MPRI 2-27-1 Exam

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

1 Right Linear Monadic CFTGs

The motivation for this section is to understand *tree insertion grammars*, a restriction of tree adjoining grammars defined by Schabes and Waters in 1995. We shall work with the more convenient (and cleaner) framework of context-free tree grammars, and study the corresponding formalism of *single-sided* linear monadic context-free tree grammars (recall that tree adjoining grammars are roughly equivalent to linear monadic context-free tree grammars). To further simplify matters, we shall work with *right* grammars.

Definition 1 (Right Contexts)**.** We work with three disjoint ranked alphabets:

- N_0 is a *nullary nonterminal* alphabet consisting of symbols of rank 0,
- N_R is a *right nonterminal* alphabet consisting of symbols of rank 1, and
- $\mathcal{F}$ is a ranked *terminal* alphabet.

We use $A_0, B_0, \dots$ to denote elements of N_0, $A_R, B_R, \dots$ for elements of N_R, and $f^{(k)}, \dots$ for elements of $\mathcal{F}_k$ the sub-alphabet of $\mathcal{F}$ with symbols of rank k. Let us define $N \stackrel{\text{def}}{=} N_0 \uplus N_R$ and $V \stackrel{\text{def}}{=} N \uplus \mathcal{F}$; then $e, e_1, \dots$ denote trees in $T(V)$ and $t, t_1, \dots$ terminal trees in $T(\mathcal{F})$.

The set of **right contexts** $\mathcal{C}_R(V)$ is made of contexts C where $\Box$ is the rightmost leaf. In other words, $\Box$ is a right context in $\mathcal{C}_R(V)$, and if $X^{(k)}$ is a symbol of arity $k > 0$ in V, C is a right context in $\mathcal{C}_R(V)$, and $e_1, \dots, e_{k-1}$ are trees in $T(V)$ then $X^{(k)}(e_1, \dots, e_{k-1}, C)$ is also a right context in $\mathcal{C}_R(V)$.

Definition 2 (Right Linear Monadic CFTGs)**.** A **right linear monadic context-free tree grammar** is a tuple $\mathcal{G} = \langle N_0, N_R, \mathcal{F}, S_0, R\rangle$ where N_0, N_R, and $\mathcal{F}$ are as above, $S_0 \in N_0$ is the *axiom*, and R is a finite set of rules of form:

- $A_0 \to e$ with $A_0 \in N_0$ and $e \in T(V)$, or
- $A_R(y) \to C[y]$ with $A_R \in N_R$ and $C \in \mathcal{C}_R(V)$; y is called the *parameter* of the rule.

The *tree language* of $\mathcal{G}$ is

$$L(\mathcal{G}) \stackrel{\text{def}}{=} \{t \in T(\mathcal{F}) \mid S_0 \stackrel{R}{\Rightarrow}^{\star} t\}\ .$$

Exercise 1 (Yields and Branches)**.** Given a tree language $L \subseteq T(\mathcal{F})$, let $\text{Yield}(L) \stackrel{\text{def}}{=} \bigcup_{t\in L} \text{Yield}(t)$ and define inductively

$$\text{Yield}(a^{(0)}) \stackrel{\text{def}}{=} a \qquad\qquad \text{Yield}(f^{(k)}(t_1,\dots,t_k)) \stackrel{\text{def}}{=} \text{Yield}(t_1)\cdots\text{Yield}(t_k)\ .$$

Hence $\text{Yield}(t) \in \mathcal{F}_0^*$ is a word over $\mathcal{F}_0$, and $\text{Yield}(L) \subseteq \mathcal{F}_0^*$ is a word language over $\mathcal{F}_0$.

[1] 1. What is the word language $\text{Yield}(L(\mathcal{G}))$ of the CFTG with rules

$$\begin{aligned}
S_0 &\to A_R(c^{(0)})\\
A_R(y) &\to f^{(2)}\big(a^{(0)}, A_R(f^{(2)}(a^{(0)}, y))\big)\\
A_R(y) &\to f^{(2)}\big(b^{(0)}, A_R(f^{(2)}(b^{(0)}, y))\big)\\
A_R(y) &\to y
\end{aligned}$$

