Information Extraction and Question-Answering Systems

Foundations and methods

Dr. Günter Neumann
LT-Lab, DFKI
neumann@dfki.de

What the lecture will cover

Machine Learning for IE

Lexical processing

Evaluation Methods

Basic Terms & Examples

Parsing of Unrestricted Text

Generic NL Core system

Domain Modelling

Question/Answering Core components

Advanced Topics

Parsing of unrestricted text

- Complexity of parsing of unrestricted text
  - Robustness
  - Large sentences
  - Speed
  - Input texts are not simply sequences of word forms
    - Textual structure (e.g., enumeration, spacing, etc.)
    - Combined with structual annotation (e.g., SGML tags)

22/02/2002 3

The majority of current information extraction systems perform a partial parsing approach following a bottom-up strategy

Major steps

lexical processing

including morphological analysis, POS-tagging, Named Entity recognition

phrase recognition

general nominal & prepositional phrases, verb groups

clause recognition via domain-specific templates

templates triggered by domain-specific predicates attached to relevant verbs;

expressing domain-specific selectional restrictions for possible argument fillers

Bottom-up chunk parsing

perform clause recognition after phrase recognition is completed

22/02/2002 4

However a bottom-up strategy showed to be problematic in case of German free text processing

Crucial properties of German

highly ambiguous morphology (e.g., case for nouns, tense for verbs);

free word/phrase order;

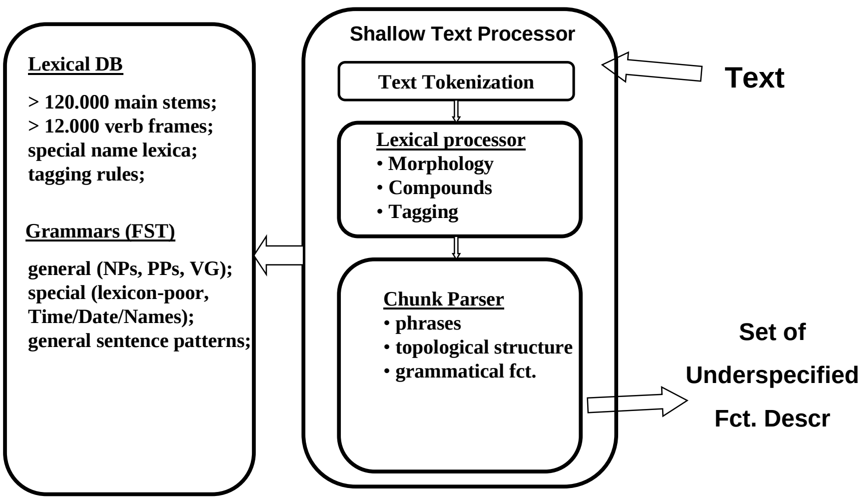
splitting of verb groups into separated parts into which arbitrary phrases and clauses can be spliced in (e.g., *Der Termin findet morgen statt. The date takes place tomorrow.)*

Main problem in case of a bottom-up parsing approach

even recognition of simple sentence structure depends heavily on performance of phrase recognition

[NPDie vom Bundesgerichtshof und den Wettbewerbern als Verstoß gegen das Kartellverbot gegeisselte zentrale TV-Vermarktung] ist gängige Praxis. [Central television marketing censured by the German Federal High Court and the guards against unfair competition as an infringement of anti-cartel legislation] is common practice.

