Introduction to Abstract Categorial Grammars

Fourth course
• Preliminary notions:
  ▸ coherence theorem
  ▸ closure properties
  ▸ definition of membership problems
• Complexity of membership problems:
  ▸ $G(3, n)$
  ▸ $G(2, n)$
• Non-linear ACGs
The coherence theorem
Properties of the linear lambda-calculus:
Properties of the linear lambda-calculus:

- **Subject expansion**
Properties of the linear lambda-calculus:

- **Subject expansion**
- **Every linear lambda-term is simply typable**
Properties of the linear lambda-calculus:

- **Subject expansion**

- **Every linear lambda-term is simply typable**

- **The principal type of a linear lambda-term specifies uniquely this term (coherence)**
Linearity and coherence

Let's take $\Sigma = (\{o\}, \{a\}, \{a : o \rightarrow o\})$

$\lambda x. a(a(x))$ has $o \rightarrow o$ as principal type
Linearity and coherence

Let’s take \( \Sigma = (\{o\}, \{a\}, \{a : o \to o\}) \)

\( \lambda x. a(a(x)) \) has \( o \to o \) as principal type

But so does \( \lambda x. a(a(a(x))) \)
Recovering linearity

Relabelings: lexicons of complexity 1
Recovering linearity

Relabelings: lexicons of complexity 1

Example:

\[ \Sigma' = (\{o_1, o_2, o_3\}, \{a_1, a_2\}, \{a_1 : o_2 \rightarrow o_1, a_2 : o_3 \rightarrow o_2\}) \]

\[ \Sigma = (\{o\}, \{a\}, \{a : o \rightarrow o\}) \]

\[ \mathcal{R}(o_i) = o \]
\[ \mathcal{R}(a_i) = a \]
Recovering linearity

Relabelings: lexicons of complexity 1

Example:

\[ \Sigma' = (\{o_1, o_2, o_3\}, \{a_1, a_2\}, \{a_1 : o_2 \rightarrow o_1, a_2 : o_3 \rightarrow o_2\}) \]

\[ \begin{array}{c}
\Sigma = (\{o\}, \{a\}, \{a : o \rightarrow o\}) \\
R(o_i) = o \\
R(a_i) = a \\
\end{array} \]

\[ \lambda x.a_1(a_2 x) \text{ is the only term of} \]

\[ \Lambda_{\Sigma'} \text{ in } \Lambda_{\Sigma'} \text{ type } o_3 \rightarrow o_1 \]
Recovering linearity

Relabelings: lexicons of complexity 1

Example:

\[ \Sigma' = (\{o_1, o_2, o_3\}, \{a_1, a_2\}, \{a_1 : o_2 \rightarrow o_1, a_2 : o_3 \rightarrow o_2\}) \]

\[ R(\Sigma') = \sum = (\{o\}, \{a\}, \{a : o \rightarrow o\}) \]

\[ R(o_i) = o \]
\[ R(a_i) = a \]

\[ \lambda x.a_1(a_2 x) \text{ is the only term of type } o_3 \rightarrow o_1 \text{ in } \Lambda_{\Sigma'} \]

\[ R(\lambda x.a_1(a_2 x)) = \lambda x.a(a x) \]
Recovering linearity

Proposition (coherence):

For every term $t$ in $\Lambda_\Sigma$ there is:

- A signature $\Sigma'$
- A relabeling $R : \Sigma' \rightarrow \Sigma$
- A term $t'$ in $\Lambda_\Sigma^\alpha$

Verifying
Recovering linearity

Proposition (coherence):

For every term \( t \) in \( \Lambda^\Sigma \) there is:

- A signature \( \Sigma' \)
- A relabeling \( \mathcal{R} : \Sigma' \rightarrow \Sigma \)
- A term \( t' \in \Lambda^{\alpha}_{\Sigma} \), verifying

\( t' \) is the unique term of type \( \alpha \) in \( \Lambda^\Sigma' \)
Recovering linearity

Proposition (coherence):

