

Tree-Adjoining Grammars: Theory and implementation

Day 3 part 2: Grammar implementation for LTAG

Kata Balogh & Simon Petitjean

Heinrich-Heine-Universität Düsseldorf, Carl von Ossietzky Universität Oldenburg

NASSLLI 2025

June 23 – 27, 2025

University of Washington, Seattle

Overview

What is grammar implementation?

Two ways of tree template implementation

- Metarules

- Metagrammars

Getting XMG 2

Last sessions

Mon: Motivation and the basic TAG

Tue: Linguistic applications and using LTAG: syntax

Wed: Linguistic applications and using LTAG: semantics

The following sessions

Wed: Introduction to grammar engineering and XMG

Thu: Grammar implementation with XMG

Fri: Parsing TAG

Overview

What is grammar implementation?

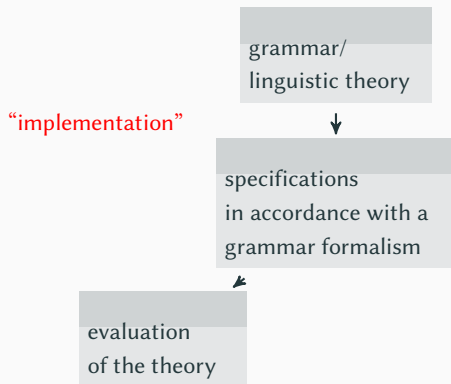
Two ways of tree template implementation

Metarules

Metagrammars

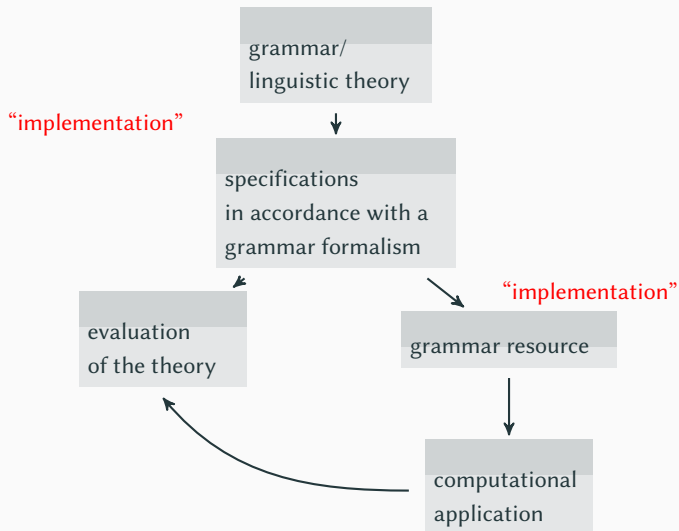
Getting XMG 2

Two kinds of grammar implementation



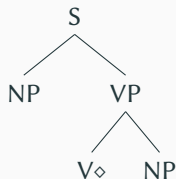
*As is frequently pointed out but cannot be overemphasized, an important goal of formalization in linguistics is to enable subsequent researchers to see the defects of an analysis as clearly as its merits; only then can **progress** be made efficiently. (Dowty 1979: 322)*

Two kinds of grammar implementation



What kind of grammar resource?

tree template



lexical insertion

anchor

repairs

The implementation task for LTAG

General task

Implement a large-coverage LTAG, as the XTAG grammar.

Subtasks:

- 1 Generate unlexicalized trees (= tree templates)
- 2 Generate a database of lexical anchors (= lexicon)
- 3 Connect the tree templates with the lexicon (= lexical insertion)

Two ways of grammar implementation with TAG

Two existing toolkits:

XTAG tools^[13]

- 1 implementation tools
⇒ **metarule approach**
- 2 editor/viewer for MorphDB and SynDB
- 3 parser

XMG + lexConverter + TuLiPA

- 1 XMG : eXtensible MetaGrammar^[5]
⇒ **metagrammar approach**
- 2 lexConverter (LEX2ALL)
- 3 TuLiPA: Tübingen Linguistic Parsing Architecture^[8]

Two ways of grammar implementation with TAG

Two existing toolkits:

XTAG tools^[13]

- 1 implementation tools
⇒ **metarule approach**
- 2 editor/viewer for MorphDB and SynDB
- 3 parser

XMG 2 + lexConverter + TuLiPA

- 1 XMG 2: eXtensible MetaGrammar^[5]
⇒ **metagrammar approach for grammar and lexicon**
- 2 lexConverter (LEX2ALL)
- 3 TuLiPA: Tübingen Linguistic Parsing Architecture^[8]

Overview

What is grammar implementation?

