Multi-dimensional Dependency Grammar as Graph Description

Ralph Debusmann and Gert Smolka

Programming Systems Lab, Saarbrücken, Germany

FLAIRS-19, May 11th, 2006

Overview

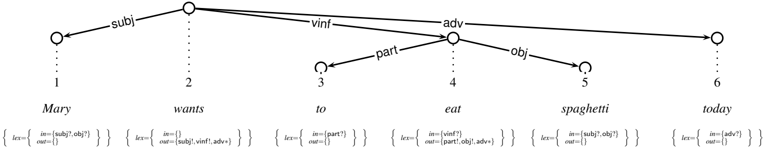
Overview

Two Trends in Natural Language Processing

- dependency grammar (Tesniere 1959), (Mel'čuk 1988)
- multi-layered linguistic description

Dependency Grammar

- collection of ideas for the analysis of natural language
- example analysis of *Mary wants to eat spaghetti today*:



- graph, 1:1-mapping nodes:words, dependency relations, valency
- e.g.: *wants*:

$$\left\{ \; lex = \left\{ \begin{array}{rcl} in & = & \{\} \\ out & = & \{\mathsf{subj!}, \mathsf{vinf!}, \mathsf{adv}*\} \end{array} \right\} \; \right\}$$

Dependency Grammar as a trend

- incorporated into grammar formalisms: CCG (Steedman 2000), HPSG (Pollard/Sag 1994), LFG (Bresnan/Kaplan 1982), TAG (Joshi 1987)
- indispensable for statistical parsing (Collins 1999)
- treebanks: Prague Dependency Treebank (Bohmova et al. 2001), Danish Dependency Bank, TiGer Dependency Bank (Forst et al. 2004)

Multi-layered Linguistic Description

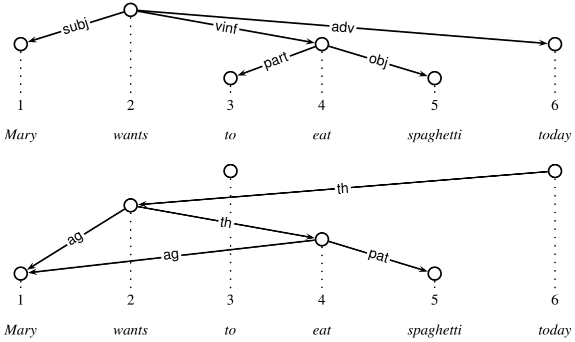
- additional layers of annotation
- predicate-argument structure: PropBank (Kingsbury/Palmer 2002), SALSA (Erk et al. 2003), tectogrammatical structure of the PDT
- information structure: PDT
- discourse structure: Penn Discourse Treebank (Webber et al. 2005)
- annotation: mostly dependency-based
- can we represent these layers as modules in one framework based on dependency grammar?

Extensible Dependency Grammar (XDG)

- new grammar formalism (Debusmann 2006 PhD)
- supports arbitrary many layers of linguistic description called "dimensions", all sharing the same set of nodes
- model-theoretic: models called "multigraphs"

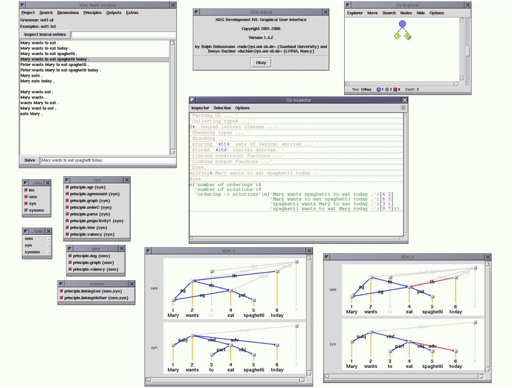
Multigraph

- syntax and predicate-argument structure:



Implementation

- concurrent constraint-based parser written in Mozart/Oz (Mozart06)
- XDG Development Kit (XDK) (Debusmann et al. 2004 MOZ)