A Robust Parser for unrestricted German Text

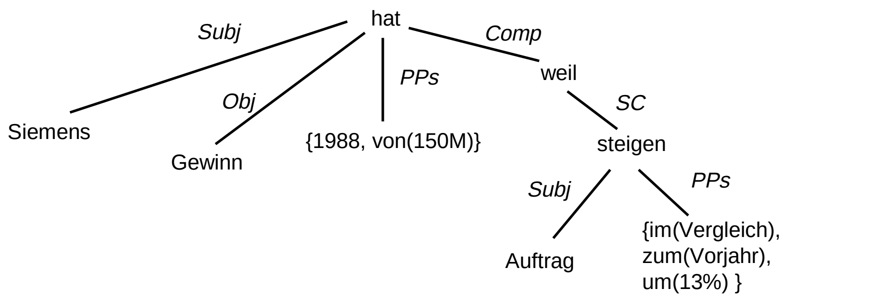


Underspecified (partial) functional descriptions UFDs

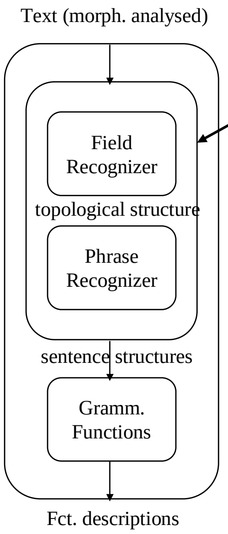
UFD: flat dependency-based structure, only upper bounds for attachment and scoping

[$_{PN}$Die Siemens GmbH] [$_{V}$hat] [$_{year}$1988][$_{NP}$einen Gewinn] [$_{PP}$von 150 Millionen DM],
[$_{Comp}$weil] [$_{NP}$die Auftraege] [$_{PP}$im Vergleich] [$_{PP}$zum Vorjahr] [$_{Card}$um 13%] [$_{V}$gestiegen sind].

"The siemens company has made a revenue of 150 million marks in 1988, since the orders increased by 13% compared to last year."



In order to overcome these problems we propose the following two phase divide-and-conquer strategy



Divide-and-conquer strategy

1. Recognize verb groups and topological structure (*fields*) of sentence domain-independently;

FrontField LeftVerb MiddleField RightVerb RestField

2. Apply general as well as domain-dependent phrasal grammars to the identified fields of the main and sub-clauses

[CoordS [CSent *Diese Angaben konnte der Bundesgrenzschutz aber nicht bestätigen*], [CSent *Kinkel sprach von Horrorzahlen,* [Relcl *denen er keinen Glauben schenke*]]].

This information couldn't be verified by the Border Police, Kinkel spoke of horrible figures that he didn't believe.

The divide-and-conquer approach offers several advantages

Improved robustness

- topological sentence structure determined on basis of simple indicators like verbgroups and conjunctions and their interplay;
- phrases need not be recognized completely

Resolution of some ambiguities

- relative pronouns vs. determiners
- subjunction vs. preposition
- clause vs. NP coordination

Modularity

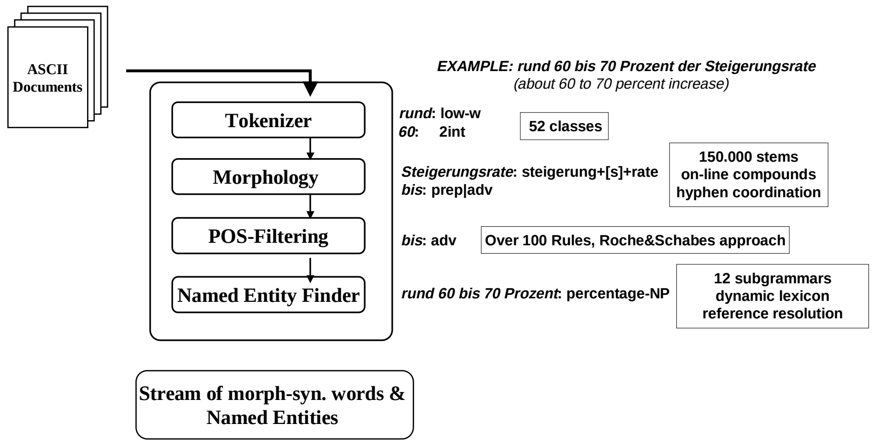
- easy exchange/extension of (domain-specific) phrase grammars

Some more examples (source text)

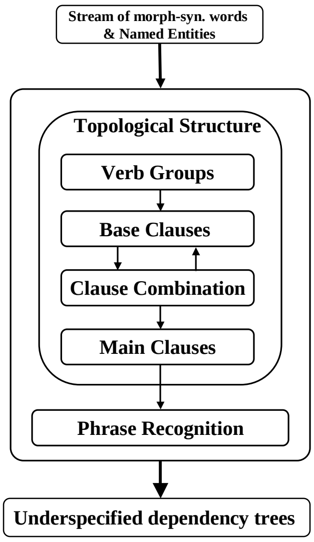
- topological structure
- plus expanded phrase structure

The divide-and-conquer parser benefits from a powerful lexical preprocessor

The lexical processor is realized on basis of state-of-the-art finite state technology, however taking care of German language specificities.



