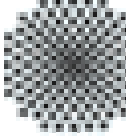
The Feature Space in Parallel Grammar Writing

Miriam Butt
Centre for Comp. Linguistics
UMIST, UK
mutt@ccl.umist.ac.uk

Martin Forst
IMS, University of Stuttgart
Germany
forst@ims.uni-stuttgart.de

Tracy Holloway King
Palo Alto Research Center
Palo Alto, USA
thking@parc.com

Jonas Kuhn
Department of Linguistics
UT Austin, USA
jonask@mail.utexas.edu



Overview

- The ParGram project and its goals
- The status of feature appropriateness conditions
- Methodology and tools applied:
  - The Feature Committee
  - The feature table
  - Technical tool for checking adherence to feature declaration(s)
  - Coordinated, systematic use of templates
- Conclusion

The ParGram project

- Involves five sites and seven languages:
  - PARC (English, French, Korean)
  - University of Stuttgart (German)
  - University of Bergen (Norwegian)
  - Fuji Xerox (Japanese)
  - UMIST (Urdu)
- Goal: Development of parallel broad-coverage LFGs for a variety of languages, processable by the grammar development platform XLE

What does "parallel" mean?

- LFGs assign two kinds of syntactic representations to a given sentence
  - A surface phrase structure tree, called *c(onstituent)-structure* – encodes precedence and dominance, highly language-specific
  - An attribute-value matrix, called *f(unctional)-structure* – encodes "universal" grammatical relations, abstracts away from most crosslinguistic differences

→ "Parallel" means "parallel at the f-structure level whenever it is linguistically defendable"

Examples of parallel f-structures

```
[PRED     'arrive<[38:letter]>'
         [PRED   'letter'
         NTYPE  [GRAIN count]
SUBJ
         SPEC   [DET [PRED     'the'
                     DET-TYPE def]]
      38 [CASE nom, NUM sg, PERS 3]
ADJUNCT {[PRED        'tomorrow'
 15      ADV-DEGREE positive]}
TNS-ASP [MOOD indicative, PERF +_, PROG -_, TENSE fut]
 83 [CLAUSE-TYPE declarative, PASSIVE -, STMT-TYPE decl, VTYPE main]]
```

English:

Tomorrow the letter will have arrived.

German:

Morgen wird der Brief angekommen sein.

Tomorrow will the letter arrived have.

```
[PRED     'an#kommen<[-5-SUBJ:Brief]>'
         [PRED   'Brief'
         NSEM   [COMMON count]
         NTYPE  [NSYN common]
SUBJ
         SPEC   [DET [PRED     'die'
                     DET-TYPE def]]
         CASE nom, GEND masc, NUM sg, PERS 3]
ADJUNCT {[PRED      'morgen'
         ADV-TYPE temp]}
TNS-ASP [MOOD indicative, PERF +_, TENSE fut_]
TOPIC    [-5-TOPIC:morgen]
 -5 [CLAUSE-TYPE decl, STMT-TYPE decl, VTYPE main]]
```

The Status of Feature Appropriateness Conditions

HPSG/Matrix

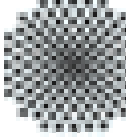
- Universal theory-driven type/sort definitions appear ideal for ensuring parallelism in multilingual grammar development
- **But:** Consideration of additional languages or phenomena may require fundamental revisions

LFG/ParGram

- LFG: type/sort definitions are not a theoretical focus
- ParGram/XLE: feature appropriateness checking similar to HPSG, but less theory-driven
- Resulting feature geometry is typically leaner and thus less prone to major revision

The Feature Committee

- Consists of all grammar writers
- Twice-yearly meetings, where new developments in the grammars are checked and discussed (mainly by f-structure comparison and feature table inspection)
- Aims at keeping the feature space as small as possible by assimilating analyses of new phenomena within the existing feature space



The Feature Table

- Columns → Languages
- Lines → Features
- Cells → Possible values of features

Created automatically by a Perl script that reads the feature declarations of all grammars concerned.

