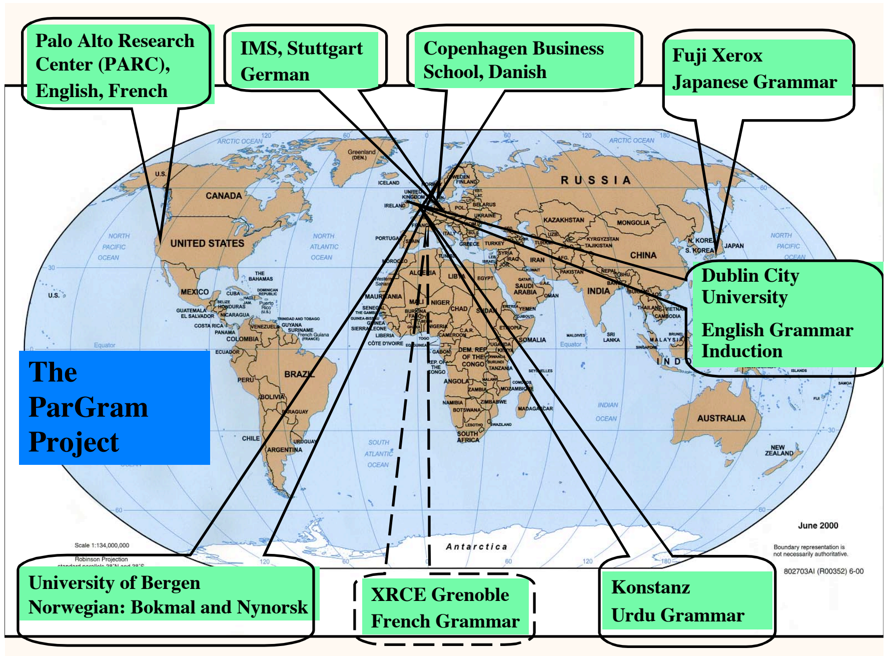
Deep CALL grammars: The LFG-OT Experiment

Rafiq Abdul Khader (Singapore)
Tracy Holloway King (PARC)
Miriam Butt (Konstanz)

DGfS 2004
AG 6, Computerlinguistik und elektronisches Lernen

Background

- **UMIST MSc Thesis Project**: Evaluate the English ParGram grammar as a possible grammar checker
- **Types of Errors**: Chinese learners of English
- **Evaluation**: Comparison to Microsoft Word Grammar Checker (and others available in the literature)



Project Structure

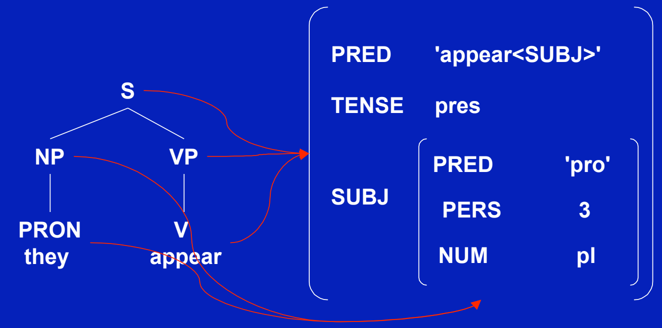
- Languages: English, Danish, French, German, Japanese, Norwegian, Urdu
- Theory: Lexical-Functional Grammar
- Platform: XLE
  - parser
  - generator
  - machine translation
- Loose organization: no common deliverables, but common interests.

Project History

- 1994: English, French, German
  - Solidified grammatical analyses and conventions
  - Expanded, hardened XLE
- 1999: Norwegian
- 2000: Japanese, Urdu
- 2002: Danish

Basic LFG

- Constituent-Structure: tree
- Functional-Structure:AVM, universal



Grammar Components

Each Grammar contains:

- Annotated Phrase Structure Rules (S --> NP VP)
- Lexicon (verb stems and functional elements)
- Finite-State Morphological Analyzer
- A version of Optimality Theory (OT):

  used as a filter to restrict ambiguities and/or parametrize the grammar.

