Grammar Engineering for Deep Linguistic Processing
Seminar 2009
Lecture 1: General Introduction

Yi Zhang

Department of Computational Linguistics & Phonetics
Saarland University

Language Technology Lab
German Research Center for Artificial Intelligence

23rd April, 2009
Grammar Engineering for Deep Linguistic Processing

Course Info
- Type: Seminar
- Time: TBA
- Location: U.15, CIP Room
- Homepage: [http://www.coli.uni-saarland.de/~yzhang/ge-ss09](http://www.coli.uni-saarland.de/~yzhang/ge-ss09)
Outline

1. Grammar engineering
   - What?
   - Why?
   - Who?
   - How?

2. HPSG and DELPH-IN

3. Structure of the course
Outline

1. Grammar engineering
   - What?
   - Why?
   - Who?
   - How?

2. HPSG and DELPH-IN

3. Structure of the course
What Is Grammar Engineering?

Goal
- To build grammar resources
What Is Grammar Engineering?

Goal
- To build computational grammar resources
What Is Grammar Engineering?

Goal

To build usable computational grammar resources
What Is Grammar Engineering?

Goal

- To build re-usable computational grammar resources
Motivation

- Linguistically motivated grammars are variable resources
- Relevant to most of the NLP tasks
- Can be used to test linguistic hypotheses
Application Scenario

Basic tasks
- Parsing
- Generation

Real-world applications
- Machine Translation
- Grammar Checking
- Information Extraction
- Question Answering
Application Scenario

Basic tasks
- Parsing
- Generation

Real-world applications
- Machine Translation
- Grammar Checking
- Information Extraction
- Question Answering
Linguists and Computer Scientists
Necessary Components

- Linguistic Theory
- Grammar Engineering Platform
- Linguistic Resources
- Methodology
Linguistic Theories

A Solid Linguistic Theory

- Rigid mathematical foundation
- Tractable computational model
- Universal to different languages
Linguistic Theories of Choice

- Tree-adjoining grammar (TAG)
- Combinatory Categorial Grammar (CCG)
- Lexical Functional Grammar (LFG)
- Head-driven Phrase Structure Grammar (HPSG)
Grammar Engineering Platform

- Implementation of the formalism (description language)
- Grammar editor
- Processor: parser, generator, etc.
- Graphical UI
- Other tools:
  - Profiling system
  - Treebanking tools
  - …
Linguistic Resources

- Corpora / Treebanks / Testsuites / Lexica
- Reference Grammars
  - Existing grammars for other languages on the same platform
  - Existing grammars for the same language on other platforms
  - Universal reusable grammar components
Hand-Crafted Grammars v.s. Treebank-Induced Grammars

Hand-crafted grammars
- Long development circle
- High linguistic precision can be achieved
- Grammar maintenance is difficult
- (Potentially) suitable for different tasks

Treebank-induced grammars
- Large annotated treebank is required
- Semi-automatic grammar adaptation and extension
- More suitable for parsing than generation
Outline

1. Grammar engineering
   - What?
   - Why?
   - Who?
   - How?

2. HPSG and DELPH-IN

3. Structure of the course
HP SG as a linguistic theory

- Highly consistent and powerful formalism ($TFS$)
- Monostratal, declarative, non-derivational, lexicalist, constraint-based
- Has been studied for many different languages
- Psycholinguistic evidence
Mathematical foundation

Typed Feature Structure

- [Carpenter, 1992]
- High expressive power
- Parsing complexity: exponential (to the input length)
  - Tractable with efficient parsing algorithms
  - Efficiency can be improved with a well designed grammar
Mathematical foundation

Typed Feature Structure

- [Carpenter, 1992]
- High expressive power
- Parsing complexity: exponential (to the input length)
  - Tractable with efficient parsing algorithms
  - Efficiency can be improved with a well designed grammar
Mathematical foundation

Typed Feature Structure

- [Carpenter, 1992]
- High expressive power
- Parsing complexity: exponential (to the input length)
  - Tractable with efficient parsing algorithms
  - Efficiency can be improved with a well designed grammar
Mathematical foundation

Typed Feature Structure

- [Carpenter, 1992]
- High expressive power
- Parsing complexity: exponential (to the input length)
  - Tractable with efficient parsing algorithms
  - Efficiency can be improved with a well designed grammar
Implementations

- Mismatch exists between theory and reality
- Trade-off has to be made for various practical reasons
- What makes a good implementation?
  - Faithfulness to the linguistic theory
  - Reliability of the software toolchain
  - Extensibility
DELPH-IN
Deep Linguistic Processing with HPSG – Initiative

- Heart-of-Gold
- [incr tsdb()]
- LKB
  - Generator
  - Parser
- PET
- Grammar
  - ERG
  - JaCY
  - GG
  - Matrix
- HPSG
- MRS
- TDL
A glance over existing grammar engineering platforms

- TAG
  - XTAG
- LFG
  - XLE
- HPSG
  - Enju (English)
  - Alpino/HDRUG (Dutch)
  - TRALE
Outline

1. Grammar engineering
   - What?
   - Why?
   - Who?
   - How?

2. HPSG and DELPH-IN

3. Structure of the course
Things to master

- HPSG
- MRS
- LKB
- [incr tsdb()]
Part I: LKB, TFS and TDL

- Hands-on introduction to LKB [Copestake, 2002]
- Type description language
Part II: Minimal Recursion Semantics (MRS)

- [Copestake et al., 2005]
- Semantically informed grammar engineering
- Compositional semantics
Part III: Matrix

- [Bender et al., 2002]
- Shared components of a HPSG across different languages
- Implemented language phenomena groups as libraries and plug-ins
Part IV: Implementations of various language phenomena

- Subcategorization
- Modification and agreements
- Long distance dependencies
- More
Questions?
The Grammar Matrix: an open-source starter-kit for the rapid development of cross-linguistically consistent
broad-coverage precision grammars.
In Proceedings of the Workshop on Grammar Engineering and Evaluation at the 19th International
Conference on Computational Linguistics, pages 8–14, Taipei, Taiwan.

The Logic of Typed Feature Structures.
Cambridge University Press, Cambridge, UK.

Implementing Typed Feature Structure Grammars.
CSLI, Stanford, USA.

Minimal recursion semantics: an introduction.