For every term \( t \) in \( \Lambda_{\Sigma} \) there is:

A signature \( \Sigma' \)

A relabeling \( \mathcal{R} : \Sigma' \to \Sigma \)

A term \( t' \) in \( \Lambda_{\Sigma'}^\alpha \), verifying

\( t' \) is the unique term of type \( \alpha \) in \( \Lambda_{\Sigma'} \)

\( \mathcal{R}(t') = t \)
Interpreting coherence

\[ a : o \rightarrow o \]
\[ \lambda x. a(a(a(a(x)))) \]

\[ a_i : o_{i+1} \rightarrow o_i \]
\[ \lambda x. a_1(a_2(a_3(a_4(a_5(x)))) : o_6 \rightarrow o_1 \]

\[ o_1 \quad o_2 \quad o_3 \quad o_4 \quad o_5 \quad o_6 \]
Interpreting coherence

\[ a : o \rightarrow o \]

\[ \lambda x.a(a(a(a(a(x)))) \]

\[ \lambda x.a_1(a_2(a_3(a_4(a_5(x)))) : o_6 \rightarrow o_1 \]

\[ o_1 \ o_2 \ o_3 \ o_4 \ o_5 \ o_6 \]

Intervals:

\[ \lambda x.a_i(\cdots(a_{i+k}x)\cdots) : o_{i+k+1} \rightarrow o_i \]
Interpreting coherence

\[ a : o \rightarrow o \]
\[ a_i : o_{i+1} \rightarrow o_i \]
\[ \lambda x.a(a(a(a(x)))) \]
\[ \lambda x.a_1(a_2(a_3(a_4(a_5(x)))) : o_6 \rightarrow o_1 \]
\[ o_1 \ o_2 \ o_3 \ o_4 \ o_5 \ o_6 \]

Intervals:

\[ \lambda x.a_i(\cdots(a_{i+k}x)\cdots) : o_{i+k+1} \rightarrow o_i \]

String contexts:

\[ \lambda s.a_i + \cdots + a_{i+k} + s + a_{i+k+l} + \cdots + a_{i+k+l+m} : \]
\[ (o_{i+k+1} \rightarrow o_{i+k+l}) \rightarrow (o_{i+k+l+m} \rightarrow o_i) \]
Closure properties
Closure under inverse relabelling

Proposition:

Let \( G = \langle \Sigma_1, \Sigma_2, \mathcal{L}, s \rangle \) of \( G(m, n) \)
\( \mathcal{R} : \Sigma_3 \to \Sigma_1 \)
\( \alpha \) a type in \( \Sigma_3 \) such that \( \mathcal{R}(\alpha) = \mathcal{L}(s) \)

There is \( G' = \langle \Sigma_1', \Sigma_3, \mathcal{L}', s' \rangle \) of \( G(m, n) \) such that

\[
\mathcal{O}(G') = \mathcal{R}^{-1}(\mathcal{O}(G)) \cap \Lambda_{\Sigma_3}^\alpha
\]
A pictorial view

\[ \Lambda \Sigma_3 \]

\[ R^{-1}(O(G)) \]

\[ \Lambda_\Sigma^\alpha \]

\[ O(G') \]
Other closure properties

$L(m,n)$ is closed under union

$L_{string}(m,n)$ is closed under intersection with regular sets of strings

$L_{tree}(m,n)$ is closed under intersection with regular sets of trees
Membership Problems
TWO KINDS OF PROBLEMS

Universal membership:

Fixed data: a class of grammar
Input: a grammar $G$ and an object $t$
Output: Yes/No when $t \in L(G) / t \notin L(G)$
Two kinds of problems

Universal membership:

Fixed data: a class of grammar
Input: a grammar \( G \) and an object \( t \)
Output: Yes/No when \( t \in L(G) \) / \( t \notin L(G) \)

Membership:

Fixed data: a grammar \( G \)
Input: an object \( t \)
Output: Yes/No when \( t \in L(G) \) / \( t \notin L(G) \)
<table>
<thead>
<tr>
<th></th>
<th>Membership</th>
<th>Universal Membership</th>
</tr>
</thead>
<tbody>
<tr>
<td>CFGs/TAGs</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>Lambek Grammars</td>
<td>P</td>
<td>NP-complete</td>
</tr>
<tr>
<td>MCFGs</td>
<td>P</td>
<td>NP-complete</td>
</tr>
</tbody>
</table>
Complexity of $G(3, n)$
The emptiness problem in \( G(3, n) \) is Turing-equivalent to proof search in MELL.

