Implementing HPSG grammars

Detmar Meurers
Universität Tübingen
dm@sfs.nphil.uni-tuebingen.de

East-West Computational Linguistics Workshop
Universität Tübingen, 20. Sept. 1996

Structure of the presentation

- The main building blocks of HPSG grammars
  - from a linguistic perspective
  - from a formal perspective
- Implementing HPSG grammars in computational systems using
  - general constraints on the domain of linguistic objects
  - definite clauses as relational backbone
  - phrase structure rules as special relational backbone
- Close to HPSG and tractable?
  - General techniques:
    - precompilation
    - improved interpretation
  - Techniques using domain-specific knowledge
- Summing Up

HPSG grammars from a linguistic perspective

From a linguistic perspective, an HPSG grammar consists of

- a lexicon licensing *basic words*
- lexical rules licensing *derived words*
- immediate dominance (ID) schemata licensing *constituent structure*
- linear precedence (LP) statements constraining *word order*
- a set of grammatical principles expressing *generalizations about linguistic objects*

Basic lexicon

$$
\textit{word} \;\rightarrow\; \begin{bmatrix} \text{PHON} & \langle \textit{she} \rangle \\ \text{SYNSEM}\,|\,\text{LOC} & \begin{bmatrix} \text{CAT} & \begin{bmatrix} \text{HEAD} & \begin{bmatrix} \textit{noun} & \\ \text{CASE} & \textit{nom} \end{bmatrix} \\ \text{SUBCAT} & \langle\rangle \end{bmatrix} \\ \text{CONT} & \begin{bmatrix} \textit{ppro} & \\ \text{INDEX} & \begin{bmatrix} \text{PER} & \textit{3rd} \\ \text{NUM} & \textit{sing} \\ \text{GEND} & \textit{fem} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix}
$$

$$
\vee \begin{bmatrix} \text{PHON} & \langle \textit{painted} \rangle \\ \text{SYNSEM}\,|\,\text{LOC} & \begin{bmatrix} \text{CAT} & \begin{bmatrix} \text{HEAD} & \begin{bmatrix} \textit{verb} & \\ \text{VFORM} & \textit{psp} \end{bmatrix} \\ \text{SUBCAT} & \langle \text{NP[nom]}_{\boxed{1}}, \text{NP[acc]}_{\boxed{2}} \rangle \end{bmatrix} \\ \text{CONT} & \begin{bmatrix} \textit{paint} & \\ \text{PAINTER} & \boxed{1} \\ \text{PAINTED} & \boxed{2} \end{bmatrix} \end{bmatrix} \end{bmatrix}
$$

$$
\vee \qquad \vdots
$$

©Detmar Meurers, SFB 340

Lexical Rules

A passive lexical rule (simplified, based on Pollard and Sag (1987, p.215)):

$$\begin{bmatrix} \text{SYNSEM} \mid \text{LOC} \mid \text{CAT} & \begin{bmatrix} \text{HEAD} & \begin{bmatrix} \textit{verb} \\ \text{VFORM} & \textit{psp} \end{bmatrix} \\ \text{SUBCAT} & \left\langle \text{NP}_{\boxed{1}}, \text{NP}_{\boxed{2}} \right\rangle \oplus \boxed{3} \end{bmatrix} \end{bmatrix}$$

$$\mapsto$$

$$\begin{bmatrix} \text{SYNSEM} \mid \text{LOC} \mid \text{CAT} & \begin{bmatrix} \text{HEAD} & \begin{bmatrix} \textit{verb} \\ \text{VFORM} & \textit{pas} \end{bmatrix} \\ \text{SUBCAT} & \left\langle \text{NP}_{\boxed{2}} \right\rangle \oplus \boxed{3} \oplus \left\langle \left(\text{PP}[\text{by}]_{\boxed{1}} \right) \right\rangle \end{bmatrix} \end{bmatrix}$$

ID Schemata and LP Statements

ID Schemata

Example: The Head-Complement Schema (Pollard and Sag 1994, p.402)

$$\begin{bmatrix} \textit{phrase} \\ \text{DTRS} & \textit{headed-structure} \end{bmatrix} \rightarrow \begin{bmatrix} \text{SYNSEM} \mid \text{LOC} \mid \text{CAT} \mid \text{SUBCAT} & \langle \textit{synsem} \rangle \\ \text{DTRS} & \begin{bmatrix} \textit{head-comp-struc} \\ \text{HEAD-DTR} & \textit{word} \end{bmatrix} \end{bmatrix}$$

