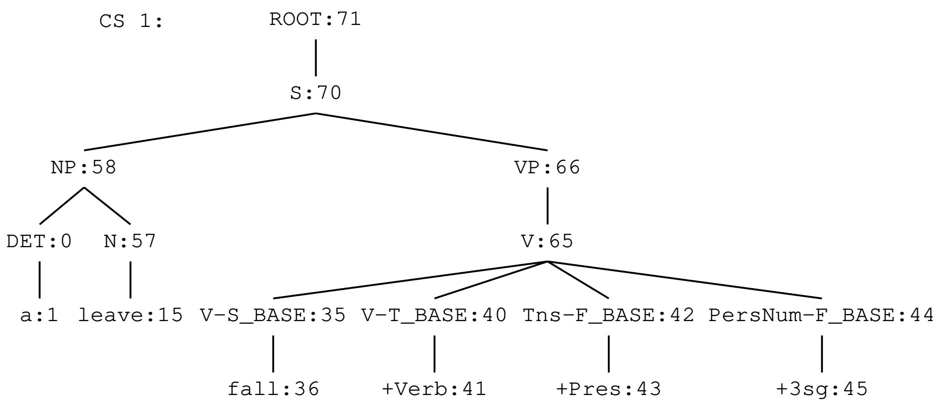
- The `_BASE` suffix for “sublexical categories” has to be used in order for XLE to know that this is sublexical
- For the lemmata and morphological tags, “sublexical lexicon entries” have to be specified (here, the `_BASE` suffix is *not* used)
  - for sublexical categories, the sequence ‘`XLE`’ replaces the ‘*’ as a separator
  - the relevant morphosyntactic information can be introduced in f-annotation as normal

```
leaf         N-S XLE    (^PRED)='leaf'.
+Noun        N-T XLE .
+Sg          Num-F  XLE (^NUM)=sg.

"old full form entry:
leaf         N * (^PRED)='leaf' (^NUM)=sg."
```

Building interface rules in the gramar

- Care should be taken in order to introduce morphosyntactic features in the right place in the f-structure
  - e.g., the person/number information given for verbs refers to the f-structure embedded under SUBJ
  - one way to capture this is by adding an annotation in the sublexical rule specification

```
"example:   falls     => fall +Verb +Pres +3sg"

 V  --> V-S_BASE
        V-T_BASE
        Tns-F_BASE
        PersNum-F_BASE:  (^ SUBJ)=!.

 +3sg     PersNum-F  XLE (^PERS)=3
                         (^NUM)=sg.
```

