Duration: 3 hours 
Paper documents are allowed. The numbers in front of questions are indicative of hardness or duration.

1 Two-level Syntax

Exercise 1 (Derivation trees). In a tree adjoining grammar \( G = \langle N, \Sigma, T_\alpha, T_\beta, S \rangle \), the trees in \( L_T(G) \) are called derived trees. We are interested here in another tree structure, called a derivation tree, for which we propose a formalisation here. Let us assume for simplicity that all the foot nodes of auxiliary trees have the "null" null adjunction annotation.

For an elementary tree \( \gamma \in T_\alpha \uplus T_\beta \), we define its contents \( c(\gamma) \) to be a finite sequence \( Q \) of labels in \( Q \). Formally, we enumerate for this the labels in \( Q \) of its nodes in position order; the nodes labelled by \( \Sigma \cup N \) are ignored.

Consider for instance the TAG \( G_1 \) with \( N \) def = \{S, NP, VP\}, \( \Sigma \) def = \{VBZ, NNP, NNS, RB\}, \( T_\alpha \) def = \{likes, Bill, mushrooms\}, \( T_\beta \) def = \{possibly\}, and \( S \) def = S, where the elementary trees are shown below:

\[
\begin{array}{c}
S \\
\downarrow NP \\
\downarrow VP \\
\end{array} \quad 
\begin{array}{c}
\downarrow NNP \circ \\
\downarrow NNS \circ \\
\end{array} \quad 
\begin{array}{c}
\downarrow RB \circ \\
\downarrow VP \ast \\
\end{array}
\]

Then \( \text{likes} \) has contents \( c(\text{likes}) = q_S, q_{NP}, q_{VP}, q_{NP}, q_{VP} \), \( c(\text{Bill}) = q_{NP}, c(\text{mushrooms}) = q_{NP} \), and \( c(\text{possibly}) = q_{VP} \).

We now define a finite ranked alphabet \( F \) def = \( T_\alpha \uplus T_\beta \uplus \{\varepsilon(0)\} \). For an elementary tree \( \gamma \in T_\alpha \uplus T_\beta \), its rank is \( r(\gamma) = |c(\gamma)| \) the length of its contents. For the symbol \( \varepsilon \), its rank is \( r(\varepsilon) = 0 \). For a TAG \( G = \langle N, \Sigma, T_\alpha, T_\beta, S \rangle \), we construct a finite tree automaton \( A_G \equiv (Q, F, \delta, Q_{S_1}) \) where \( Q \) and \( F \) are defined as above and

\[
\delta \equiv \{(q_A, \alpha^{(r(\alpha))}, c(\alpha)) \mid A \downarrow \alpha \in T_\alpha, r(\alpha) = A \}
\cup \{(q_A, \beta^{(r(\beta))}, c(\beta)) \mid A \in N, \beta \in T_\beta, r(\beta) = A \}
\cup \{(q_A, \varepsilon(0)) \mid A \in N \}
\]

where 'rl' returns the root label of the tree.
[1] 1. Give the finite automaton $A_{g_1}$ associated with the example TAG $G_1$.

[1] 2. Modify your automaton in order to also handle the trees $\text{someone} \in T_\alpha$ and $\text{real}, \text{fake}, \text{thinks} \in T_\beta$ shown below, where $PN \circ, JJ \circ, VB \circ \in \Sigma$:

\[
\begin{align*}
&\text{NP} \quad \text{NP} \quad \text{NP} \quad \text{S} \\
&\text{PN} \quad \text{JJ} \quad \text{NP}^\ast \quad \text{JJ} \quad \text{NP}^\ast \quad \text{NP} \quad \text{VP} \\
&(\text{someone}) \quad (\text{real}) \quad (\text{fake}) \quad (\text{thinks})
\end{align*}
\]

[1] 3. The intention that our finite automaton generates the \textit{derivation} language $L_D(G) \overset{\text{def}}{=} L(A_G)$ of $G$. Can you figure out what should be the derivation tree of ‘Someone possibly thinks Bill likes mushrooms’?

[2] 4. Give a PDL node formula $\varphi_1$ such that $L(A_{g_1}) = \{ t \in T(F) \mid t, \text{root} \models \varphi_1 \}$.