Grammar Sizes (2003)

Lang.	Rules	States	Arcs
German	254	2323	4906
English	353	5712	15758
French	132	1116	2674
Japanese	51	351	1224
Norwegian	47	258	814
Urdu	26	112	172

The Parallel in ParGram

- There is usually more than one way to analyze a construction
- The same theoretical analysis may have different possible implementations
- The ParGram Project decides on common analyses and implementations (via meetings and the feature committee)

The Parallel in ParGram

- Analyses at the level of c-structure are allowed to differ (variance across languages)
- Analyses at f-structure are held as parallel as possible across languages (crosslinguistic invariance).
- **Theoretical Advantage**: This models the idea of UG.
- **Applicational Advantage**: machine translation is made easier; applications are more easily adapted to new languages (e.g., Kim et al. 2003).

Current English Grammar

- Broad coverage with robustness mechanisms: fragments/chunk parsing, skimming, guessers, stochastic disambiguation
- ~380 rules

  ~25000 lexical entries (mainly verbs)
- Input: sentence string
- Output: tree (c-structure)

  avm (f-structure)

Research Question

Can this broad-coverage grammar be used for grammar checking in Computer-Assisted Language Learning (CALL) settings?

CALL Background

- Incorporation of computers in writing classes.
  - This facilitated writing processes via the ability to add, delete and rearrange texts via word processing.
- But more tools/applications could easily be envisioned for the teaching/editing process.
  - Creation of the first style/grammar checkers.

Existing Grammar Checkers

a. Writer's Workbench (Wong,1996)
b. Ruskin (William, 1992)
c. Expert Editor (Johnson, 1985)
d. Epistle Program (Wong, 1996)
e. FROG (Imlah & du Boulay, 1985)
f. LINGER (Yazdani, 1991)
g. E-linger (Lawler,1991)
h. Microsoft Word 2000 (Markoff, 2002)
i. Grammatik (Yu and Davies, 1996)
j. Native English (Tschichold, 1999)

NLP Based Grammar Checkers

- Liou et al. (1992)
- Liou (1992)
- Holland et al. (1993)
- Coniam (1991)
- Webster (1991)
- Park et al. Xu (1994)
  - to highlight errors made no substantial difference

Grammar Checkers: capability

- The grammar checkers listed (a-j)
  - use relatively low-level syntactic knowledge
  - employ Part of Speech (POS) tags
  - have some morphological analysis
  - are based on a general knowledge of adjacency of syntactic elements

Grammar Checkers: capability

- The grammar checkers
  - identify only part of the possible range of syntax errors (Halio 1995; Tschichold 1999)
  - highlight errors with only a minimal suggestion for
    - spelling and punctuation (mechanical errors)
    - syntactic errors
      - use of passives
      - split infinitives
      - sentence length

Grammar Checkers: Limitations

- Difficult to detect verb tense/aspect errors
- Can only put marks near suspect word or phrases
- Tend to flag words that writers actually do not want changed

Grammar Checkers: Limitations

- Factors working against the development of programs to identify faulty syntax:
  - Lexical and structural complexities
  - Range of problems within and across phrase and clause edges
  - Huge range of potential solutions make structural analysis, identification and reformulation difficult (Bolt 1996)
  - Language itself (complex, not well understood)

Grammar Checkers: Limitations

- NLP-based grammar checkers:
  - accept grammatical sentences
  - reject ungrammatical ones
- Need a program to
  - accept ungrammatical input
  - identify it as ungrammatical
  - provide functional feedback to improve it

Method

- Cull corpus from Chinese learners of English
- Identification of
  - Mechanical (i.e., spelling, punctuation)
  - Lexical (lexical semantic errors)
  - Grammatical errors (syntax, Zheng 1993)

- Adapt English ParGram grammar to deal with the learner corpus via OT-marks

Method

- Introduce special UNGRAMMATICAL feature at f-structure for feedback as to the type of error
- Generate back possible corrections
- Evaluate on developed and unseen corpus
  - i. accuracy of error detection
  - ii. value of suggestions or possible feedback
  - iii. range of language problems/errors covered
  - iv. speed of operation

Adapting the English Grammar

- The standard ParGram English grammar was augmented with:
  - OT marks for ungrammatical constructions
  - Information for feedback (e.g., missing-be, subj-verb-agr)
- Parametrization of the Generator to allow for corrections based on ungrammatical input.

