SMOR: A German Computational Morphology Covering Derivation, Composition and Inflection

Helmut Schmid, Arne Fitschen, Ulrich Heid

1 Introduction

We present a morphological analyser for German inflection and word formation implemented in finite state technology. Unlike purely lexicon-based approaches, it can account for productive word formation. The implementation is based on the Stuttgart Finite State Transducer Tools (SFST-Tools), a non-commercial FST platform.

German word formation morphology is characterized by a considerable number of productive compounding and derivation processes (both suffixation and prefixation). A morphological analyser for real text (e.g. newspapers) must be able to cope with complex words not contained as such in its lexicon, as they are built ad hoc according to productive word formation rules. Examples include particle and prefix verbs (*hinein-quietschen, ent-eisenen*), suffixation, e.g. with *-bar* (*enteisenbar*) and certain types of compounding, such as noun+noun (*Enteisenungsapparat*).

Existing (FST-based) morphology systems for German are either mainly based on large lexicons (cf. the public version of WordManager [Domenig and Hsiung, 1996] and GerTWOL [Haapalainen and Majorin, 1995]), or they cover only parts of German word formation (cf. DMOR [Schiller, 1996] which lacks a derivation component, and DeKo [Schmid et al., 2001, Heid et al., 2002] which lacks inflection). With the exception of an experimental system described in [Lorenz, 1997], we are thus not aware of any other computational morphology for German covering productive word formation and inflection in one system.

2 SMOR

SMOR is designed for the morphological analysis of German word forms. For the input in (1), it produces analyses consisting of a sequence of morphemes with feature decorations, as shown in (2):

(1) *unübersetzbarstes* (most untranslatable)

(2) un<PREF>übersetzen<V>bar<SUFF><+ADJ><Sup><Neut><Nom><Sg>
un<PREF>übersetzen<V>bar<SUFF><+ADJ><Sup><Neut><Akk><Sg>

The linguistic modelling principles behind SMOR are the following:

- SMOR implements a concatentative approach to morphology, postulating that affixes have their own lexical entries which encode selection constraints. Affixes select their base in terms of word class, stem type, origin and complexity (cmp. [Lüdeling and Fitschen, 2002]).

- Affixation is implemented as concatenation with feature checking; features encode the properties of bases and the selection constraints of affixes; this applies to suffixation and prefixing.
- Morphophonological rules are implemented using replacement operations (similar to two-level rules) which map analysis strings to surface strings (and vice versa).
- Inflection is handled via continuation classes, as in two-level morphology.
- The lexicon plays a central role in SMOR. It encodes the properties of bases with respect to
  - Type of lexicon entry: <BaseStem> <DerivStem> <CompStem> <Suffix> <Prefix>
  - Word class: <V> <ADJ> <NN> ...
  - Stem type: <base> <deriv> <compound>
  - Origin: <native> <foreign> <classical> ...
  - Complexity: <simplex> <prefderiv> <suffderiv>
  - Inflectional class: <Adj> <Adj+> ...

  The affix lexicon encodes the respective selectional constraints in affixes. In (3), we represent a lexicon entry for *-bar*

  (3) <Suffix><simplex><native><deriv><V>bar<ADJ><SUFF><base><native><Adj+>

  To minimize the need for morphophonological rules and in line with the concatenative approach, the lexicon contains different types of stems (base, deriv, compound).

3 Implementation

The implementation of SMOR was written in the SFST transducer specification language which is based on extended regular expressions with variables and operators for concatenation, conjunction, disjunction, repetition, composition, negation, and replacement. A compiler translates transducer specifications to minimised finite state transducers to be used by the analyser.

The basic operations of the SMOR implementation are concatenation of morphemes, filtering of morpheme sequences (by checking feature agreement), and mapping of the resulting analysis strings to surface realisations (by applying phonological rules).

The SMOR transducer is created incrementally. Stems, prefixes and affixes are listed in the lexicon. Derived forms are generated by adding suffixes to stems. The resulting transducer is composed with a filter transducer to check and delete the suffix agreement features. We call the result S_0. Prefix derivations are generated by adding prefixes in front of S_0 and checking features agreement, resulting in a transducer P_1. Further suffixes are added to P_1 with feature checking to obtain S_1. The disjunction of S_0 and S_1 forms the set of simplex and derived stems.