A Sample Feature Table

Feature	ENGLISH	GERMAN	JAPANESE
CASE	acc gen nom	acc dat gen nom	acc gen nom
GEND		fem masc neut	
TNS-ASP	MOOD PERF PROG TENSE	MOOD PASS-SEM PERF TENSE	MOOD PROG TENSE
VTYPE	copular main modal	copular main modal	copular main

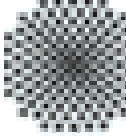
Feature Declarations and Feature Declaration Checking

- Declaration of possible atomic values

  $\texttt{FEAT:}\rightarrow\texttt{\$\{value}_1\ \texttt{value}_2\ \texttt{...}\ \texttt{value}_n\texttt{\}}$

- Declaration of complex f-structure values

  $\texttt{FEAT}_\texttt{A}\texttt{:}\rightarrow\texttt{<<[FEAT}_\texttt{B1}\ \texttt{FEAT}_\texttt{B2}\ \texttt{...}\ \texttt{FEAT}_\texttt{Bn}\texttt{]}$



Feature Declarations (Example)

```
TNS-ASP:-> << [MOOD PERF PROG
  TENSE].
MOOD:-> ${imperative indicative
  subjunctive}.
PERF:-> ${+ -}.
PROG:-> ${+ -}.
TENSE:-> ${fut past pres}.
```


Feature Declaration Checking

When grammar is loaded, compiled grammar rules are checked against feature declaration.

- If any undefined feature is found:

```
Feature declaration violation near line
162, column 28 in file dmor1.lex.lfg:
Unknown feature: PERFECT.
```

- If any value not declared for the feature concerned is found:

```
Feature declaration violation near line
364, column 27 in file dmor1.lex.lfg:
TENSE cannot be equal to present.
```

Further possibilities of the XLE feature declaration mechanism

- Combination of various feature declarations for grammars of different "dialects"

```
FEATURES  STANDARD-ENGLISH EUREKA-ENGLISH.
```

- Combination of a universal and a language-specific feature declaration (particularly interesting in multilingual grammar development)

```
FEATURES  COMMON STANDARD-FRENCH.
```

Combining various feature declarations

XLE provides three edit operators for the combination of various feature declarations: &(union), +(disjunction) and !(overwriting)

→ A feature declaration common to all grammars can be modified and completed by language-specific feature declarations.

→ Feature Committee can concentrate on changes in the language-specific feature declarations and then

- Integrate changes into common feature declaration.
- Reject feature as result of aberrant analysis
- Accept feature as reflecting a language-particular characteristic

Combining various feature declarations (examples)

COMMON:
```
CASE:-> ${acc dat erg gen
         inst loc nom obl}.
GEND:-> ${fem masc neut}.
TENSE:-> ${fut past pres}.
```

NORW.:
```
&CASE:-> ${gen nom obl}.
```

ENGLISH:
```
!GEND:-> ${}.
```

FRENCH:
```
+TENSE:-> ${passesimple}.
```

Capturing Generalizations through Templates

- Main goal: Parallelism at the level of f-structure *representations*
  - Facilitates further use of analyses obtained, particularly in multilingual applications such as MT
- But also: Parallelism at the level of f-structure *description*, especially templates
  - Facilitates reuse of (more or less) large portions of a grammar for grammar porting

Capturing Generalizations through Templates (Example)

- **V-SUBJ-COMP_PRED**(_stem) =
  (↑PRED)='_stem<(↑SUBJ)(↑COMP)>'.

Capturing Generalizations through Templates (Example)

- **V-SUBJ-COMP_PRED**(_stem) =
  (↑PRED)='_stem<(↑SUBJ)(↑COMP)>'.
- **V-SUBJ-COMP-FIN**(_stem) =
  @(**V-SUBJ-COMP_PRED** _stem)
  @COMP-FIN @NO-PRT @NO-CL-SUBJ.

