Tree Adjoining Grammars
TAG: The syntax-semantics interface

Laura Kallmeyer & Benjamin Burkhardt

HHU Düsseldorf

WS 2017/2018
Outline

1. LTAG semantics: Overview
   - Synchronous TAGs for semantics
   - Unification-based LTAG semantics with predicate logic
   - Unification-based LTAG semantics with frames

2. Introduction to frame semantics

3. Case study: Directed motion construction
**Goal:** an LTAG architecture of the syntax-semantics interface that

- is compositional: the meaning of a complex expression can be computed from the meaning of its subparts and its composition operation.
- pairs entire elementary trees with meaning components.
LTAG semantics: overview

Three principal approaches:

1. LTAG semantics with synchronous TAG (STAG) (Shieber, 1994; Nesson and Shieber, 2006, 2008)

2. Unification based LTAG semantics with predicate logic (Kallmeyer and Joshi, 2003; Gardent and Kallmeyer, 2003; Kallmeyer and Romero, 2008)

3. Unification based LTAG semantics with frames (Kallmeyer and Osswald, 2013; Kallmeyer et al., 2016)

We will use the third approach in this course and only briefly present the other two.
Idea:

- pair two TAGs, one for syntax and one for L(ogical) F(orm) (= typed predicate logic),
- and do derivations in parallel.

Formalism used for this: **synchronous TAG (STAG)** Shieber and Schabes (1990); Shieber (1994).

STAG = two TAGs $G_1$, $G_2$ whose trees are related to each other. More precisely, it contains pairs $\langle \gamma_1, \gamma_2, \text{link} \rangle$ where $\gamma_1$ is an elementary tree from $G_1$, $\gamma_2$ an elementary tree from $G_2$, and link is a set of pairs of node addresses from $\gamma_1$ and $\gamma_2$ respectively.
The non-terminals of the semantic TAG are types \( t, e, \langle e, t \rangle, \ldots \).

The semantic TAG describes the syntactic structure of typed predicate logical formulas.

The links in this example tell us, for instance, that the subject NP corresponds to the \( e \) argument of \texttt{laugh}.
LTAG semantics: STAG

STAG derivation proceeds as in TAG, except that all operations must be paired: In every derivation step:

- A new elementary tree pair $\langle \gamma_1, \gamma_2 \rangle$ is picked.
- $\gamma_1$ is attached (substituted or adjoined) to the syntactic tree while $\gamma_2$ is attached to the semantic tree.
- The nodes that the two trees attach to must be linked.
- The link that is used in this derivation step disappears while all other links involving the attachment sites are inherited by the root of the attaching tree.
LTAG semantics: STAG

Logical form:

\[
\text{sometimes}(\text{laugh}(\text{John}))
\]
LTAG semantics: STAG

Logical form: 

sometimes (laugh (john))
Laughter:  

S₁  

NP  VP₃  

John  V  laughed  

VP  Adv  VP*  ⟨t, t⟩  t*  

sometimes  

Logical form:  

sometimes (laugh (john))
LTAG semantics: STAG

Logical form:

\[
\text{sometimes} (\text{laugh} (\langle \text{e}, \text{t} \rangle, \text{John}))
\]
Logical form: \( \text{sometimes}(\text{laugh}(\text{john})) \)

Idea: Each elementary tree is paired with
Unification-based LTAG semantics with predicate logic


Idea: Each elementary tree is paired with

- A set of typed predicate logic expressions and of scope constraints (i.e., constraints on sub-term relations)

Idea: Each elementary tree is paired with

- A set of typed predicate logic expressions and of scope constraints (i.e., constraints on sub-term relations)
- Interface features that characterizes a) which arguments need to be filled, b) which elements are available as arguments for other elementary trees and c) the scope behaviour.

