New Methods in Data-Based
Computational Linguistics

Winter Term 2002/03

Lexicon-Based Annotation of Complements

Frank Henrik Müller

fhm@sfs.uni-tuebingen.de

SfS Tübingen,

12th November 2002

Lexicon-Based Annotation of Complements – Overview of the talk

- Motivation
- Conversion of Syntactic Frames
- Annotation of Complements
- Problems and Sources of Errors
- Future Work

Motivation

Why lexicon-based annotation of complements?

Unlike chunks, which are dominated by syntactic restrictions, SC frames are subject to lexical selection, i.e. they depend on the lemma (e.g. the verb).

Example (tags taken from STTS):

```
APPR ART ADV ADJA NN --> PP
NP VVFIN NP PP --> ???
```

- Consequences for NLP:
  - annotation of complements impossible without lexical information
  - need for a machine-readable lexicon containing (in the ideal case) all relevant lemmas

Conversion of Syntactic Frames

In order to apply a syntactic lexicon for parsing, lexical entries have to be translated into rules of a format which can be used by the respective formalism (here: finite state formalism).

Automization of translation because of:

- updates of lexicon
- changes in grammar

Conversion of Syntactic Frames

- It is possible to apply a syntactic lexicon for parsing using a finite state approach.
  - lexical rules (like passivization) are non-recursive and thus transferrable into a finite state automaton;
  - the transfer of lexical entries themselves is also possible using a finite state automaton
  - prestructured data (topological fields, chunks and clauses) allows faster and more accurate annotation of syntactic functions

Conversion of Syntactic Frames

Two sample entries (TSNLP format) from the IMSLex syntactic lexicon:

```
ab#geben haben (subj(NP_nom),arg(PRON_refl-acc),p-obj(PP_mit_dat)) HGC
ab#geben haben (subj(NP_nom),arg(PRON_refl-acc),p-obj(PP_mit_dat)) SADAW
ab#geben haben (subj(NP_nom),obj(NP_acc)) HGC
ab#geben haben (subj(NP_nom),obj(NP_acc)) SADAW
ab#geben haben (subj(NP_nom),obj(NP_acc),p-obj(PP_an_acc)) SADAW
ab#geben haben (subj(NP_nom),obj(NP_acc),p-obj(PP_für_acc)) HGC
ab#geben haben (subj(NP_nom),obj(NP_acc),p-obj(PP_zu_dat)) HGC
ab#geben haben (subj(NP_nom)) HGC

bangen haben (subj(NP_nom),arg(PRON_refl-acc)) SADAW
bangen haben (subj(NP_nom),arg(PRON_refl-acc),p-obj(PP_um_acc)) SADAW
bangen haben (subj(NP_nom),arg(NP_acc)) SADAW
bangen haben (subj(NP_nom),corr-pobj(PAV_um_acc),p-obj(C_decl-vsec)) HGC
bangen haben (subj(NP_nom),p-obj(PP_um_acc)) HGC
bangen haben (subj(NP_nom),s-comp(C_daß)) RULE-SEM
bangen haben (subj(NP_nom),s-comp(C_decl-vsec)) HGC
bangen haben (subj(NP_nom),s-comp(C_ob)) HGC
bangen haben (subj(NP_nom),s-comp(C_wh)) RULE-SEM
bangen haben (subj(NP_nom),s-comp(C_wh/ob)) HGC
bangen haben (subj(NP_nom),v-comp(VP_zu-inf-pres)) RULE-SEM
```


Conversion of Syntactic Frames

An initial idea for the conversion of lexical entries was to convert each entry into a single rule file.

First step: annotation of phrasal complements in subclauses.

Rules are converted
1. into plain text rules files
2. and then into XML format for *fsgmatch*

More than 2000 rule files are generated; process of conversion takes four minutes.