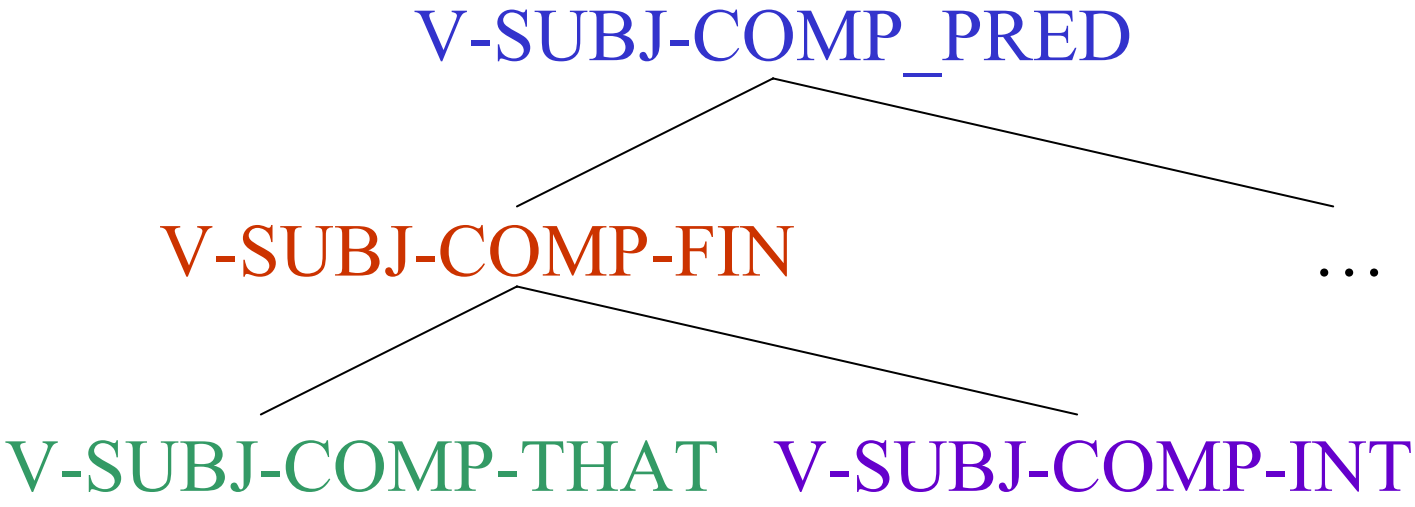
Capturing Generalizations through Templates (Example)

- `V-SUBJ-COMP_PRED(_stem) =`
  `(↑PRED)='_stem<(↑SUBJ)(↑COMP)>'.`
- `V-SUBJ-COMP-FIN(_stem) =`
  `@(V-SUBJ-COMP_PRED _stem)`
  `@COMP-FIN @NO-PRT @NO-CL-SUBJ.`
- `V-SUBJ-COMP-THAT(_stem) =`
  `@(V-SUBJ-COMP-FIN _stem)`
  `@(COMP-FORM that).`

Capturing Generalizations through Templates (Example)

- `V-SUBJ-COMP_PRED(_stem) =`
  `(↑PRED)='_stem<(↑SUBJ)(↑COMP)>'.`
- `V-SUBJ-COMP-FIN(_stem) =`
  `@(V-SUBJ-COMP_PRED _stem)`
  `@COMP-FIN @NO-PRT @NO-CL-SUBJ.`
- `V-SUBJ-COMP-THAT(_stem) =`
  `@(V-SUBJ-COMP-FIN _stem)`
  `@(COMP-FORM that).`
- `V-SUBJ-COMP-INT(_stem) =`
  `@(V-SUBJ-COMP-FIN _stem)`
  `@(COMP-FORM whether).`

Capturing Generalizations through Templates (Example)



Discussion and Conclusions

- Parallel grammar development needs a common feature space; two tools in XLE:
  - Feature declaration: manipulate common feature space in a transparent manner
  - Templates encode generalizations within the grammars
- Grammars for new languages can be bootstrapped by use of the existing declared feature space and templates