The features are linked to positions in the elementary tree.
Unification-based LTAG semantics with predicate logic

$$S \rightarrow NP^{[1=1]} \rightarrow VP^{[p=2]} \rightarrow V \rightarrow \text{laughed}$$

$$l_1: \text{laugh}(\overline{1})$$
Unification-based LTAG semantics with predicate logic

$$S \rightarrow \text{NP}_{[I=1]} \cdot \text{VP}_{[p=2]} \cdot l_1 : \text{laugh}(1)$$

$$l_3 : \text{john}(x) \rightarrow \text{NP}_{[I=x]} \cdot \text{VP}_{[p=\_]}$$

$$\text{VP}_{[p=\_]} \rightarrow \text{V}$$

$$\text{V} \rightarrow \text{laughed}$$

Signifies that the formula labeled $l_1$ is a subformula of the formula that has to be placed in the hole. Disambiguation leads to $\text{john}(x) \land \text{sometimes}((\text{laugh}(x)))$.
Unification-based LTAG semantics with predicate logic

\[ l_1 : \text{laugh}(\square) \]

\[ l_3 : \text{john}(x) \]

\[ \text{NP}[I=1] \]

\[ \text{NP}[I=x] \]

\[ \text{VP}[p=2] \]

\[ \text{VP}[p=l_1] \]

\[ \text{V} \]

\[ \text{laughed} \]

\[ \text{John} \]
Unification-based LTAG semantics with predicate logic

\[
S \\
\text{NP}[I=x] \quad \text{VP}[p=2] \\
\mid \quad \mid \\
\text{John} \quad \text{V} \\
\mid \\
\text{laughed} \\
\]

\[
l_1 : \text{laugh}(x), \\
l_3 : \text{john}(x)
\]
Unification-based LTAG semantics with predicate logic


$\begin{align*}
l_1 &: \text{laugh}(x), \\
l_3 &: \text{john}(x) \\
l_2 &: \text{sometimes}(6), \\
6 &: \triangleleft^* 7
\end{align*}$

Signifies that the formula labeled $l_1$ is a subformula of the formula that has to be placed in the hole $6$. Disambiguation leads to $\text{john}(x) \land \text{sometimes}(\text{laugh}(x))$. 
Unification-based LTAG semantics with predicate logic

$$l_1 : \text{laugh}(x), \quad l_3 : \text{john}(x)$$

$$l_2 : \text{sometimes}(6), \quad 6 \triangleleft^* 7$$

The formula labeled $l_1$ is a subformula of the formula that has to be placed in the hole.

Disambiguation leads to:

$$\text{john}(x) \land \text{sometimes}(\text{laugh}(x))$$
Unification-based LTAG semantics with predicate logic

NP[\text{John}] \quad \text{VP}^{[\text{laughed}]} \quad \text{Adv} \quad \text{VP}^{[\text{sometimes}]} \quad \text{V} \quad \text{NP}^{[\text{sometimes}]} 

\text{VP}^{[\text{laugh}]} \quad \text{NP}^{[\text{John}]} 

\text{l}_1 : \text{laugh}(x), \quad \text{l}_3 : \text{john}(x), \quad \text{l}_2 : \text{sometimes}(6), \quad 6 \triangleleft^* 7

\text{signifies that the formula labeled } l_1 \text{ is a subformula of the formula that has to be placed in the hole. Disambiguation leads to } \text{John}(x) \land \text{sometimes}(\text{laugh}(x)).
Unification-based LTAG semantics with predicate logic

\[ l_1 : \text{laugh}(x), \]
\[ l_3 : \text{john}(x), \]
\[ l_2 : \text{sometimes}(\overline{6}), \]
\[ \overline{6} \triangleleft l_1 \]
Unification-based LTAG semantics with predicate logic

\[ l_1 : \text{laugh}(x), \]
\[ l_3 : \text{john}(x), \]
\[ l_2 : \text{sometimes}(\llbracket 6 \rrbracket), \]
\[ \llbracket 6 \rrbracket \triangleleft^* l_1 \]

\[ \llbracket 6 \rrbracket \triangleleft^* l_1 \] signifies that the formula labeled \( l_1 \) is a subformula of the formula that has to be placed in the hole \( \llbracket 6 \rrbracket \).

Disambiguation leads to \( \text{john}(x) \land \text{sometimes}(\text{laugh}(x)) \)
Unification-based LTAG semantics with frames

- Semantic representations are linked to entire elementary trees (as in the previous approaches).

- Semantic representations: frames, expressed as typed feature structures.

- Interface features relate nodes in the syntactic tree to nodes in the frame graph.