The divide-and-conquer parser is realized by means of a series of finite state grammars



Weil die Siemens GmbH, die vom Export lebt, Verluste erlitt, mußte sie Aktien verkaufen.
Because the Siemens Corp which strongly depends on exports suffered from losses they had to sell some shares.

Weil die Siemens GmbH, die vom Export Verb-FIN, Verluste Verb-FIN, Modv-FIN sie Aktien FV-Inf.

Weil die Siemens GmbH, Rel-Clause Verluste Verb-FIN, Modv-FIN sie Aktien FV-Inf.

Subconj-Clause,
Modv-FIN sie Aktien FV-Inf.

<u>Clause</u>

The Shallow Text Processor has several Important Characteristics

Modularity: each subcomponent can be used in isolation;
Declarativity: lexicon and grammar specification tools;
High coverage: more than 93 % lexical coverage of unseen text;
high degree of subgrammars
Efficiency: finite state technology in all components;
specialized constrained solvers
(e.g. agreement checks & grammatical functions);
Run-time: 4.5 msec real time per token (Standard PC environment)
Available for research:
http://www.dfki.de/~neumann/pd-smes/pd-smes.html

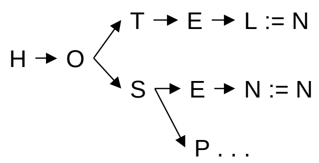
Morphological Processing

- Performed by the Morphix package http://www.dfki.de/~neumann/morphix/morphix.html

- Morphix performs:
  - ➢Inflectional analysis
  - ➢Compound analysis
  - ➢Generation of word forms

Dynamic tries as basic data structure for lexical data

- Dynamic tries (letter tries)
  - ➢ sole storage device for all sorts of lexical information
  - ➢ Robust specialized regular matcher
  - ➢ Dynamic memory allocation (based on access frequency and access time)



Basic processing strategy of Morphix

- Recursive trie traversal of lexicon
- Application of finite state automata for handling inflectional regularities
- Preprocessing
  - ➢ Each word form is fristly transformed into a set of tripples <prefix, lemma, suffix>
    - Prefix: (complex) verb prefix or GE-
    - Lemma: possible lexical stem, where possible umlauts are reduced (e.g., Mädchen vs. Häusern)
    - Suffix: longest matching inflection ending (using a inflection lexicon)

22/02/2002 15

Representation of results

- Set of tripple <stem, inflection, POS>
- Compound processing handles words with
  - ➢nominal root (*Häuserblock “block of houses”*)
  - ➢adjectival root (*tiefschwarz “deep black”*)
  - ➢verbal root (*blaugefärbt “blue colored”*)
- Compund processing
  - ➢a recursive trie traversal
  - ➢Identification of allowable infixes

22/02/2002 16

Flexible output interface

Compute DNF for the compactly represented disjunctive morpho-syntactic output. User can choose different forms of DNF representation:

disjunctive output for the form “die Häuser” (“*the houses*”)
("haus" (cat noun) (flexion ((ntr ((pl (nom gen acc)))))))

as symbol list (e.g., used in case of lexical tagging)
("haus" (ntr-pl-nom ntr-pl-gen ntr-pl-acc) . :n)
as feature term (e.g., used in case of shallow parsing)
("haus"
(((:tense . :no) (:person . :no) (:gender . :ntr) (:number . :pl) (:case . :nom))
((:tense . :no) (:person . :no) (:gender . :ntr) (:number . :pl) (:case . :gen))
((:tense . :no) (:person . :no) (:gender . :ntr) (:number . :pl) (:case . :acc)))
. :n)

Morphix comes with a very flexible output interface

- Finite set of possible morpho-syntatic output structures
  - ➢ DNF computation can be done off-line and on-line using memorization techniques
- User can select interactively subset from possible morpho-syntactic feature set {:cat :mact :sym :comp :comp-f :det :tense :form :person :gender :number :case}

e.g. ("haus"
(((:number . :pl) (:case . :nom))
((:number . :pl) (:case . :gen))
((:number . :pl) (:case . :acc)))
. :n)

