Language modeling with tree-adjoining grammars

Day1: Introduction to TAG

Kata Balogh & Simon Petitjean

University of Düsseldorf

NASSLLI 2018

SFB 991

erc TreeGraSP

HEINRICH HEINE UNIVERSITÄT DÜSSELDORF

What this course is about

Language modeling with Tree-Adjoining Grammars

- language modeling → trying to implement syntactic theories
  - implement[1]: general concepts → mathematical objects
  - implement[2]: paper & pencil → electronic resource
- Why 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)]

  - incentive for rigor
  - check for consistency
  - applications (→ NLP)

What this course is **not** about

Details of …

- formal language theory
- parsing with mildly context-sensitive formalisms (LCFRS, 2-MCFG, 2-ACG) [Kallmeyer 2010]

… However, this is highly relevant for motivating TAG!

- complexity of a language
  - ⇒ determined by the weakest formal grammar that generates it
- expressive power of the formalism
  - ⇒ TAG: The formalism is part of the theory, so let's try to make it both convenient and minimally expressive!

Why working with TAG? (in a nutshell)

- formal complexity of natural languages → gain insights into
  - ⇒ the general structure of natural language
  - ⇒ the general human language capacity
  - ⇒ the adequacy of grammar formalisms
  - ⇒ lower bound of the computational complexity of NLP tasks

TAG exactly provides the expressive power needed to treat NL.

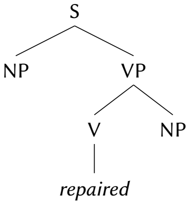
TAG: The formalism is part of the theory, so let's try to make it both convenient and minimally expressive!

Expressive power in terms of a specific generative capacity:

- weak generative capacity → to generate **string languages**
- strong generative capacity → to generate **tree languages**
- derivational generative capacity

Why working with TAG? (some linguistic reasons)

- extended domain of locality



- long-distance dependencies / discontinuous constituents

(1) Who did Mary say that Tom claimed …repaired the fridge?

- multi-word expressions

(2) to kick the bucket ('to die')

Schedule

Day 1: Motivation and the basic TAG

Day 2: Linguistic applications and using LTAG: syntax

Day 3: Linguistic applications and using LTAG: semantics

Day 4: Grammar implementation with XMG

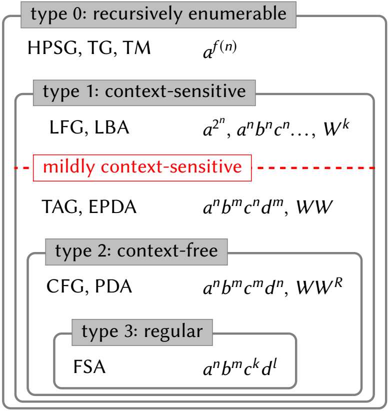
Day 5: Parsing TAG

- lecturers:
  - Kata Balogh (`Katalin.Balogh@hhu.de`)
  - Simon Petitjean (`petitjean@phil.hhu.de`)
- course page:
  - https://tinyurl.com/ycwje6ma

From CFG to TAG

Grammar Formalisms

- aim: find an adequate formal system for natural language analysis
  - mathematically concise representation of a grammar theory
  - a formal system for linguistic analyses



- theory of formal languages (Chomsky-hierarchy)
  - finite-state models
    $\Rightarrow$ not plausible enough
  - context-free grammars
    $\Rightarrow$ almost plausible, just not enough

Chomsky-hierarchy

A grammar (N, T, S, R) is a

- Type 0 or unrestricted (phrase structure) grammar iff every production is of the form $\alpha \rightarrow \beta$ with $\alpha \in (N \cup T)^* \setminus T^*$ and $\beta \in (N \cup T)^*$;
  generates a recursively enumerable language (RE).
- Type 1 or context-sensitive grammar iff every production is of the form $\gamma A \delta \rightarrow \gamma \beta \delta$ with $\gamma, \delta, \beta \in (N \cup T)^*$, $A \in N$ and $\beta \neq \epsilon$;
  generates a context-sensitive language (CS).
- Type 2 or context-free grammar iff every production is of the form $A \rightarrow \beta$ with $A \in N$ and $\beta \in (N \cup T)^* \setminus \{\epsilon\}$;
  generates a context-free language (CF).
- Type 3 or right-linear grammar iff every production is of the form $A \rightarrow \beta B$ or $A \rightarrow \beta$ with $A, B \in N$ and $\beta \in T^* \setminus \{\epsilon\}$;
  generates a regular language (REG).