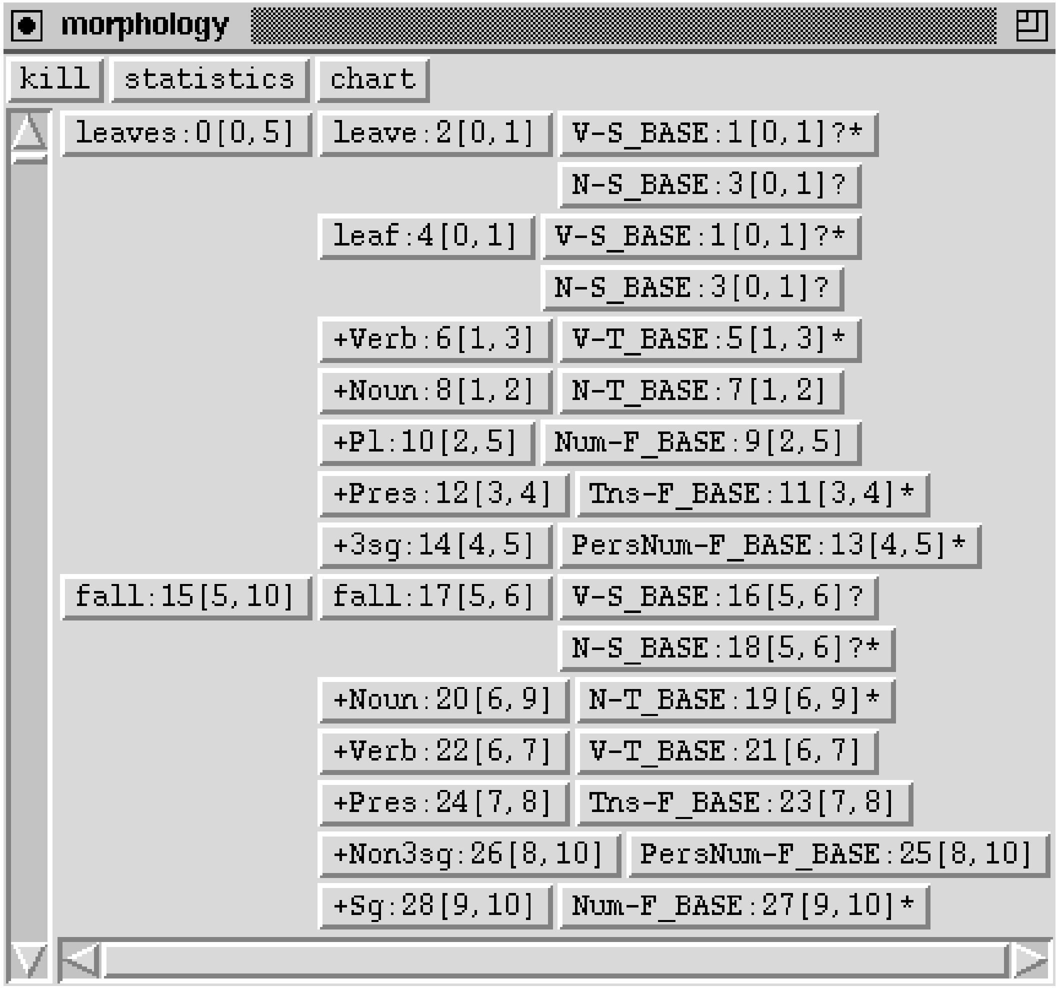


"a leave falls"

```
43[PRED   'fall<[0:leave]>'                 ]
42[                                          ]
41[     45[PRED 'leave'     ]                ]
40[     44[PERS 3, NUM sg   ]                ]
36[SUBJ 57[                 ]                ]
35[      1[                 ]                ]
65[      0[                 ]                ]
66[     58[                 ]                ]
70[VFORM fin, TENSE pres                     ]
71[                                          ]
```

Building interface rules in the gramar

The “sublexical categories” are also shown in the XLE morphology display (rightmost column):



Building interface rules in the gramar

Defaults for the lemma entries

- Specifying the lexicon entries for the lemma forms for all words in the morphology would still mean an enormous amount of hand-coding

  ```
  leaf          N-S XLE    (^PRED)='leaf'.
  ```

- For most word classes, all entries will look exactly the same, apart from the specification of the PRED value

- XLE provides a mechanism to deal with lemma forms, for which no explicit lexicon entry exists:

```
-Lunknown    N-S XLE    (^PRED)='%stem';
             V-S XLE    (^PRED)='%stem<(^SUBJ)>'.
```

- `-Lunknown` takes up any string of lower-case letters
- `-LUnknown` takes up any string with a capital initial (this is e.g. required for the lemma form of proper names or nouns in German)

- the identifier `%stem` can be generally used to refer to the string that matches the lexicon entry

Defaults for the lemma entries

```
"example:  leaves      => leaf +Noun +Pl"

-Lunknown    N-S XLE    (^PRED)='%stem';
             V-S XLE    (^PRED)='%stem<(^SUBJ)>'.
+Noun        N-T XLE .
+Sg          Num-F  XLE (^NUM)=sg.
```

- The `-Lunknown` overgenerates, but it is restricted by the immediate context (e.g., the following category tag)

- Subcategorization information is not normally given in the output of morphological analyzers
  - for word classes with differing subcategorization behaviour, like verbs, the default mechanism for lemma entries cannot be used for all cases
  - explicit lexica for these lemma forms and their subcategorization are required
  - machine-readable dictionaries can be exploited, but usually additional lists are required (e.g., for special domains)
  - to a considerable extent, such lists can be acquired automatically from text corpora (cf. Eckle and Heid 1996; Eckle 1998, and Kuhn, Eckle, and Rohrer 1998, which is in the reader)

References

Eckle, J. (1998). Methods for quality assurance in semi-automatic lexicon acquisition from corpora. In *Proceedings of EURALEX*.

Eckle, J. and U. Heid (1996). Extracting raw material for a german subcategorization lexicon from newspaper text. In *Proceedings of the 4th International Conference on Computational Lexicography, COMPLEX '96*, Budapest.

Kaplan, R. and M. Kay (1994). Regular models of phonological rule systems. *Computational Linguistics 20*(3), 331–378.

Karttunen, L., R. Kaplan, and A. Zaenen (1992). Two-level morphology with composition. In *COLING-92*, pp. 141–148.

Koskenniemi, K. (1983). *Two-level morphology: A general computational model for word-form recognition and production*. Ph. D. thesis, University of Helsinki.

Kuhn, J., J. Eckle, and C. Rohrer (1998). Lexicon acquisition with and for symbolic NLP-systems – a bootstrapping approach. In *Proceedings of the First International Conference on Language Resources and Evaluation (LREC98)*, Granada, Spain, pp. 89–95.