Exam: Abstract Categorial Grammars

Duration: 3 hours.
Written documents are allowed. The numbers in front of questions are indicative of hardness or duration. Please put your answers to sections 1 and 2 on separate sheets; do not forget to write your name on both.

Quick Course Recap. Recall from the course that a *higher-order linear signature* is a triple $\Sigma = \langle A, C, \tau \rangle$ where A is a finite set of atomic types, C is a finite set of constants, and $\tau : C \to \mathcal{T}(A)$ is a function that assigns each constant in C to a *linear implicative type* α built over A, according to the syntax

$$\alpha ::= a \mid \alpha \multimap \alpha$$

where a ranges over A. By convention we consider $\multimap$ to be right-associative, i.e. we write $\alpha \multimap \beta \multimap \gamma$ for $\alpha \multimap (\beta \multimap \gamma)$. The *order* of a linear type is defined inductively as

$$\mathsf{ord}(a) = 1 \qquad\qquad \mathsf{ord}(\alpha \multimap \beta) = \max(\mathsf{ord}(\alpha) + 1, \mathsf{ord}(\beta)) \; .$$

Given a higher-order linear signature Σ, each linear lambda term of $\Lambda^\circ(\Sigma)$ can be assigned a type in $\mathcal{T}(A)$ by the typing system

$$\frac{}{\vdash_\Sigma c : \tau(c)}\,(\mathsf{Cons}) \qquad \frac{}{x : \alpha \vdash_\Sigma x : \alpha}\,(\mathsf{Var}) \qquad \frac{\Gamma, x : \alpha \vdash_\Sigma t : \beta}{\Gamma \vdash_\Sigma \lambda x.t : \alpha \multimap \beta}\,(\mathsf{Abs})$$

$$\frac{\Gamma \vdash_\Sigma t : \alpha \multimap \beta \quad \Delta \vdash_\Sigma u : \alpha}{\Gamma, \Delta \vdash_\Sigma tu : \beta}\,(\mathsf{App})$$

Note that x occurs free in t exactly once in (Abs) and the environments Γ and Δ are disjoint in (App).

Given two higher-order linear signatures Σ_1 and Σ_2, a *linear higher-order homomorphism* is generated by two functions $\eta : A_1 \to \mathcal{T}(A_1)$ on types and $\theta : C_1 \to \Lambda^\circ(\Sigma_2)$ on constants such that $\vdash_{\Sigma_2} \theta(c) : \eta(\tau_1(c))$ for all c in C_1, where η and θ are lifted in a natural way by $\eta(\alpha \multimap \beta) = \eta(\alpha) \multimap \eta(\beta)$ on the one hand, and $\theta(x) = x$, $\theta(\lambda x.t) = \lambda x.\theta(t)$, and $\theta(tu) = \theta(t)\theta(u)$ on the other hand.

An *abstract categorial grammar* is a tuple $\mathcal{G} = \langle \Sigma_1, \Sigma_2, \mathcal{L}, s \rangle$ where $\mathcal{L}$ is a linear higher-order homomorphism from Σ_1 to Σ_2 and s is a distinguished type in $\mathcal{T}(A_1)$. The *abstract language* generated by $\mathcal{G}$ is

$$\mathscr{A}(\mathcal{G}) = \{t \in \Lambda^\circ(\Sigma_1) \mid \vdash_{\Sigma_1} t : s\}$$

while its object language is the image of the abstract language by the homomorphism:

$$\mathscr{L}(\mathcal{G}) = \{t \in \Lambda^\circ(\Sigma_2) \mid \exists u \in \mathscr{A}(\mathcal{G}). t = \mathcal{L}(u)\} \; .$$

1 Second-Order ACGs and Tree Languages

Exercise 1 (Ground Lambda Terms)**.** Let Σ be a second-order linear signature, i.e. a signature such that the type of any constant c is of form

$$\tau(c) = a_1 \multimap \cdots \multimap a_n \multimap a_0$$

for atomic a_i's in A. Consider the normalized typing system with a single rule

$$\frac{\vdash'_\Sigma c : \tau(c) = a_1 \multimap \cdots \multimap a_n \multimap a_0 \quad \vdash'_\Sigma t_1 : a_1 \ \ldots\ \vdash'_\Sigma t_n : a_n}{\vdash'_\Sigma c\, t_1 \cdots t_n : a_0}\ (\mathsf{App}')$$

We want to show that, for all ground terms t and atomic types a, $\vdash_\Sigma t : a$ if and only if $\vdash'_\Sigma t : a$.