- Frame composition by unification, triggered by the unifications on the interface features that are in turn triggered by substitution, adjunction and final top-bottom unification on the derived tree.
(1) Adam ate an apple.
(1) Adam ate an apple.

\[
S 
\quad \xrightarrow{e} \quad \text{[eating, actor \(x\), theme \(y\)]}
\]

\[
NP[I=x] 
\quad \xrightarrow{\text{NP[I=u]}} 
\quad \text{[person, name 'Adam']}
\]

\[
VP[I=e] 
\quad \xrightarrow{V} 
\quad \text{['ate']}
\]

\[
NP[I=y] 
\quad \xrightarrow{\text{'Adam'}} 
\quad \text{[person, name 'Adam']}
\]
(1) Adam ate an apple.
(1) Adam ate an apple.
(1) Adam ate an apple.
Introduction to frame semantics

Frames as used in LTAG

- A representation format for rich lexical and constructional content.
- Can nicely capture semantic composition and decomposition.
- Can be formalized as generalized feature structures with types, relations and node labels.
Introduction to frame semantics

Frames as used in LTAG

- A representation format for rich lexical and constructional content.
- Can nicely capture semantic composition and decomposition.
- Can be formalized as generalized feature structures with types, relations and node labels.

Basic assumptions

- Attributes (features, functional roles/relations) play a central role in the organization of semantic and conceptual knowledge and representation.
- Semantic components (participants, subevents) can be (recursively) addressed via attributes (from some “base” node).

→ inherently structured representations (models); composition by unification (under constraints)
Introduction to frame semantics

Example

*locomotion*:

- **MANNER**: walking
- **PATH**: man
- **MOVEMENT**: part-of

*house*:

- **IN-REGION**: region

Core property:

Every node is reachable from some labeled "base" node via attributes.
Introduction to frame semantics

Example

Ingredients

- Attributes (funct. relations): ACTOR, MOVER, PATH, MANNER, IN-REGION, …
Introduction to frame semantics

Example

Ingredients

- Attributes (funct. relations): ACTOR, MOVER, PATH, MANNER, IN-REGION, …
- Type symbols: locomotion, man, path, walking, region, …
Introduction to frame semantics