  - ➢ supports lexical tagging (use of different tag sets)
  - ➢ supports feature relaxation (ignore uninteresting features)

Specialized Unifier

- Currently, constraints are mainly used to express morpho-syntactical agreement
- Feature checking performed by a simple but fast specialized unifier
  - Feature vector representation
  - Special symbol :no used as anonymous variable
  - Example

```
s1=(((:TENSE . :NO) (:FORM . :NO) (:NUMBER . :S) (:CASE . :N))
    ((:TENSE . :NO) (:FORM . :NO) (:NUMBER . :S) (:CASE . :A))
    ((:TENSE . :NO) (:FORM . :NO) (:NUMBER . :P) (:CASE . :N))
    ((:TENSE . :NO) (:FORM . :NO) (:NUMBER . :P) (:CASE . :A)))
s2=(((:TENSE . :NO) (:FORM . :XX) (:NUMBER . :S) (:CASE . :N))
    ((:TENSE . :NO) (:FORM . :NO) (:NUMBER . :S) (:CASE . :G))
    ((:TENSE . :NO) (:FORM . :NO) (:NUMBER . :S) (:CASE . :D)))
unify(s1,s2)=
   (((:TENSE . :NO) (:FORM . :XX) (:NUMBER . :S) (:CASE . :N)))
```


Writing grammars with SMES

- Finite state transducers FST
  <identifier, recognition part, output description, compiler options>
- Recognition part is a regular expression where alphabet is implicitly expressed via basic edges
  - Predicate or a specific class of tokens, e.g. (:morphix-cat *partikel pre*)
  - :morphix-cat is a predicate which checks whether the current token‘s POS equals *partikel*, and if so, bound the token to the variable *pre*

Example of simple NP rule

```
(:conc
  (star<=n (:morphix-cat det det) 1)
  (:star (:morphix-cat adj adj))
  (:morphix-cat n noun))
```

Thus defined, a nominal phrase is the concatenation of one optional determiner (expressed by the loop operator :star<=n, where n starts from 0 and ends by 1), followed by zero or more adjectives followed by a noun.

NP with feature vector unification

```
(compile-regexp
 '(:conc
   (:current-pos start)
   (:alt
    (:star<=n (:morphix-unify :indef  NIL agr det) 1)
    (:star<=n (:morphix-unify :def  NIL agr det) 1))
   (:star<=n (:morphix-unify :a agr agr adj) 1)
   (:morphix-unify :n  agr agr noun)
   (:current-pos end))
 :output-desc
 '(:lisp (build-item
            :type :np :start start :end end :agr agr
            :det det :adj adj :noun noun))
 :name 'small-np)
```

Special basic edge → (:current-pos start)

Empty feature vector → NIL

Output description (typed based) → '(:lisp (build-item

Phrase recognition

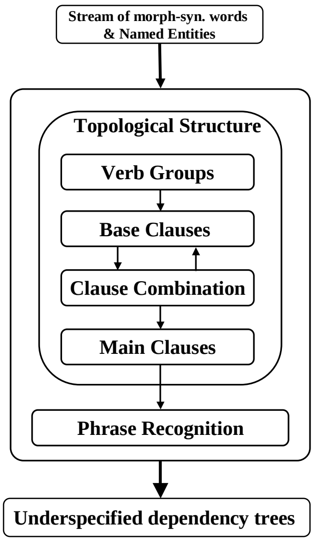
- Nominal phrases NP
  - *dem Fernrohr*
- Prepositional phrases PP
  - *mit dem Fernrohr*
- Verb groups VG
  - *glaubt mit dem Fernrohr sehen zu können*
- NE grammars
  - *Kanzler Schröder glaubt mit dem Fernrohr sehen zu können.*