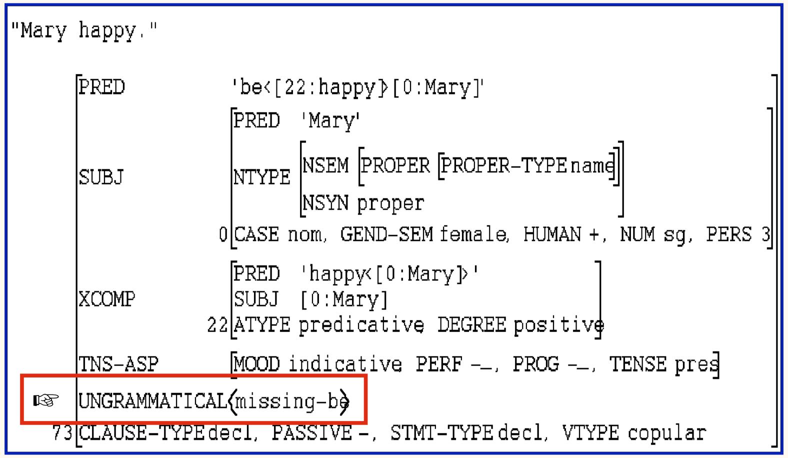
Adapting the English Grammar

- Effort Required:
  - 3 working days by Tracy Holloway King (PARC)

Markup for Feedback

- Indicate ungrammaticality in f-structure
  – Type of problem
  – Where the problem is
- Example: Mary happy.
  – UNGRAMMATICAL {missing-be}
  – top level f-structure

F-structure: Mary happy.



Modifying the grammar

- Optimality Theory (OT) marks control ranking of rule application
- Loosen grammar to allow ungrammatical constructions
- Ungrammatical constructions put in feature indicating source of ungrammaticality
- Slightly disprefer ungrammatical constructions with OT mark

Example modifications

- Missing copula (Mary happy.)
- No subj-verb agreement (The boys leaves.)
- Missing specifier on count noun (Boy leaves.)
- Missing puntuation (Mary is happy)
- Bad adverb placement (Mary quickly leaves.)
- Non-fronted wh-words (You saw who?)
- Missing *to* infinitive (I want disappear.)

Using OT Marks

- OT marks allow one analysis to be prefered over another
- The marks are introduced in rules and lexical entries

  @(OT-MARK ungrammatical)
- The parser is given a ranking of the marks
- Only the top ranked analyses appear

OT Marks in the CALL grammar

- A correct sentence triggers no marks
- A sentence with a known error triggers a mark ungrammatical
- A sentence with an unknown error triggers a mark fragment
- no mark < ungrammatical < fragment
  - the grammar first tries for no mark
  - then for a known error
  - then a fragment if all else fails

No Subj-Verb Agreement

- Original template for V3SG =

```
(^ SUBJ NUM)=sg
(^ SUBJ PERS)=3
```

- CALL template for V3SG =

```
{ (^ SUBJ NUM)=sg
   (^ SUBJ PERS)=3
 |~[(^ SUBJ NUM)=sg
   (^ SUBJ PERS)=3]
   (^ UNGRAMMATICAL)=subj-verb-agr
   @(OT-MARK ungrammatical) }.
```

Missing copula

- Old VPcop rule:

```
VPcop --> Vcop
          PREDLINKP.
```

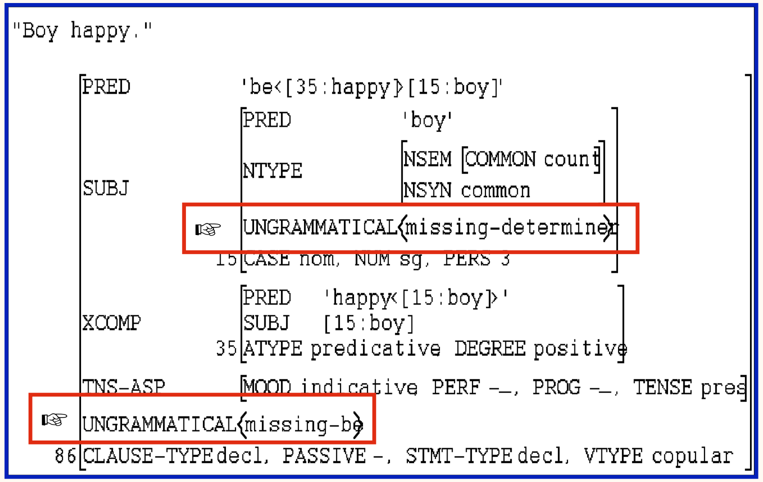
- CALL VPcop rule:

```
VPcop --> { Vcop
            |e: @MISSING-BE-RULES
                (^ UNGRAMMATICAL)=missing-be
                @(OT-MARK ungrammatical) }
          PREDLINKP.
```

Multiple errors

- Multiple errors can be parsed
- Record in relevant part of f-structure
- Example:
  - Boy happy.
  - missing-be
  - missing-determiner

F-structure: Boy happy.



Unknown error

- Ungrammatical sentences with unknown problems produce a Fragment parse
- This fragment is UNGRAMMATICAL

- Problem: no correction via generation
- Example: The the boy arrived.

Generation of corrections

- But XLE generally allows the generation of correct sentences from ungrammtical input.
- Method:
  - Parse ungrammatical sentence
  - Remove UNGRAMMATICAL feature for generation
  - Generate from stripped down ungrammatical f-structure

Generation in LFG

- **Basic Idea:** The grammar should be bidirectional.
  - If you have written a parser, you should be able to reverse the grammar for generation.
  - **Input to Generator:** an f-structure (Prolog)
  - Shemtov (1997) is the basis for the current implementation, based on ideas by Kay (1996).

Generation in ParGram

- Use of the same grammars (bidirectionality).
- Use of OT (Optimality Theory) to control generation:
  - (unwanted) parses such as mismatched subject-verb agreement or optional punctuation (either prefer to have it or prefer it is gone).
- Use a different generation tokenizer to generate a single space between words, etc.

Generation from Underspecified Input

- XLE allows the specification of information to be removed from or added to an f-structure.
- Example: generation from an f-structure without tense/aspect information.

John sleeps → All tense/aspect variations

```
John
 {  { will be
      |was
      |is
      |{has|had} been}
    sleeping
   |{{will have|has|had}|} slept
   |sleeps
   |will sleep}
```

XLE

CALL Generation example

- parse "Mary happy."
  generate back:
  Mary is happy.
- parse "boy arrives."
  generate back:
  { This | That | The | A } boy arrives.
- Demo: parse sentences from Chinese learner corpus.

XLE

Evaluation

- Preliminary Evaluation promising:
  - Word 10 out of 50=20% (bad user feedback)
  - XLE 29 out of 50=58% (better user feedback)
- Unseen Real Life Student Production
  - Word 5 out of 11 (bad user feedback)
  - XLE 6 out 11 (better user feedback)

Future work

- More testing/development needed
- Evaluation of what kind of feedback is most useful for the user.
  - better suggestions for error reporting
  - user-friendly XML interface
- Better correction suggestions for unknown errors

References

Butt, M., H. Dyvik, T.H. King, H. Masuichi and C. Rohrer. 2002 The Parallel Grammar Project. In J. Carroll, N. Oostdijk and R. Sutcliffe (eds.) *Proceedings of the Workshop on Grammar Engineering and Evaluation,* COLING, Taipei, Taiwan, September, 1-7.

Butt, M., T.H. King, M.-E. Niño, F. Segond. 1999. *A Grammar Writer's Cookbook*. Stanford, CA: CSLI Publications.

Khader, Rafiq. 2003. *Evaluation of an English LFG-based Grammar as Error Checker*. UMIST MSc Thesis, Manchester.

Kim, R., M. Dalrymple, R.M. Kaplan, T.H. King, H. Masuichi, and T. Ohkuma. 2003, Multilingual Grammar Development via Grammar Porting. In *ESSLLI 2003 Workshop on Ideas and Strategies for Multilingual Grammar Development,* August, Vienna.

Shemtov, Hadar. 1997. *Ambiguity Management in Natural Language Generation*. PhD thesis, Stanford University.

Contact Information

- Rafiq Khader: isarafiq@tp.edu.sg
- Tracy Holloway King: thking@parc.com
- Miriam Butt: miriam.butt@uni-konstanz.de