Grammar Engineering for Deep Linguistic Processing

Lecture 2: TFS and LKB

Yi Zhang

Department of Computational Linguistics & Phonetics
Saarland University

Language Technology Lab
German Research Center for Artificial Intelligence

May, 2012
Outline

1. Typed Feature Structures (TFS) – A DELPH-IN View
2. Linguistic Knowledge Builder (LKB)
3. LKB Grammar Files
Typed Feature Structures

- Recall the sessions on the typed feature structures in the Syntactic Theory class
  http://www.coli.uni-saarland.de/courses/syntactic-theory-09/slides/tfs.pdf
- Refer to [Carpenter, 1992] for details, or [Copestake, 2000] for a brief summary
Typed Feature Structures In Unification-Based Grammar Development

- Feature structure and feature structure description
- Type inheritance hierarchy
- Unification
Feature Structure

- A feature structure is a set of attribute-value pairs
- Each attribute (or feature) is an atomic symbol
- The value of each attribute can be either atomic, or complex (a feature structure, a list, or a set)

```
CATEGORY   noun-phrase
AGREEMENT  
            [PERSON  3rd]
            [NUMBER  sing]
```
Feature Structure & Description

Theoretically . . .

- A feature structure is essentially a directed (acyclic) graph with complete description of the underlying linguistic object
- A (partial) feature structure description can be given in the form of attribute-value matrix (AVM)

Practically . . .

- In constraint-based descriptive grammar engineering, two terms are used interchangeably: grammar descriptions are always partial
A typed feature structure is composed of two parts

- A type
- A (possibly empty) set of attribute-value pairs with each value being a TFS
Properties of Typed Feature Structure

- **Finiteness:** a typed feature structure has a finite number of nodes
- **Unique root and connectedness:** a typed feature structure has a unique root node; apart from the root, all nodes have at least one parent
- **No cycles:** no node has an arc that points back to the root node or to another node that intervenes between the node itself and the root
- **Unique features:** no node has two features with the same name and different values
- **Typing:** each node has single type which is defined in the hierarchy
Multiple Inheritance Type Hierarchy

In the view of constraint-based grammar

- A unique most general type: \(*_{\text{TOP}}* \top []\)
- Each non-top type has one or more parent type(s)
- Two types are compatible iff they share at least one offspring type
- Each non-top type is associated with optional constraints
  - Constraints specified in ancestor types are monotonically inherited
  - Constraints (either inherited, or newly introduced) must be compatible
Unification

The unification result on two TFSes $TFS_a$ and $TFS_b$ is:

- $\bot$, if either one of the following:
  - type $a$ and $b$ are incompatible
  - unification of values for attribute $X$ in $TFS_a$ and $TFS_b$ returns $\bot$

- a new TFS, with:
  - the most general shared subtype of $a$ and $b$
  - a set of attribute-value pairs being the results of unifications on sub-TFSes of $TFS_a$ and $TFS_b$
GLB Types

- In case of multiple inheritance, two types can have more than one shared subtype that neither is more general than the others
- Non-deterministic unification results
- Type hierarchy can be modified to avoid this

```
*TOP*
   /  \
  a   b
 /   /  \     \  /   /
c   d   e   glb(a,b)   c   d   e
```

⇒

```
*TOP*
   /  \
  a   b
 /     \
 e     glb(a,b)
   /   /  \     \  /   /
c   d   e   c   d   e
```
An Impressionist’s Slide
Outline

1. Typed Feature Structures (TFS) – A DELPH-IN View

2. Linguistic Knowledge Builder (LKB)

3. LKB Grammar Files
LKB—Linguistic Knowledge Builder

- [Copestake, 2002]
- A grammar and lexicon development environment for use with unification-based linguistic formalisms
- http://wiki.delph-in.net/moin/LkbTop
General Information

- Implemented in Common Lisp
  - Core works for different CL implementations
  - GUI requires Allegro CL
- Extensible architecture including the following components:
  - Reference implementation of the formalism (TDL)
  - MRS library
  - Bottom-up chart parser
  - MRS-based text generator
  - Finite-state pre-processor
  - Lexical database (Lex-DB)
  - User interface
    - Old-fashioned GUI based on CLIM
    - New GUI called LUI
    - Emacs integration
- ...
A short tour in LKB . . .
Outline

1. Typed Feature Structures (TFS) – A DELPH-IN View

2. Linguistic Knowledge Builder (LKB)

3. LKB Grammar Files
What Makes A LKB Grammar?

- Types and constraints (*types.tdl*)
- Lexical entries (*lexicon.tdl*)
- Grammar rules (*rules.tdl*)
- Lexical and morphological rules (*lrules.tdl irules.tdl*)
- Start symbol descriptions (*start.tdl*)
- Parse node descriptions (*parse-nodes.tdl*)

Auxiliary settings

- *script* which loads various files in the grammar
- *globals.lsp* which contains global settings
- *user-fns.lsp* which contains user defined lisp functions
- *user-prefs.lsp* which contains user preference settings
What Makes A LKB Grammar?

- Types and constraints (*types.tdl*)
- Lexical entries (*lexicon.tdl*)
- Grammar rules (*rules.tdl*)
- Lexical and morphological rules (*lrules.tdl* *irules.tdl*)
  * Start symbol descriptions (*start.tdl*)
  * Parse node descriptions (*parse-nodes.tdl*)
- Auxiliary settings
  - *script* which loads various files in the grammar
  - *globals.lsp* which contains global settings
  - *user-fns.lsp* which contains user defined lisp functions
  - *user-prefs.lsp* which contains user preference settings
The Script File

(lkb-load-lisp (this-directory) "globals.lsp")
(lkb-load-lisp (this-directory) "user-fns.lsp")
(load-lkb-preferences (this-directory) "user-prefs.lsp")
(read-tdl-type-files-aux
   (list (lkb-pathname (this-directory) "types.tdl")))
(read-tdl-lex-file-aux
   (lkb-pathname (this-directory) "lexicon.tdl"))

(read-tdl-grammar-file-aux
   (lkb-pathname (this-directory) "rules.tdl"))
(read-tdl-start-file-aux
   (lkb-pathname (this-directory) "start.tdl"))
(read-tdl-parse-node-file-aux
   (lkb-pathname (this-directory) "parse-nodes.tdl"))

... ...
References I

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

Definitions of typed feature structures.
*Natural Language Engineering (appendix to special issue on efficient processing with HPSG),* 6(1).

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