Example

- Der Mann sieht die Frau mit dem Fernrohr.
  The man sees the woman with the telescope.

```
((:SEM (:HEAD "mann") (:QUANTIFIER "d-det"))
 (:AGR
  ((:TENSE . :NO) ... (:CASE . :NOM)))
 (:END . 2) (:START . 0) (:TYPE . :NP))
((:SEM (:HEAD "frau") (:QUANTIFIER "d-det"))
 (:AGR
  ((:TENSE . :NO) ... (:GENDER . :F) (:NUMBER . :S)
   (:CASE . :NOM))
  ((:TENSE . :NO) ... (:GENDER . :F) (:NUMBER . :S)
   (:CASE . :AKK)))
 (:END . 5) (:START . 3) (:TYPE . :NP))
((:SEM (:HEAD "mit")
      (:COMP (:QUANTIFIER "d-det") (:HEAD "fernrohr")))
 (:AGR
  ((:TENSE . :NO) ... (:GENDER . :NT) (:NUMBER . :S)
   (:CASE . :DAT)))
 (:END . 8) (:START . 5) (:TYPE . :PP)))
```

The divide-and-conquer parser is realized by means of a series of finite state grammars



Weil die Siemens GmbH, die vom Export lebt, Verluste erlitt, mußte sie Aktien verkaufen.
Because the Siemens Corp which strongly depends on exports suffered from losses they had to sell some shares.

Weil die Siemens GmbH, die vom Export Verb-FIN, Verluste Verb-FIN, Modv-FIN sie Aktien FV-Inf.

Weil die Siemens GmbH, Rel-Clause Verluste Verb-FIN, Modv-FIN sie Aktien FV-Inf.

Subconj-Clause,
Modv-FIN sie Aktien FV-Inf.

<u>Clause</u>

Verb grammar

- A verb grammar recognizes all
  - ➢ single occurrences of verbforms (in most cases corresponding to LeftVerb)
  - ➢ all closed verbgroups (in general RightVerb)
- Discontinuous verb groups (separated LeftVerb and RightVerb) are not put together
- Major problem here is not a structural one but the massive morphosyntactic ambiguity of verbs

Verb Grammars

- The verb rules solve most of these problems on the basis of feature value occurence (e.g., a rule is only triggered if the current verb form is finite).
- Feature checking is performed through unification.
- The different rules assign to each recognized expression its type for example on the basis of time and active/passive information (e.g., whether it is final, modal perfect active).

Example output

- nicht gelobt haben kann
  could not have been praised

Type	VG-final
Subtype	Mod-Perf-Ak
Modal-stem	Koenn
Stem	Lob
Form	nicht gelobt haben kann
Neg	T
Agree	...

Base clauses

- Subclauses of type
  - Subjunctive (e.g., als, als ob, soweit, ...)
  - Subordinate (e.g., relative clauses)
- Simply be recognized on the basis
  - Commas
  - initial elements (like complementizer)
  - interrogative or relative item
- The different types of subclauses are described very compactly as finite state expressions

22/02/2002 29

Snapshot of Base clause grammar

Base-clause ::=
 Inf-Cl | Subj-Cl | w-Cl | Rel-Cl | Parenthese

Sub-Cl ::=
 (, | Cl-Beg){funct-word} Subjunctor verb-final-cl

Subjunktor ::= als | als dass | sooft | ...

Verb-final-cl ::= ...

22/02/2002 30

In order to deal with embedded clauses, two sorts of recursions are identified

Middle-field recursion

embedded base clause is located in the middle field of the embedding sentence

…, weil die Firma, nachdem sie expandiert hatte, größere Kosten hatte.

(**..., because the company, after it expanded had, increased costs had.*)

➽ …, weil die Firma [Subclause], größere Kosten hatte.

➽ … [Subclause].

Rest-field recursion

embedded clause follows the right verb part of the embedding sentence

…, weil die Firma größere Kosten hatte, nachdem sie expandiert hatte.

