

Introduction to Tree Adjoining Grammar

Natural Language Syntax with TAG

Wolfgang Maier and Timm Lichte
University of Düsseldorf

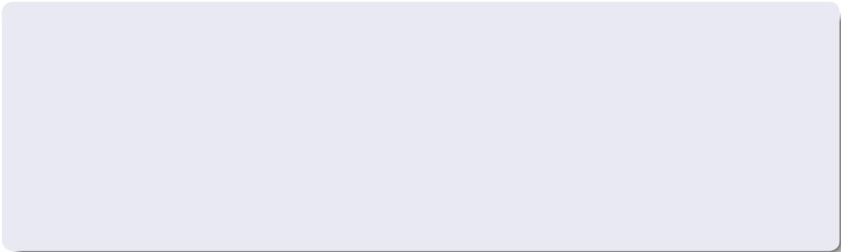
DGfS-CL Fall School 2011

1st week, 3rd session
Aug 31, 2011



- 1 Elementary trees for natural languages
- 2 Constituency and Dependency

Important features of TAG when used for natural languages:



Important features of TAG when used for natural languages:

- Grammar is **lexicalized**

Elementary trees for natural languages (1)

Important features of TAG when used for natural languages:

- Grammar is **lexicalized**
- Recursive parts are put into separate elementary trees that can be adjoined (**Factoring of recursion, FR**)

Elementary trees for natural languages (1)

Important features of TAG when used for natural languages:

- Grammar is **lexicalized**
- Recursive parts are put into separate elementary trees that can be adjoined (**Factoring of recursion, FR**)
- Elementary trees can be arbitrarily large, in particular (because of FR) they can contain elements that are far apart in the final derived tree (**Extended domain of locality**)

Elementary trees for natural languages (1)

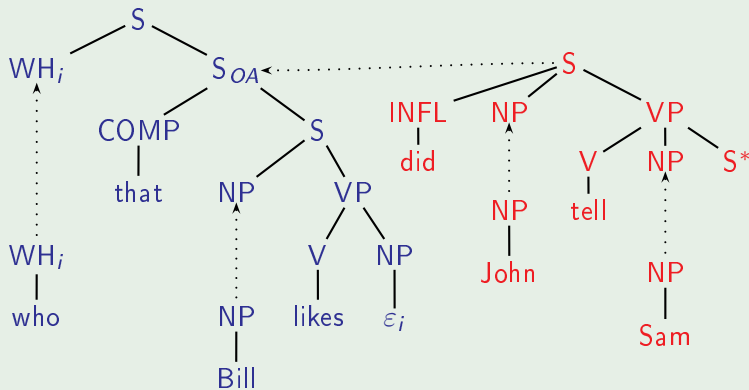
Important features of TAG when used for natural languages:

- Grammar is **lexicalized**
- Recursive parts are put into separate elementary trees that can be adjoined (**Factoring of recursion, FR**)
- Elementary trees can be arbitrarily large, in particular (because of FR) they can contain elements that are far apart in the final derived tree (**Extended domain of locality**)

For natural language syntax and TAG see
[Kroch, 1987, Abeillé, 1988, Abeillé, 2002, Frank, 2002,
XTAG Research Group, 2001].

Elementary trees for natural languages (2)

- (1) a. *who_i did John tell Sam that Bill likes t_i*
b. *who_i did John tell Sam that Mary said that Bill likes t_i*



Elementary trees for natural languages (3)

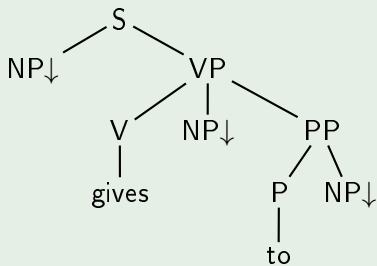
- Elementary trees are extended projections of lexical items.
- Recursion is factored away \Rightarrow finite set of elementary trees.
- The elementary tree of a lexical predicate contains slots for all arguments of the predicate, for nothing more.

Besides lexical predicates, there are functional elements (complementizers, determiners, auxiliaries, negation) whose treatment in LTAG is less clear. They can be

- either in separate elementary trees (e.g., XTAG grammar)
- or in the elementary tree of the lexical item they are associated with.

