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 $\square$ is the rightmost leaf. In other words, $\square$ 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 $\mathrm{Yield}(L) \stackrel{\text{def}}{=} \bigcup_{t \in L} \mathrm{Yield}(t)$ and define inductively

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

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

[1] 1. What is the word language $\mathrm{Yield}(L(\mathcal{G}))$ 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 $\mathcal{F} \stackrel{\text{def}}{=} \{f^{(2)}, a^{(0)}, b^{(0)}, c^{(0)}\}$?

Solution: This is the language of even-length palindromes over $\{a, b\}$ suffixed with a c: $\mathrm{Yield}(L(\mathcal{G})) = \{ww^R c \mid w \in \{a, b\}^*\}$ where $\cdot^R$ denotes the mirror operation on words.

[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 $\mathrm{Branches}(L)$ is a regular word language over $\mathcal{F}$. We define $\mathrm{Branches}(L) \subseteq \mathcal{F}^*$ by $\mathrm{Branches}(L) \stackrel{\text{def}}{=} \bigcup_{t \in L} \mathrm{Branches}(t)$ and in turn

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

Solution: Consider the right linear monadic CFTG with rules

$$\begin{aligned} S_0 &\to A_R(c^{(0)}) \\ A_R(y) &\to a^{(1)}\big(A_R(a^{(1)}(y))\big) \\ A_R(y) &\to b^{(1)}\big(A_R(b^{(1)}(y))\big) \\ A_R(y) &\to y \end{aligned}$$

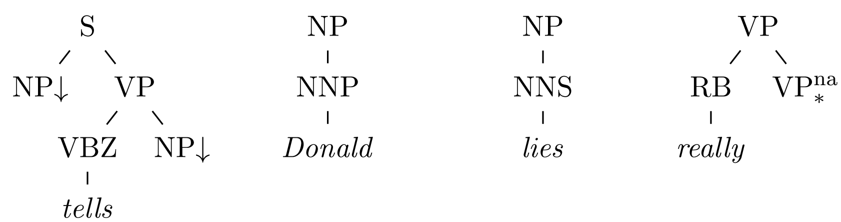
where $N_0 \stackrel{\text{def}}{=} \{S_0\}$, $N_R \stackrel{\text{def}}{=} \{A_R\}$, and $\mathcal{F} \stackrel{\text{def}}{=} \{a^{(1)}, b^{(1)}, c^{(0)}\}$.

Its yield language $\{c\}$ is uninteresting, but

$$\mathrm{Branches}(L(\mathcal{G})) = \{ww^R c \mid w \in \{a,b\}^*\}$$

is not a regular word language, and thus $L(\mathcal{G})$ is not a regular tree language. This could be generalised to arbitrary context-free word languages (assuming $\varepsilon^{(0)}$ belongs to $\mathcal{F}$).

Exercise 2 (Tree Insertion Grammars)**.** Consider the tree adjoining grammar depicted below. Note that its sole auxiliary tree β_1 is of the form $C[\mathrm{VP}_*^{\mathrm{na}}]$ 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.

Solution: It suffices to apply the translation from TAGs to linear monadic CFTG from Section 5.1.3 of the lecture notes:

$$\begin{aligned}
S{\downarrow} &\to \mathrm{S}^{(2)}\big(\mathit{NP}{\downarrow}, \overline{\mathit{VP}}(\mathrm{VP}^{(2)}(\mathrm{VBZ}^{(1)}(\mathit{tells}^{(0)}), \mathit{NP}{\downarrow}))\big)\\
\mathit{NP}{\downarrow} &\to \mathrm{NP}^{(1)}(\mathrm{NNP}^{(1)}(\mathit{Donald}^{(0)}))\\
\mathit{NP}{\downarrow} &\to \mathrm{NP}^{(1)}(\mathrm{NNS}^{(1)}(\mathit{lies}^{(0)}))\\
\overline{\mathit{VP}}(y) &\to \overline{\mathit{VP}}(\mathrm{VP}^{(2)}(\mathrm{RB}^{(1)}(\mathit{really}^{(0)}), y))\\
\overline{\mathit{VP}}(y) &\to y\ ,
\end{aligned}$$