So is the membership problem.
Proof search in MELL is Turing-equivalent to the emptiness problem of \( \Lambda^\alpha_\Sigma \).

The emptiness problem in \( G(3, n) \) is Turing-equivalent to proof search in MELL.

So is the membership problem.
Membership and MELL

\[ A(\mathcal{G}) \]

\[ \Lambda_{\Sigma_1}^{s} \]

\[ \Lambda_{\Sigma_2}^{\mathcal{L}(s)} \]

\[ \mathcal{L} \]

\[ t \in \Lambda_{\Sigma_2}^{\mathcal{L}(s)} \]

\[ t \in \mathcal{O}(\mathcal{G}) \]
Membership and MELL

Coherence

\[ \Lambda^s_{\Sigma_1} \]
\[ \Lambda_{\Sigma_2}^{L(s)} \]
\[ \mathcal{A}(\mathcal{G}) \]
\[ \mathcal{L} \]
\[ \mathcal{O}(\mathcal{G}) \]

\[ t \in \Lambda_{\Sigma_2}^{L(s)} \]
\[ t \in \mathcal{O}(\mathcal{G}) \]
\[ \mathcal{R}(t') = t \]

\[ \Lambda^\alpha_{\Sigma_3} = \{ t' \} \]
Membership and MELL

Coherence

Closure under inverse relabeling

\[ \Lambda_{\Sigma_1}^{s} \subseteq A(G) \]

\[ \Lambda_{\Sigma_2}^{L(s)} \subseteq O(G) \]

\[ \Lambda_{\Sigma_3}^{\alpha} = \{ t' \} \]

\[ \mathcal{R}^{-1}(O(G)) \]

\[ t \in \Lambda_{\Sigma_2}^{L(s)} \quad t \in O(G) \quad \mathcal{R}(t') = t \]
Membership and MELL

Exercise:

Reduce emptiness of an ACG to the membership problem
Membership and MELL

Exercise:

Reduce emptiness of an ACG to the membership problem

Hint 1: prove that sets of pure terms of a given type is finite
Membership and MELL

Exercise:

Reduce emptiness of an ACG to the membership problem

Hint 1: prove that sets of pure terms of a given type is finite

Hint 2: prove that the sets of pure terms built with types of the form

\[ T ::= (\sigma \rightarrow \sigma) \mid (T \rightarrow T) \]

are not empty
Lexicalized grammars

\[ G = \langle \Sigma_1, \Sigma_2, \mathcal{L}, s \rangle \] is lexicalized if for every \( c \in \Sigma_1 \)

\( \mathcal{L}(c) \) contains at least one object constant
Lexicalized grammars

\[ G = \langle \Sigma_1, \Sigma_2, \mathcal{L}, s \rangle \text{ is lexicalized if for every } c \in \Sigma_1, \mathcal{L}(c) \text{ contains at least one object constant} \]

<table>
<thead>
<tr>
<th></th>
<th>Membership</th>
<th>Universal Membership</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>G(3, 1)</strong></td>
<td>NP-complete</td>
<td>NP-complete</td>
</tr>
<tr>
<td><strong>G(3, n)</strong></td>
<td>NP-complete</td>
<td>NP-complete</td>
</tr>
</tbody>
</table>
Complexity of $G(2, n)$
<table>
<thead>
<tr>
<th></th>
<th>Membership</th>
<th>Universal Membership</th>
</tr>
</thead>
<tbody>
<tr>
<td>$G(2,1)$</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>$G(2,n) ; n &gt; 1$</td>
<td>P</td>
<td>NP-complete</td>
</tr>
</tbody>
</table>
Let \( G = (N, T, R, s) \) be a CFG
\[ a_1 \ldots a_n \]
CYK algorithm for CFGs

