Exam: Synchronous Grammars

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

Synchronous grammars consist of pairs of grammars whose productions (or trees) are synchronized through the use of variables. Various kinds of grammars can be synchronised. These grammars mainly used to define syntactic translations and syntax-semantics interfaces.

Notations. Let $\mathcal{X}$ be a countable infinite set of variables. For a finite alphabet Σ, we will consider finite **linear strings** α over $\Sigma \uplus \mathcal{X}$, i.e. strings where each variable from $\mathcal{X}$ occurs at most once. Similarly, given a finite ranked alphabet V, we will consider **linear terms** t in $T(V, \mathcal{X})$, where variables in $\mathcal{X}$ have rank 0, and each variable occurs at most once.

A **synchronised string** is a pair α, α' that contains the same variables. We can thus write a synchronised string as

$$w_0 x_1 w_1 \cdots w_{m-1} x_m w_m,\ w'_0 x_{\pi(1)} w'_1 \cdots w'_{m-1} x_{\pi(m)} w'_m \qquad (\dagger)$$

for some w_i and w'_i in Σ^*, x_i in $\mathcal{X}$, and π a permutation of $\{1, \ldots, m\}$. We denote by $\mathcal{X}(\alpha) = \mathcal{X}(\alpha') = \{x_1, \ldots, x_m\}$ the associated set of variables. Similarly, a **synchronised term** is a pair t, t' that contains the same variables $\mathcal{X}(t) = \mathcal{X}(t')$.

Exercise 1 (Synchronous Context-Free Grammars)**.** A **synchronous CFG** (SCFG) is defined as a tuple $\mathcal{G} = \langle N, \Sigma, P, S \rangle$ where, as usual, N is a finite alphabet of non-terminal symbols, Σ is a finite alphabet of terminal symbols disjoint from N, P is a finite set of productions of the form

$$A \to \alpha,\ \alpha',\ \sigma$$

where α, α' is a synchronised string and σ is a substitution $\mathcal{X}(\alpha) \to N$.[1]

Let $V \overset{\text{def}}{=} (N \uplus \Sigma)$. A **sentential form** of $\mathcal{G}$ is a triple δ, δ', s where δ, δ' is a synchronised string and s is a substitution $\mathcal{X}(\delta) \to N$. The **derivation relation** $\Rightarrow$ between sentential forms is defined by

$$\beta x \gamma, \beta' x \gamma', s \Rightarrow \beta \nu(\alpha) \gamma, \beta' \nu(\alpha') \gamma', (\sigma \circ \nu^{-1}) \cup (s \setminus \{(x, A)\})$$

[1] In the literature, people often write such productions as

$$A \to w_0 A_1^{[1]} w_1 \cdots A_m^{[m]} w_m,\ w'_0 A_{\pi(1)}^{[\pi(1)]} w'_1 \cdots A_{\pi(m)}^{[\pi(m)]} w'_m$$

where the superscripts between square brackets denote the variable underlying a nonterminal, i.e. x_i is the variable associated to a nonterminal $A^{[i]}$. As in $(\dagger)$, we have here w_i and w'_i in Σ^* and π a permutation of $\{1, \ldots, m\}$.

where $s(x) = A$, $A \to \alpha, \alpha', \sigma$ is in P, and $\nu\colon \mathcal{X}(\alpha) \to \mathcal{X} \setminus \mathcal{X}(\beta\gamma)$ injectively substitutes fresh variables instead of those found in $\mathcal{X}(\alpha)$, and is lifted to a homomorphism over $(\Sigma \uplus \mathcal{X}(\alpha))^*$ by setting $\nu(a) \overset{\text{def}}{=} a$ for all a in Σ. Applying ν to α and α' ensures that the new pair $\beta\nu(\alpha)\gamma$, $\beta'\nu(\alpha')\gamma'$ is again a synchronised string.

The *string language* defined by a SCFG is the set of pairs of strings[2]

$$L(\mathcal{G}) \overset{\text{def}}{=} \{w, w' \in \Sigma^* \times \Sigma^* \mid x, x, \{(x, S)\} \Rightarrow^\star w, w', \emptyset\}\ .$$

Example 1. The SCFG with rules[3] $S \to x_1x_2, x_1x_2, \{(x_1, A), (x_2, A)\}$ and $A \to a, a, \emptyset \mid b, b, \emptyset$ allows for instance the derivation

$$\begin{aligned} x_0, x_0, \{(x_0, S)\} &\Rightarrow x_1x_2, x_1x_2, \{(x_1, A), (x_2, A)\} \\ &\Rightarrow ax_2, ax_2, \{(x_2, A)\} \\ &\Rightarrow ab, ab, \emptyset \end{aligned}$$

Its langage is $\{(cd, cd) \mid c, d \in \{a, b\}\}$. In particular, the pair of strings (ab, ba) is not in the language.

[1] 1. Give a SCFG for the string language $L_{\text{cross}} \overset{\text{def}}{=} \{a^nb^m, c^nd^m \mid n, m \geq 0\}$.