Elementary trees for natural languages: NP complements

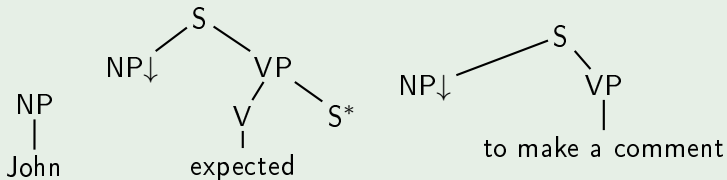
(2) John gives a book to Mary



Elementary trees for natural languages: Sentential complements

(3) John expected Mary to make a comment

expected selects for a subject NP and an infinitival sentence:

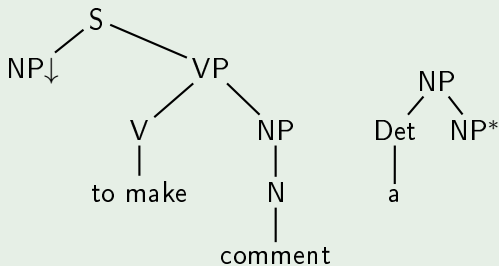


The sentential object is realised as a foot node in order to allow extractions:

(4) whom does John expect to come?

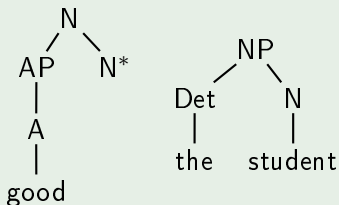
Elementary trees for natural languages: Multiple anchors

to make a comment: *make* and *comment* in the same elementary tree since they form a light verb construction:

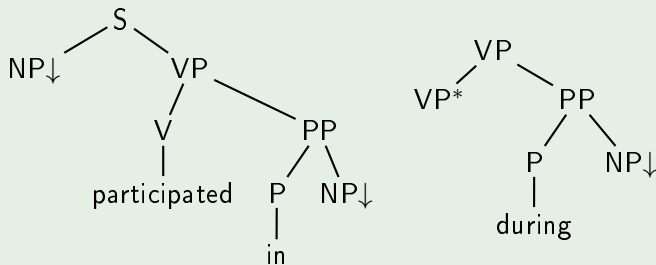


Example with modifiers:

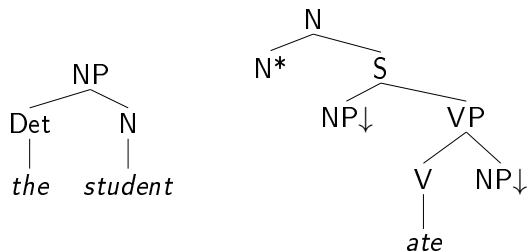
(5) the good student participated in every course during the semester



Elementary trees for natural languages: Modifiers



(1) the dog [who ate the cake]



Problem: Extraposed relative clauses:

(2) Somebody_i lives nearby [who_i has a CD-burner].

- Constraints on larger structures (constraints on “unbounded dependencies”) need not be stipulated.
- Instead, they follow from the possibilities of adjunction in the extended projections.

- Constraints on larger structures (constraints on “unbounded dependencies”) need not be stipulated.
- Instead, they follow from the possibilities of adjunction in the extended projections.

Fundamental LTAG hypothesis

Every syntactic dependency is expressed locally within a single elementary tree.

- Constraints on larger structures (constraints on “unbounded dependencies”) need not be stipulated.
- Instead, they follow from the possibilities of adjunction in the extended projections.

Fundamental LTAG hypothesis

Every syntactic dependency is expressed locally within a single elementary tree.

Non-local dependency corollary

Non-local dependencies always reduce to local ones once recursive structure is factored away.

(6) which book did Harvey say Cecile had read

How do the elementary trees look like?

Constituency and Dependency (1)

- The **derived tree** gives the **constituent structure**.
- The **derivation tree** records the history of how the elementary trees are put together.

⇒ the edges in the derivation tree represent **predicate-argument dependencies**; the derivation tree is close to a semantic dependency graph.