### 1.1 Macro Tree Transducers

Let $\mathcal{X}$ be a countable set of variables and $\mathcal{Y}$ a countable set of parameters; we assume $\mathcal{X}$ and $\mathcal{Y}$ to be disjoint. For $Q$ a ranked alphabet with arities greater than zero, we abuse notations and write $Q(\mathcal{X})$ for the alphabet of pairs $(q, x) \in Q \times \mathcal{X}$ with $\text{arity}(q, x) \overset{\text{def}}{=} \text{arity}(q) - 1$. This is just for convenience, and $(q, x)(t_1, \ldots, t_n)$ is really the term $q(x, t_1, \ldots, t_n)$.

**Syntax.** A \textit{macro tree transducer} (NMTT) is a tuple $\mathcal{M} = (Q, F, F', \Delta, I)$ where $Q$ is a finite set of states, all of arity $\geq 1$, $F$ and $F'$ are finite ranked alphabets, $I \subseteq Q_1$ is a set of root states of arity one, and $\Delta$ is a finite set of term rewriting rules of the form $q(f(x_1, \ldots, x_n), y_1, \ldots, y_p) \rightarrow e$ where $q \in Q_{1+p}$ for some $p \geq 0$, $f \in F_n$ for some $n \in \mathbb{N}$, and $e \in T(F' \cup Q(\mathcal{X}), \mathcal{Y}_{p})$. Note that this imposes that any occurrence in $e$ of a variable $x \in \mathcal{X}$ must be as the first argument of a state $q \in Q$.

**Inside-Out Semantics.** Given a NMTT, the \textit{inside-out} rewriting relation over trees in $T(F \cup F' \cup Q)$ is defined by: $t \overset{\text{IO}}{\rightarrow} t'$ if there exist a rule $q(f(x_1, \ldots, x_n), y_1, \ldots, y_p) \rightarrow e$ in $\Delta$, a context $C \in C(F \cup F' \cup Q)$, and two substitutions $\sigma: \mathcal{X} \rightarrow T(F)$ and $\rho: \mathcal{Y} \rightarrow T(F')$ such that $t = C[q(f(x_1, \ldots, x_n), y_1, \ldots, y_p)\sigma\rho]$ and $t' = C[e\sigma\rho]$. In other words, in inside-out rewriting, when applying a rewriting rule $q(f(x_1, \ldots, x_n), y_1, \ldots, y_p) \rightarrow e$, the parameters $y_1, \ldots, y_p$ must be mapped to trees in $T(F')$, with no remaining states from $Q$.

Similarly to context-free tree grammars, the \textit{inside-out} transduction $[\mathcal{M}]_{\text{IO}}$ realised by $\mathcal{M}$ is defined through inside-out rewriting semantics:

$$[\mathcal{M}]_{\text{IO}} \overset{\text{def}}{=} \{(t, t') \in T(F) \times T(F') \mid \exists q \in I . q(t) \overset{\text{IO}}{\rightarrow}^* t' \}.$$
Example 1. Let $F \overset{\text{def}}{=} \{a^{(1)}, s^{(0)}\}$ and $F' \overset{\text{def}}{=} \{f^{(3)}, a^{(1)}, b^{(1)}, s^{(0)}\}$. Consider the NMTT $M = (\{q^{(1)}, q'^{(3)}\}, F, F', \Delta, \{q\})$ with $\Delta$ the set of rules
\begin{align*}
q(a(x_1)) & \rightarrow q'(x_1, s, s) & q'(s, y_1, y_2) & \rightarrow f(y_1, y_1, y_2) \\
q'(a(x_1), y_1, y_2) & \rightarrow q'(x_1, a(y_1), a(y_2)) & q'(a(x_1), y_1, y_2) & \rightarrow q'(x_1, a(y_1), b(y_2)) \\
q'(a(x_1), y_1, y_2) & \rightarrow q'(x_1, b(y_1), a(y_2)) & q'(a(x_1), y_1, y_2) & \rightarrow q'(x_1, b(y_1), b(y_2))
\end{align*}

Then we have for instance the following derivation:
\begin{align*}
q(a(a(a($)))) & \overset{\mathcal{IO}}{\rightarrow} q'(a(a($)), s, s) \\
& \overset{\mathcal{IO}}{\rightarrow} q'(a($), b($), b($)) \\
& \overset{\mathcal{IO}}{\rightarrow} q'(s, a(b($)), b(b($))) \\
& \overset{\mathcal{IO}}{\rightarrow} f(a(b($)), a(b($)), b(b($)))
\end{align*}
showing that $(a(a(a($))), f(a(b($)), a(b($)), b(b($)))) \in [M]$.