LP Statements

Example: A restriction on the linearization of indefinite NPs in the German Mittelfeld (based on (Lenerz 1977))

$$\text{NP[dat]} < \text{NP[acc,indef]}$$

©Detmar Meurers, SFB 340

General grammatical principles

Example 1: The Head-Feature Principle (Pollard and Sag 1994, p.399)

$$\begin{bmatrix} \textit{phrase} \\ \text{DTRS} \quad \textit{headed-struc} \end{bmatrix} \rightarrow$$

$$\begin{bmatrix} \text{SYNSEM} \mid \text{LOC} \mid \text{CAT} \mid \text{HEAD} & \boxed{1} \\ \text{DTRS} \mid \text{HEAD-DTR} \mid \text{SYNSEM} \mid \text{LOC} \mid \text{CAT} \mid \text{HEAD} & \boxed{1} \end{bmatrix}$$

Example 2: The Clausal Rel Prohibition (Pollard and Sag 1994, p.401)

$$\begin{bmatrix} \textit{synsem} \\ \text{LOC} \mid \text{CAT} \begin{bmatrix} \text{HEAD} & \textit{verb} \\ \text{SUBCAT} & \langle\rangle \end{bmatrix} \end{bmatrix} \rightarrow \begin{bmatrix} \text{NONLOC} \mid \text{INHER} \mid \text{REL} \ \{\} \end{bmatrix}$$

HPSG grammars – formal perspective

From a formal perspective, an HPSG grammar consists of:

- The signature as declaration of the domain:
  - type hierarchy (which kind of objects exist)
  - appropriateness conditions (which objects have which properties)
- The theory constraining the domain:
  - A theory is a set of description language statements, the constraints, which single out the grammatical objects from the ungrammatical ones.
    - The description language statements consist of: type assignment, path equality, conjunction, disjunction, negation.
    - Most of the theory – the lexicon, ID schemata, and principles – is already expressed using such statements.
    - The other components can also be formalized so that it uses the same logical basis:
      - · LP statements (Richter and Sailer 1995)
      - · Lexical rules (Meurers 1995)
  - A linguistic object is admissible with respect to a theory iff it satisfies each of the descriptions in the theory and so does each of its substructures.

Computational systems based on type constraints

TFS (Emele and Zajac 1990)

- A TFS theory is a set of implicational statements with type antecedents.
- Any description can be entered as a query and the system returns a description subsumed by the query, such that the descriptions satisfies each theory constraint and so does each of its parts.

Pro:

- The organization of constraints is the same as in the HPSG architecture.

Cons:

- Only implicational statements with type antecedent are allowed. No general treatment of negation.
- Very severe control problems: efficiency, termination

Computational systems using a relational backbone

CUF (Dörre and Eisele 1991; Dörre and Dorna 1993)

- A theory is expressed using definite clauses as a relational extension of the description language. Basically CUF is like PROLOG, but with feature terms used as arguments of the relations instead of the first order terms.
- An HPSG grammar is implemented by rewriting it as a logic program.
- A query is a call to one of the relations and the system returns the instantiation of the arguments required by the relation.

©Detmar Meurers, SFB 340

Relational backbone – example

An implementation of an HPSG theory using a relational backbone could look as follows. (Note that CUF actually uses a functional notation instead of the relational one.)

```
well_formed_phrase(1) :-
          id_schema(1,2,3),
          well_formed_sign(2),
          well_formed_signs(3),
          head_feature_principle(1,2),
          ...
```

% id_schema(Phrase,Head-Dtr,Other-Dtr-List)

$$\texttt{id\_schema(}\boxed{1}\begin{bmatrix} \ldots\text{SUBCAT} & \langle \textit{synsem} \rangle \\ \text{DTRS} & \begin{bmatrix} \textit{headed\_structure} & \\ \text{HEAD\_DTR} & \boxed{2}\textit{word} \\ \text{COMP\_DTRS} & \boxed{3} \end{bmatrix} \end{bmatrix},\boxed{2},\boxed{3}\rangle\texttt{).}$$

$$\texttt{head\_feature\_principle(}\begin{bmatrix}\text{SYNSEM}\,|\,\text{LOC}\,|\,\text{CAT}\,|\,\text{HEAD} & \boxed{1}\end{bmatrix}, \begin{bmatrix}\text{SYNSEM}\,|\,\text{LOC}\,|\,\text{CAT}\,|\,\text{HEAD} & \boxed{1}\end{bmatrix}\texttt{).}$$