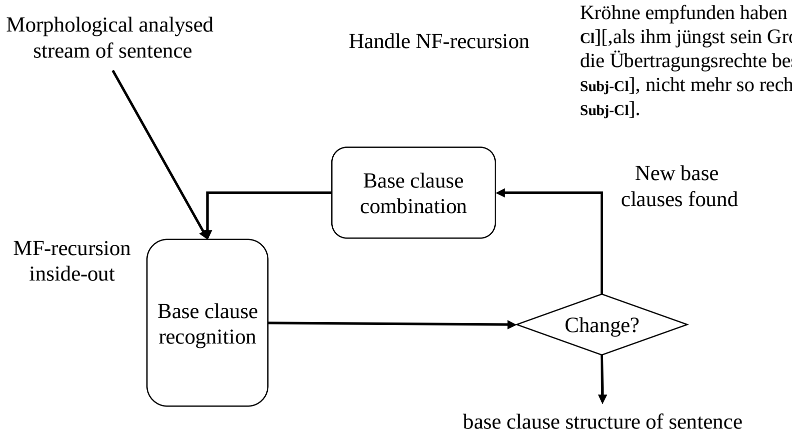
(**..., because the company increased costs had, after it expanded had.*)

➽ … [Subclause] [Subclause].

➽ … [Subclause].

These recursions are treated as iterations which destructively substitute recognized embedded base clauses with their type

...*[daß das Glück [, das Jochen Kröhne empfunden haben soll **Rel-Cl**][,als ihm jüngst sein Großaktionär die Übertragungsrechte bescherte **Subj-Cl**], nicht mehr so recht erwärmt **Subj-Cl**].



Main clauses

- Builds the complete topological structure of the input sentence on the basis of
  - ➢recognized (remaining) verb groups
  - ➢base clauses
  - ➢word form information (punctuations and coordinations)

Main clause grammar

Csent	::=	... LVP ... [RVP] ...
Ssent	::=	LVP [RVP] ...
CoordS	::=	CSent (, CSent)* Coord CSent | CSent (, SSent)* Coord SSent
AsyndSent	::=	CSent {,} CSent
ComplexCSent	:: =	CSent {,} SSent | CSent , CSent
AsyndCond	::=	SSent {,} SSent

Evaluation on unseen test data (press releases)

Lexical pre-processor (20.000 tokens)

	Recall **%**	Precision %	
compound analysis	99.01	99.29	
part-of-speech-filtering	74.50	97.90	
named entity (incl. dynamic lexicon)	85.00	95.77	
fragments (NPs, PPs):	76.11	91.94	

Divide-and-conquer parser (400 sentences, 6306 words)

verb module	98.10	98.43	
base-clause module	93.08 (94.61)	93.80 (93.89)	
main-clause module	89.00 (93.00)	94.42 (95.62)	
complete analysis	84.75	89.68	F=87.14

22/02/2002 35

Preliminary summary

Divide-and-conquer parsing strategy

- free German text processing
- suited for free worder languages
- high modularity

Main experience

- full text processing necessary even if only some parts of a text are of interest;
- application-oriented depth of text understanding;
- the difference between shallow and deep NLP seen as a continuum

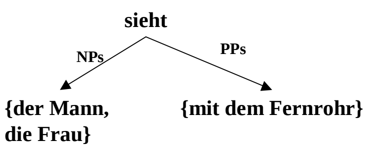
22/02/2002 36

Underspecified dependency tree

- After topological parsing, the phrase grammars are applied to the elements of the identified fields
- Then an underspecified dependency tree is computed by collecting
  - ➢ the elements from the verb groups which define the head of the tree
  - ➢ all NPs directly governed by the head into a set NP modifiers
  - ➢ all PPs directly governed by the head into a set PP modifiers
- This process is recursively applied to all embedded clauses
- The resulting structure is underspecified because only upper bounds for attachment are defined