Exercise 2 (Monadic trees). An NMTT $M$ is called linear and non-deleting if, in every rule $q(f(x_1, \ldots, x_n), y_1, \ldots, y_p) \rightarrow e$ in $\Delta$, the term $e$ is linear in $\{x_1, \ldots, x_n\}$ and $\{y_1, \ldots, y_p\}$, i.e. each variable and each parameter occurs exactly once in the term $e$.

Let $F' \overset{\text{def}}{=} \{a^{(1)}, b^{(1)}, s^{(0)}\}$. Observe that trees in $T(F')$ are in bijection with contexts in $C(F')$ and words over $\{a, b\}^*$. For a context $C$ from $C(F')$, we write $C^R$ for its mirror context, read from the leaf to the root. For instance, if $C = a(b(a(a(\square)))$, then $C^R = a(a(b(a(\square))))$. Formally, let $n \in \mathbb{N}$ be such that $\text{dom}(C) = \{0^m | m \leq n\}$; then $C(0^n) = \square$ and $C(0^m) \in \{a, b\}$ for $m < n$. Then $C^R$ is defined by $\text{dom}(C^R) = \text{dom}(C)$, $C^R(0^n) = \square$, and $C^R(0^m) = C^R(0^{n-m})$ for all $m < n$.

[2] 1. Give a linear and non-deleting NMTT $M$ from $F'$ to $F$ such that $[M]_{\mathcal{IO}} = \{(C[s], C[C^R[s]] | C \in C(F'))\}$. In terms of words over $\{a, b\}^*$, this transducer maps $w$ to the palindrome $ww^R$. Is $[M]_{\mathcal{IO}}(T(F))$ a recognisable tree language?

Exercise 3 (From derivation to derived trees). Consider again the tree adjoining grammar $G_1$ from Exercise 1.

[3] 1. Give a linear non-deleting NMTT $M_1$ that maps the derivation trees of $G_1$ to its derived trees. Formally, we want $\text{dom}(M_1)_{\mathcal{IO}} = L_D(G_1)$ and $[M_1]_{\mathcal{IO}}(T(F)) = L_T(G_1)$.

Exercise 4 (Context-free tree grammar). Let $M = (Q, F, F', \Delta, I)$ be an NMTT and $A = (Q', F, \delta, I')$ be an NFTA.

[5] 1. Show that $L = [M]_{\mathcal{IO}}(L(A)) = \{t' \in T(F') | \exists t \in L(A) . (t, t') \in [M]_{\mathcal{IO}}\}$ is an inside-out context-free tree language, i.e., show how to construct a CFTG $G = (N, F', S, R)$ such that $L_{\mathcal{IO}}(G) = L$. 
2 Scope ambiguities and propositional attitudes

Exercise 5. One considers the two following signatures:

\[
(\Sigma_{\text{ABS}}) \quad \begin{align*}
\text{SUZY} : & \ NP \\
\text{BILL} : & \ NP \\
\text{MUSHROOM} : & \ N \\
A : & \ N \rightarrow (NP \rightarrow S) \rightarrow S \\
A_{inf} : & \ N \rightarrow (NP \rightarrow S_{inf}) \rightarrow S_{inf} \\
\text{EAT} : & \ NP \rightarrow NP \rightarrow S_{inf} \\
\text{TO} : & \ (NP \rightarrow S_{inf}) \rightarrow VP \\
\text{WANT} : & \ VP \rightarrow NP \rightarrow S
\end{align*}
\]

\[
(\Sigma_{S\text{-FORM}}) \quad \begin{align*}
\text{Suzy} : & \ \text{string} \\
\text{Bill} : & \ \text{string} \\
\text{mushroom} : & \ \text{string} \\
\text{a} : & \ \text{string} \\
\text{eat} : & \ \text{string} \\
\text{to} : & \ \text{string} \\
\text{wants} : & \ \text{string}
\end{align*}
\]

where, as usual, \textit{string} is defined to be \(o \rightarrow o\) for some atomic type \(o\).