[2] 2. Let # be a fresh symbol. Justify why $L_\#(\mathcal{G}) = \{w_1\#w_2 \in (\Sigma \uplus \{\#\})^* \mid x, x, \{(x, S)\} \Rightarrow^\star w_1, w_2, \emptyset\}$ is a mildly context-sensitive language and more precisely a 2-MCFL (multiple context-free language), i.e. a language generated by a sRCG with predicates of arity at most 2, or by a LCFRS with tuples of size at most 2.

[6] 3. A variant of SCFGs, dubbed **inverse transduction grammars** (ITG), is frequently employed. They are defined like SCFGs with an additional constraint: for all synchronised strings of form (†) in P, either π is the identity permutation ($\pi(i) = i$ for all $1 \leq i \leq m$) or π is the reverse permutation ($\pi(i) = m + 1 - i$ for all $1 \leq i \leq m$).

   (a) Show that ITGs can be **binarised** to use only synchronised strings of the forms (T) w_0, w'_0, or (I) x_1x_2, x_1x_2, or (R) x_1x_2, x_2x_1 in its productions.

   (b) Propose a CKY-like deductive system for recognising pairs of strings w, w' using binarised ITGs, i.e. for telling whether the pair belongs to $L(\mathcal{G})$.

   (c) Deduce a worst-case complexity upper bound for ITG parsing.

[2] 4. As in the case of context-free grammars, we can associate a *tree language* to a SCFG.

   To simplify matters, let us assume that our SCFG $\mathcal{G}$ has no ε productions, i.e. $|\alpha| > 0$ and $|\alpha'| > 0$ in the productions in P. Formally, we associate a kind of synchronous term rewriting system to $\mathcal{G}$. We treat for this V as a ranked alphabet, where the symbols in Σ are nullary, and those in N have rank at least 1. We define a disjoint copy

[2] Using superscripts, this can be written as $L(\mathcal{G}) = \{w, w' \in \Sigma^* \times \Sigma^* \mid S^{[1]}, S^{[1]} \Rightarrow^\star w, w'\}$.

[3] The alternate notation for the productions of this SCFG gives $S \to A^{[1]}A^{[2]}, A^{[1]}A^{[2]}$ and $A \to a, a|b, b$.

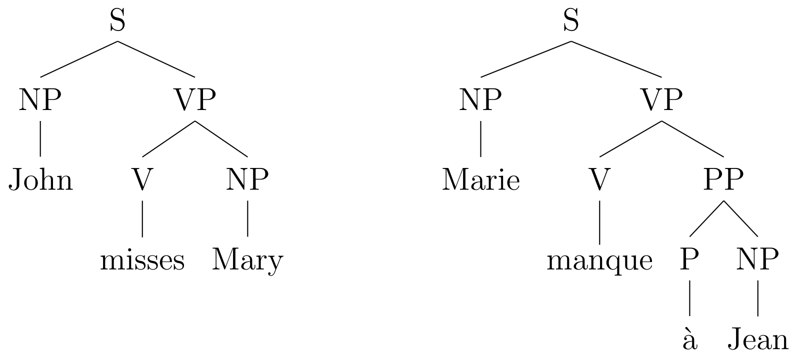
$\bar{N} = \{\bar{A} \mid A \in N\}$ of N all with rank 0. The term rewriting relation $\Rightarrow_{\mathcal{G}}$ associated to $\mathcal{G}$ works on triples t, t', s where t, t' is a synchronised term and $s\colon \mathcal{X}(t) \to \bar{N}$ is a substitution. We rewrite $C[x], C'[x], s$ where $s(x) = \bar{A}$ and $A \to \alpha, \alpha', \sigma$ is a production in P, into a new triple

$$C[A^{(|\alpha|)}(\nu(\alpha))],\ C'[A^{(|\alpha'|)}(\nu(\alpha'))],\ (\sigma \circ \nu^{-1}) \cup (s \setminus \{(x, \bar{A})\})$$

where ν is as before an injective substitution from $\mathcal{X}(\alpha)$ to $\mathcal{X} \setminus \mathcal{X}(C[x])$. The **tree language** of $\mathcal{G}$ is then

$$T(\mathcal{G}) \stackrel{\text{def}}{=} \{t, t' \in T(V) \times T(V) \mid x, x, \{(x, \bar{S})\} \Rightarrow^{\star}_{\mathcal{G}} t, t', \emptyset\}\ .$$

Show that no SCFG can induce a translation between the following French and English structures:



Exercise 2 (Synchronous Tree Substitution Grammars)**.** In a similar way, we can synchronise the derivations of two tree substitution grammars. To make the notations consistent with the previous exercise, we define a **synchronous TSG** (STSG) $\mathcal{S} = \langle V, N^{\downarrow}, R, S^{\downarrow}\rangle$ as a kind of synchronous term rewriting system. It features:

- A finite ranked alphabet V, where we denote again the nullary part of V as Σ and the remainder as N.
- A finite set of *substitution variables* $N^{\downarrow}$, which are all nullary symbols.
- A finite set of rules R, which consists of triples $A^{\downarrow} \to t, t', \sigma$ where t, t' is a synchronised term over $T(V, \mathcal{X})$ and $\sigma\colon \mathcal{X}(t) \to N^{\downarrow}$ is a substitution.
- An initial variable $S^{\downarrow}$ in $N^{\downarrow}$.