Application

- German syntax (Duchier/Debusmann 2001 ACL), (Debusmann 2001), (Bader et al. 2004)
- Arabic syntax (Odeh 2004)
- English syntax (Debusmann 2006 PhD)
- relational syntax-semantics interface (Debusmann et al. 2004 COLING)
- prosodic account of information structure (Debusmann et al 2005 CICLING)

Two Stumbling Blocks

1. no complete formalization (Debusmann et al. 2005 FG-MOL)
2. no efficient large-scale parsing (Bojar 2004), (Moehl 2004), (Narendranath 2004)

Overview

A Description Language for Multigraphs

- formalization as a description language for multigraphs in higher order logic
- expressed in simply typed lambda calculus extended with finite domains and records
- types, given set of atoms At:

$$
\begin{array}{llll}
a \in At & & & \\
T \in Ty & ::= & \mathsf{B} & \text{boolean} \\
 & \mid & \mathsf{V} & \text{node} \\
 & \mid & T_1 \to T_2 & \text{function} \\
 & \mid & \{a_1, \ldots, a_n\} & \text{finite domain } (n \geq 1) \\
 & \mid & \{a_1 : T_1, \ldots, a_n : T_n\} & \text{record}
\end{array}
$$

- interpretation: $\mathsf{B} = \{0, 1\}$, $\mathsf{V} = \{1, 2, \ldots, n\}$ given n nodes, i.e., both base types finite

Multigraph Type

- signature of XDG varies according to the dimensions, words, edge labels and attributes of the described multigraphs
- multigraph type: $MT = (Dim, Word, lab, attr)$
- domains of dimensions and words must be finite

Signature

- multigraph constants, given multigraph type $MT = (Dim, Word, lab, attr)$:

$$
\begin{array}{rcll}
\stackrel{\cdot}{\longrightarrow}_d & : & \mathsf{V} \rightarrow \mathsf{V} \rightarrow lab\ d \rightarrow \mathsf{B} & \text{labeled edge } (d \in Dim) \\
< & : & \mathsf{V} \rightarrow \mathsf{V} \rightarrow \mathsf{B} & \text{precedence} \\
(W\ \cdot) & : & \mathsf{V} \rightarrow Word & \text{node-word mapping} \\
(d\ \cdot) & : & \mathsf{V} \rightarrow attr\ d & \text{node-attributes mapping } (d \in Dim)
\end{array}
$$

- logical constant:

$$
\doteq_T \quad : \quad T \rightarrow T \rightarrow \mathsf{B} \quad \text{equality (for each type } T\text{)}
$$

Grammar, models and string language

- grammar: $G = (MT, P)$
- P set of formulas called “principles”, i.e., the well-formedness conditions
- models: all multigraphs with multigraph type MT and which satisfy P
- string language: set of all strings $s = w_1 \dots w_n$ such that:
  1. there are as many nodes as words: $V = \{1, \dots, n\}$
  2. concatenating the words of the nodes yields s: $(W\ 1) \dots (W\ n) = s$

Tree Principle

- three conditions:
  1. There are no cycles.
  2. There is precisely one root.
  3. Each node has at most one incoming edge.
- principle definition:

$$
\begin{aligned}
tree_d \quad = \quad & \forall v : \neg(v \rightarrow_d^+ v) \quad \wedge \\
& \exists^1 v : \neg \exists v' : v' \rightarrow_d v \quad \wedge \\
& \forall v : (\neg \exists v' : v' \rightarrow_d v) \vee (\exists^1 v' : v' \rightarrow_d v)
\end{aligned}
$$

Other Principles

- DAG
- valency
- order
- projectivity
- agreement
- linking
- etc. (Debusmann 2006 PhD)

Overview

Recognition Problems

- universal recognition problem: given a pair (G, s) where G is a grammar and s a string, is s in $L(G)$?
- fixed recognition problem: let G be a fixed grammar. Given a string s, is s in $L(G)$?
- plan: prove NP-hardness of the fixed recognition problem, NP-hardness of the universal then falls out

Reduction

- proof by reducing the NP-complete SAT problem to the fixed XDG recognition problem
- SAT: does a propositional formula f have an assignment that evaluates to true?
- propositional formula:

$$\begin{array}{rll} f ::= & X, Y, Z, \ldots & \text{variable} \\ \mid & 0 & \text{false} \\ \mid & f_1 \Rightarrow f_2 & \text{implication} \end{array}$$

Input Preparation

- 2 challenges:
  1. propositional formulas can be ambiguous
  2. can contain arbitrary many variables, but an XDG grammar only has a finite set of words
- input preparation function: $prep : f \longrightarrow Word$
- example formula: $(X \Rightarrow Y) \Rightarrow Y$
  1. prefix notation:

$$\Rightarrow \Rightarrow X\ Y\ Y$$

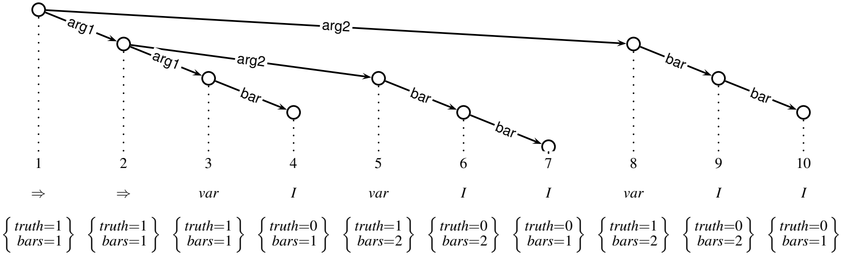
  2. unary encoding:

$\Rightarrow \Rightarrow$ *var I var I I var I I*

Models

- representation of the example formula $(X \Rightarrow Y) \Rightarrow Y$:

$\Rightarrow \Rightarrow$ *var I var I I var I I*



Coreference

- which type for the “bars” attribute?
- idea: use V, whose interpretation is a finite interval of the natural numbers starting with 1, because:
  1. there are always more nodes in the analysis than variables in the formula, i.e., V always includes enough elements to distinguish all variables
  2. bars can be counted by emulating incrementation with the precedence predicate:

$$incr \quad = \quad \lambda v, v'. \quad v < v' \; \wedge \; \neg \exists v'' : v < v'' \; \wedge \; v'' < v'$$

NP-hardness of the Fixed Recognition Problem

- Given a formula f and the fixed XDG grammar G defined above, f is satisfiable if and only if $prep\ f \in L(G)$, i.e., SAT is reducible to the fixed recognition problem for XDG.
- as the reduction is polynomial, the fixed recognition problem for XDG is NP-hard
- universal recognition problem: generalization of the fixed recognition problem, thus also NP-hard

Upper Bounds

- principles first order: upper bound in PSPACE
- principles testable in polynomial time: upper bound in NP (all principles defined so far)

Overview

Summary

- XDG is a showcase for two trends in NLP: dependency grammar and multi-layered linguistic description
- but: two stumbling blocks: no complete formalization, no efficient large-scale parsing
- this talk: first complete formalization of XDG as a description language for multigraphs
- complexity: NP-hard, upper bound: with realistic restrictions: in NP

Future Work

- XDG parser: constraint-based parser, complete, concurrent, efficient for handcrafted grammars
- but does not yet scale up to large-scale parsing
- future work:
  1. optimizing the constraint-based parser: find global constraints, Gecode (Schulte/Stuckey 2004), (Schulte/Tack 2005), statistical support (supertagging)
  2. finding polynomially parsable fragments of XDG, e.g. related to TAG, STAG or GMTG (Melamed et al. 2004)

Thanks for your attention!

References

Regine Bader, Christine Foeldesi, Ulrich Pfeiffer, and Jochen Steigner.
Modellierung grammatischer Phänomene der deutschen Sprache mit Topologischer Dependenzgrammatik, 2004.
Softwareprojekt, Saarland University.

Alena Böhmová, Jan Hajič, Eva Hajičová, and Barbora Hladká.
The Prague Dependency Treebank: Three-level annotation scenario.
In *Treebanks: Building and Using Syntactically Annotated Corpora*. Kluwer Academic Publishers, 2001.

References

Ondrej Bojar.
Problems of inducing large coverage constraint-based dependency grammar.
In *Proceedings of the International Workshop on Constraint Solving and Language Processing*, Roskilde/DK, 2004.

Joan Bresnan and Ronald Kaplan.
Lexical-Functional Grammar: A formal system for grammatical representation.
In Joan Bresnan, editor, *The Mental Representation of Grammatical Relations*, pages 173–281. The MIT Press, Cambridge/US, 1982.

References

Michael Collins.
Head-Driven Statistical Models for Natural Language Parsing.
PhD thesis, University of Pennsylvania, 1999.

Ralph Debusmann.
Extensible Dependency Grammar: A Modular Grammar Formalism Based On Multigraph Description.
PhD thesis, Universität des Saarlandes, 4 2006.

References

Ralph Debusmann, Denys Duchier, Alexander Koller, Marco Kuhlmann, Gert Smolka, and Stefan Thater.
A relational syntax-semantics interface based on dependency grammar.
In *Proceedings of COLING 2004*, Geneva/CH, 2004.

Ralph Debusmann, Denys Duchier, and Joachim Niehren.
The XDG grammar development kit.
In *Proceedings of the MOZ04 Conference*, volume 3389 of *Lecture Notes in Computer Science*, pages 190–201, Charleroi/BE, 2004. Springer.

References

Ralph Debusmann, Denys Duchier, and Andreas Rossberg.
Modular Grammar Design with Typed Parametric Principles.
In *Proceedings of FG-MOL 2005*, Edinburgh/UK, 2005.

Ralph Debusmann, Oana Postolache, and Maarika Traat.
A modular account of information structure in Extensible Dependency Grammar.
In *Proceedings of the CICLING 2005 Conference*, Mexico City/MX, 2005. Springer.

References

Raph Debusmann.
A declarative grammar formalism for dependency grammar.
Diploma thesis, Saarland University, 2001.
http://www.ps.uni-sb.de/Papers/abstracts/da.html.

Denys Duchier and Ralph Debusmann.
Topological dependency trees: A constraint-based account of linear precedence.
In *Proceedings of ACL 2001*, Toulouse/FR, 2001.

References

Katrin Erk, Andrea Kowalski, Sebastian Pado, and Manfred Pinkal.
Towards a resource for lexical semantics: A large German corpus with extensive semantic annotation.
In *Proceedings of ACL 2003*, Sapporo/JP, 2003.

Martin Forst, Nuria Bertomeu, Berthold Crysmann, Frederik Fouvry, Silvia Hansen-Schirra, and Valia Kordoni.
Towards a dependency-based gold standard for German parsers—the TiGer dependency bank.
In *Proceedings of the 5th Int. Workshop on Linguistically Interpreted Corpora*, Geneva/CH, 2004.

References

Aravind K. Joshi.
An introduction to tree-adjoining grammars.
In Alexis Manaster-Ramer, editor, *Mathematics of Language*, pages 87–115. John Benjamins, Amsterdam/NL, 1987.

Paul Kingsbury and Martha Palmer.
From Treebank to PropBank.
In *Proceedings of LREC-2002*, Las Palmas/ES, 2002.

References

I. Dan Melamed, Giorgio Satta, and Benjamin Wellington.
Generalized Multitext Grammars.
In *Proceedings of ACL 2004*, Barcelona/ES, 2004.

Mathias Möhl.
Modellierung natürlicher Sprache mit Hilfe von Topologischer Dependenzgrammatik, 2004.
Fortgeschrittenenpraktikum, Saarland University, http://www.ps.uni-sb.de/ rade/papers/related/Moehl04.pdf.

References

Mozart Consortium.
The Mozart-Oz website, 2006.
http://www.mozart-oz.org/.

Renjini Narendranath.
Evaluation of the stochastic extension of a constraint-based dependency parser, 2004.
Bachelorarbeit, Saarland University.

References

Marwan Odeh.
Topologische Dependenzgrammatik fürs Arabische, 2004.
Forschungspraktikum, Saarland University.

Carl Pollard and Ivan A. Sag.
Head-Driven Phrase Structure Grammar.
University of Chicago Press, Chicago/US, 1994.

References

Christian Schulte and Peter J. Stuckey.
Speeding up constraint propagation.
In *Tenth International Conference on Principles and Practice of Constraint Programming*, volume 3258 of *Lecture Notes in Computer Science*, pages 619–633, Toronto/CA, 2004. Springer-Verlag.

Christian Schulte and Guido Tack.
Views and iterators for generic constraint implementations.
In Christian Schulte, Fernando Silva, and Ricardo Rocha, editors, *Proceedings of the Fifth International Colloqium on Implementation of Constraint and Logic Programming Systems*, pages 37–48, Sitges/ES, 2005.

References

Mark Steedman.
The Syntactic Process.
MIT Press, Cambridge/US, 2000.

Bonnie Webber, Aravind Joshi, Eleni Miltsakaki, Rashmi Prasad, Nikhil Dinesh, Alan Lee, and Katherine Forbes.
A short introduction to the Penn Discourse TreeBank.
Technical report, University of Pennsylvania, 2005.

Notational Conveniences

- strict dominance:

$$v \rightarrow_d^+ v' \stackrel{\text{def}}{=} v \rightarrow_d v' \vee (\exists v'' : v \rightarrow_d v'' \wedge v \rightarrow_d^+ v'')$$

Principles: Roots, Implications and Zeros

- roots:

$$\begin{array}{l} plRoots = \forall v : \\ \quad \neg\exists v' : v' \rightarrow_{\mathsf{PL}} v \ \Rightarrow \ (\mathsf{PL}\ v).truth \doteq 1 \end{array}$$

- implications:

$$\begin{array}{l} plImpls = \forall v, v', v'' : \\ \quad (v \xrightarrow{\mathsf{arg1}}_{\mathsf{PL}} v' \ \wedge \ v \xrightarrow{\mathsf{arg2}}_{\mathsf{PL}} v'' \ \Rightarrow \\ \quad (\mathsf{PL}\ v).truth \doteq ((\mathsf{PL}\ v').truth \ \Rightarrow \ (\mathsf{PL}\ v'').truth)) \wedge \\ \quad (\mathsf{PL}\ v).bars \doteq 1 \end{array}$$

- zeros:

$$\begin{array}{l} plZeros = \forall v : \\ \quad (W\ v) \doteq 0 \ \Rightarrow \\ \quad (\mathsf{PL}\ v).truth \doteq 0 \wedge \\ \quad (\mathsf{PL}\ v).bars \doteq 1 \end{array}$$

Principles: Variables and Bars

- variables:

$$
\begin{array}{l}
plVars = \forall v, v' : \\
\quad (W\ v) \doteq \textit{var} \Rightarrow \\
\quad v \xrightarrow{\mathsf{bar}}_{\mathsf{PL}} v' \ \Rightarrow \ (\mathsf{PL}\ v).bars \doteq (\mathsf{PL}\ v').bars
\end{array}
$$

- bars:

$$
\begin{array}{l}
plBars = \forall v : \\
\quad (W\ v) \doteq I \ \Rightarrow \\
\quad (\mathsf{PL}\ v).truth \doteq 0 \wedge \\
\quad \neg\exists v' : v \rightarrow_{\mathsf{PL}} v' \Rightarrow (\mathsf{PL}\ v).bars \doteq 1 \wedge \\
\quad (\forall v' : v \xrightarrow{\mathsf{bar}}_{\mathsf{PL}} v' \Rightarrow incr\ v'\ v)
\end{array}
$$

Principles: Coreference

- coreference:

$$
\begin{array}{l}
plCoref = \forall v, v' : \\
\quad (W\ v) \doteq \textit{\textsf{var}} \wedge (W\ v') \doteq \textit{\textsf{var}} \Rightarrow \\
\quad (\textsf{PL}\ v).bars \doteq (\textsf{PL}\ v').bars \Rightarrow (\textsf{PL}\ v).truth \doteq (\textsf{PL}\ v').truth
\end{array}
$$