Mildly Context-Sensitive Grammar Formalisms:

Tree Adjoining Grammars

Laura Kallmeyer
Heinrich-Heine-Universität Düsseldorf

Sommersemester 2011

Overview

1. Adjunction and substitution
2. Tree Adjoining Grammar
3. Adjunction constraints
4. Derivation trees
5. TAG and natural languages

Adjunction and substitution (1)

We have seen that

1. Using larger trees and allowing only substitution does not extend the weak capacity of CFG.
2. With TSGs, we cannot build satisfying lexicalized grammars.

The reason for the latter is that we cannot add modifiers in the middle of elementary trees, only the leaves can be replaced with new trees.

Therefore, we now add a second operation on trees called adjunction.

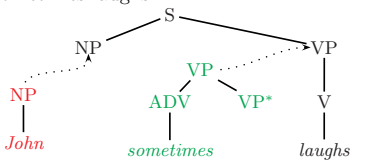
Adjunction and substitution (2)

Tree Adjoining Grammars (TAG)
[Joshi et al., 1975, Joshi and Schabes, 1997]: Tree-rewriting system: set of *elementary* trees with two operations:

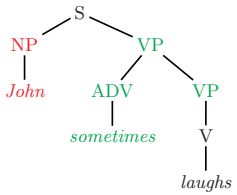
- adjunction: replacing an internal node with a new tree. The new tree is an auxiliary tree and has a special leaf, the foot node.
- substitution: replacing a leaf with a new tree. The new tree is an initial tree

Adjunction and substitution (3)

(1) John sometimes laughs



derived tree:



Adjunction and substitution (4)

Definition 1 (Auxiliary and initial trees)

1. *An auxiliary tree is a syntactic tree $\langle V, E, r\rangle$ such that there is a unique leaf f marked as foot node with $l(r) = l(f)$. We write this tree as $\langle V, E, r, f\rangle$.*
2. *An initial tree is a non-auxiliary syntactic tree.*

As a convention, the foot node is marked with a "*".

Adjunction and substitution (5)

Definition 2 (Adjunction)

Let $\gamma = \langle V, E, r\rangle$ be a syntactic tree, $\gamma' = \langle V', E', r', f\rangle$ an auxiliary tree and $v \in V$. $\gamma[v, \gamma']$, the result of adjoining *γ' into γ at node v is defined as follows:*

- *if $l(v) \neq l(r')$, then $\gamma[v, \gamma']$ is undefined;*
- *otherwise, $\gamma[v, \gamma'] = \langle V'', E'', r''\rangle$ with $V'' = V \cup V' \setminus \{v\}$ and $E'' = (E \setminus \{\langle v_1, v_2\rangle \mid v_1 = v \text{ or } v_2 = v\}) \cup E' \cup \{\langle v_1, r'\rangle \mid \langle v_1, v\rangle \in E\} \cup \{\langle f, v_2\rangle \mid \langle v, v_2\rangle \in E\}$.*

Tree Adjoining Grammar (1)

Definition 3 (Tree Adjoining Grammar) *A Tree Adjoining Grammar (TAG) is a tuple $G = \langle N, T, S, I, A\rangle$ such that*

- *T and N are disjoint alphabets, the terminals and nonterminals,*
- *$S \in N$ is the start symbol,*
- *I is a finite set of initial trees, and*
- *A is a finite set of auxiliary trees.*

The trees in $I \cup A$ are called elementary trees.

G is lexicalized iff each elementary tree has at least one leaf with a terminal label.

Tree Adjoining Grammar (2)

- Every elementary tree is considered a derived tree in a TAG.
- In every derivation step, we pick a fresh instance of an elementary tree from the grammar and we add derived trees (by substitution or adjunction) to certain nodes in this tree.
- The trees in the tree language are the derived initial trees with root label S.

Tree Adjoining Grammar (3)

Definition 4 (Derived tree)

Let $G = \langle N, T, S, I, A\rangle$ be a TAG.

1. *Every instance γ of a $\gamma_e \in I \cup A$ is a derived tree in G.*
2. *For pairwise disjoint $\gamma_1, \ldots, \gamma_n, \gamma$ such that $\gamma_1, \ldots, \gamma_n$ are derived trees in G $(1 \leq i \leq n)$ and $\gamma = \langle V, E, r\rangle$ is an instance of a $\gamma_e \in I \cup A$ such that $v_1, \ldots, v_n \in V$ are pairwise different: if $\gamma' = \gamma[v_1, \gamma_1] \ldots [v_n, \gamma_n]$ is defined then γ' is a derived tree in G.*
3. *These are all derived trees in G.*

Note that this definition adopts a bottom-up perspective: derived trees are added to elementary trees.

Tree Adjoining Grammar (4)

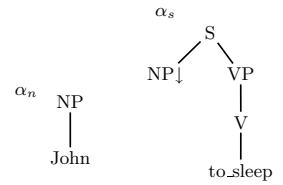
Definition 5 (TAG language)

Let $G = \langle N, T, S, I, A\rangle$ be a TAG.

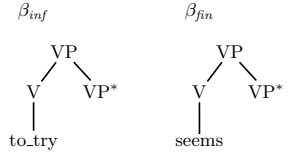
1. *The tree language of G is the set of all derived trees $\gamma = \langle V, E, r\rangle$ in G such that*
   - *$l(r) = S$, and*
   - *$l(v) \in T \cup \{\varepsilon\}$ for every leaf $v \in V$.*
2. *The string language of G is $\{w \mid$ there is a $\gamma \in L_T(G)$ such that $w = yield(\gamma)\}$.*

Tree Adjoining Grammar (5)

Initial trees:



Auxiliary trees:



Adjunction constraints (1)

TAG as defined above are more powerful than CFG but they cannot generate the copy language.

In order to increase the expressive power, adjunction constraints are introduced that specify for each node

1. whether adjunction is mandatory and
2. which trees can be adjoined.

For a given node,

1. the function f_{OA} specifies whether adjunction is obligatory (value 1) or not (value 0) and
2. the function f_{SA} gives the set of auxiliary trees that can be adjoined.

Adjunction constraints (2)

Definition 6 (TAG with adjunction constraints) *A TAG with adjunction constraints is a tuple $G = \langle N, T, S, I, A, f_{OA}, f_{SA}\rangle$ where*

- *$\langle N, T, S, I, A\rangle$ is a TAG as defined above and*
- *$f_{OA} : \{v \mid v \text{ is a node in some } \gamma \in I \cup A\} \rightarrow \{0, 1\}$ and $f_{SA} : \{v \mid v \text{ is a node in some } \gamma \in I \cup A\} \rightarrow P(A)$ where $P(A)$ is the set of subsets of A are functions such that $f_{OA}(v) = 0$ and $f_{SA}(v) = \emptyset$ for every leaf v.*

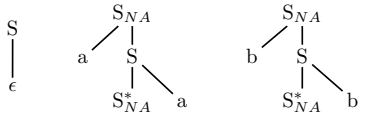
Adjunction constraints (3)

Three types of constraints are distinguished:

- A node v with $f_{OA}(v) = 1$ is said to carry a obligatory adjunction (OA) constraint.
- A node v with $f_{OA}(v) = 0$ and $f_{SA}(v) = \emptyset$ is said to carry a null adjunction (NA) constraint.
- A node v with $f_{OA}(v) = 0$ and $f_{SA}(v) \neq \emptyset$ and $f_{SA}(v) \neq A$ is said to carry a selective adjunction (SA) constraint.

Adjunction constraints (4)

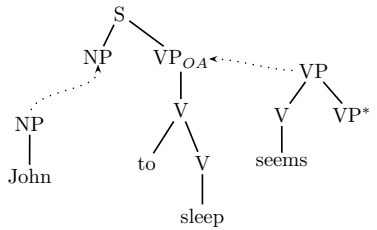
Example: TAG for the copy language:



Adjunction constraints (5)

Example:

(2) John seems to sleep



Adjunction constraints (6)

Definition 7 (Derived tree)

Let $G = \langle N, T, S, I, A, f_{OA}, f_{SA}\rangle$ be a TAG with adjunction constraints.

1. *Every instance of a $\gamma_e \in I \cup A$ is a derived tree obtained from γ_e in G.*
2. *For pairwise disjoint $\gamma_1, \ldots, \gamma_n, \gamma$ such that a) $\gamma_1, \ldots, \gamma_n$ are derived trees obtained from $\gamma_1^e, \ldots, \gamma_n^e$ in G respectively, and b) $\gamma = \langle V, E, r\rangle$ is an instance of a $\gamma_e \in I \cup A$ such that $v_1, \ldots, v_n \in V$ are pairwise different nodes: If*
   - *$\gamma' = \gamma[v_1, \gamma_1] \ldots [v_n, \gamma_n]$ is defined, and*
   - *$\gamma_i^e \in f_{SA}(v_i)$ for all $1 \leq i \leq n$*

   then γ' is a derived tree obtained from γ_e in G
3. *These are all derived trees in G.*

Adjunction constraints (7)

Definition 8 (Tree language)

Let $G = \langle N, T, S, I, A, f_{OA}, f_{SA}\rangle$ be a TAG with adjunction constraints.

The tree language of G is the set of all derived trees $\gamma = \langle V, E, r\rangle$ in G such that

- *$l(r) = S$,*
- *$f_{OA}(v) = 0$ for all $v \in V$, and*
- *$l(v) \in T \cup \{\varepsilon\}$ for every leaf $v \in V$.*

In the following, whenever we use the term "TAG", this means "TAG with adjunction constraints".

Derivation trees (1)

TAG derivations are described by derivation trees:

For each derivation in a TAG there is a corresponding derivation tree. This tree contains

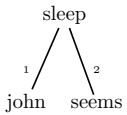
- nodes for all elementary trees used in the derivation, and
- edges for all adjunctions and substitutions performed throughout the derivation.

Whenever an elementary tree γ was attached to the node at address p in the elementary tree γ', there is an edge from γ' to γ labeled with p.

Derivation trees (2)

Example:

derivation tree for the derivation of (2) *John seems to sleep*



Derivation trees (3)

We can define the derivation trees in parallel to the definition of a derived trees:

Definition 9 (Derived tree)

Let $G = \langle N, T, S, I, A, f_{OA}, f_{SA} \rangle$ be a TAG.

1. *Every instance of a $\gamma_e \in I \cup A$ is a derived tree in G.*
   The corresponding derivation tree is $\langle \{v\}, \emptyset, v \rangle$ with $l(v) = \gamma_e$.

2. *For pairwise disjoint $\gamma_1, \ldots, \gamma_n, \gamma$ such that a) $\gamma_1, \ldots, \gamma_n$ are derived trees with derivation trees $D_i = \langle V_i, E_i, r_i \rangle$ $(1 \leq i \leq n)$ and $\gamma = \langle V, E, r \rangle$ is an instance of a $\gamma_e \in I \cup A$ such that $v_1, \ldots, v_n \in V$ are pairwise different with Gorn addresses $p_1, \ldots, p_n$: if*
   - *$\gamma' = \gamma[v_1, \gamma_1] \ldots [v_n, \gamma_n]$ is defined and*
   - *$l(r_i) \in f_{SA}(v_i)$ for all $1 \leq i \leq n$,*

   then γ' is a derived tree in G with a corresponding derivation tree $D = \langle V_D, E_D, r_D \rangle$ such that $V_D = \bigcup_{i=1}^{n} V_i \cup \{r_D\}$ where $r_D \notin \bigcup_{i=1}^{n} V_i$ is a new node,
   $E_D = \bigcup_{i=1}^{n} E_i \cup \{\langle r_D, r_1 \rangle, \ldots, \langle r_D, r_n \rangle\}$ and $l(r_D) = \gamma_e$ and $g(\langle r_D, r_i \rangle) = p_i$ for $1 \leq i \leq n$.
3. *These are all pairs of derived trees and derivation trees in G.*

Derivation trees (4)

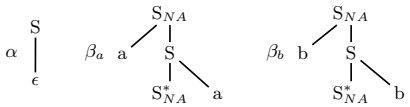
Derivation trees are unordered trees.

They

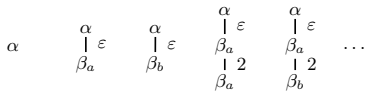
- uniquely determine a derived tree (but not vice versa), and
- are context-free, i.e., for every TAG, one can find a CFG whose tree language is, except for adding ε-leaves and ordering the trees, the set of derivation trees of the TAG.

Derivation trees (5)

Example:



Derivation trees:



CFG: $\alpha \to \varepsilon$ $\alpha \to \beta_a$ $\alpha \to \beta_b$ $\beta_a \to \varepsilon$ $\beta_a \to \beta_a$
$\beta_a \to \beta_b$ $\beta_b \to \varepsilon$ $\beta_b \to \beta_a$ $\beta_b \to \beta_b$

TAG and natural languages (1)

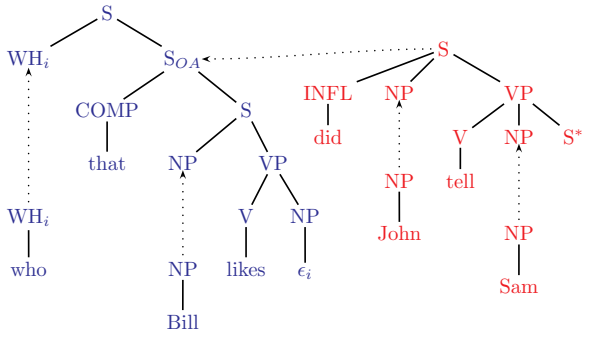
Important features of TAG when used for natural languages:

- 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].

TAG and natural languages (2)

(3) 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



TAG and natural languages (3)

Elementary trees are extended projections of lexical items.
Recursion is factored away ⇒ finite set of elementary trees.

The elementary tree of a lexical predicate contains slots for all arguments of the predicate, for nothing more.

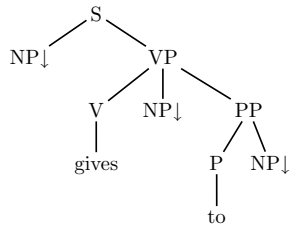
Besides lexical predicates, there are functional elements (complementizers, determiners, auxiliaries, negation) whose treatment in LTAG is less clear. They can be

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

TAG and natural languages (4)

Example:

(4) John gives a book to Mary

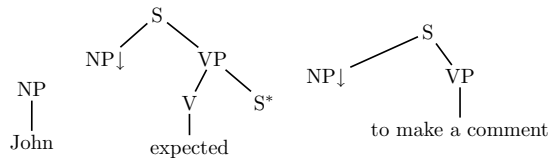


TAG and natural languages (5)

Example:

(5) John expected Mary to make a comment

expected selects for a subject NP and an infinitival sentence:

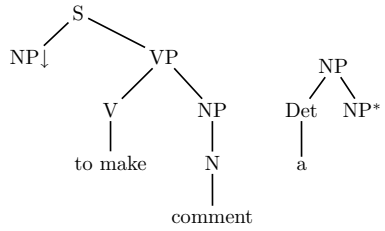


The sentential object is realised as a foot node in order to allow extractions:

(6) whom does John expect to come?

TAG and natural languages (6)

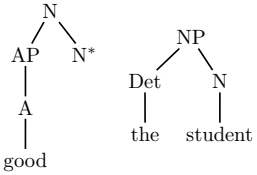
to make a comment: *make* and *comment* in the same elementary tree since they form a light verb construction:



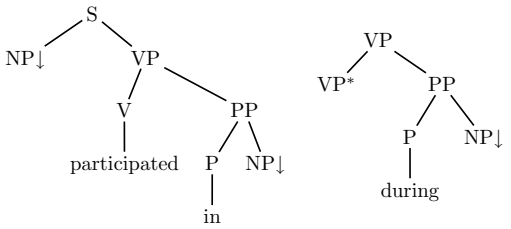
TAG and natural languages (7)

Example with modifiers:

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



TAG and natural languages (8)



References

[Abeillé, 1988] Abeillé, A. (1988). Parsing French with tree adjoining grammar: some linguistic accounts. In *Proceedings of COLING*, pages 7–12, Budapest.

[Abeillé, 2002] Abeillé, A. (2002). *Une Grammaire Électronique du Français*. CNRS Editions, Paris.

[Frank, 2002] Frank, R. (2002). *Phrase Structure Composition and Syntactic Dependencies*. MIT Press, Cambridge, Mass.

[Joshi et al., 1975] Joshi, A. K., Levy, L. S., and Takahashi, M. (1975). Tree Adjunct Grammars. *Journal of Computer and System Science*, 10:136–163.

[Joshi and Schabes, 1997] Joshi, A. K. and Schabes, Y. (1997). Tree-Adjoining Grammars. In Rozenberg, G. and Salomaa, A., editors, *Handbook of Formal Languages*, pages 69–123. Springer, Berlin.

[Kroch, 1987] Kroch, A. S. (1987). Unbounded dependencies and subjacency in a Tree Adjoining Grammar. In Manaster-Ramer, A., editor, *Mathematics of Language*, pages 143–172. John Benjamins, Amsterdam.

[XTAG Research Group, 2001] 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`.