Conversion of Syntactic Frames

An instantiation of one syntactic frame:

```
Verb: ab#geben
Prep: an
CF_rel_nom_W OA_pron_W OPP_an_W VCR_ab#geben_T
CF_rel_acc_W ON_pron_W OPP_an_W VCR_ab#geben_T
CF_rel_nom_W OA_W OPP_an_W VCR_ab#geben_T
CF_rel_acc_W ON_W OPP_an_W VCR_ab#geben_T
CF_rel_an_W ON_pron_W OA_pron_W VCR_ab#geben_T
CF_rel_an_W ON_pron_W OA_W VCR_ab#geben_T
CF_rel_an_W ON_W OA_pron_W VCR_ab#geben_T
CF_rel_an_W ON_W OA_W VCR_ab#geben_T
CF_rel_nom_W FOPP_von_W OPP_an_W VCR_pass_ab#geben_T
CF_rel_nom_W OPP_an_W VCR_pass_ab#geben_T
CF_rel_an_W OA_W FOPP_von_W VCR_pass_ab#geben_T
CF_rel_an_W OA_W VCR_pass_ab#geben_T
CF_rel_an_W OA_pron_W FOPP_von_W VCR_pass_ab#geben_T
CF_rel_fopp_von_W ON_pron_W OPP_an_W VCR_pass_ab#geben_T
CF_rel_fopp_von_W ON_W OPP_an_W VCR_pass_ab#geben_T
CF_T ON_pron_W OA_pron_W OPP_an_W VCR_ab#geben_T
CF_T ON_pron_W OA_W OPP_an_W VCR_ab#geben_T
CF_T OA_pron_W ON_W OPP_an_W VCR_ab#geben_T
CF_T ON_W OA_W OPP_an_W VCR_ab#geben_T
CF_T ON_W OPP_an_W OA_W VCR_ab#geben_T
CF_T ON_pron_W OPP_an_W OA_W VCR_ab#geben_T
CF_T ON_pron_W FOPP_von_W OPP_an_W VCR_pass_ab#geben_T
CF_T ON_pron_W OPP_an_W FOPP_von_W VCR_pass_ab#geben_T
CF_T ON_pron_W OPP_an_W VCR_pass_ab#geben_T
CF_T ON_W FOPP_von_W OPP_an_W VCR_pass_ab#geben_T
CF_T ON_W OPP_an_W VCR_pass_ab#geben_T
```

Conversion of Syntactic Frames

One rule file per verb:

- advantages
  - grammar close to lexicon
  - very transparent grammar
- disadvantages
  - extremely slow (more than three hours for 1,000 sentences)
  - little generalization (approx. 90,000 rules for verbs)

Conversion of Syntactic Frames

The new conversion of rules generalizes over the different syntactic frames.

Just one rule file for all syntactic frames (including adjectives now).

- disadvantages
  - lexicon has to be used during parsing process
  - rules are no longer as transparent as they used to be
- advantages
  - higher degree of generalization (approx. 1,000 rules for verbs and adjectives)
  - a lot faster (less than three minutes for 1,000 sentences)

Conversion of Syntactic Frames

Generalized syntactic frame for verb with ON, OA and OPP (with 'an').

```
CF_rel_ON_W OA_pron_W OPP_an_W VCR_akt_ONOAOPP_an_T
CF_rel_OA_W ON_pron_W OPP_an_W VCR_akt_ONOAOPP_an_T
CF_rel_ON_W OA_W OPP_an_W VCR_akt_ONOAOPP_an_T
CF_rel_OA_W ON_W OPP_an_W VCR_akt_ONOAOPP_an_T
CF_rel_OPP_an_W ON_pron_W OA_pron_W VCR_akt_ONOAOPP_an_T
CF_rel_OPP_an_W ON_pron_W OA_W VCR_akt_ONOAOPP_an_T
CF_rel_OPP_an_W ON_W OA_pron_W VCR_akt_ONOAOPP_an_T
CF_rel_OPP_an_W ON_W OA_W VCR_akt_ONOAOPP_an_T
CF_rel_ON_W FOPP_von_W OPP_an_W VCR_pass_ONOAOPP_an_T
CF_rel_ON_W OPP_an_W VCR_pass_ONOAOPP_an_T
CF_rel_OPP_an_W OA_W FOPP_von_W VCR_pass_ONOAOPP_an_T
CF_rel_OPP_an_W OA_W VCR_pass_ONOAOPP_an_T
CF_rel_OPP_an_W OA_pron_W FOPP_von_W VCR_pass_ONOAOPP_an_T
CF_rel_FOPP_von_W ON_pron_W OPP_an_W VCR_pass_ONOAOPP_an_T
CF_rel_FOPP_von_W ON_W OPP_an_W VCR_pass_ONOAOPP_an_T
CF_T ON_pron_W OA_pron_W OPP_an_W VCR_akt_ONOAOPP_an_T
CF_T ON_pron_W OA_W OPP_an_W VCR_akt_ONOAOPP_an_T
CF_T OA_pron_W ON_W OPP_an_W VCR_akt_ONOAOPP_an_T
CF_T ON_W OA_W OPP_an_W VCR_akt_ONOAOPP_an_T
CF_T ON_W OPP_an_W OA_W VCR_akt_ONOAOPP_an_T
CF_T ON_pron_W OPP_an_W OA_W VCR_akt_ONOAOPP_an_T
CF_T ON_pron_W FOPP_von_W OPP_an_W VCR_pass_ONOAOPP_an_T
CF_T ON_pron_W OPP_an_W FOPP_von_W VCR_pass_ONOAOPP_an_T
CF_T ON_pron_W OPP_an_W VCR_pass_ONOAOPP_an_T
CF_T ON_W FOPP_von_W OPP_an_W VCR_pass_ONOAOPP_an_T
CF_T ON_W OPP_an_W VCR_pass_ONOAOPP_an_T
```