For Type 1-3 languages a rule $S \rightarrow \epsilon$ is allowed if S does not occur in any rule's right-hand side.

Chomsky-hierarchy: overview

type	grammar	rules	word problem
RE	phrase structure	$\alpha \longrightarrow \beta$	undecidable
CS	context-sensitive	$\gamma A\delta \longrightarrow \gamma\beta\delta$	exponential
CF	context-free	$A \longrightarrow \beta$	cubic
REG	right-linear	$A \longrightarrow aB\|b$	linear

Languages as problems:
"Can we decide for every word whether it belongs to L?"

Limits of CFG

- for natural languages context-free grammars are just not ‘enough’
  - **expressivity challenge**: cannot describe all NL phenomena
    - ★ cross-serial dependencies ($a^n b^m c^n d^m$); Schwyzerdütsch
    - ★ duplication (yy); Bambara (spoken in Mali)
    - ★ multiple agreement ($a^n b^n c^n$); Bantu languages
  - **low descriptive power**: problems with certain linguistic phenomena e.g. subcategorization, number agreement, case marking
  - **only weak-lexicalization** possible
- natural languages are almost context-free

mildly context sensitive languages

RL ⊂ CFL ⊂ **MCSL** ⊂ CSL ⊂ RE

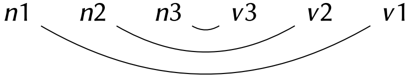
[Joshi, 1985]

- for natural languages we need grammars, that are somewhat richer than context-free grammars, but more restricted than context-sensitive grammars

Limits of CFG: expressivity challenge

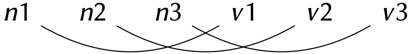
- German: **nested dependency** (subordinate clauses)

(3) er die Kinder dem Hans das Haus streichen helfen ließ.
he the children the Hans the house paint help let.
'(that) he let the children to help Hans to paint the house.'



- Schwyzerdütsch: **cross-serial dependency**

(4) mer d'chind em Hans es huus lönd hälfe aastriiche.
we children.acc the Hans.dat the house.acc let help paint.
'(that) we let the children to help Hans to paint the house.'



(5) *mer d'chind de Hans es huus lönd hälfe aastriiche.
we children.acc the Hans.acc the house.acc let help paint.

Limits of CFG: expressivity challenge

Proof by Schieber

- Jan säit das mer d'chind em Hans es huus lönd hälfe aastriiche.
- homomorphism f:

f(d'chind) = a	f(em Hans) = b	f(laa) = c
f(hlfe) = d	f(aastriiche) = y	f(es huus haend wele) = x
f(Jan sit das mer) = w	$f(s) = z$ *otherwise*	

- f(Schwyzerdütsch) $\cap\ wa^*b^*xc^*d^*y = wa^mb^nxc^md^ny$
  - CFLs are closed under intersection with regular languages: $L1_{CF} \cap L2_{REG} = L3_{CF}$
  - $wa^*b^*xc^*d^*y$ is regular
  - by Pumping Lemma: $wa^mb^nxc^md^ny$ is not context-free
- $\Rightarrow$ Schwyzerdütsch is not context-free

Limits of CFG: low descriptive power

- take a simple CFG
  - string rewriting
  - replace non-terminals by strings of terminals and non-terminals

$$G_{\mathrm{CFG}} = \langle N, T, S, P \rangle$$

$$P = \{$$

$$\mathrm{S} \longrightarrow NP\ VP$$

$$VP \longrightarrow V\ NP \mid V$$

$$V \longrightarrow likes \mid like \mid sleeps$$

$$NP \longrightarrow she \mid her \mid they$$

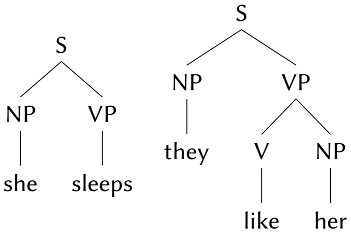
$$\}$$

Example derivations:

$\mathrm{S} \rightarrow NP\ VP \rightarrow she\ VP \rightarrow she\ V \rightarrow she\ sleeps$

$S \rightarrow NP\ VP \rightarrow they\ VP \rightarrow they\ V\ NP \rightarrow$
$they\ like\ NP \rightarrow they\ like\ her$

Example derivation history:



Limits of CFG: low descriptive power

- subcategorization / argument selection
  (1) She sleeps. / She likes her. / *She likes.
  S⇒ NP VP⇒ Joe VP ⇒ Joe V ⇒ Joe sleeps
  S⇒ NP VP⇒ Joe VP ⇒ Joe V ⇒ Joe likes
- number agreement
  (2) They like her. / *They likes her.
- case marking
  (3) She likes her. / *She likes they.
- encode necessary information in the non-terminals?

Limits of CFG: low descriptive power

- extend for number agreement, argument selection (transitive vs. non-transitive) and case marking

  $S \rightarrow NP_{3sg/nom}\ VP_{3sg/itr}$, $S \rightarrow NP_{3pl/nom}\ VP_{3pl/itr}$,
  $S \rightarrow NP_{3sg/nom}\ VP_{3sg/tr}$, $S \rightarrow NP_{3pl/nom}\ VP_{3pl/tr}$,
  $VP_{3sg/tr} \rightarrow V_{3sg/tr}\ NP_{3sg/acc}$, $VP_{3pl/tr} \rightarrow V_{3pl/tr}\ NP_{3sg/acc}$,
  $VP_{3sg/itr} \rightarrow V_{3sg/itr}$, $VP_{3pl/itr} \rightarrow V_{3pl/itr}$,
  $NP_{3sg/nom} \rightarrow$ she, $NP_{3sg/acc} \rightarrow$ her, $NP_{3pl/nom} \rightarrow$ policemen,
  $V_{3sg/itr} \rightarrow$ sleeps, $V_{3pl/itr} \rightarrow$ sleep, $V_{3sg/tr} \rightarrow$ likes, $V_{3pl/tr} \rightarrow$ like

- every possible combination of arguments selection (e.g. transitive/non-transitive), number agreement and case marking must have a separate non-terminal and a separate re-write rule
- grammar writing is quite error prone (and boring)
- linguistic generalizations are difficult to express, e.g.
  - subject and verb must have the same number
  - the object of a transitive verb must be in accusative case
- solution: feature structures, unification, underspecification (see later)

Lexicalization

Lexicalized grammar

A lexicalized grammar consists of:

(i) a finite set of structures each associated with a lexical item (anchor),
(ii) operation(s) for composing these structures.

Lexicalization

A formalism F can be lexicalized by another formalism F',
if for any finitely ambiguous grammar G in F there is a grammar G' in F',
such that (i) G' is a lexicalized grammar; and
(ii) G and G' generate the same set.

weak vs. strong lexicalization

- weak lexicalization: preserve the string language
- strong lexicalization: preserve the tree structure

Limits of CFG: lexicalization

- **Formally interesting:**
  - a finite lexicalized grammar provides finitely many analyses for each string (finitely ambiguous)
- **Linguistically interesting:**
  - syntactic properties of lexical items can be accounted for more directly
  - each lexical item comes with the possibility of certain partial syntactic constructions
- **Computationally interesting:**
  - the search space during parsing can be delimited (grammar filtering)
  - use of corpora in NLP

Lexicalization of CFG's

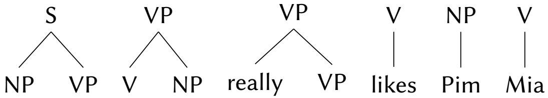
- lexicalize CFGs:
  - recursive ($X \Rightarrow^* X$) and elementary ($X \rightarrow X$) rules are disallowed
  - each rule must consist at least one terminal on the RHS
- lexicalized CFG $\rightsquigarrow$ e.g. Greibach normal-form: $A \longrightarrow aX$ or $A \longrightarrow a$ ($a \in V_T$; $A \in V_N$; $X \in (V_N)^*$) [Greibach, 1965]
- example:
  - a CFG G: $S \longrightarrow SS$, $S \longrightarrow a$
  - lexicalize $G \Rightarrow G'$: $S \longrightarrow aS$, $S \longrightarrow a$
- same string language, but not the same tree set
- only weak lexicalization possible

Lexicalization of CFG's

- take the following (very simple) CFG

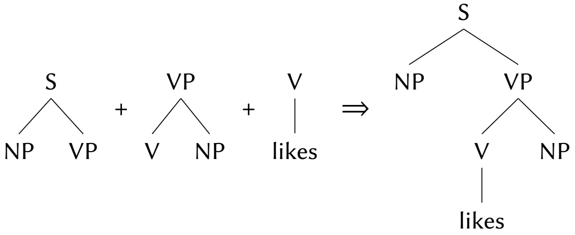
  G = { S → NP VP   VP → really VP   NP → Joe
  VP → V NP   V → likes   NP → Cleo }

- step 1: take trees as elementary structures



- step 2: combine the elementary structures

  ⇒ lexical items appear as part of the elementary structures



Tree Substitution Grammar (TSG)

- a CFG rule corresponds to a tree
  - lhs as the root node / rhs as the daughter nodes
  - e.g. S $\longrightarrow$ NP VP
- tree rewriting
- **substitution:** replace a non-terminal leaf with a tree
- grammar on trees + substitution $\longrightarrow$ **Tree Substitution Grammar**

  A TSG is a quadruple $TSG = \langle \Sigma, NT, I, S \rangle$, where

  Σ is a set of terminal symbols;
  NT is a set of non-terminal symbols;
  $S \in NT$ is a distinguished non-terminal symbol;
  I is a finite set of initial trees.

From CFG to TAG: Tree Substitution Grammar

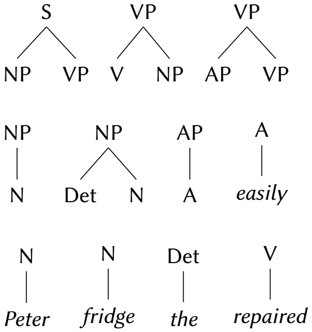
$G_{\text{CFG}} = \langle N, T, S, P \rangle$
$P = \{$

$$
\begin{aligned}
S &\rightarrow NP\ VP \\
VP &\rightarrow V\ NP \mid AP\ VP \\
NP &\rightarrow N \mid Det\ N \\
AP &\rightarrow A \\
N &\rightarrow Peter \mid fridge \\
Det &\rightarrow the \\
A &\rightarrow easily \\
V &\rightarrow repaired
\end{aligned}
$$

$\}$

$\approx$

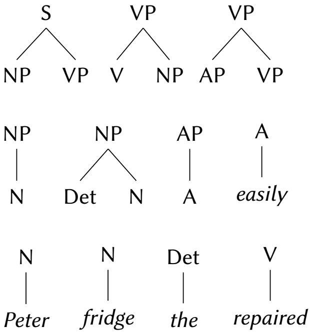
$G_{\text{TSG}} = \langle N, T, S, I \rangle$
$I = \{$



$\}$

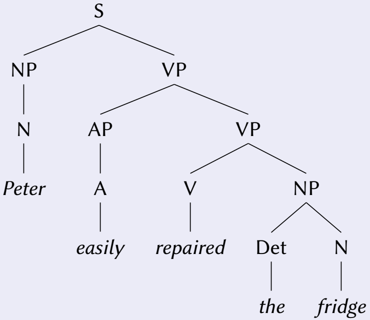
From CFG to TAG: Tree Substitution Grammar

$G_{\mathrm{TSG}} = \langle N, T, S, I \rangle$
$I = \{$



$\}$

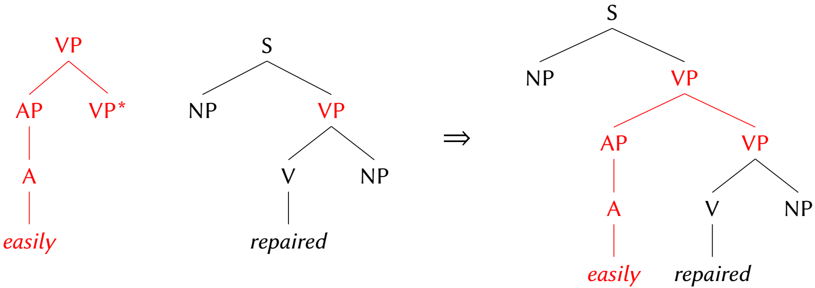
Example derivation:



Lexicalize this TSG!

TSG + Adjunction

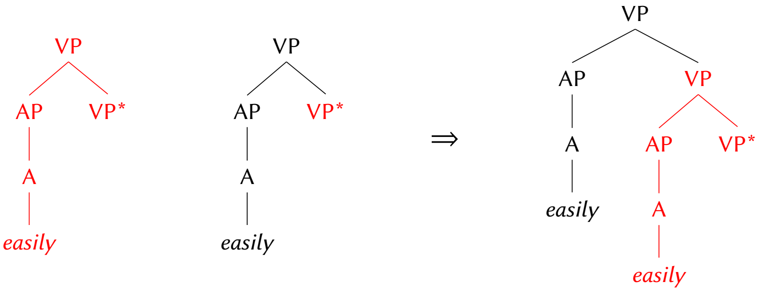
- lexicalization of CFG in a linguistically meaningful way
- TSG: still no strong lexicalization of CFG, no cross-serial dependencies etc.
- add **adjunction:**
  - replace a **non-terminal node** with an “auxiliary” tree
  - put the subtree of the replaced node under the **footnode** (*)



TSG + Adjunction

⇒ Adjunction at footnodes causes spurious ambiguities in derivations.

⇒ Therefore, this is usually forbidden.

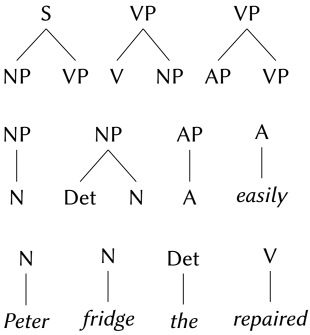


From CFG to TAG: Example with adjunction

- tree rewriting
- **Substitution:** replace a non-terminal **leaf** with a tree
- **Adjunction:** replace a non-terminal **node** with an “auxiliary” tree

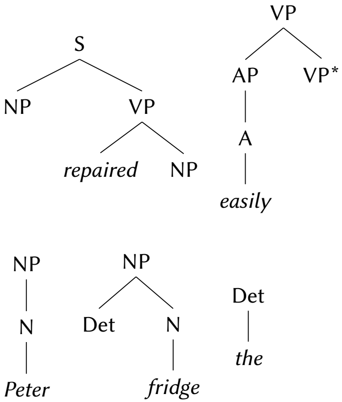
$G_{\mathrm{TSG}} = \langle N, T, S, I \rangle$

$I = \{$



$\}$

$\approx$



From CFG to TAG: Restrictions on adjunction (I)

Restrictions on the shape of auxiliary trees:

- The root node and the footnode must carry the same non-terminal.

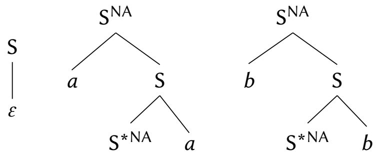
Specific adjunction constraints on target nodes:

- obligatory adjunction (OA): true/false
- null adjunction (NA): no adjoinable auxiliary tree
- selective adjunction (SA): a nonempty set of adjoinable auxiliary trees

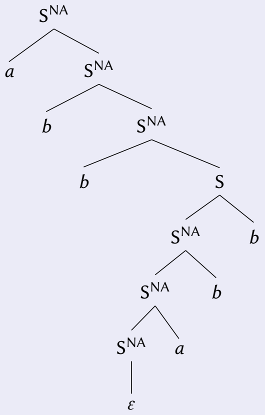
Adjunction constraints are essential in generating non-context-free languages (e.g. the copy language $\{ww | w \in \{a, b\}^*\}$)!

From CFG to TAG: Restrictions on adjunction (I)

Example grammar for the copy language $\{ww | w \in \{a, b\}^*\}$:



Example derivation of *abbabb*:



$\Longrightarrow$ TAG = TSG + adjunction + adjunction constraints