One then defines a morphism \((\mathcal{L}_{\text{SYNT}} : \Sigma_{\text{ABS}} \rightarrow \Sigma_{S\text{-FORM}})\) as follows:

\[
(\mathcal{L}_{\text{SYNT}}) \quad \begin{align*}
\text{NP} : & \ \text{string} \\
\text{N} : & \ \text{string} \\
\text{S} : & \ \text{string} \\
S_{inf} : & \ \text{string} \\
\text{VP} : & \ \text{string} \\
\text{SUZY} : & \text{Suzy} \\
\text{BILL} : & \text{Bill} \\
\text{MUSHROOM} : & \text{mushroom} \\
A : & \lambda xy. y (a + x) \\
A_{inf} : & \lambda xy. y (a + x) \\
\text{EAT} : & \lambda xy. y + \text{eat} + x \\
\text{TO} : & \lambda x. \text{to} + (x \epsilon) \\
\text{WANT} : & \lambda xy. y + \text{wants} + x
\end{align*}
\]

where, as usual, the concatenation operator (+) is defined as functional composition, and the empty word (\(\epsilon\)) as the identity function.

1. Give two different terms, say \(t_0\) and \(t_1\), such that:

\[
\mathcal{L}_{\text{SYNT}}(t_0) = \mathcal{L}_{\text{SYNT}}(t_1) = \text{Bill} + \text{wants} + \text{to} + \text{eat} + a + \text{mushroom}
\]
Exercise 6. One considers a third signature:

\[(\Sigma_{L\text{-FORM}})\]

- \(\text{suzy} : \text{ind}\)
- \(\text{bill} : \text{ind}\)
- \(\text{mushroom} : \text{ind} \to \text{prop}\)
- \(\text{eat} : \text{ind} \to \text{ind} \to \text{prop}\)
- \(\text{want} : \text{ind} \to \text{prop} \to \text{prop}\)

One then defines a morphism \((\mathcal{L}_{\text{SEM}} : \Sigma_{\text{ABS}} \to \Sigma_{L\text{-FORM}})\) as follows:

\[(\mathcal{L}_{\text{SEM}})\]

- \(\text{NP} := \text{ind}\)
- \(\text{N} := \text{ind} \to \text{prop}\)
- \(\text{S} := \text{prop}\)
- \(\text{S}_{\text{inf}} := \text{prop}\)
- \(\text{VP} := \text{ind} \to \text{prop}\)
- \(\text{SUZY} := \text{suzy}\)
- \(\text{BILL} := \text{bill}\)
- \(\text{MUSHROOM} := \text{mushroom}\)
- \(\text{A} := \lambda xy. \exists z. (xz) \land (yz)\)
- \(\text{A}_{\text{inf}} := \lambda xy. \exists z. (xz) \land (yz)\)
- \(\text{EAT} := \lambda xy. \text{eat} y x\)
- \(\text{TO} := \lambda x. x\)
- \(\text{WANT} := \lambda xy. \text{want} y (x y)\)

1. \([1]\) Compute the different semantic interpretations of the sentence Bill wants to eat a mushroom, i.e., compute \(\mathcal{L}_{\text{SEM}}(t_0)\) and \(\mathcal{L}_{\text{SEM}}(t_1)\).

Exercise 7. One extends \(\Sigma_{\text{ABS}}\) and \(\mathcal{L}_{\text{SYNT}}\), respectively, as follows:

\[(\Sigma_{\text{ABS}})\]

\(\text{WANT2} : \text{VP} \to \text{NP} \to \text{S}\)

\[(\mathcal{L}_{\text{SYNT}})\]

\(\text{WANT2} := \lambda xyz. z + \text{wants} + x + y\)

1. \([1]\) Extend \(\mathcal{L}_{\text{SEM}}\) accordingly in order to allow for the analysis of a sentence such as Bill wants Suzy to eat a mushroom.

Exercise 8. One extends \(\Sigma_{\text{ABS}}\) as follows:

\[(\Sigma_{\text{ABS}})\]

\(\text{EVERYONE} : (\text{NP} \to \text{S}) \to \text{S}\)

\(\text{THINK} : \text{S} \to \text{NP} \to \text{S}\)

in order to allow for the analysis of the following sentence:

\((1)\) everyone thinks Bill wants to eat a mushroom.
1. Extend $\Sigma_{\text{S-FORM}}$, $\mathcal{L}_{\text{SYNT}}$, $\Sigma_{\text{L-FORM}}$, and $\mathcal{L}_{\text{SEM}}$ accordingly.

2. Give the several $\lambda$-terms that correspond to the different parsings of sentence (1).