Relational backbone – evaluation

Pros:

- The recursive constraints are encoded on a different level (relations) than the simple constraints on the linguistic data structure (arguments to the relations).
- The grammar writer determines the order of execution of goals by the way in which he/she encodes the grammar (= program).
- A definite clause encoding allows the efficient processing techniques of logic programming to be used (e.g., clause indexing, earley deduction, goal freezing, ...).

Con:

- Since the organization and expression of constraints is completely different, a CUF grammar is only related to the original linguistic theory on a very intuitive level.

Computational systems using a phrase structure backbone

ALE (Carpenter 1992; Carpenter and Penn 1994),
Troll (Gerdemann & Götz)

- The implemented grammar is organized around a special kind of relation, phrase structure rules, with relational attachments. Basically ALE grammars are like DCGs, but with feature terms used as arguments of the phrase structure rules and definite clauses instead of the first order terms.
- phrase structure:
  - fixed number of daughters per rule (at run-time)
  - phon(mother) = phon(dtr1) $\oplus \ldots \oplus$ phon(dtrN)

head_complement_schema rule

$$\boxed{1}\left[\text{SYNSEM}\,|\,\text{LOC}\,|\,\text{CAT}\,|\,\text{SUBCAT}\ \langle \boxed{2} \rangle\right]$$

===>

cat> $\boxed{3}\begin{bmatrix} \textit{word} \\ \text{SYNSEM}\,|\,\text{LOC}\,|\,\text{CAT}\,|\,\text{SUBCAT}\ \langle \boxed{2} \mid \boxed{4} \rangle \end{bmatrix}$,

cats> $\boxed{4}$,

goal> head_feature_principle($\boxed{1}$,$\boxed{3}$),

...

Phrase structure backbone – evaluation

Pros:

- most important recursive structure singled out
- very efficient algorithms for parsing are available since indexing on the phonology can be used

Cons:

- As with the general relational encoding, the organization and expression of constraints is different from the linguistic theory.
- The additional restrictions on phrase structure (compared to general relations) usually require the grammar writer to encode each constraint specifying constituent structure in several phrase structure rules.

©Detmar Meurers, SFB 340

Close to HPSG and tractable?

ConTroll (SFB 340, Project B4)

- Motivation: Only when a clear, formally meaningful, and computationally usable link is defined, we can use implemented grammars to provide feedback for a rigid and complete formalization of linguistic theories.
- Idea: Bridge the gap between the linguistic theory and the efficient implementations by
  - compiling HPSG theories as specified by the linguist into definite clause programs (Götz and Meurers 1995).
  - advance the resulting program encoding
    - ∗ without employing domain specific knowledge
      - · magic compilation: add predicates providing top down guidance for bottom up processing (Minnen 1996)
      - · constraint propagation: locate specifications at those place in a program that make a goal fail early, thus pruning the search space (Meurers and Minnen 1996)

    - ∗ employing domain specific knowledge
      - · identifying phrase structure sub-fragments (Aldag 1996)
      - · Precompiling the lexical rules: determining possible lexical rule orders, hiding disjunctive possibilities in relations, lifting common factors up to be available at lexical lookup (Meurers and Minnen 1995).
  - use special interpretation to process efficiently
    - ∗ lazy interpretation: only check constraints on those nodes which are specific enough to cause conflicts instead of enumerating models (Aït-Kaci, Podelski, and Goldstein 1993; Götz and Meurers 1996)
    - ∗ tabelling methods (Earley deduction) to store intermediate results (Minnen, Gerdemann, and Götz 1995)

Summing Up

- We have provided an overview of the main ingredients of HPSG grammars from a linguistic and a formal perspective.
- Three kind of computational setups were discussed, which differ in the organization of constraints they require/use:
  - only universal constraints on well formed objects are expressed
  - a relational backbone is used
  - a phrase structure backbone is used
- Finally, the approach pursued with the ConTroll system was described:
  - compiling HPSG theories as specified by the linguist into more efficient relational encodings,
  - advancing the resulting program in precompilation steps, and
  - using specific modes of interpretation.