[2] 1. Show that, if $\tau(c) = a_1 \multimap \cdots \multimap a_n \multimap a_0$, $0 \le i \le n$, and $\vdash_\Sigma t_j : a_j$ for all $1 \le j \le i$, then $\vdash_\Sigma c\, t_1 \cdots t_i : a_{i+1} \multimap \cdots \multimap a_n \multimap a_0$. Deduce that $\vdash'_\Sigma t : a$ implies $\vdash_\Sigma t : a$ if t is ground and a atomic.

[2] 2. Show that, if $\vdash_\Sigma t : \alpha$ for a ground term t and type α, then $t = c\, t_1 \cdots t_i$ for some constant c with $\tau(c) = a_1 \multimap \cdots \multimap a_n \multimap a_0$, some $0 \le i \le n$, and some ground terms $t_1, \ldots, t_i$ such that $\alpha = a_{i+1} \multimap \cdots \multimap a_n \multimap a_0$ and $\vdash_\Sigma t_j : a_j$ for $0 \le j \le i$ for some atomic types a_j's.

[1] 3. Deduce that $\vdash_\Sigma t : a$ implies $\vdash'_\Sigma t : a$ whenever t is a ground term and a an atomic type.

Exercise 2 (Local Tree Languages)**.** For a second-order constant c with type $\tau(c) = a_1 \multimap \cdots \multimap a_n \multimap a_0$, we call n its *arity* (and thus can see C as a ranked alphabet) and associate to the *ground* lambda term $t = c\, t_1 \cdots t_n$ the unique tree $\bar{t} = c^{(n)}(\bar{t}_1, \ldots, \bar{t}_n)$. Given a second-order signature Σ and a distinguished atomic type s, we define the *tree language*

$$\mathscr{G}(\Sigma, s) = \{\bar{t} \in T(C) \mid \vdash_\Sigma t : s \text{ where } t \text{ is ground}\}\ .$$

[1] 1. Consider the second-order linear signature Σ_0 with atomic types $A_0 = \{np, s, c\}$, constants $C_0 = \{\textsc{alice}, \textsc{believe}, \textsc{left}, \textsc{someone}, \textsc{that}\}$, and typing

$$\tau_0(\textsc{alice}) = np \qquad \tau_0(\textsc{believe}) = c \multimap np \multimap s$$
$$\tau_0(\textsc{left}) = np \multimap s \qquad \tau_0(\textsc{someone}) = np$$
$$\tau_0(\textsc{that}) = s \multimap c$$

The corresponding ranked alphabet is $\mathcal{F}_0 = \{\textsc{alice}^{(0)}, \textsc{believe}^{(2)}, \textsc{left}^{(1)}, \textsc{someone}^{(0)}, \textsc{that}^{(1)}\}$.

Give a tree automaton over $\mathcal{F}_0$ for $\mathscr{G}(\Sigma_0, s)$.

[2] 2. Let $\mathcal{F}$ be a ranked alphabet. A deterministic top-down tree automaton $\mathcal{A} = \langle Q, \mathcal{F}, \delta, \{q_0\}\rangle$ is *local* if there exists a function $\ell{:}\mathcal{F} \to Q$ such that the rules in δ are all of the form $(\ell(f^{(n)}), f^{(n)}, q_1, \ldots, q_n)$.

