Grammar Implementation with TAG

Overview

Timm Lichte

HHU Düsseldorf

SS 2011

06.04.2011

“Grammar”, “Implementation”, “TAG”

Grammar:

rules for well-formed structures of natural language

Implementation:

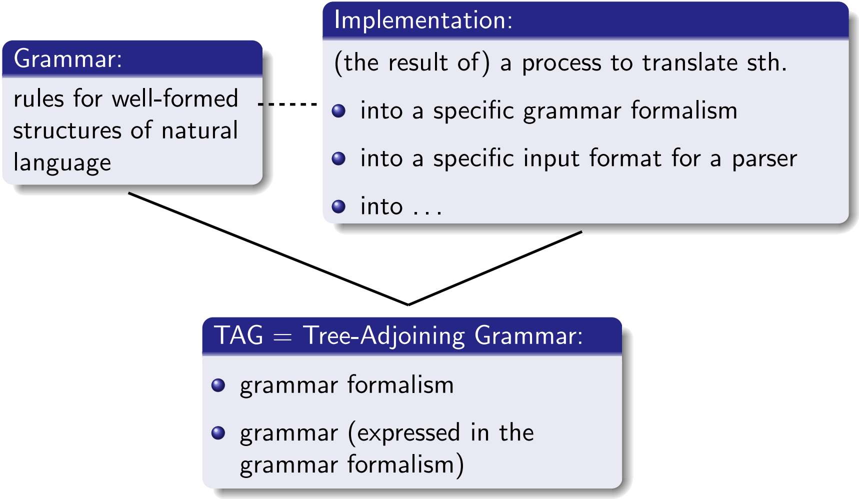
(the result of) a process to translate sth.

- into a specific grammar formalism
- into a specific input format for a parser
- into . . .

TAG = Tree-Adjoining Grammar:

- grammar formalism
- grammar (expressed in the grammar formalism)

“Grammar”, “Implementation”, “TAG”



Grammar:

rules for well-formed structures of natural language

Implementation:

(the result of) a process to translate sth.

- into a specific grammar formalism
- into a specific input format for a parser
- into . . .

TAG = Tree-Adjoining Grammar:

- grammar formalism
- grammar (expressed in the grammar formalism)

Grammar Formalisms

Grammar:

rules for well-formed structures of natural language

⇒ Grammars without formalisms ?

Formalisms:

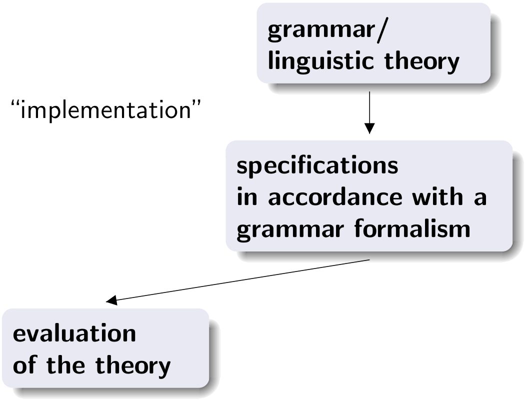
mathmatically concise specification language

Informal:
A wellformed sentence consists of a subject, a verb and an object, where the subject precedes the verb and the object follows the verb.

Formal:

$\texttt{S} \longrightarrow \texttt{SU V OB}$

The benefit of grammar formalisms and implementation



grammar/ linguistic theory

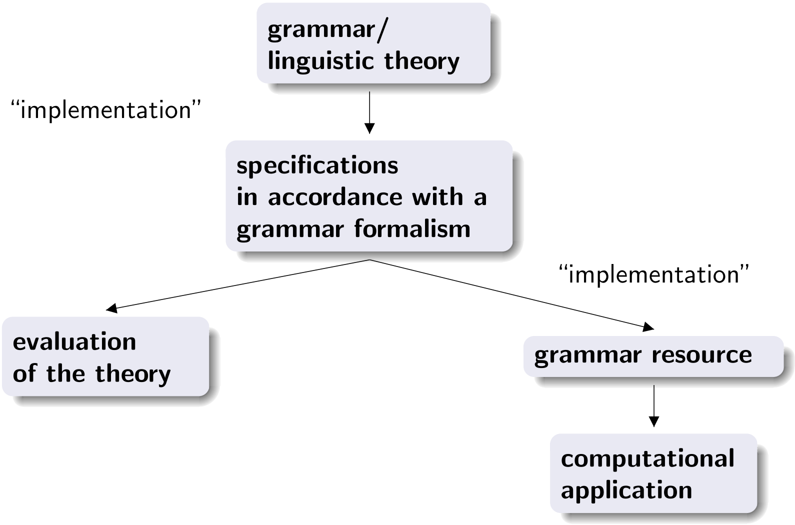
“implementation”

specifications in accordance with a grammar formalism

evaluation of the theory

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)

The benefit of grammar formalisms and implementation



grammar/
linguistic theory

"implementation"

specifications
in accordance with a
grammar formalism

"implementation"

evaluation
of the theory

grammar resource

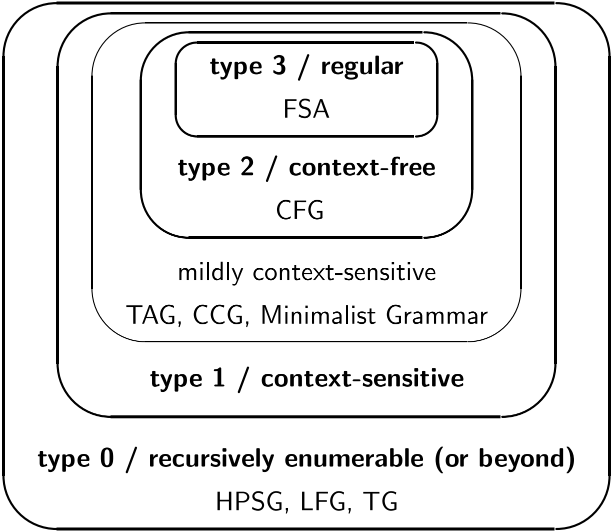
computational
application

The landscape of Grammar Formalisms (1)

