# From “hand-written” to computationally implemented HPSG theories

Nurit Melnik

Caesarea Rothschild Institute
for Interdisciplinary Applications
of Computer Science
Haifa University
nurit@eyron.com

August 24, 2005

# Motivations

**Why implement an HPSG theory?**

- Practical NLP applications (e.g,. The DELPHIN Collaboration)
- Language documentation (e.g., The Montage project (Bender et al., 2004))
- Evaluation of linguistic hypotheses

# Motivations

**Why implement an HPSG theory?**

- Practical NLP applications (e.g,. The DELPHIN Collaboration)

- Language documentation (e.g., The Montage project (Bender et al., 2004))

- Evaluation of linguistic hypotheses
  - internal consistency
  - interaction of a set of hypotheses
  - test suites
  - ‘real’ corpus data

# The “Hand-written” Theory

**Verb-initial Constructions in Modern Hebrew** (Melnik, 2002)

- SV-VS word order alternations
- Subject-verb agreement patterns
- Valence alternations: canonical and subjectless
- Possessive Dative Construction

# Platforms

**The LKB system** (The Linguistic Knowledge Building; Copestake, 2002)

- Primary engineering environment of the LinGO English Resource Grammar (ERG; (Copestake & Flickinger, 2000))
- Implemented in Common Lisp

**TRALE** (Meurers et al., 2002)

- An extension of the Attribute Logic Engine (ALE; (Carpenter and Penn, 1995 & 1999)
- Implemented in SICStus Prolog 3.8.6.

# Dimensions of Comparison

- Type definition
- Exhaustive Typing and Subtype Covering
- Principles
- Lexical rules
- Grammar rules
- Definite relations
- Semantic representation
- Grammar evaluation

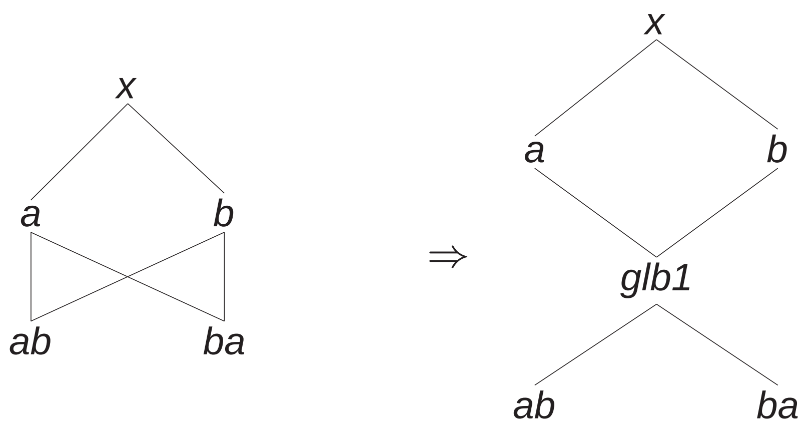
# Type Definition

## Properties

- The type’s hierarchical relation to other types
- Appropriateness Conditions
- Type Constraints

# The Type Hierarchy

**The glb condition**: Every set of types which are compatible must have a unique greatest subtype (*greatest lower bound* (glb) or *most general unifier* (mgu)).