with $N_0 \stackrel{\text{def}}{=} \{S{\downarrow}, \mathit{NP}{\downarrow}\}$, $N_R \stackrel{\text{def}}{=} \{\overline{\mathit{VP}}\}$, and $\mathcal{F} \stackrel{\text{def}}{=} \{\mathrm{S}^{(2)}, \mathrm{VP}^{(2)}, \mathrm{VBZ}^{(1)}, \mathit{tells}^{(0)}, \mathrm{NP}^{(1)}, \mathrm{NNP}^{(1)}, \mathit{Donald}^{(0)}, \mathrm{NNS}^{(1)}, \mathit{lies}^{(0)}, \mathrm{RB}^{(1)}, \mathit{really}^{(0)}\}$.

Of course, the language of the TAG is regular, so other solutions are possible—but somewhat less elegant. For instance,

$$\begin{aligned} q_S &\to \mathrm{S}^{(2)}(q_{NP}, q_{VP}) \\ q_{VP} &\to \mathrm{VP}^{(2)}(\mathrm{RB}^{(1)}(\mathit{really}^{(0)}), q_{VP}) \\ q_{VP} &\to \mathrm{VP}^{(2)}(\mathrm{VBZ}^{(1)}(\mathit{tells}^{(0)}), q_{NP}) \\ q_{NP} &\to \mathrm{NP}^{(1)}(\mathrm{NNP}^{(1)}(\mathit{Donald}^{(0)})) \\ q_{NP} &\to \mathrm{NP}^{(1)}(\mathrm{NNS}^{(1)}(\mathit{lies}^{(0)})) \end{aligned}$$

with $N_0 \stackrel{\text{def}}{=} \{q_S, q_{NP}, q_{VP}\}$ and $N_R \stackrel{\text{def}}{=} \emptyset$.

[1] 2. Complete the TIG or your CFTG (in a linguistically informed manner) in order to also generate the sentence ‘Donald tells the best lies.’

Solution: It’s quicker to modify the right TIG with an additional auxiliary tree $\beta_2 \stackrel{\text{def}}{=} \mathrm{NP}^{(3)\mathrm{na}}(\mathrm{DT}^{(1)}(\mathit{the}^{(0)}), \mathrm{JJS}^{(1)}(\mathit{best}^{(0)}), \mathrm{NP}_*^{\mathrm{na}})$; it makes sense to force the presence of ‘the’ before a superlative, though it does not capture e.g. ‘his best efforts’. Adding null adjunction annotations forbids to stack superlatives (one would expect a coordination for this, as in ‘the best and cleverest lies’).

Modifying the CFTG involves introducing new right nonterminals $\overline{NP}$ in several places.

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 $L \setminus \{\varepsilon\} = \mathrm{Yield}(L(\mathcal{G}'))$.

Solution: This can be argued from well-known theorems: if L is context-free, then $L \setminus \{\varepsilon\}$ is the yield $\mathrm{Yield}(L(\mathcal{A}))$ of some finite tree automaton $\mathcal{A}$ (c.f. Definition 3.6 of the lecture notes, where ε is also handled by having $\varepsilon^{(0)}$ in $\mathcal{F}$), which in turn is a right linear monadic CFTG with $N_0 \stackrel{\text{def}}{=} Q$, $N_R \stackrel{\text{def}}{=} \emptyset$ and the same set of rules. Alternatively, we can re-prove it from scratch:

Without loss of generality, we can assume we are given a CFG $\mathcal{G} = \langle N, \Sigma, P, S\rangle$ in Chomsky normal form with $L \setminus \{\varepsilon\} = L(\mathcal{G})$: the productions in P are of the form $A \to BD$ or $A \to a$ with $A, B, D \in N$ and $a \in \Sigma$. We define the CFTG

$\mathcal{G}' = \langle N, \emptyset, \mathcal{F}, S, R\rangle$ with $\mathcal{F} \stackrel{\text{def}}{=} \Sigma \uplus \{f^{(2)}\}$ where the symbols in Σ are nullary, and the set of rules

$$R \stackrel{\text{def}}{=} \{A \to f^{(2)}(B, D) \mid A \to BD \in P\}$$
$$\cup\ \{A \to a^{(0)} \mid A \to a \in P\}\ .$$