Example dependency tree

Der Mann sieht die Frau
mit dem Fernrohr.




```
(((:PPS
  ((:SEM (:HEAD "mit")
       (:COMP (:QUANTIFIER "d-det") (:HEAD "fernrohr")))
   (:AGR
    ((:TENSE . :NO) ... (:CASE . :DAT)))
   (:END . 8) (:START . 5) (:TYPE . :PP)))
 (:NPS
  ((:SEM (:HEAD "mann") (:QUANTIFIER "d-det"))
   (:AGR
    ((:TENSE . :NO) ... (:CASE . :NOM)))
   (:END . 2) (:START . 0) (:TYPE . :NP))
  ((:SEM (:HEAD "frau") (:QUANTIFIER "d-det"))
   (:AGR
    ((:TENSE . :NO) ... (:CASE . :NOM))
    ((:TENSE . :NO) ... (:CASE . :AKK)))
   (:END . 5) (:START . 3) (:TYPE . :NP)))
 (:VERB
  (:COMPACT-MORPH
   ((:TEMPUS . :PRAES) ... (:PERSON . 3)
    (:GENUS . :AKTIV)))
  (:MORPH-INFO
   ((:TENSE . :PRES) (:FORM . :FIN) ... (:CASE . :NO)))
  (:ART . :FIN) (:STEM . "seh")
  (:FORM . "sieht") (:C-END . 3) (:C-START . 2)
  (:TYPE . :VERBCOMPLEX))
 (:END . 8) (:START . 0) (:TYPE . :VERB-NODE)))
```

Grammatical function recognition GFR

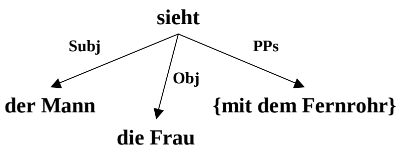
- In the final step of parsing process, the grammatical functions are determined for all subtrees of the dependency tree
- Main knowledge source is a huge subcategorization lexicon for verb
- During a recursive traversal of the dependency tree the longest matching subcat frame is checked to identify the head and modifier elements

Main steps of GFR

- Identification of possible *arguments* on the basis of the lexical subcategorization information available for the local head (the verb group)
- Marking of the other non-head elements of the dependence tree as *adjuncts*, possibly by applying a distinctive criterion for standard and specialized adjuncts.
- Adjuncts - opposed to arguments, for which an attachment resolution is attempted - have to be considered underspecified wrt. attachment, even after GFR
  - in other words, their dependency relation to the head counts as an *upper border* rather than an attachment

Example of GFR output

Der Mann sieht die Frau
mit dem Fernrohr.




```
(((:SYN
   (:SUBJ
    (:RANGE (:SEM (:HEAD "mann") (:QUANTIFIER "d-det"))
     (:AGR
      ((:PERSON . 3) (:GENDER . :M)
       (:NUMBER . :S) (:CASE . :NOM)))
     (:END . 2) (:START . 0) (:TYPE . :NP)))
   (:OBJ
    (:RANGE (:SEM (:HEAD "frau") (:QUANTIFIER "d-det"))
     (:AGR
      ((:PERSON . 3) (:GENDER . :F)
       (:NUMBER . :S) (:CASE . :NOM))
      ((:PERSON . 3) (:GENDER . :F)
       (:NUMBER . :S) (:CASE . :AKK)))
     (:END . 5) (:START . 3) (:TYPE . :NP)))
   (:NP-MODS)
   (:PP-MODS
    ((:SEM (:HEAD "mit")
           (:COMP (:QUANTIFIER "d-det") (:HEAD "fernrohr")))
     (:AGR ((:PERSON . 3) (:GENDER . :NT)
            (:NUMBER . :S) (:CASE . :DAT)))
     (:END . 8) (:START . 5) (:TYPE . :PP)))
   (:PROCESS
    (:COMPACT-MORPH
     ((:TEMPUS . :PRAES) ... (:GENUS . :AKTIV)))
    (:MORPH-INFO
     ((:TENSE . :PRES) ... (:NUMBER . :S) (:CASE . :NO)))
    (:ART . :FIN) (:STEM . "seh") (:FORM . "sieht")
    (:TYPE . :VERBCOMPLEX))
   (:SC-FRAME ((:NP . :NOM) (:NP . :AKK)))
   (:START . 0) (:END . 8)
   (:TYPE . :SUBJ-OBJ))))
```


The subcategorization lexicon

- more than 25500 entries for German verbs
- the information conveyed by the verb subcategorization lexicon we use, includes subcategorization patterns, like arity, case assigned to nominal arguments, preposition/ subconjunction form for other classes of complements
- Example subcat for the verb fahr (to drive):
  1. {<np,nom>}
  2. {<np,nom>, <pp, dat, mit>}
  3. {<np,nom>, <np,acc>}

Shallow strategy

- Given a set of different subcategorization frames that the lexicon associates to a verbal stem, the structure chosen as the final (disambiguated) solution is the one corresponding to the *maximal subcategorization frame* available in the set, which is the frame mentioning the largest number of arguments that may be succesfully applied to the input dependence tree.