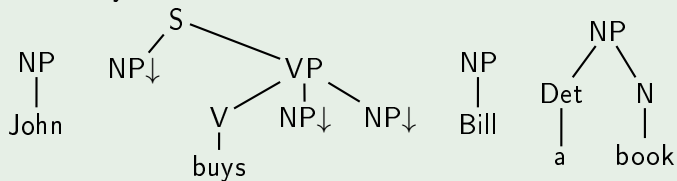
⇒ **compute semantics on derivation tree**

[Gardent and Kallmeyer, 2003, Kallmeyer and Joshi, 2003, Kallmeyer and Romero, 2008, Nesson and Shieber, 2006]

Constituency and Dependency (2)

(7) John buys Bill a book

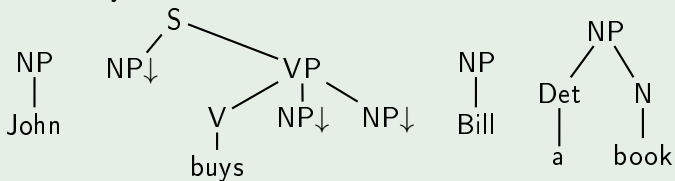
Elementary trees:



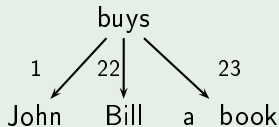
Constituency and Dependency (2)

(7) John buys Bill a book

Elementary trees:

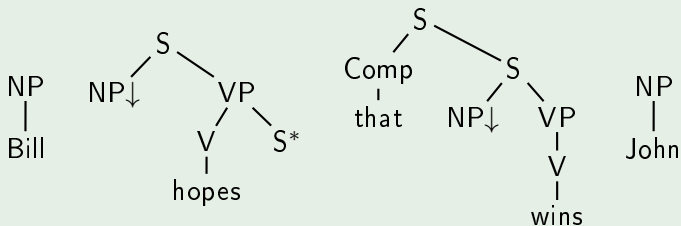


Derivation tree:



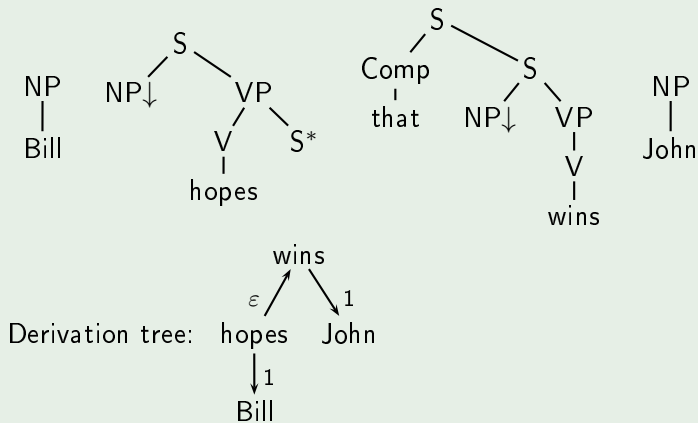
Constituency and Dependency (3)

(8) Bill hopes that John wins



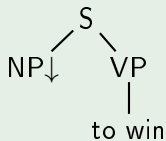
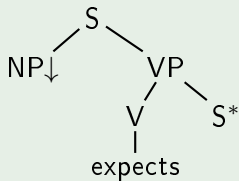
Constituency and Dependency (3)

(8) Bill hopes that John wins



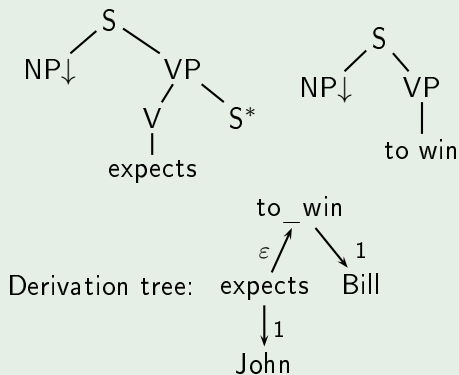
Constituency and Dependency (4)

(9) John expects [Bill to win]



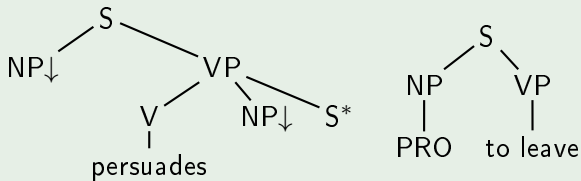
Constituency and Dependency (4)

(9) John expects [Bill to win]