Let us show that $L(\mathcal{G}) \subseteq \text{Yield}(L(\mathcal{G}'))$: we prove by induction over n that, for all $A \in N$ and $w \in \Sigma^*$, if $A \Rightarrow^\star w$ in $\mathcal{G}$, then there exists $t \in T(\mathcal{F})$ such that $A \stackrel{R}{\Rightarrow}^\star t$ in $\mathcal{G}'$ and $\text{Yield}(t) = w$. This will show that, for any $w \in L(\mathcal{G})$, there exists $t \in L(\mathcal{G}')$ with $\text{Yield}(t) = w$.

base case for $n = 1$: then $A \Rightarrow a = w \in \Sigma$, and $t = a^0$ fits;

induction step for $n > 1$: then we have a derivation $A \Rightarrow BD \Rightarrow^{n-1} w$ for a production $A \to BD \in P$. Thus $B \Rightarrow^{n_1} w_1$ and $D \Rightarrow^{n_2} w_2$ with $n_1 + n_2 = n - 1$ and $w_1 w_2 = w$. By induction hypothesis on $n_1, n_2 < n$, there exist $t_1, t_2 \in T(\mathcal{F})$ such that $B \stackrel{R}{\Rightarrow}^\star t_1$, $D \stackrel{R}{\Rightarrow}^\star t_2$, $\text{Yield}(t_1) = w_1$, and $\text{Yield}(t_2) = w_2$. Therefore, $t \stackrel{\text{def}}{=} f^{(2)}(t_1, t_2)$ fits since $A \stackrel{R}{\Rightarrow} f^{(2)}(B, D) \stackrel{R}{\Rightarrow}^\star f^{(2)}(t_1, D) \stackrel{R}{\Rightarrow}^\star f^{(2)}(t_1, t_2) = t$ and $\text{Yield}(t) = \text{Yield}(t_1) \cdot \text{Yield}(t_2) = w_1 w_2 = w$.

Conversely, let us show that $L(\mathcal{G}) \supseteq \text{Yield}(L(\mathcal{G}'))$: we prove by induction over n that, for all $A \in N$ and $t \in T(\mathcal{F})$, if $A \stackrel{R}{\Rightarrow}^n t$ in $\mathcal{G}'$, then $A \Rightarrow \text{Yield}(t)$ in $\mathcal{G}$. This will show that, for any $t \in L(\mathcal{G}')$, $\text{Yield}(t) \in L(\mathcal{G})$.

base case for $n = 1$: then $A \stackrel{R}{\Rightarrow} a^{(0)} = t$, and $A \Rightarrow a$ holds in $\mathcal{G}$.

induction step for $n > 1$: then $A \stackrel{R}{\Rightarrow} f^{(2)}(B, D) \stackrel{R}{\Rightarrow}^{n-1} t$ for a production $A \to BD \in P$. Thus $t = f^{(2)}(t_1, t_2)$ such that $B \stackrel{R}{\Rightarrow}^{n_1} t_1$, $D \stackrel{R}{\Rightarrow}^{n_2} t_2$, and $n_1 + n_2 = n - 1$. By induction hypothesis, $B \Rightarrow^\star \text{Yield}(t_1)$ and $D \Rightarrow^\star \text{Yield}(t_2)$ in $\mathcal{G}$. Hence $A \Rightarrow BD \Rightarrow^\star \text{Yield}(t_1)\text{Yield}(t_2) = \text{Yield}(t)$.

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

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

Solution: We proceed by induction over terminal right contexts:

for the base case $c = \square$: then $c[t] = t$ and thus $\text{Yield}(c[t]) = \text{Yield}(t) = \text{Yield}(c)\text{Yield}(t)$;

for the induction step $c = f^{(k)}(t_1, \ldots, t_{k-1}, c')$ for some $k > 0$, $f^{(k)} \in \mathcal{F}_k$, $c' \in \mathcal{C}_R(\mathcal{F})$, and $t_1, \ldots, t_{k-1} \in T(\mathcal{F})$: by induction hypothesis, for all $t \in \mathcal{C}_R(\mathcal{F}) \cup T(\mathcal{F})$, $\mathrm{Yield}(c'[t]) = \mathrm{Yield}(c')\mathrm{Yield}(t)$. Thus for all $t \in \mathcal{C}_R(\mathcal{F}) \cup T(\mathcal{F})$,