- **generative rewriting formalisms:**
  - Context-Free Grammar (CFG)
  - Tree-Adjoining Grammar (TAG)
  - Lexical Functional Grammar (LFG)
  - Transformational Grammar (GB), Minimalism
- **proof-theoretic formalisms:**
  - Combinatorial Categorial Grammar (CCG)
- **model-theoretic/constraint-based formalisms:**
  - Head-Driven Phrase Structure Grammar (HPSG)

The landscape of Grammar Formalisms (2)

Within **Chomsky hierarchy**:



type 3 / regular

FSA

type 2 / context-free

CFG

mildly context-sensitive

TAG, CCG, Minimalist Grammar

type 1 / context-sensitive

type 0 / recursively enumerable (or beyond)

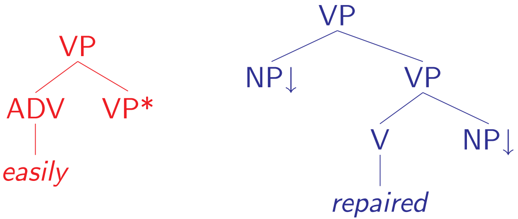
HPSG, LFG, TG

Tree-Adjoining Grammar - Basics

A **Tree Adjoining Grammar (TAG)** is a set of elementary trees:

- a finite set of initial trees
- a finite set of auxiliary trees

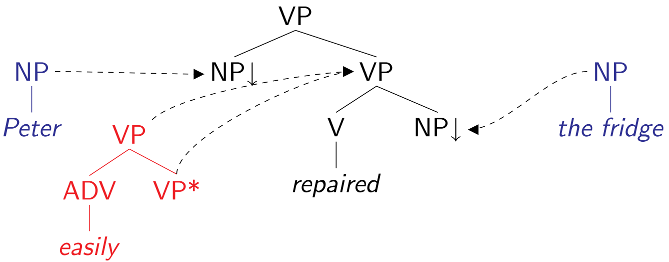
E.g.:



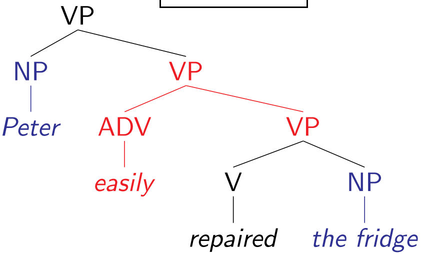
Combinatorial operations:

- substitution: replacing a non-terminal leaf with an initial tree
- adjunction: replacing an internal node with an auxiliary tree

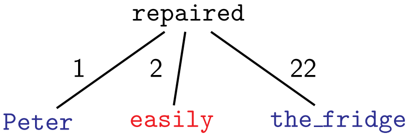
Tree-Adjoining Grammar - Example



derived tree



derivation tree



Tree-Adjoining Grammar - Basics

TAGs are **mildly context-sensitive:**

1) Polynomial time parsing complexity
2) Generation of limited crossing dependencies
3) Constant growth property (semilinearity)

Mild context-sensitivity characterizes the generative capacity needed for the analysis of natural language syntax.

Large TAG grammars:

- English and Korean (XTAG, UPenn)
- French TAG (Benoit Crabbé's PhD-thesis)
- German (GerTT)
- . . .

Tree-Adjoining Grammar - Basics

TAGs are **mildly context-sensitive:**

1) Polynomial time parsing complexity
2) Generation of limited crossing dependencies
3) Constant growth property (semilinearity)

Mild context-sensitivity characterizes the generative capacity needed for the analysis of natural language syntax.

Large TAG grammars:

- English and Korean (XTAG, UPenn)
- French TAG (Benoit Crabbé's PhD-thesis)
- German (GerTT)
- ...

Two ways of grammar implementation with TAG

1) **XTAG tools** (UPenn)
   - parser, editor, viewer, . . .
2) **XMG + TuLiPA**
   - XMG: eXtensible MetaGrammar (Duchier et al, 2004)
   - TuLiPA: Tübingen Linguistic Parsing Architecture (Parmentier et al, 2008)

Inside and outside this lecture

- **What we are going to cover:**
  1. Grammar formalism: Tree Adjoining Grammar (TAG)
  2. Phenomena + analysis from the XTAG grammar (syntax, few semantics)
  3. Implementation: XTAG tools, XMG + TuLiPA

- **What is not part of our concerns in this lecture:**
  - pragmatics, morphology, phonetics/phonology , . . .
  - Head Driven Phrase Structure Grammar (HPSG),
    Combinatorial Categorial Grammar (CCG),
    Lexical Functional Grammar (LFG),
    Transformational Grammar (GB), Minimalism
  - corpus-driven approaches (quantitative linguistics)

Inside and outside this lecture

- **What we are going to cover:**
  1. Grammar formalism: Tree Adjoining Grammar (TAG)
  2. Phenomena + analysis from the XTAG grammar (syntax, few semantics)
  3. Implementation: XTAG tools, XMG + TuLiPA

- **What is not part of our concerns in this lecture:**
  - pragmatics, morphology, phonetics/phonology , . . .
  - Head Driven Phrase Structure Grammar (HPSG), Combinatorial Categorial Grammar (CCG), Lexical Functional Grammar (LFG), Transformational Grammar (GB), Minimalism
  - corpus-driven approaches (quantitative linguistics)