Let \( G = (N, T, R, s) \) be a CFG

\[ a_1 \ldots a_n \]

Items \( (Y, i, j) \) meaning \( Y \to^* a_{i+1} \ldots a_j \)
**CYK Algorithm for CFGs**

**Let** $G = (N, T, R, s)$ **be a CFG**

$$a_1 \ldots a_n$$

**Items** $(Y, i, j)$ **meaning** $Y \rightarrow^* a_{i+1} \ldots a_j$

$X \rightarrow w_1 X_1 \ldots w_n X_n w_{n+1}$

$w_1 = a_i \ldots a_j$ $(X_1, j + 1, k)$

$\vdots$

$w_n = a_l \ldots a_m$ $(X_n, m + 1, p)$

$w_{n+1} = a_{p+1} \ldots a_q$ $(X, i, q)$
**CYK algorithm for** \( G(2, n) \)

**Let** \( G = \langle \Sigma_1, \Sigma_2, \mathcal{L}, s \rangle \) **be in** \( G(2, n) \) **and** \( t \in \Lambda_{\Sigma_2}^{\mathcal{L}(s)} \)

**Coherence:** \( \Sigma_3, \mathcal{R}, \alpha, t' \)

**Item:** \((\beta, \gamma)\) with \( \beta \in \Sigma_1, \gamma \in \Sigma_2 \) and \( \mathcal{L}(\beta) = \mathcal{R}(\gamma) \)

**Meaning:** there is \( t \in \Lambda_{\Sigma_1}^{\beta} \) such that \( \mathcal{R}^{-1}(\mathcal{L}(t)) \cap \Lambda_{\Sigma_3}^{\gamma} \neq \emptyset \)

\[
(\beta_1, \gamma_1) \cdots (\beta_n, \gamma_n) \\
\mathcal{R}^{-1}(\mathcal{L}(c)) \cap \Lambda_{\Sigma_3}^{\gamma_1} \rightarrow \cdots \rightarrow \gamma_n \rightarrow \gamma \\
\neq \emptyset
\]

\[
(\beta, \gamma)
\]
Non linear ACGs
ABSTRACT
SYNTACTIC
STRUCTURE
\[ \Sigma_0 \]
ABSTRACT
SYNTACTIC
STRUCTURE

$\Sigma_0$

SYNTACTIC
FORM

$\Sigma_1$
ABSTRACT
SYNTACTIC
STRUCTURE

\[ \Sigma_0 \]

\[ \Sigma_1 \]
SYNTACTIC
FORM

\[ \Sigma_2 \]
SEMANTIC
FORM
The diagram illustrates the relationship between abstract syntactic structure, syntactic form, semantic form, and a set $\Sigma_0$.

- $\Sigma_0$: Abstract syntactic structure
- $\Sigma_1$: Syntactic form
- $\Sigma_2$: Semantic form
- $L_{\text{synt}}$: Set representing syntactic structure
Abstract syntactic structure

\[ \Sigma_0 \]

\[ L_{\text{synt}} \]
\[ \Sigma_1 \]
Syntactic form

\[ L_{\text{sem}} \]
\[ \Sigma_2 \]
Semantic form
Decidability and non-linearity

$G_{\text{non-lin}}(3, n)$: undecidable membership problem

$G_{\text{non-lin}}(2, n)$: decidable membership problem

$\exists x. \text{ unicorn } x \land \text{ find } j x$
Decidability of $G_{\text{non-lin}}(2, n)$

- Recognizability for lambda-terms
- Singleton sets are recognizable
- Recognizable sets are closed under:
  - Boolean operations
  - Inverse homomorphisms (lexicons)