$$\begin{aligned}\mathrm{Yield}(c[t]) &= \mathrm{Yield}(f^{(k)}(t_1, \ldots, t_{k-1}, c'[t]))\\ &= \mathrm{Yield}(t_1) \cdots \mathrm{Yield}(t_{k-1})\mathrm{Yield}(c'[t])\\ &= \mathrm{Yield}(t_1) \cdots \mathrm{Yield}(t_{k-1})\mathrm{Yield}(c')\mathrm{Yield}(t)\\ &= \mathrm{Yield}(c) \cdot \mathrm{Yield}(t)\ .\end{aligned}$$

[6] 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, $L(\mathcal{G}) = L_{\mathrm{IO}}(\mathcal{G})$.

Solution: Let $\mathcal{G} = \langle N_0, N_R, \mathcal{F}, S_0, R\rangle$ be a right linear monadic CFTG. We let E denote the set of subtrees and subcontexts appearing inside right-hand-sides of rules in R: formally,

$$E \stackrel{\text{def}}{=} \mathrm{Sub}(\{e \in T(V) \mid A_0 \to e \in R\} \cup \{C \in \mathcal{C}_R(V) \mid A_R(y) \to C[y] \in R\})$$

where for any $S \subseteq \mathcal{C}_R(V) \cup T(V)$

$$\mathrm{Sub}(S) \stackrel{\text{def}}{=} \{e \in \mathcal{C}_R(V) \cup T(V) \mid \exists C \in \mathcal{C}_R(V).C[e] \in S\}\ .$$

We define $\mathcal{G}' \stackrel{\text{def}}{=} \langle N', \mathcal{F}_0, [S_0], P\rangle$ a word context-free grammar with nonterminals $N' \stackrel{\text{def}}{=} \{[e] \mid e \in E\} \cup \{[S_0]\}$ and with productions:

$$\begin{aligned}P \stackrel{\text{def}}{=}\ & \{[a^{(0)}] \to a \mid a^{(0)} \in \mathcal{F}_0 \cap E\}\\ \cup\ & \{[\Box] \to \varepsilon\}\\ \cup\ & \{[f^{(k)}(e_1, \ldots, e_k)] \to [e_1] \cdots [e_k] \mid k > 0, f^{(k)}(e_1, \ldots, e_k) \in E, e_1 \in T(V) \cup \mathcal{C}_R(V),\\ & \qquad e_2, \ldots, e_k \in T(V)\}\\ \cup\ & \{[A_0] \to [e] \mid A_0 \to e \in R\}\\ \cup\ & \{[A_R(e)] \to [C][e] \mid A_R(e) \in E, A_R(y) \to C[y] \in R, e \in T(V) \cup \mathcal{C}_R(V)\}\ .\end{aligned}$$

Let us show that $\mathrm{Yield}(L(\mathcal{G})) \supseteq L(\mathcal{G}')$. We prove for this by induction over n that, for all $e \in \mathcal{C}_R(\mathcal{F}) \cap E$ (resp. $e \in T(V) \cap E$ or $e = S_0$), if $[e] \Rightarrow^n w$ in $\mathcal{G}'$, then there exists $t \in \mathcal{C}_R(\mathcal{F})$ (resp. $t \in T(\mathcal{F})$) such that $e \overset{R}{\Rightarrow}{}^{\star}\ t$ and $\mathrm{Yield}(t) = w$. Then, by setting $e = S_0$, the statement follows.

base case $n = 1$ for $e = a^{(0)}$: then $w = a$, and $t \stackrel{\text{def}}{=} a^{(0)}$ fits.

base case $n = 1$ for $e = \Box$: then $w = \varepsilon$, and $c' \stackrel{\text{def}}{=} \Box$ fits.