Two ways of tree template implementation

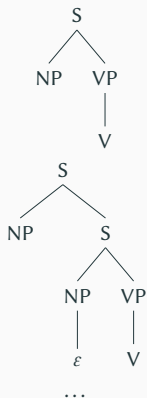
Metarules

Metagrammars

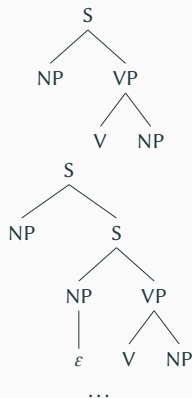
Getting XMG 2

The situation

12 templates for intransitive verbs



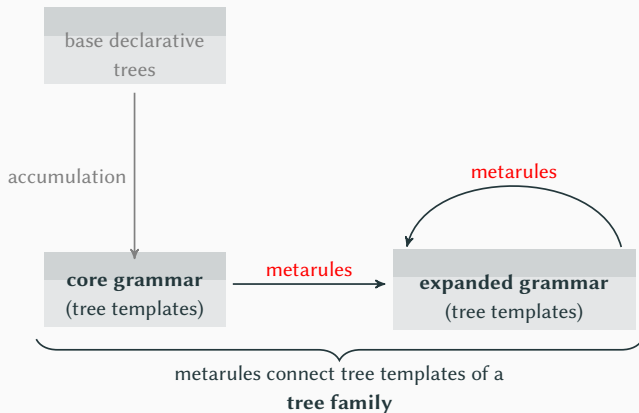
39 tree templates for transitive verbs



XTAG defines a set of 1008 unrelated tree templates.

Metarules for LTAG

Idea from GPSG^[7], later applied to XTAG^[1,2,9]



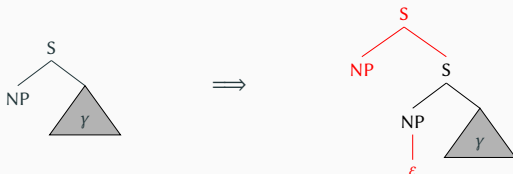
Metarules for LTAG: Example

extraction:

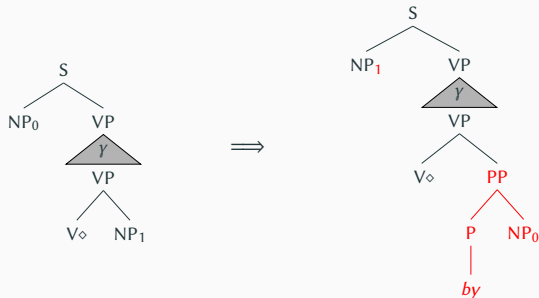


Metarules for LTAG: Example

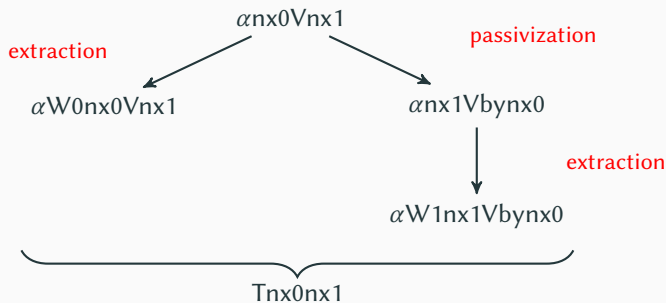
extraction:



passivization:



Metarules for LTAG: Example



Metarules for LTAG: Problems^[1]

Metarules are very powerful:

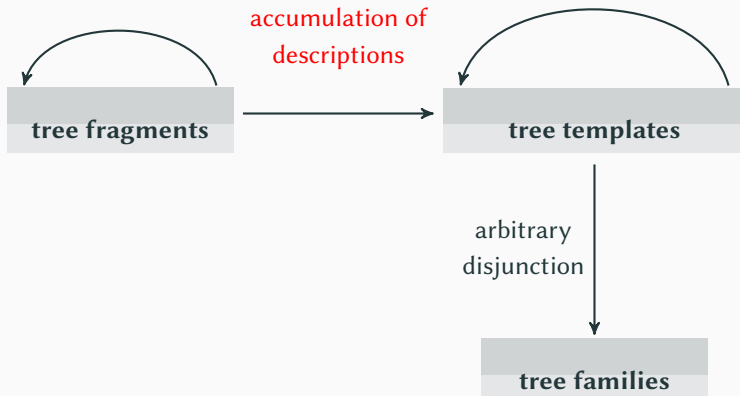
- deletion, copying, recursive application, metavariables over trees
- order sensitive
- in the unrestricted case: undecidable^[11]