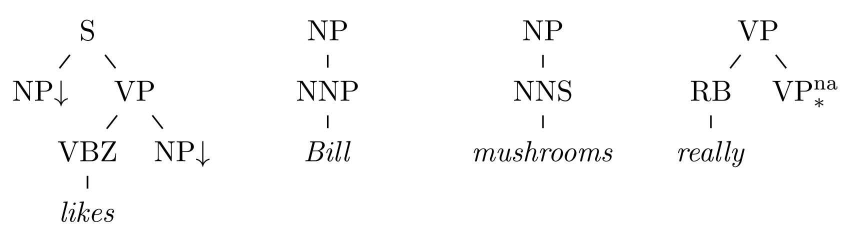
where $N_0 \stackrel{\text{def}}{=} \{S_0\}$, $N_R \stackrel{\text{def}}{=} \{A_R\}$, and $\mathcal{F} \stackrel{\text{def}}{=} \{f^{(2)}, a^{(0)}, b^{(0)}, c^{(0)}\}$?

[2] 2. Show that there exists a right linear monadic CFTG $\mathcal{G}$ such that $L(\mathcal{G})$ is not a regular tree language.

Hint: Recall that, if $L \subseteq T(\mathcal{F})$ is a regular tree language, then its set of branches $\text{Branches}(L)$ is a regular word language over $\mathcal{F}$. We define $\text{Branches}(L) \subseteq \mathcal{F}^*$ by $\text{Branches}(L) \stackrel{\text{def}}{=} \bigcup_{t\in L} \text{Branches}(t)$ and in turn

$$\text{Branches}(a^{(0)}) \stackrel{\text{def}}{=} \{a\} \qquad \text{Branches}(f^{(k)}(t_1,\dots,t_k)) \stackrel{\text{def}}{=} \bigcup_{1\leq j\leq k} \{f\}\cdot\text{Branches}(t_j)\ .$$

Exercise 2 (Tree Insertion Grammars)**.** Consider the tree adjoining grammar depicted below. Note that its sole auxiliary tree β_1 is of the form $C[\text{VP}_*^{\text{na}}]$ where C is a right context; this grammar is actually a *right* tree insertion grammar.



(α_1) (α_2) (α_3) (β_1)

[1] 1. Provide an equivalent right linear monadic CFTG.

[1] 2. Complete the TIG or your CFTG (in a linguistically informed manner) in order to also generate the sentence ‘Bill likes black mushrooms.’

Exercise 3 (Context-Free Word Languages)**.** We show in this exercise that, although right linear monadic CFTGs can generate non-regular tree languages, their expressive power is just as limited as that of finite tree automata when it comes to word languages.

[3] 1. Show for any context-free language L, there is a right linear monadic context-free tree grammar $\mathcal{G}'$ with $L \setminus \{\varepsilon\} = \mathrm{Yield}(L(\mathcal{G}'))$.

[1] 2. Let us extend $\mathrm{Yield}(\cdot)$ to terminal contexts $c \in \mathcal{C}(\mathcal{F}) \subseteq T(\mathcal{F} \uplus \{\Box\})$ by $\mathrm{Yield}(\Box) \stackrel{\text{def}}{=} \varepsilon$. Show that, for all terminal right contexts $c \in \mathcal{C}_R(\mathcal{F})$ and all $t \in \mathcal{C}_R(\mathcal{F}) \cup T(\mathcal{F})$,

$$\mathrm{Yield}(c[t]) = \mathrm{Yield}(c) \cdot \mathrm{Yield}(t) \ .$$

[5] 3. Show the converse: for any right linear monadic CFTG, $\mathrm{Yield}(L(\mathcal{G}))$ is a context-free word language over $\mathcal{F}_0$.

Hint: You might use the fact that $\mathcal{G}$ is linear to restrict your attention to IO derivations: by Theorem 5.9 and Proposition 5.13 of the lecture notes, $L(\mathcal{G}) = L_{\mathrm{IO}}(\mathcal{G})$.

[1] 4. Show that, the **word membership problem** for right linear monadic CFTG can be solved in polynomial time (this problem is, given $w \in \mathcal{F}_0^*$ and $\mathcal{G}$ a right linear monadic CFTG, whether $w \in \mathrm{Yield}(L(\mathcal{G}))$).

2 Scope ambiguities and covert moves in ACGs

Exercise 4. One considers the two following signatures:

$$(\Sigma_{\mathrm{ABS}}) \quad \begin{aligned} \textsc{trace} &: NP_{NP} \\ \textsc{move} &: NP_{NP} \to (NP \to S) \to S_{NP} \\ \textsc{man} &: N \\ \textsc{help} &: N \\ \textsc{every} &: N \to S_{NP} \to S \\ \textsc{some} &: N \to S_{NP} \to S \\ \textsc{needs} &: NP \to NP \to S \end{aligned}$$

$$(\Sigma_{\text{S-FORM}}) \quad \begin{aligned} /man/ &: string \\ /help/ &: string \\ /every/ &: string \\ /some/ &: string \\ /needs/ &: string \end{aligned}$$

where, as usual, *string* is defined to be $o \rightarrow o$ for some atomic type o.