Deep grammatical functions

- Obliquity hierarchy (implicitly assuming an ordering of the subcat elements; but only used for assigning a deep case label)
  - SUBJ: deep subject;
  - OBJ: deep object;
  - OBJ1: indirect object;
  - P-OBJ: prepositional object;
  - XCOMP: subcategorized subclause
- The subject and object does not necessarily correspond to the surface subject and direct object in the sentence, e.g., in case of passivization

Processing strategy of GFR

1. Retrieve the subcategorization frames for the verbal head of the root node of the input dependency tree;
2. Apply lexical rules in order to determine deep case information depending on the verb diathesis; since frames are expressed for active sentences only, a passivation rule exists which transforms NP-nominative to NP-accusative, and NP-nominative to PP-accusative with preposition von and durch
3. For each subcat frame sc do:
    1. match sc with the dependent elements; if matching succeeds, then call sc a valid subcat frame; otherwise sc is discarded;
    2. if sc is a valid subcat frame and sc_p is the current active subcat frame compute in the previous step of the loop, then if $|sc| > | sc_p|$ select sc as the current active subcat frame;
    3. insert the domain-specific information found for the verbal head of the root (if available); this information can be retrieved from the domain lexicon using the stem entry of the head verb (template triggering)
4. the same method is recursively applied on all sub-clauses
5. finally return the new dependency tree marked for deep grammatical functions; we call such dependency tree an underspecified functional description

Unification of subcat elements

- Expand subcat frame element to corresponding feature vector and unify it with the feature structure found for verbal head
- Example: *Der Mann sieht die Frau.*
    - subcat frame for *seh (to see):* {<np,nom>, <np,acc>}.
    - Fvect from input:
((:tense . :pres) (:form . :fin) (:person . 3)
(:gender . :no)(:number . :s) (:case . :no))
    - Expanded and unified fvec:
{((:tense . :pres) (:form . :fin) (:person . 3)
(:gender . :no) (:number . :s) (:case . :nom)),
    - ((:tense . :no) (:form . :no) (:person . :no)
(:gender . :no) (:number . :no) (:case . :acc))}
- Expanded fvec now used for unification with elements from NPs to assign subject and object.

Adjuncts are further grouped into type compatible subsets

- All elements which are not assigned grammatical functions are considered as adjuncts
- All elements of same type (e.g., date-np, loc-pp) are collected into disjunctive subsets (actually based on NE recognition):
  - {LOC-PP, LOC-NP, RANGE-LOC-PP} maps to LOC-MODS
  - {DATE-PP, DATE-NP} maps to DATE-MODS
- All others retain in their respective generic phrasals sets
  - NPS
  - PPS
  - SClause

22/02/2002 47

Summary

- SMES is a *mildly* deep parsing system
  - Combining shallow approaches with generic linguistic resources
  - Finite state backbone with feature constraints
  - Topological structure for coarse-grained sentence structure
  - Identification of grammatical functions

22/02/2002 48

Publications

(check http://www.dfki.de/~neumann/publications/neumann-ref.html)

- G. Neumann, C. Braun and J. Piskorski: A Divide-and-Conquer Strategy for Shallow Parsing of German Free Texts. In proceedings of ANLP-2000, Seattle, Washington, April, 2000.
- G. Neumann and G. Mazzini: Domain adaptive information extraction. Technical Report, 1999. A detailed description of SMES, especially
  - grammatical function recognition
  - use and integration of TDL (typed feature structure formalism orginally developed for HPSG but in SMES used for domain modelling)
- C. Braun: Flaches und robustes Parsing deutscher Satzgefüge. Diplomarbeit Computerlinguistik, Universität des Saarlandes, Oktober, 1999.
- G. Neumann, R. Backofen, J. Baur, M. Becker, C. Braun: An Information Extraction Core System for Real World German Text Processing. In Proceedings of 5th ANLP, Washington, March, 1997.