Restrictions (GPSG):^[10]

- finite closure: apply every metarule at most once
⇒ still NP-complete
- biclosure: apply at most two metarules in a row
⇒ insufficient for LTAG metarules^[1]
- explicit rule ordering (by means of finite state automata)^[9]

Metagrammars for LTAG

Candito (1996)^[4,5,12]



- Observation: too many trees, but a lot of redundancies
- Idea: instead of trees, consider (reusable) tree fragments
- Trees obtained by assembling tree fragments
- No transformations \rightarrow accumulation of descriptions
- Monotonic, not order sensitive

Overview

What is grammar implementation?

Two ways of tree template implementation

Metarules

Metagrammars

Getting XMG 2

Getting XMG 2

Check out the documentation (in construction):

`https://xmg-hhu.github.io/documentation`

Three options for using XMG 2 (see *Getting Started* section)

- Follow the steps (Ubuntu), or
- Install VirtualBox and get the XMG 2 image, or
- Install Docker and get the container (recommended)

- [1] Becker, Tilman. 1994. ***Hytag: a new type of tree adjoining grammars for hybrid syntactic representations of free word order languages.*** Universität des Saarlandes dissertation.
<http://www.dfki.de/~becker/becker.diss.ps.gz>.
- [2] Becker, Tilman. 2000. **Patterns in metarules for TAG.** In Anne Abeillé & Owen Rambow (eds.), *Tree Adjoining Grammars: Formalisms, linguistic analyses and processing* (CSLI Lecture Notes 107), 331–342. Stanford, CA: CSLI Publications.
- [3] Candito, Marie-Hélène. 1996. **A principle-based hierarchical representation of LTAGs.** In *Proceedings of the 16th international Conference on Computational Linguistics (COLING 96)*. Copenhagen.
<http://aclweb.org/anthology-new/C/C96/C96-1034.pdf>.
- [4] Crabbé, Benoît. 2005. **Représentation informatique de grammaires d'arbres fortement lexicalisées: Le cas de la grammaire d'arbres adjoints.** Université Nancy 2 dissertation.
- [5] Crabbé, Benoît, Denys Duchier, Claire Gardent, Joseph Le Roux & Yannick Parmentier. 2013. **XMG: eXtensible MetaGrammar.** *Computational Linguistics* 39(3). 1–66. <http://hal.archives-ouvertes.fr/hal-00768224/en/>.
- [6] Dowty, David R. 1979. **Word meaning and Montague Grammar.** Reprinted 1991 by Kluwer Academic Publishers. Dordrecht: D. Reidel Publishing Company.
- [7] Gazdar, Gerald. 1981. **Unbounded dependencies and coordinated structure.** *Linguistic Inquiry* 12. 155–182.
- [8] Parmentier, Yannick, Laura Kallmeyer, Wolfgang Maier, Timm Lichte & Johannes Dellert. 2008. **TuLiPA: A syntax-semantics parsing environment for mildly context-sensitive formalisms.** In *Proceedings of the ninth international workshop on Tree Adjoining Grammars and related formalisms (TAG+9)*, 121–128. Tübingen, Germany.

- [9] Prolo, Carlos A. 2002. **Generating the XTAG English grammar using metarules**. In *Proceedings of the 19th international Conference on Computational Linguistics (COLING 2002)*, 814–820. Taipei. Taiwan.
- [10] Ristad, Eric Sven. 1987. **Revised General Phrase Structure Grammar**. In *Proceedings of the 25th annual meeting of the Association for Computational Linguistics*, 243–250. Stanford, CA. <http://www.aclweb.org/anthology/P87-1034>.
- [11] Uszkoreit, Hans & Stanley Peters. 1987. **On some formal properties of metarules**. English. In Walter J. Savitch, Emmon Bach, William Marsh & Gila Safran-Naveh (eds.), *The formal complexity of natural language* (Studies in Linguistics and Philosophy 33), 227–250. Dordrecht, The Netherlands: D. Reidel Publishing. http://dx.doi.org/10.1007/978-94-009-3401-6_9.
- [12] Xia, Fei. 2001. **Automatic grammar generation from two different perspectives**. University of Pennsylvania dissertation. http://faculty.washington.edu/fxia/papers_from_penn/thesis.pdf.
- [13] XTAG Research Group. 2001. **A Lexicalized Tree Adjoining Grammar for English**. Tech. rep. Philadelphia, PA: Institute for Research in Cognitive Science, University of Pennsylvania.