These stems are concatenated to form compounds. A filter ensures that all but the last stem are compound stems. Inflectional endings are generated by a strategy similar to the continuation classes in two-level morphology and added to the compounds. A filter eliminates incorrect endings. Finally, phonological rules are applied to map analysis strings to surface forms.

We will now present more implementational details. The command `$LEX$ = "lexicon"` reads in the lexicon from the file *lexicon*, generates a transducer which recognises each lexicon entry and assigns it to the variable LEX. The lexicon is split into sublexica. The command `$Prefix$ = $LEX$ || <Prefix> .*` e.g. extracts prefixes. We add suffixes to the stems and compose the result with a suffix filter transducer:

```
$S0$ = $Stems$ $SimplexSuffix$? $SuffDerivSuffix$* || $SuffixFilter$
```

The filter is implemented as a cascade of mappings examining the origin, stem type, and category features. The complexity feature is treated differently. Suffixes are split into subsets according to their complexity feature and then concatenated in the right order as shown above. The next step adds prefixes with feature checking:

```
$P1$ = $Prefix$ $S0$ || $PrefixFilter$
```

Only one prefix is allowed because the compilation of multiple prefixes turned out to be intractable. The reason is that the compared features are separated by arbitrarily many suffixes. During analysis, the transducer must remember the prefix features in its state while analysing the intervening material until it reaches the matching suffix. With multiple prefixes, the number of states explodes. A theoretical justification for the proposed limitation is given in [Erben, 2000] where it is argued that in apparent counterexamples like *unübersetzbar*, the second prefix (*über*) is actually part of a listed word stem (*übersetz*).

Further suffixes are added to the prefix derivations:

```
$S1$ = $P1$ $PrefDerivSuffix$? $SuffDerivSuffix$* || $SuffixFilter$
```

Compounds are formed by concatenating (derived) word stems and checking that all but the last stem are compound stems.

```
$Compounds$ = ($S0$ | $S1$)+ || $CompoundFilter$
```

Finally, inflection is added and phonological rules are applied:

```
$Base$ = $Compounds$ $Inflection$ || $InflectionFilter$ || $PhonRules$
```

The filter eliminates incorrect endings. Individual phonological rules are combined with composition rather than conjunction to simplify development by reducing rule interference.

The whole source code (without lexicon) comprises about 1500 lines (without comments).

4 Summary

We described the development of a German morphological analyser covering prefix and suffix derivation, compounding, and inflection. The analyser is fast and achieves good coverage. The final paper will contain details about lexicon size, coverage and speed. A demonstration is envisaged.

References

[Domenig and Hsiung, 1996] Domenig, M. and Hsiung, A. (1996). Concepts and tools for lexical knowledge acquisition. *AI communications*, 9(2):79–82.

[Erben, 2000] Erben, J. (2000). *Einführung in die deutsche Wortbildungslehre*. Erich Schmidt Verlag, Berlin, Germany, 4 edition.

[Haapalainen and Majorin, 1995] Haapalainen, M. and Majorin, A. (1995). Gertwol: Ein System zur automatischen Wortformerkennung deutscher Wörter. Technical report, Lingsoft Inc.

[Heid et al., 2002] Heid, U., Säuberlich, B., and Fitschen, A. (2002). Using descriptive generalisations in the acquisition of lexical data for word formation. In *Proceedings of the 3rd Conference on Language Resources and Evaluation*, volume IV, pages 86–92, Las Palmas de Gran Canaria, Spain.

[Lüdeling and Fitschen, 2002] Lüdeling, A. and Fitschen, A. (2002). An integrated lexicon for the automatic analysis of complex words. In *Proceedings of the Tenth EURALEX International Congress*, volume I, pages 145–152, Copenhagen, Denmark.

[Lorenz, 1997] Lorenz, O. (1997). Automatische Wortformerkennung für das Deutsche im Rahmen von MALAGA. Master's thesis, Friedrich-Alexander-Universität, Erlangen, Germany.

[Schiller, 1996] Schiller, A. (1996). Deutsche Flexions- und Kompositionsmorphologie mit PC-KIMMO. In Hausser, R., editor, *Proceedings, 1. Morpholympics, Erlangen, 7./8. März 1994*, Tübingen. Niemeyer.

[Schmid et al., 2001] Schmid, T., Lüdeling, A., Säuberlich, B., Heid, U., and Möbius, B. (2001). DeKo: Ein System zur Analyse komplexer Wörter. In *GLDV - Jahrestagung 2001*, pages 49–57.