induction step $n > 0$ for $e = f^{(k)}(e_1, \dots, e_k)$: if $[e] = [f^{(k)}(e_1, \dots, e_k)] \Rightarrow [e_1] \cdots [e_k] \Rightarrow^{n-1} w$, then for all $1 \leq j \leq k$, $[e_j] \Rightarrow^{n_j} w_j$ with $n_1 + \cdots + n_k = n-1$ and $w_1 \cdots w_k = w$. By induction hypothesis on $n_j < n$, there exists $t_j \in \mathcal{C}_R(\mathcal{F}) \cup T(\mathcal{F})$ with $e_j \stackrel{R}{\Rightarrow}^{\star} t_j$ for each $1 \leq j \leq k$. Therefore, $t \stackrel{\text{def}}{=} f^{(k)}(t_1, \dots, t_k)$ fits.

induction step $n > 0$ for $e = A_0$: then $[e] = [A_0] \Rightarrow [e'] \Rightarrow^{n-1} w$ for some $A_0 \to e$ in R. By induction hypothesis, there exists $t' \in \mathcal{C}_R(\mathcal{F}) \cup T(\mathcal{F})$ with $e' \stackrel{R}{\Rightarrow}^{\star} t'$ and $\text{Yield}(t') = w$, hence $t \stackrel{\text{def}}{=} t'$ fits.

induction step $n > 0$ for $e = A_R(e')$: if $[e] = [A_R(e')] \Rightarrow [C][e'] \Rightarrow^{n-1} w$ for some $A_R(y) \to C[y] \in R$, then $[C] \Rightarrow^{n_1} w_1$ and $[e'] \Rightarrow^{n_2} w_2$ for some $n_1 + n_2 = n - 1$ and $w_1 w_2 = w$. By induction hypothesis, there exist $c_1 \in \mathcal{C}_R(\mathcal{F})$ and $t_2 \in \mathcal{C}_R(\mathcal{F}) \cup T(\mathcal{F})$ such that $C \stackrel{R}{\Rightarrow}^{\star} c_1$, $\text{Yield}(c_1) = w_1$, $e' \stackrel{R}{\Rightarrow}^{\star} t_2$, and $\text{Yield}(t_2) = w_2$. Thus letting $t \stackrel{\text{def}}{=} c_1[t_2]$ fits: $A_0(e') \stackrel{R}{\Rightarrow} C[e'] \stackrel{R}{\Rightarrow}^{\star} C[t_2] \stackrel{R}{\Rightarrow}^{\star} c_1[t_2]$ and $\text{Yield}(c_1[t_2]) = \text{Yield}(c_1)\text{Yield}(t_2) = w_1 w_2 = w$ by Question 2 above.

Conversely, let us show that $\text{Yield}(L(\mathcal{G})) \subseteq L(\mathcal{G}')$. We prove for this by induction over $(e, n) \in (E \cup S_0) \times \mathbb{N}$ ordered lexicographically (with n being most significant) that, if $e \in \mathcal{C}_R(V) \cap E$ (resp. $T(V) \cap E$ or $e = S_0$) and for all $t \in \mathcal{C}_R(\mathcal{F})$ (resp. $T(\mathcal{F})$), if $e \stackrel{R}{\Rightarrow}^{n} t$ using IO derivations in $\mathcal{G}$, then $[e] \Rightarrow^{\star} \text{Yield}(t)$ in $\mathcal{G}'$.

case $e = a^{(0)}$ and $n = 0$: then $[e] = [a^{(0)}] \Rightarrow a = \text{Yield}(e)$ in $\mathcal{G}'$.

case $e = \Box$ and $n = 0$: then $[e] = [\Box] \Rightarrow \varepsilon = \text{Yield}(e)$ in $\mathcal{G}'$.

case $e = f^{(k)}(e_1, \dots, e_k)$ and $n \geq 0$: then $e \stackrel{R}{\Rightarrow}^{n} t$ using IO derivations implies $e_j \stackrel{R}{\Rightarrow}^{n_j} t_j$ for $1 \leq j \leq k$ with $n = n_1 + \cdots + n_k$ and $t = f^{(k)}(t_1, \dots, t_j)$. Using the induction hypothesis on (e_j, n_j) shows $[e_j] \Rightarrow^{\star} \text{Yield}(t_j)$ in $\mathcal{G}'$, hence $[e] \Rightarrow [e_1] \cdots [e_k] \Rightarrow^{\star} \text{Yield}(t_1) \cdots \text{Yield}(t_k) = \text{Yield}(t)$.

case $e = A_0$ and $n > 0$: then $e = A_0 \stackrel{R}{\Rightarrow} e' \stackrel{R}{\Rightarrow}^{n-1} t$ using rule $A_0 \to e'$ in R. As $e' \in E$, we can apply the induction hypothesis on $(e', n-1)$ to show $[e'] \Rightarrow^{*} \text{Yield}(t)$ in $\mathcal{G}'$, and using the production $[A_0] \to [e']$ we get $[e] = [A_0] \Rightarrow^{*} \text{Yield}(t)$.

case $e = A_R(e')$ and $n > 0$: then $e = A_R(e') \stackrel{R}{\Rightarrow}^{n_1} A_R(c_1) \stackrel{R}{\Rightarrow} C[c_1] \stackrel{R}{\Rightarrow}^{n_2} c_2[c_1] = t$ since we are using IO derivations, with $n_1 + n_2 = n - 1$ and $A_R(y) \to C[y] \in R$. As $e' \in E$ and $e' \stackrel{R}{\Rightarrow}^{n_1} c_1$, by induction hypothesis on (e', n_1), $[e'] \Rightarrow^{*} \text{Yield}(c_1)$

in $\mathcal{G}'$. Similarly, $C \in E$ and $C \overset{R}{\Rightarrow}^{n_2} c_2$, and by induction hypothesis on (C, n_2), $[C] \Rightarrow^\star \text{Yield}(c_2)$ in $\mathcal{G}'$. Finally, $[A_R(e')] \to [C][e']$ is a production of P, hence $[e] = [A_R(e')] \Rightarrow [C][e'] \Rightarrow^\star \text{Yield}(c_2)\text{Yield}(c_1) = \text{Yield}(c_2[c_1]) = \text{Yield}(t)$ by Question 2 above.

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

Solution: It suffices to observe that the previous construction results in a CFG $\mathcal{G}'$ of quadratic size in $|\mathcal{G}|$, on which we can apply the $O(|\mathcal{G}'| \cdot |w|^3)$ algorithm seen in class (c.f. Lemma 3.8 in the lecture notes, where the word automaton for $\{w\}$ has $|Q| = |w| + 1$ states). Beware that one could imagine that Question 3 holds but that the CFGs we obtain are not of polynomial size (or even not constructible at all!), so it's not enough to just assume Question 3.

The quadratic blow-up in the construction of $\mathcal{G}'$ can be avoided by the usual trick: add N_R to N' and split the productions $[A_R(e)] \to [C][e]$ into $[A_R(e)] \to A_R\,[e]$ and $A_R \to [C]$.

Note that, by Theorem 5.9 and Proposition 5.13 of the lecture notes, $L(\mathcal{G}) = L_{\text{IO}}(\mathcal{G})$ since $\mathcal{G}$ is linear, and we could try to apply Proposition 5.14 and Proposition 5.15 of the lecture notes to obtain an algorithm running in $O(|\mathcal{G}| \cdot |Q|^{M+D+1})$, by constructing a tree automaton with $|Q| = O(|w|^2)$ states with $\text{Yield}(L(\mathcal{A})) = \{w\}$. This is not polynomial due to the D and M in the exponent, and quite a bit of work would be involved in order to show that we can bound those.

2 Scope ambiguities and covert moves in ACGs

Exercise 4. One considers the two following signatures:

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

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

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

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

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

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:

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

Solution:

$$
\begin{aligned}
t_0 &= \textsc{every man}\,(\textsc{move trace}\,(\lambda x.\, \textsc{some help}\,(\textsc{move trace}\,(\lambda y.\, \textsc{needs}\, y\, x)))) \\
t_1 &= \textsc{some help}\,(\textsc{move trace}\,(\lambda y.\, \textsc{every man}\,(\textsc{move trace}\,(\lambda x.\, \textsc{needs}\, y\, x))))
\end{aligned}
$$

Exercise 5. One considers a third signature :

$$
\begin{array}{lrl}
(\Sigma_{\text{L-FORM}}) & \mathbf{man} & : \mathsf{ind} \to \mathsf{prop} \\
& \mathbf{help} & : \mathsf{ind} \to \mathsf{prop} \\
& \mathbf{needs} & : \mathsf{ind} \to \mathsf{ind} \to \mathsf{prop}
\end{array}
$$

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 $(\mathcal{L}_{\mathrm{SEM}} : \Sigma_{\mathrm{ABS}} \to \Sigma_{\mathrm{L\text{-}FORM}})$ as follows:

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

[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*.

Solution:

$$
\begin{aligned}
NP &:= \mathsf{ind} \\
\textsc{trace} &:= \lambda x.\, x \\
\textsc{move} &:= \lambda xyz.\, y\, (x\, z) \\
\textsc{needs} &:= \lambda xy.\, \mathbf{needs}\, y\, x
\end{aligned}
$$

Then:

$$
\begin{aligned}
\mathcal{L}_{\mathrm{SEM}}(t_0) &= \forall x.\, (\mathbf{man}\, x) \to (\exists y.\, (\mathbf{help}\, y) \land (\mathbf{need}\, x\, y)) \\
\mathcal{L}_{\mathrm{SEM}}(t_1) &= \exists y.\, (\mathbf{help}\, y) \land (\forall x.\, (\mathbf{man}\, x) \to (\mathbf{need}\, x\, y))
\end{aligned}
$$

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

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

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

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

Solution: There are six such terms:

$$u_0 = \textsc{possibly}\,(\textsc{every man}\,(\textsc{move trace}\,(\lambda x.\,\textsc{some help}\,(\textsc{move trace}\,(\lambda y.\,\textsc{needs}\,y\,x)))))$$
$$u_1 = \textsc{every man}\,(\textsc{move trace}\,(\lambda x.\,\textsc{possibly}\,(\textsc{some help}\,(\textsc{move trace}\,(\lambda y.\,\textsc{needs}\,y\,x)))))$$
$$u_2 = \textsc{every man}\,(\textsc{move trace}\,(\lambda x.\,\textsc{some help}\,(\textsc{move trace}\,(\lambda y.\,\textsc{possibly}\,(\textsc{needs}\,y\,x)))))$$
$$u_3 = \textsc{possibly}\,(\textsc{some help}\,(\textsc{move trace}\,(\lambda y.\,\textsc{every man}\,(\textsc{move trace}\,(\lambda x.\,\textsc{needs}\,y\,x)))))$$
$$u_4 = \textsc{some help}\,(\textsc{move trace}\,(\lambda y.\,\textsc{possibly}\,(\textsc{every man}\,(\textsc{move trace}\,(\lambda x.\,\textsc{needs}\,y\,x)))))$$
$$u_5 = \textsc{some help}\,(\textsc{move trace}\,(\lambda y.\,\textsc{every man}\,(\textsc{move trace}\,(\lambda x.\,\textsc{possibly}\,(\textsc{needs}\,y\,x)))))$$

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

Solution:

$$\mathcal{L}_{\text{SEM}}(u_0) = \Diamond(\forall x.\,(\mathbf{man}\,x) \to (\exists y.\,(\mathbf{help}\,y) \land (\mathbf{need}\,x\,y)))$$
$$\mathcal{L}_{\text{SEM}}(u_1) = \forall x.\,(\mathbf{man}\,x) \to \Diamond(\exists y.\,(\mathbf{help}\,y) \land (\mathbf{need}\,x\,y))$$
$$\mathcal{L}_{\text{SEM}}(u_2) = \forall x.\,(\mathbf{man}\,x) \to (\exists y.\,(\mathbf{help}\,y) \land \Diamond(\mathbf{need}\,x\,y))$$
$$\mathcal{L}_{\text{SEM}}(u_3) = \Diamond(\exists y.\,(\mathbf{help}\,y) \land (\forall x.\,(\mathbf{man}\,x) \to (\mathbf{need}\,x\,y)))$$
$$\mathcal{L}_{\text{SEM}}(u_4) = \exists y.\,(\mathbf{help}\,y) \land \Diamond(\forall x.\,(\mathbf{man}\,x) \to (\mathbf{need}\,x\,y))$$
$$\mathcal{L}_{\text{SEM}}(u_5) = \exists y.\,(\mathbf{help}\,y) \land (\forall x.\,(\mathbf{man}\,x) \to \Diamond(\mathbf{need}\,x\,y))$$