Conversion of Syntactic Frames

Extract from the syntactic lexicon in fsgmatch format.

```
ab#feuern :: ONOA ON
ab#finden :: ONOA_refl ONOA_reflOPP_mit ONOA ONOAOPP_mit
ab#geben :: ONOA_reflOPP_mit ONOA ONOAOPP_an ONOAOPP_für ONO
ab#gewinnen :: ONOA ONOAOD
ab#grenzen :: ONOA_refl ONOA_reflOPP_gegen ONOA_reflOPP_von
ab#hacken :: ONOA
ab#haken :: ONOA ONOAPREDalsADJ ONOAOPP_mit
ab#halten :: ONOA ONOAOPP_von
ab#handeln :: ONOA ONOAOD ONOAOPP_für ONOAOPP_in
ab#hängen :: ONOA ONOAOPP_von ONOPP_von
ab#hauen :: ONOA ONOAOPP_von ON
ab#holen :: ONOA
ab#knallen :: ONOA
ab#koppeln :: ONOA_refl ONOA_reflOPP_von ONOA
ab#laden :: ONOA ON
```

Annotation of Complements

Features of fsgmatch:

- *fsgmatch* can only wrap elements which are siblings

–> DEREKO format is converted into a format which allows *fsgmatch* to be used - XSL is used for this conversion

Annotation of Complements

```
(REL
    {CF
        [NC
            .PRELS   die ] }
    {MF
        [AVC
            .ADV     derzeit ]
        [PC
            .APPR    von
            [NC
                .ART     der
                .NN      Firma ] ]
        [NC
            .NE      Mosolf ]
        [PC
            .APPR    in
            [NC
                .NE      Bremerhaven ] ] }
    [VCRAFVP
        .VVPP    abgeladen
        .VAFIN   werden ] )
```


Annotation of Complements

```
(REL
    [ ON
        [NC  CF
            .PRELS   die ] ]
    [AVC  MF
        .ADV     derzeit ]
    [ FOPP
        [PC  MF
            .APPR    von
            [NC
                .ART     der
                .NN      Firma ] ] ]
    [NC  MF
        .NE      Mosolf ]
    [PC  MF
        .APPR    in
        [NC
            .NE      Bremerhaven ] ]
    [VCRAFVP
        .VVPP    abgeladen
        .VAFIN   werden ] )
```


Annotation of Complements

Generation of finite state grammar:

- conversion of IMSLex into fsgmatch format

- generation of plain text rule files

- conversion of fsgmatch rules (which are in XML format) from plain text rules

Annotation of Complements

Disambiguation of ambiguous structures is done by the following mechanisms.

- annotate as many complements as you can i.e. try syntactic frames with most complements first

- try most plausible word orders first (relies on intuition so far)

Annotation of Complements

Annotation order of syntactic frames:

```
ON OPP PRED(adj) gegen Pollen allergisch sein
ON OA PRED(adj)  das Wetter gewohnt sein
ON OD PRED(adj)  ihm auskunftspflichtig sein
ON PRED(adj)     sicher sein
ON OA OPP        ihn mit dem Bau beauftragen
ON OA(refl) OPP  sich mit einem Problem arrangieren
ON OA OD         ihm das Bild beschreiben
ON OA PRED(als_ADJ) ihn als ungeeignet bezeichnen
ON OA PRED(als_NN)  ihn als Bewerber akzeptieren
ON OD            dem Lehrer antworten
ON OPP           um den Freund bangen
ON OA            den Himmel sehen
ON OA (refl)     sich benehmen
ON PRED(als_NN)  als Fensterputzer arbeiten
ON               singen
```


Annotation of Complements

Results:

- 71 of 101 relative clauses get analysis = 70.3%

- 129 of 182 general subordinate clauses get analysis = 70.9%

- 200 of 283 all subclauses get analysis = 70.7%

- accuracy not checked, yet

Problems and Sources of Errors

with gold morphology

- wrong morphology

- no morphological analysis

with automatic morphology

- ambiguous morphology

- no morphological analysis (less than 3.5% of all tokens, including numbers, acronyms, etc.)

Problems and Sources of Errors

- syntactic frame not in lexicon
  –> accept as characteristic of the approach?
- headword not in lexicon
  –> find backup strategy
- named entities
  –> NE component
- coordination
  –> ???
- function verbs
  –> list of function verbs

Future Work

- integrate automatic morphology into system
- find back-up strategies for missing headwords/frames
- extend system to cover whole sentences
- extend system to cover sentential complements