Show that, if L is recognized by a local deterministic top-down tree automaton, then there is a second order linear signature Σ and a distinguished atomic type s such that $L = \mathscr{G}(\Sigma, s)$.

[2] 3. Show that, conversely, given a second-order signature Σ and a distinguished atomic type s, there exists a local top-down deterministic tree automaton $\mathcal{A}$ such that $L(\mathcal{A}) = \mathscr{G}(\Sigma, s)$.

[1] 4. Give an example of a regular tree language, which cannot be expressed as $\mathscr{G}(\Sigma, s)$ for any second-order linear signature Σ and distinguished atomic type s.

Exercise 3 (Regular Tree Languages)**.** Fix some ranked alphabet $\mathcal{F}$. We define the generic *tree signature* $\Sigma_{\mathcal{F}} = \langle\{\sigma\}, \mathcal{F}, \tau_{\mathcal{F}}\rangle$ by $\tau_{\mathcal{F}}(f^{(n)}) = \overbrace{\sigma \multimap \cdots \multimap \sigma}^{n \text{ times}} \multimap \sigma = \sigma^n \multimap \sigma$.

Let $\mathcal{G} = \langle\Sigma_1, \Sigma_{\mathcal{F}}, \mathcal{L}, s\rangle$ be an ACG with Σ_1 a second-order linear signature and s an atomic type of A_1. We define the *tree language* of $\mathcal{G}$ as

$$\mathscr{T}(\mathcal{G}) = \{\bar{t} \in T(\mathcal{F}) \mid \exists t \text{ ground}.\exists u \text{ ground}. \vdash_{\Sigma_1} u : s \wedge \mathcal{L}(u) \to_\beta^* t\}\ .$$

[1] 1. Give an ACG $\mathcal{G}$ s.t. $\mathscr{T}(\mathcal{G}) = \{f(g(a), g(b))\}$.

[1] 2. Assume that $\max_{a \in A_1} \mathsf{ord}(a) = 1$. Justify why $\mathscr{T}(\mathcal{G})$ is a regular tree language. Hint: Recall that linear tree homomorphisms preserve regularity.

Exercise 4 (Tree Adjoining Languages)**.**

[1] 1. Give a TAG $\mathcal{G}$ with word language $L(\mathcal{G}) = \{a^n b^m c^n d^m \mid n, m \geq 0\}$.

[2] 2. Give an ACG $\mathcal{G}'$ that generates the same *trees* as your answer to the previous question: $\mathscr{T}(\mathcal{G}') = L_T(\mathcal{G})$.

2 ACGs for Semantics

Exercise 5 (Covert Movements and Spurious Ambiguities)**.** Consider again the signature of Exercice 2.1, to which we add a constant QR, i.e., $\Sigma_0 = \langle A_0, C_0, \tau_0\rangle$ where:

$$A_0 = \{np, s, c\} \qquad C_0 = \{\text{ALICE}, \text{BELIEVE}, \text{LEFT}, \text{SOMEONE}, \text{THAT}, \text{QR}\}$$

$$\begin{aligned}
\tau_0(\text{ALICE}) &= np & \tau_0(\text{BELIEVE}) &= c \multimap np \multimap s\\
\tau_0(\text{LEFT}) &= np \multimap s & \tau_0(\text{SOMEONE}) &= np\\
\tau_0(\text{THAT}) &= s \multimap c & \tau_0(\text{QR}) &= np \multimap (np \multimap s) \multimap s
\end{aligned}$$