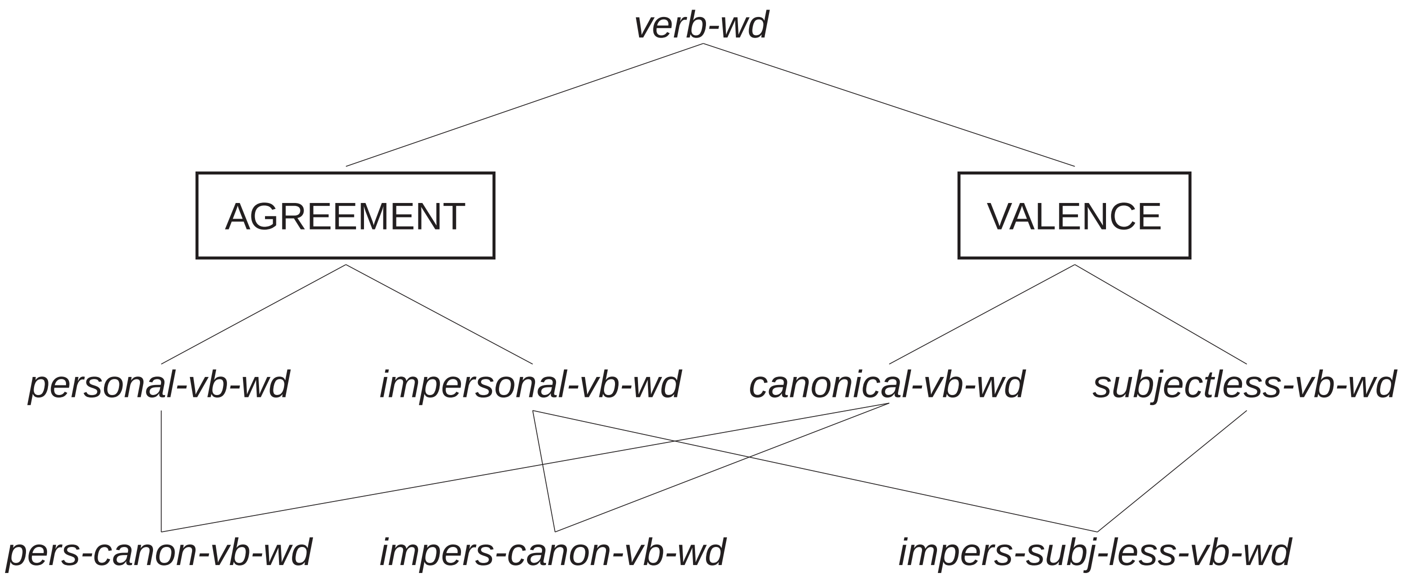


- The LKB automatically restructures a violating hierarchy by inserting *glb* types (see above).

- TRALE produces an error message.

# The Type Hierarchy

## Multi-dimensional inheritance



- The LKB and TRALE do not provide a way for implementing multi-dimensional inheritance.

# Type Definition

## Properties

- The type's hierarchical relation to other types
- Appropriateness Conditions: The features which a type has and the values these features can have
- Type Constraints: Values of embedded features and path equations

## Type Inheritance

- In TRALE inheritance is monotonic.
- The LKB allows default inheritance in the type hierarchy (Lascarides and Copestake, 1999).

# Type Definition - The LKB

```
sign := *top* &
    [ORTH   string,
     SYNSEM synsem,
     ARGS *list*
    ].

agr-pers-vb-wd := verb-wd &
   [SYNSEM.LOC.CAT [HEAD.AGR #agr,
                    VAL.SUBJ < synsem &
                                 [LOC [CAT.HEAD noun &
                                        [AGR #agr]]] > ]].

impers-canon-vb-wd := val-canon-vb-wd &
                      agr-impers-vb-wd &
    [SYNSEM.LOC.CAT [VAL.SUBJ < synsem &
                      LOC.CAT.HEAD non-nom-pos ] > ]].
```

# Type Definition - TRALE

## The signature

```
bot
  sign phon:ne_list synsem:synsem
      lex_item
          word
              verb_wd
                  agr_personal_vb_wd
                      personal_canonical_vb_wd
                  agr_impersonal_vb_wd
                      impersonal_canonical_vb_wd
                      impersonal_subjectless_vb_wd
                  val_canonical_vb_wd
                      &personal_canonical_vb_wd
                      &impersonal_canonical_vb_wd
                  val_subjectless_vb_wd
                      &impersonal_subjectless_vb_wd
```

# Type Definition - TRALE

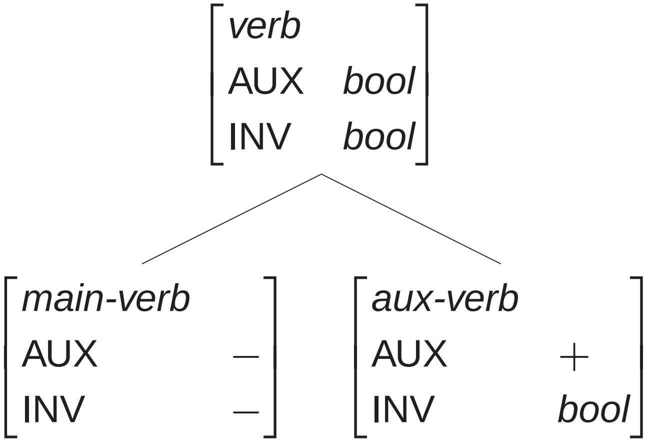
## The theory

```
agr_personal_vb_wd *>
         (synsem:loc:cat:(head:agr:Agr,
                          val:subj:[loc:cat:head:(noun,agr:Agr)])).

impersonal_canonical_vb_wd *>
         (synsem:loc:cat:val:subj:[loc:cat:head:non_nom_pos]).
```

# Exhaustive Typing and Subtype Covering

(*aka Open vs. Closed World Reasoning*)



$$\begin{bmatrix} \textit{verb} & \\ \text{AUX} & \textit{bool} \\ \text{INV} & \textit{bool} \end{bmatrix}$$

$$\begin{bmatrix} \textit{main-verb} & \\ \text{AUX} & - \\ \text{INV} & - \end{bmatrix} \quad \begin{bmatrix} \textit{aux-verb} & \\ \text{AUX} & + \\ \text{INV} & \textit{bool} \end{bmatrix}$$

- The LKB accepts $\begin{bmatrix} \textit{verb} & \\ \text{AUX} & - \\ \text{INV} & + \end{bmatrix}$

- TRALE rejects $\begin{bmatrix} \textit{verb} & \\ \text{AUX} & - \\ \text{INV} & + \end{bmatrix}$ and promotes $\begin{bmatrix} \textit{verb} & \\ \text{AUX} & - \\ \text{INV} & \textit{bool} \end{bmatrix}$ to $\begin{bmatrix} \textit{verb} & \\ \text{AUX} & - \\ \text{INV} & - \end{bmatrix}$

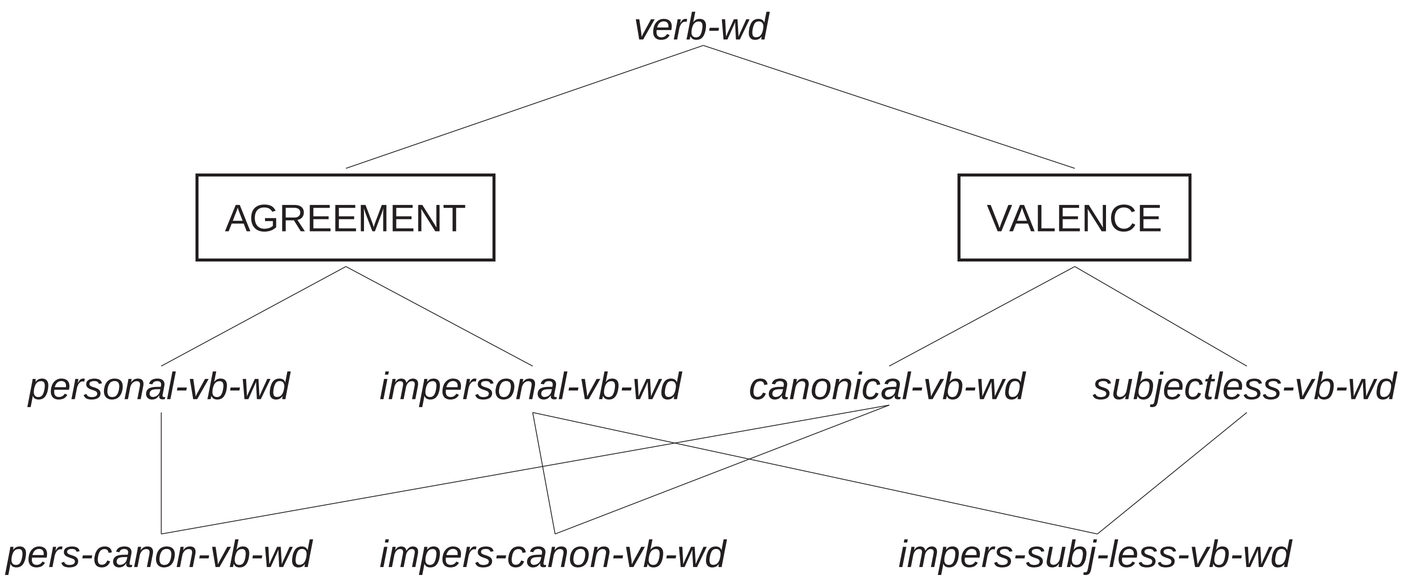
# Principles

**Implicational Constraints:**
**Type Antecedents vs. Complex Antecedents**

Example:
The Verbal Agreement Principle in Modern Hebrew

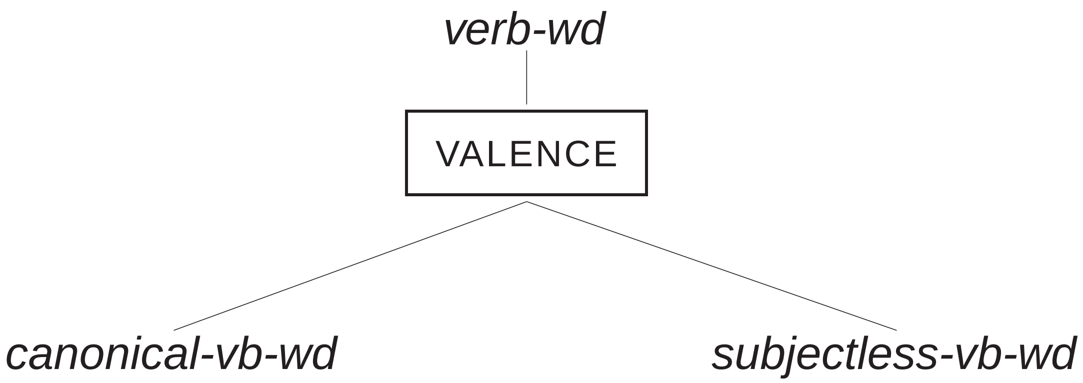
- Verbs with NP subjects exhibit full agreement with the subject.
- Verbs with non-NP subjects exhibit impersonal 3SM agreement.
- Subjectless verbs exhibit impersonal 3SM agreement.

# Implicational Constraints with Type Antecedents



- ***pers-canon-vb-wd***: Agreeing verbs with NP subjects
- ***impers-canon-vb-wd***: Impersonal-agreement verbs with non-NP subjects
- ***impers-subj-less-vb-wd***: Subjectless impersonal-agreement verbs

# Implicational Constraints with Complex Antecedents



- Agreeing verbs with NP subjects

$$\begin{bmatrix} \textit{verb-wd} \\ \text{SYNSEM} \mid \ldots \mid \text{SUBJ} \left\langle \begin{bmatrix} \text{LOC} \mid \text{CAT} \mid \text{HEAD } \textit{noun} \end{bmatrix} \right\rangle \end{bmatrix} \longrightarrow \begin{bmatrix} \textit{verb-wd} \\ \text{SYNSEM} \mid \text{LOC} \mid \text{CAT} \begin{bmatrix} \text{HEAD} \mid \text{AGR } \boxed{1} \\ \text{VAL} \mid \text{SUBJ} \left\langle \begin{bmatrix} \text{LOC} \mid \text{CAT} \mid \text{HEAD} \mid \text{AGR } \boxed{1} \end{bmatrix} \right\rangle \end{bmatrix} \end{bmatrix}$$

- Impersonal-agreement verbs

$$\begin{bmatrix} \textit{verb-wd} \\ \text{SYNSEM} \mid \ldots \mid \text{SUBJ } \textit{elist} \vee \left\langle \begin{bmatrix} \textit{LOC} \mid \textit{CAT} \mid \textit{HEAD non-noun} \end{bmatrix} \right\rangle \end{bmatrix} \longrightarrow \begin{bmatrix} \textit{verb-wd} \\ \text{SYNSEM} \mid \text{LOC} \mid \text{CAT} \mid \text{HEAD} \mid \text{AGR } \textit{3SM} \end{bmatrix}$$

# Principles

## Type Antecedents vs. Complex Antecedents

- The LKB support implicational constraints with type antecedents.
- TRALE support implicational constraints with type antecedents and complex antecedents.

## Disjunction

- TRALE's description language includes disjunction.
- The LKB's description language does not include disjunction.

# Lexical Rules

## The LKB

- Lexical rules are viewed as unary grammar rules
- LRs relate a mother structure (the output) to its daughter (the input)
- Explicit specification of information that is copied over from input to output
- Hierarchical rules with defeasible default values

## TRALE - two mechanisms

- The traditional ALE mechanism
  - Requires explicit copying over from input to output
- Description Level LRs (DLRs; Meurers & Minnen, 1997)
  - Employs default reasoning to copy over from input to output

# Fixed vs. Variable Arity in Grammar Rules

## The challenge

$$\left[\begin{array}{ll} \textit{hd-comp-ph} & \\ \text{COMPS} & \langle\rangle \\ \text{HD-DTR} & \left[\text{COMPS}\left\langle \boxed{1},\ldots,\boxed{n}\right\rangle\right] \\ \text{NON-HD-DTRS} & \left\langle \left[\text{SYNSEM}\ \boxed{1}\right],\ldots,\left[\text{SYNSEM}\ \boxed{n}\right]\right\rangle \end{array}\right]$$

## The solutions

- In the LKB the number of daughters in a grammar rule is fixed
  - Separate rule for each arity
    OR
  - Binary branching

- TRALE provides a special `cats>` operator for variable arity rules

# Definite Relations

TRALE provides a Prolog-like definite logic programming language with which the grammar writer can encode definite relations.

- A grammar rule

```
head_complement_schema_rule ##
  (hd_comp_phrase,
   hd_dtr:Head,
   non_hd_dtr:CompDtrs)
 ===>
   cat> (Head,word, synsem:loc:cat:val:comps:CompDtrsSynsem),
   goal> list_sign_to_synsem(CompDtrs,CompDtrsSynsem),
   cats> CompDtrs.
```

- A definite relation

```
sign_to_synsem(synsem:Synsem,Synsem) if true.
list_sign_to_synsem([],[]) if true.
list_sign_to_synsem([H|T], [S|R]) if
    sign_to_synsem(H,S),
    list_sign_to_synsem(T,R).
```

# Semantic Representation

- The LKB contains a module for processing Minimal Recursive Semantics (MRS) representations (Copestake and Flickinger, 2000).

- TRALE provides a module which is an implementation of Lexical Resource Semantics (Penn and Richter, 2004).

# Grammar evaluation

## Grammar evaluation tools

- Batch parsing of test suites (The LKB and TRALE)
- The `[incr tsdb()]` package (The LKB and TRALE)

## Platform performance comparison

- No evidence from the test case grammar
- Comparison of TRALE's MERGE (Ver. 0.9.6) and the LKB's ERG (Penn, 2004)
  - Similar coverage
  - Different internal systems
  - MERGE runs 300 times slower than the ERG

# Conclusion

## Expressiveness

- TRALE
  - √ Implicational constraints with complex antecedents
  - √ Variable arity in grammar rules
  - √ Definite relations
  - √ Exhaustive Typing and Subtype Covering
- LKB
  - √ Default inheritance

# Conclusion

**Accessibility - computational skills required**

- LKB
  - √ Automatic correction of glb violations
  - √ Interactive type hierarchy display
  - √ Matrix open-source starter-kit (Bender et al,. 2002)
  - √ Runs on Windows
- TRALE
  - √ Grisu's user-friendly feature-structure and syntactic-tree display BUT
  - ∼ Programming Prolog definite relations
  - ∼ Parametric macros
  - ∼ Does not run on Windows (modulo the Grammix CD Rom; Müller)

# Final Note

**Should “hand-written” HPSG theories**

**=**

**computationally implemented HPSG theories**?

# Thank You