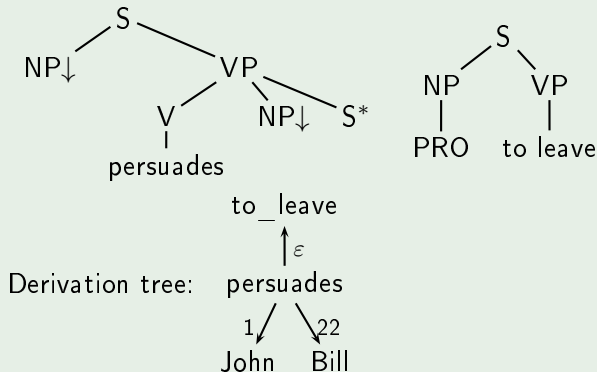
Constituency and Dependency (5)

(10) John persuades Bill [PRO to leave]



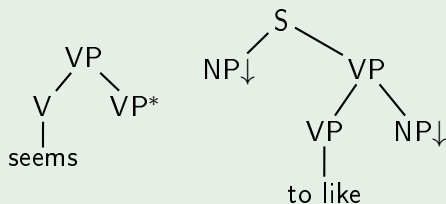
Constituency and Dependency (5)

(10) John persuades Bill [PRO to leave]



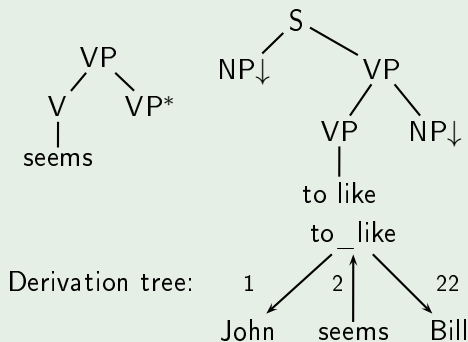
Constituency and Dependency (6)

(11) John seems to like Bill

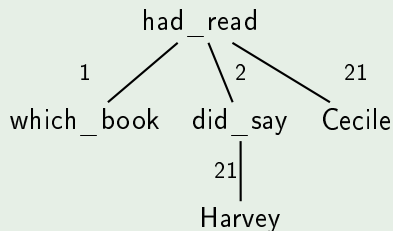


Constituency and Dependency (6)

(11) John seems to like Bill



(12) which book did Harvey say Cecile had read



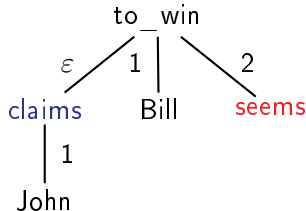
Constituency and Dependency (7)

The derivation tree is not always the semantic dependency structure, due to:

- indiscernibility of complementation and modification in adjunction, and
- missing links.

Example for a missing link:

(3) John **claims** [Bill **seems** to win]





Abeillé, A. (1988).

Parsing French with tree adjoining grammar: some linguistic accounts.
In [Proceedings of COLING](#), pages 7–12, Budapest.



Abeillé, A. (2002).

Une grammaire électronique du français.
CNRS Editions, Paris.



Frank, R. (2002).

Phrase Structure Composition and Syntactic Dependencies.
MIT Press, Cambridge, Mass.



Gardent, C. and Kallmeyer, L. (2003).

Semantic Construction in FTAG.
In [Proceedings of EACL 2003](#), pages 123–130, Budapest.



Kallmeyer, L. and Joshi, A. K. (2003).

Factoring Predicate Argument and Scope Semantics: Underspecified Semantics with LTAG.
[Research on Language and Computation](#), 1(1–2):3–58.



Kallmeyer, L. and Romero, M. (2008).

Scope and Situation Binding in LTAG using Semantic Unification.
[Research on Language and Computation](#), 6(1):3–52.



Kroch, A. S. (1987).

Unbounded dependencies and subadjacency in a Tree Adjoining Grammar.
In Manaster-Ramer, A., editor, [Mathematics of Language](#), pages 143–172. John Benjamins, Amsterdam.



Nesson, R. and Shieber, S. M. (2006).

Simpler TAG semantics through synchronization.
In [Proceedings of the 11th Conference on Formal Grammar](#), Malaga, Spain.



XTAG Research Group (2001).

A Lexicalized Tree Adjoining Grammar for English.

Technical report, Institute for Research in Cognitive Science, Philadelphia.

Available from <ftp://ftp.cis.upenn.edu/pub/xtag/release-2.24.2001/tech-report.pdf>.