References

Aït-Kaci, H., A. Podelski, and S. C. Goldstein (1993). Order-Sorted Theory Unification. Technical Report 32, Digital Equipment Corporation.

Aldag, B. (1996). Compiling Head-Driven Phrase Structure Grammars in to Context Free Grammars. Ms., Universität Tübingen, Germany.

Carpenter, B. (1992). ALE – The Attribute Logic Engine, User's Guide. Laboratory for Computational Linguistics Report CMU-LCL-92-1, Laboratory for Computational Linguistics, Carnegie Mellon University.

Carpenter, B. and G. Penn (1994). ALE – The Attribute Logic Engine, User's Guide, Version 2.0.1, December 1994. Technical report, Computational Linguistics Program, Philosopy Department, Carnegie Mellon University.

Dörre, J. and M. Dorna (1993, August). CUF - a formalism for linguistic knowledge representation. In J. Dörre (Ed.), *Computational aspects of constraint based linguistic descriptions I*, pp. 1–22. Universität Stuttgart: DYANA-2 Deliverable R1.2.A.

Dörre, J. and A. Eisele (1991, January). A comprehensive unification based formalism. DYANA Deliverable R3.1.B, Universität Stuttgart.

Emele, M. and R. Zajac (1990). Typed Unification Grammars. In H. Karlgreen (Ed.), *Proceedings of the 13th Conference on Computational Linguistics (COLING)*, Helsinki.

Götz, T. and W. D. Meurers (1995). Compiling HPSG type constraints into definite clause programs. In *Proceedings of the Thirty-Third Annual Meeting of the ACL*, Boston. Association for Computational Linguistics.

Götz, T. and W. D. Meurers (1996). The importance of being lazy - Using lazy evaluation to process queries to HPSG grammars. In *Proceedings of TALN 96 (joint session with the Third International Conference on HPSG)*, Marseille, France.

Lenerz, J. (1977). *Zur Abfolge nominaler Satzglieder im Deutschen.* Tübingen: TBL Verlag Gunter Narr.

Meurers, W. D. (1994). On Implementing an HPSG theory – Aspects of the logical architecture, the formalization, and the implementation of head-driven phrase structure grammars. In: Erhard W. Hinrichs, W. Detmar Meurers, and Tsuneko Nakazawa: *Partial-VP and Split-NP Topicalization in German – An HPSG Analysis and its Implementation*. Arbeitspapiere des SFB 340 Nr. 58, Universität Tübingen.

Meurers, W. D. (1995). Towards a Semantics for Lexical Rules as used in HPSG. In *Proceedings of the Conference on Formal Grammar*, Barcelona. Also in: Proceedings of the ACQUILEX II Workshop on Lexical Rules, 1995, Cambridge, UK.

Meurers, W. D. and G. Minnen (1995). A computational treatment of HPSG lexical rules as covariation in lexical entries. In *Proceedings of the Fifth International Workshop on Natural Language Understanding and Logic Programming*, Lisbon, Portugal.

Meurers, W. D. and G. Minnen (1996). Off-line Constraint Propagation for Efficient HPSG Processing. In *HPSG/TALN Proceedings*, Marseille, France.

Minnen, G. (1996). Magic for Filter Optimization in Dynamic Bottom-up Processing. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, Santa Cruz, USA.

Minnen, G., D. Gerdemann, and T. Götz (1995). Off-line Optimization for Earley-style HPSG Processing. In *Proceedings of the Seventh Conference of the European Chapter of the Association for Computational Linguistics*, Dublin, Ireland.

Pollard, C. and I. A. Sag (1987). *Information-based Syntax and Semantics, Vol. 1.* Number 13 in Lecture Notes. Stanford University: CSLI Publications. Distributed by University of Chicago Press.

Pollard, C. and I. A. Sag (1994). *Head-Driven Phrase Structure Grammar*. Chicago: University of Chicago Press.

Richter, F. and M. Sailer (1995). Remarks on Linearization: Reflections on the Treatment of LP-Rules in HPSG in a Typed Feature Logic. Master's thesis, University of Tübingen.

Many of the cited papers are available from the respective author's homepages which are accessible from:

http://www.sfs.nphil.uni-tuebingen.de/sfb/b4home.html and b8home.html.