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

$$(\mathcal{L}_{\text{SYNT}}) \qquad \begin{aligned} N &:= \textit{string} \\ NP &:= \textit{string} \\ S &:= \textit{string} \\ NP_{NP} &:= \textit{string} \rightarrow \textit{string} \\ S_{NP} &:= \textit{string} \rightarrow \textit{string} \\ \textsc{trace} &:= \lambda x.\, x \\ \textsc{move} &:= \lambda xyz.\, y\,(x\,z) \\ \textsc{man} &:= /\textit{man}/ \\ \textsc{help} &:= /\textit{help}/ \\ \textsc{every} &:= \lambda xy.\, y\,(/\textit{every}/ + x) \\ \textsc{some} &:= \lambda xy.\, y\,(/\textit{some}/ + x) \\ \textsc{needs} &:= \lambda xy.\, y + /\textit{needs}/ + x \end{aligned}$$

where, as usual, the concatenation operator (+) is defined as functional composition.

[1] 1. Give two different terms, say t_0 and t_1, such that:

$$\mathcal{L}_{\text{SYNT}}(t_0) = \mathcal{L}_{\text{SYNT}}(t_1) = /\textit{every}/ + /\textit{man}/ + /\textit{needs}/ + /\textit{some}/ + /\textit{help}/$$

Exercise 5. One considers a third signature :

$$(\Sigma_{\text{L-FORM}}) \qquad \begin{aligned} \mathbf{man} &: \mathsf{ind} \rightarrow \mathsf{prop} \\ \mathbf{help} &: \mathsf{ind} \rightarrow \mathsf{prop} \\ \mathbf{needs} &: \mathsf{ind} \rightarrow \mathsf{ind} \rightarrow \mathsf{prop} \end{aligned}$$

where the intended intuitive interpretation of the binary relation **needs** is that ($\mathbf{needs}\, a\, b$) means that b is needed by a.

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

$$
\begin{aligned}
(\mathcal{L}_{\mathrm{SEM}}) \qquad N &:= \mathsf{ind} \to \mathsf{prop} \\
NP &:= \cdots \\
S &:= \mathsf{prop} \\
NP_{NP} &:= \mathsf{ind} \to \mathsf{ind} \\
S_{NP} &:= \mathsf{ind} \to \mathsf{prop} \\
\textsc{trace} &:= \cdots \\
\textsc{move} &:= \cdots \\
\textsc{man} &:= \mathbf{man} \\
\textsc{help} &:= \mathbf{help} \\
\textsc{every} &:= \lambda xy.\, \forall z.\, (x\, z) \to (y\, z) \\
\textsc{some} &:= \lambda xy.\, \exists z.\, (x\, z) \wedge (y\, z) \\
\textsc{needs} &:= \cdots
\end{aligned}
$$

[2] 1. Complete the above semantic interpretation (i.e., provide interpretations for NP, TRACE, MOVE, and NEEDS) in such a way that $\mathcal{L}_{\mathrm{SEM}}(t_0)$ and $\mathcal{L}_{\mathrm{SEM}}(t_1)$ yield two different plausible semantic interpretations of the sentence *every man needs some help.*

Exercise 6. One extends Σ_{ABS}, $\Sigma_{\text{S-FORM}}$, $\mathcal{L}_{\mathrm{SYNT}}$, and $\mathcal{L}_{\mathrm{SEM}}$, respectively, as follows:

$$
\begin{array}{ll}
(\Sigma_{\mathrm{ABS}}) & \textsc{possibly} : \; S \to S \\
(\Sigma_{\text{S-FORM}}) & /possibly/ : string \\
(\mathcal{L}_{\mathrm{SYNT}}) & \textsc{possibly} := \lambda x.\, x + /possibly/ \\
(\mathcal{L}_{\mathrm{SEM}}) & \textsc{possibly} := \lambda x.\, \Diamond\, x
\end{array}
$$

[2] 1. How many terms u are there such that:

$$\mathcal{L}_{\mathrm{SYNT}}(u) = /every/ + /man/ + /needs/ + /some/ + /help/ + /possibly/$$

[2] 2. Give three such terms together with their semantic interpretations.