Consider the signatures $\Sigma_1 = \langle A_1, C_1, \tau_1 \rangle$ and $\Sigma_2 = \langle A_2, C_2, \tau_2 \rangle$, which are respectively defined as follows:

$$A_1 = \{\sigma\} \qquad C_1 = \{/\mathrm{Alice}/, /\mathrm{believes}/, /\mathrm{left}/, /\mathrm{someone}/, /\mathrm{that}/\}$$

$$\begin{array}{rlrl} \tau_1(/\mathrm{Alice}/) &= \sigma \multimap \sigma & \tau_1(/\mathrm{believes}/) &= \sigma \multimap \sigma \\ \tau_1(/\mathrm{left}/) &= \sigma \multimap \sigma & \tau_1(/\mathrm{someone}/) &= \sigma \multimap \sigma \\ \tau_1(/\mathrm{that}/) &= \sigma \multimap \sigma & & \end{array}$$

$$A_2 = \{\iota, o\} \qquad C_2 = \{\mathbf{a}, \mathbf{left}, \mathsf{B}, \exists\}$$

$$\begin{array}{rlrl} \tau_2(\mathbf{a}) &= \iota & \tau_2(\mathbf{left}) &= \iota \multimap o \\ \tau_2(\mathsf{B}) &= \iota \multimap o \multimap o & \tau_2(\exists) &= (\iota \multimap o) \multimap o \end{array}$$

Finally, define two linear higher-order homomorphisms $\mathcal{L}_1$ and $\mathcal{L}_2$ as follows:

$$\mathcal{L}_1(np) = \sigma \multimap \sigma \qquad \mathcal{L}_1(s) = \sigma \multimap \sigma \qquad \mathcal{L}_1(c) = \sigma \multimap \sigma$$

$$\begin{array}{rlrl} \mathcal{L}_1(\textsc{alice}) &= /\mathrm{Alice}/ & \mathcal{L}_1(\textsc{believe}) &= \lambda xy.\, y + /\mathrm{believes}/ + x \\ \mathcal{L}_1(\textsc{left}) &= \lambda x.\, x + /\mathrm{left}/ & \mathcal{L}_1(\textsc{someone}) &= /\mathrm{someone}/ \\ \mathcal{L}_1(\textsc{that}) &= \lambda x.\, /\mathrm{that}/ + x & \mathcal{L}_1(\textsc{qr}) &= \lambda xp.\, p\, x \end{array}$$

where $a + b$ is defined as $\lambda x.\, a\,(b\,x)$,

$$\mathcal{L}_2(np) = (\iota \multimap o) \multimap o \qquad \mathcal{L}_2(s) = o \qquad \mathcal{L}_2(c) = o$$

$$\begin{array}{rlrl} \mathcal{L}_2(\textsc{alice}) &= \lambda k.\, k\, \mathbf{a} & \mathcal{L}_2(\textsc{believe}) &= \lambda pk.\, k\,(\lambda x.\, \mathsf{B}\, x\, p) \\ \mathcal{L}_2(\textsc{left}) &= \lambda k.\, k\,(\lambda x.\, \mathbf{left}\, x) & \mathcal{L}_2(\textsc{someone}) &= \lambda k.\, \exists x.\, k\, x \\ \mathcal{L}_2(\textsc{that}) &= \lambda x.\, x & \mathcal{L}_2(\textsc{qr}) &= \dots \end{array}$$

[1] 1. Show that the two following terms

$$\begin{aligned} t_1 &= \textsc{believe}\,(\textsc{that}\,(\textsc{left}\,\textsc{someone}))\,\textsc{alice} \\ t_2 &= \textsc{qr}\,\textsc{someone}\,(\lambda x.\, \textsc{believe}\,(\textsc{that}\,(\textsc{left}\, x))\,\textsc{alice}) \end{aligned}$$

get the same interpretation under $\mathcal{L}_1$.

[1] 2. Compute $\mathcal{L}_2(t_1)$.

[2] 3. Define $\mathcal{L}_2(\textsc{qr})$ in such a way that $\mathcal{L}_2(t_2)$ yields the *de re* interpretation (i.e., the interpretation where the existential quantifier takes wide scope over the modal operator).

[3] 4. Show that there is an infinity of terms $u_0, u_1, u_2, \dots$ such that:

$$\mathcal{L}_1(u_i) = /\mathrm{Alice}/ + /\mathrm{believes}/ + /\mathrm{that}/ + /\mathrm{someone}/ + /\mathrm{left}/$$