Example

```
<table>
<thead>
<tr>
<th>e</th>
<th>locomotion</th>
<th>x</th>
<th>man</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>MANNER</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>PATH</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACTOR</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>MOVER</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>e</th>
<th>walking</th>
<th>x</th>
<th>path</th>
<th>region</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>e</th>
<th>region</th>
<th>z</th>
<th>house</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Ingredients

- Attributes (funct. relations): ACTOR, MOVER, PATH, MANNER, IN-REGION, …
- Type symbols: *locomotion*, *man*, *path*, *walking*, *region*, …
- Proper relations: *part-of*
Introduction to frame semantics

Example

Ingredients

- Attributes (funct. relations): ACTOR, MOVER, PATH, MANNER, IN-REGION, …
- Type symbols: locomotion, man, path, walking, region, …
- Proper relations: part-of
- Node labels (variables): e, x, z
Introduction to frame semantics

Example

Ingredients

- Attributes (funct. relations): ACTOR, MOVER, PATH, MANNER, IN-REGION, …
- Type symbols: locomotion, man, path, walking, region, …
- Proper relations: part-of
- Node labels (variables): e, x, z

Core property

- Every node is reachable from some labeled “base” node via attributes.
Introduction to frame semantics

Example

(2) Anna ran
Introduction to frame semantics

Example

(2) Anna ran
Example

(2) Anna ran to the station.
Introduction to frame semantics

Example

(2) Anna ran to the station.

`bounded-motion` running

`loc-stage` `station`

`ACTOR` `NAME` ‘Anna’

`FINAL` `THEME` `LOC` `person` `station`
Introduction to frame semantics

Example

(2) Anna ran to the station.

\[
\text{running} \land \text{bounded-motion} \\
E \left[ \begin{array}{c}
\text{actor} \ 1 \\
\text{name} \ 
\end{array} \right] \\
\text{loc-stage} \\
\text{theme} \ 1 \\
\text{loc} \ 1 \\
\text{station} \left[ \right]
\]
Introduction to frame semantics

Example

(2) Anna ran to the station.

Attribute-value logic

\[ e \cdot (\text{running} \land \text{bounded-motion} \land \text{ACTOR} : (\text{person} \land \text{NAME} \equiv \text{Anna}')) \]

\[ \text{ACTOR} \equiv \text{FINAL} \land \text{THEME} \land \text{FINAL} : (\text{loc-stage} \land \text{LOC} : \text{station}) \]
Introduction to frame semantics

Example

(2) Anna ran to the station.

Attribute-value logic

\[ e \cdot (\text{running} \land \text{bounded-motion} \land \text{ACTOR}: (\text{person} \land \text{NAME} \triangleq \text{‘Anna’}) \]

\[ \text{ACTOR} \models \text{FINAL} \land \text{THEME} \land \text{FINAL}: (\text{loc-stage} \land \text{LOC}: \text{station}) \]

Translation into first-order logic

\[ \exists x \exists s \exists y (\text{running}(e) \land \text{bounded-motion}(e) \land \text{ACTOR}(e, x) \land \text{person}(x) \land \text{NAME}(x, \text{‘Anna’}) \]

\[ \land \text{FINAL}(e, s) \land \text{loc-stage}(s) \land \text{THEME}(s, x) \land \text{LOC}(s, y) \land \text{station}(y)) \]
Introduction to frame semantics

Example

(2) Anna ran to the station.

Attribute-value logic

\[ e \cdot (\text{running} \land \text{bounded-motion} \land \text{ACTOR} : (\text{person} \land \text{NAME} \triangleq \text{‘Anna’})) \]

Constraints

\[ \text{running} \supset \text{activity} \text{ (short for } \forall e(\text{running}(e) \rightarrow \text{activity}(e))) \]
\[ \text{loc-stage} \supset \text{THEME} : \top \land \text{LOC} : \top, \ldots \]
Case study: directed motion construction

Intransitive:

(3) a. Mary walked to the house.
    b. The ball rolled into the goal.
Case study: directed motion construction

Intransitive:

(3) a. Mary walked to the house.
    b. The ball rolled into the goal.

Transitive:

(4) a. John threw/kicked the ball into the goal.
    b. John pushed/pulled the cart to the station.
    c. John rolled the ball into the hole.
Case study: directed motion construction

Intransitive:

(3) a. Mary walked to the house.
   b. The ball rolled into the goal.

Transitive:

(4) a. John threw/kicked the ball into the goal.
   b. John pushed/pulled the cart to the station.
   c. John rolled the ball into the hole.

Directional specifications are not restricted to goal expressions but can also describe the source or the course of the path in more detail.
Case study: directed motion construction

Intransitive:

(3) a. Mary walked to the house.
   b. The ball rolled into the goal.

Transitive:

(4) a. John threw/kicked the ball into the goal.
   b. John pushed/pulled the cart to the station.
   c. John rolled the ball into the hole.

Directional specifications are not restricted to goal expressions but can also describe the source or the course of the path in more detail. Moreover, path descriptions can be iterated to some extent:

(5) a. John walked through the gate along the fence to the house.
   b. John threw the ball over the fence into the yard.
Case study: directed motion construction

**Question:** Syntactic treatment of directional PPs?

- Construction (⇝ elementary tree)
- Syntactic composition (⇝ adjunction)

Arguments for treating goal (or bounded) PPs constructionally, in contrast to path (or unbounded) PPs:

- Goal PPs cannot be iterated.
- They affect the Aktionsart of the expression:
  
  (6) a. She walked (*in half an hour / for half an hour).
  
  b. She walked to the brook (*in half an hour / *for half an hour).
  
  c. She walked along the brook (*in half an hour / *for half an hour).
Case study: directed motion construction

**Question:** Syntactic treatment of directional PPs?

- Construction (⇝ elementary tree)
- Syntactic composition (⇝ adjunction)

Arguments for treating goal (or **bounded**) PPs constructionally, in contrast to path (or **unbounded**) PPs:

- Goal PPs cannot be iterated.
Question: Syntactic treatment of directional PPs?

- Construction (⇝ elementary tree)
- Syntactic composition (⇝ adjunction)

Arguments for treating goal (or bounded) PPs constructionally, in contrast to path (or unbounded) PPs:

- Goal PPs cannot be iterated.
- They affect the Aktionsart of the expression:

(6) a. She walked (*in half an hour/for half an hour).
   b. She walked to the brook (in half an hour/*for half an hour).
   c. She walked along the brook (*in half an hour/for half an hour).
Case study: directed motion construction

Unanchored construction for intransitive directed motion ($n0Vpp(dir)$):

$S$ [i.sc=$x$] $VP$ [e.sc=$e$]

$VP$ [e.sc=$e$] $PP$ [i.sc=$z$, e.sc=$e$]

$V$ ◊ [e.sc=$e$]

$e$ [bounded-translocation]

MOVER $x$

GOAL $z$

PATH [path]
Case study: directed motion construction

Unanchored construction for intransitive directed motion ($\text{n0Vpp(dir)}$):

Elementary tree for ‘into’:
Case study: directed motion construction

Example (intransitive directed motion)

(7) John walked into the house.
Case study: directed motion construction

Example (intransitive directed motion)

(7) John walked into the house.
Case study: directed motion construction

Example (intransitive directed motion)

(7) John walked into the house.
Case study: directed motion construction

Example (intransitive directed motion)

(7) John walked into the house.
Case study: directed motion construction

Lexical anchoring (non-directed case)

**morph entry**

'walked'

pos: V

**lemma entry**

walk:

FAM: n0V, ...

**Syn**1:

\[
\begin{cases}
\text{AGR} = \begin{bmatrix}
\text{PERS} = 3 \\
\text{NUM} = \text{sg}
\end{bmatrix}
\end{cases}
\]

lemma: walk

**Syn**2:

\[
\begin{cases}
E = e_0
\end{cases}
\]

**Sem**:

\[
e_0 \left[ \text{locomotion} \mid \text{MANNER} \left[ \text{walking} \right] \right]
\]

\[
\sim \quad \sim
\]

Constraints:

- \( \text{locomotion} \Rightarrow \text{activity} \land \text{translocation} \)
- \( \text{translocation} \Rightarrow \text{motion} \land \text{PATH} : \text{path} \)
- \( \text{activity} \Rightarrow \text{ACTOR} : \top \)
- \( \text{motion} \Rightarrow \text{MOVER} : \top \)
- \( \text{activity} \land \text{motion} \Rightarrow \text{ACTOR} \neq \text{MOVER} \)

**S**

\[
\begin{align*}
\text{S} & \quad \text{N} \left[ \text{I} = \text{x} \right] \\
& \quad \text{V} \left[ \text{AGR} = \ldots, E = e_0 \right] \\
& \quad \text{V} \left[ \text{AGR} = \ldots, E = e \right] \\
& \quad \text{V} \left[ \text{AGR} = \ldots, E = e \right]
\end{align*}
\]

\[
\begin{align*}
\text{S} & \quad \text{N} \left[ \text{I} = \text{x} \right] \\
& \quad \text{V} \left[ \text{AGR} = \ldots, E = e \right] \\
& \quad \text{V} \left[ \text{AGR} = \ldots, E = e \right]
\end{align*}
\]

\[
\begin{align*}
\text{S} & \quad \text{N} \left[ \text{I} = \text{x} \right] \\
& \quad \text{V} \left[ \text{AGR} = \ldots, E = e \right]
\end{align*}
\]
Case study: directed motion construction

Example

(8) John walked along the brook.
Case study: directed motion construction

Example

(8) John walked along the brook.
Case study: directed motion construction

Example

(8) John walked along the brook.
Case study: directed motion construction

Example (causative directed motion)

(9) Mary threw/kicked/rolled the ball into the room.
Case study: directed motion construction

**Example** (causative directed motion)

(9) Mary threw/kicked/rolled the ball into the room.

Unanchored construction (\(n0Vn1pp(dir)\)):
Case study: directed motion construction

Example (causative directed motion)

(9) Mary threw/kicked/rolled the ball into the room.

Unanchored construction ($n0Vn1pp(dir)$):

(Partial) lexical entry for ‘threw’:


