Towards a competitive dependency grammar formalism

Ralph Debusmann

rade@ps.uni-sb.de

Programming Systems Lab

Universität des Saarlandes

Dependency grammar

- idea: words in a sentence depend on each other
- dates back to the middle ages
- modern dependency grammar: Tesniere (1959) (also: Melcuk, Sgall)
- dependency analysis = dependency tree (*1)

Comparison with context-free grammar

(*2)

	CFG	DG
nodes	phrasal and terminal	only terminal
gr. function	no, only categories	first class citizen
word order	rules	no, unconstrained
lexicalization	no, rule-based	lexicalized (valency)

Valency

- in the lexicon, each word specifies its daughters, similar to subcategorization in HPSG
- e.g. *liebt* requires a subject and an object:

$$\textit{liebt} \quad = \quad \left[\ \mathsf{valency\_syn} \quad : \quad \{\mathsf{subj!}, \mathsf{obj!}\}\ \right]$$

- *Frau* requires a determiner and may be modified by arbitrary many adjectives:

$$\textit{Frau} \quad = \quad \left[\ \mathsf{valency\_syn} \quad : \quad \{\mathsf{det!}, \mathsf{adj}*\}\ \right]$$

Word Order

- so far: no assumptions about word order
- but: need to state word order constraints to prevent overgeneration
- existing formalisms often invent non-declarative extensions to constrain word order

The failure of dependency grammar

- some of the most well-known formalisms:
  - Meaning Text Theory, Melcuk (1988)
  - Functional Generative Description, Sgall et al (1986)
  - Word Grammar, Hudson (1990)
  - Slot Grammar, McCord (1990)
- none of these as successful as e.g. HPSG in the linguistic mainstream
- why? lack of declarativity (esp. word order)

Topological Dependency Grammar

- new dependency grammar formalism, described in Duchier/Debusmann 2001 (ACL) and Debusmann 2001 (Diplomarbeit)
- includes a new, declarative way of adding word order constraints to dependency grammar
- dependency relations clearly separated from word order
- makes use of topological fields theory

Topological fields theory

- traditional descriptive theory (Herling 1821, Erdmann 1886) for German word order
- divides a sentence into contiguous substrings and assigns to them so-called *topological fields*
- topological fields: Vorfeld, left sentence bracket, Mittelfeld, right sentence bracket, Nachfeld
- e.g.:

Vorfeld	(	Mittelfeld	)
Peter	*liebt*	*eine blonde Frau.*	
Peter	*versucht*	*eine Frau*	*zu lieben.*

Topological trees

- idea: add to the dependency tree a second level of analysis: the topology tree (*3)
- two differences
  1. edge labels = topological fields
  2. daughters of each node ordered by the label of their incoming edge
- the topology tree is a flattening of the dependency tree
- same notion of valency

Topological valency

- lexical entry now specifies *syntactic valency* and *topological valency*
- e.g. *liebt* may have at most one daughter in the Vorfeld and an arbitrary number of daughters in the Mittelfeld:

$$\textit{liebt} :$$

$$\left[\begin{array}{lll} \mathsf{valency\_syn} & : & \{\mathsf{subj}!, \mathsf{obj}!\} \\ \mathsf{valency\_top} & : & \{\mathsf{vf}?, \mathsf{mf}*, \mathsf{nf}?\} \end{array}\right]$$

TDG overview

- copes well with languages with a high degree of word order freedom (e.g. German, Dutch)
- highly lexicalized, lexical inheritance to state linguistic generalizations
- efficient parser, uses constraint technology developed at Programming Systems Lab
- but: so far only covers syntax

Advancing to semantics

- plan of my doctoral thesis: extend TDG grammar formalism with a concurrent syntax-semantics interface
- concurrency = syntax inferences can rule out semantic readings and vice versa
- falls out rather naturally from the constraint-based approach to parsing
- facilitates incorporation of preferences (CHORUS)

Recovering predicate-argument structure

- problem: dependency trees reflect only syntactic argument structure
- does not always match semantic predicate-argument structure, e.g. passive, control/raising
- need a more semantically oriented structure to recover predicate-argument structure

Thematic graphs

- idea: add a new level of representation to the dependency and topology tree levels: thematic graphs (*4)
- represent semantic predicate-argument structure
- edge labels = thematic roles (agent, patient...)
- again: notion of valency re-used

Thematic valency

- each lexical entry now specifies syntactic, topological and thematic valencies
- e.g. *liebt* must have an agent and a patient:

$$\textit{liebt} : \left[\begin{array}{rcl} \mathsf{valency\_syn} & : & \{\mathsf{subj!}, \mathsf{obj!}\} \\ \mathsf{valency\_top} & : & \{\mathsf{vf?}, \mathsf{mf}*\} \\ \mathsf{valency\_them} & : & \{\mathsf{ag!}, \mathsf{pt!}\} \end{array}\right]$$

Linking

- problem: need to link the thematic graph and the dependency tree
- idea: thematic roles realized as dependency relations
- e.g. for *liebt*: agent realized as the subject, and patient realized as the object:

$$
\textit{liebt} : \left[\begin{array}{rcl}
\mathsf{valency\_syn} & : & \{\mathsf{subj!}, \mathsf{obj!}\} \\
\mathsf{valency\_top} & : & \{\mathsf{vf?}, \mathsf{mf}*\} \\
\mathsf{valency\_them} & : & \{\mathsf{ag!}, \mathsf{pt!}\} \\
\mathsf{link} & : & \{\mathsf{ag} : \mathsf{subj}, \mathsf{pt} : \mathsf{obj}\}
\end{array}\right]
$$

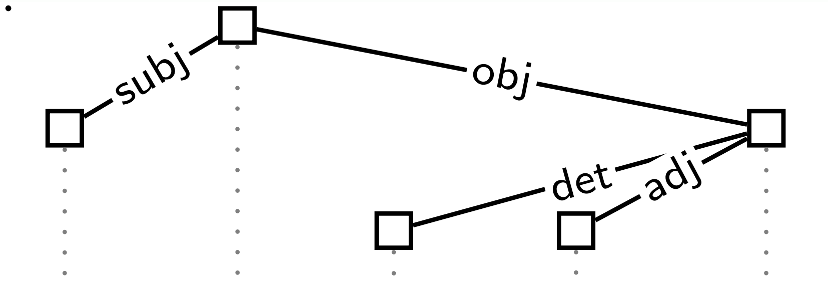
State of the art

- new implementation of the dependency parser including the thematic graph level
- also includes a scope tree-level to represent quantifier scope (not talked about here)
- no breakdown in parsing performance

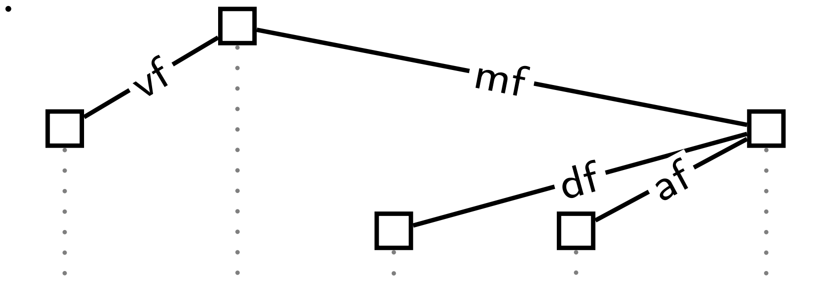
Still to do

- construct semantics (e.g. CLLS) from the thematic graph and the scope tree
- incorporate preferences (CHORUS)
- improve the grammar formalism (e.g. lexical rules)
- improve grammar coverage
- improve implementation (parser, GUI)

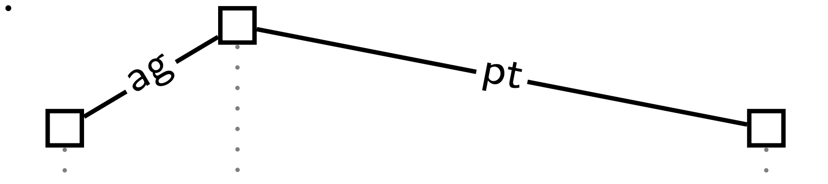
X1: Levels of analysis



- (*1) Peter liebt eine blonde Frau



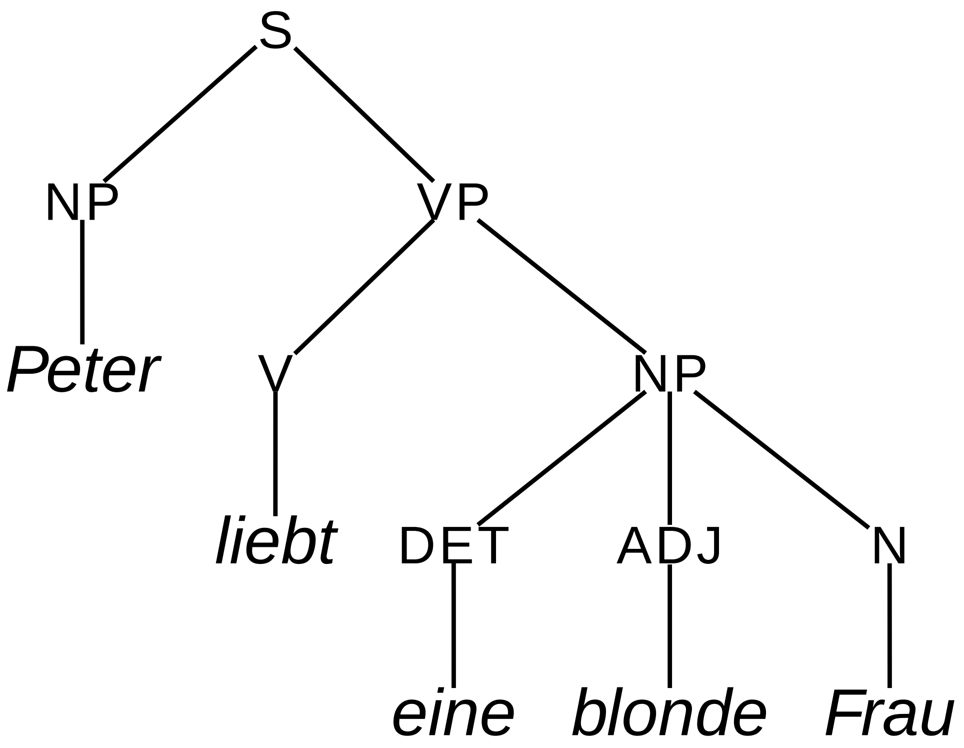
- (*3) Peter liebt eine blonde Frau



- (*4) Peter liebt eine blonde Frau

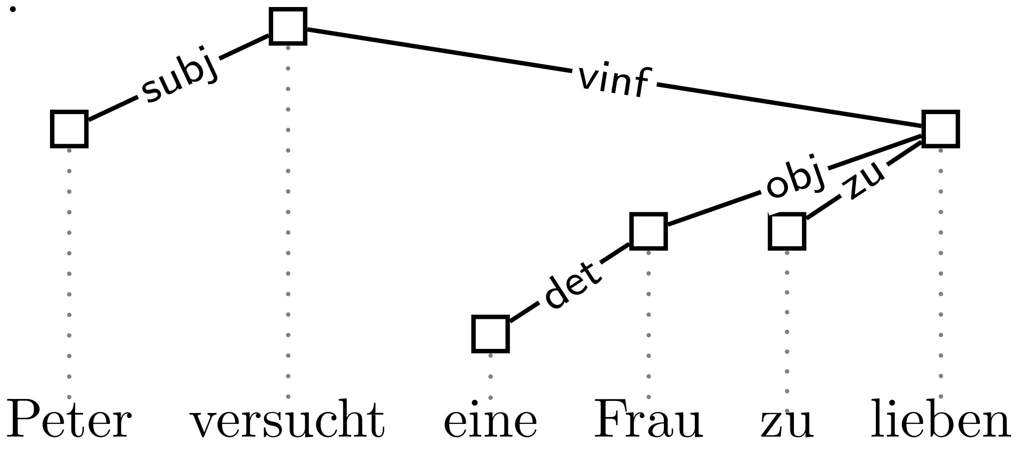
X2: Phrase structure analysis

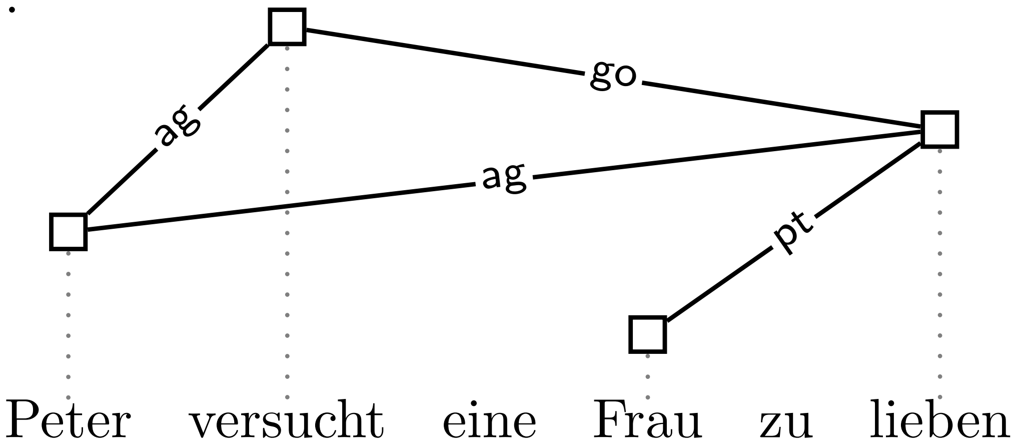
- (*2)



X3: Control and raising

- subjects need not be realized locally but also higher in the dependency tree:


subj
vinf
obj
zu
det
Peter
versucht
eine
Frau
zu
lieben


go
ag
ag
pt
Peter
versucht
eine
Frau
zu
lieben