A STSG induces a synchronised *derivation relation* $\Rightarrow_{\mathcal{S}}$ over triples t, t', s, where t, t' is a synchronised term and $s\colon \mathcal{X}(t) \to N^{\downarrow}$ is a substitution. This relation rewrites a triple $C[x], C'[x], s$ with $s(x) = A^{\downarrow}$ into $C[\nu(t)], C[\nu(t')], (\sigma \circ \nu^{-1}) \cup (s \setminus \{(x, A^{\downarrow})\})$ whenever

$A^\downarrow \to t, t', \sigma$ is a rule in R, where $\nu\colon \mathcal{X}(t) \to \mathcal{X} \setminus \mathcal{X}(C[x])$ is an injective substitution that freshens the variables in t and t'. The **derived language** of $\mathcal{S}$ is

$$T(\mathcal{S}) \overset{\text{def}}{=} \{t, t' \in T(V) \times T(V) \mid x, x, \{(x, S^\downarrow)\} \Rightarrow^\star_{\mathcal{S}} t, t', \emptyset\}\ .$$

We want to study the expressiveness of STSGs:

[1] 1. Show that, for every language of tree pairs $T \subseteq T(V) \times T(V)$ over a ranked alphabet V, if there exists a SCFG $\mathcal{G}$ with $T(\mathcal{G}) = T$, then there exists a STSG $\mathcal{S}$ with $T(\mathcal{S}) = T$.

[0] 2. Show that there exists a language of tree pairs $T = T(\mathcal{S})$ for some STSG $\mathcal{S}$, such that no SCFG generates T.

[1] 3. Can you give an example for the previous question where additionally every tree pair t, t' in T is such that $t = t'$?

[5] 4. Consider two ranked alphabets Δ and V. A **tree homomorphism** from $T(\Delta)$ to $T(V)$ is defined by a function h that maps symbols $f^{(r)}$ in Δ to trees $h(f^{(r)}) = C[y_1, \ldots, y_r]$ in $T(V, \{y_1, \ldots, y_r\})$. Such a homomorphism is lifted to trees $t = f^{(r)}(t_1, \ldots, t_r)$ by $h(t) \overset{\text{def}}{=} h(f^{(r)})[y_1/h(t_1), \ldots, y_r/h(t_r)]$. A homomorphism is **linear non-deleting** if the images $h(f^{(r)}) = C[y_1, \ldots, y_r]$ are linear, i.e. every variable y_i for $1 \leq i \leq r$ occurs exactly once in $C[y_1, \ldots, y_r]$.

A **tree bimorphism** is a tuple $\mathcal{B} = \langle \Delta, V, D, h_1, h_2 \rangle$ where Δ and V are ranked alphabets, D is a regular tree language included in $T(\Delta)$, and h_1 and h_2 are two tree homomorphisms from $T(\Delta)$ to $T(V)$. The tree language defined by $\mathcal{B}$ is

$$T(\mathcal{B}) \overset{\text{def}}{=} \{h_1(t), h_2(t) \in T(V) \times T(V) \mid t \in D\}\ .$$

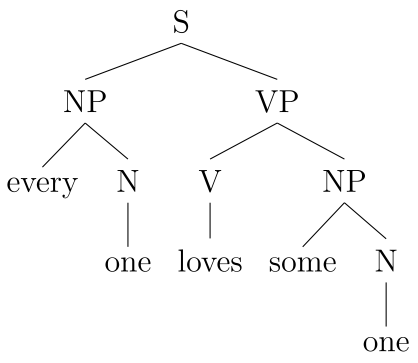
A tree bimorphism is **linear non-deleting** if both h_1 and h_2 are linear non-deleting.

Show that, for any language of tree pairs $T \subseteq T(V) \times T(V)$, there exists a STSG $\mathcal{S}$ with $T(\mathcal{S}) = T$ if and only if there exists a linear non-deleting bimorphism $\mathcal{B}$ with $T = T(\mathcal{B})$.

[1] 5. We measure the size of a STSG as $|\mathcal{S}| \overset{\text{def}}{=} \sum_{A^\downarrow \to t, t', \sigma \in R} |t| + |t'|$. What is the time complexity of testing whether $T(\mathcal{S}) = \emptyset$ for a STSG $\mathcal{S}$?

Exercise 3 (Higher-Order Semantics)**.** The purpose of this exercise is to model the syntax/semantics interface of a tiny fragment of English using a STSG $\mathcal{S}$. The first component of its derived language will be a syntactic constituent analysis of a sentence, while the second component will be a λ-term interpreted in Church's simple theory of types.

On the syntactic side, we want our STSG to derive at least the following tree:



On the semantic side, we consider a higher-order signature with atomic types $A = \{\iota, o\}$, constants $C = \{\bot, \supset, (\forall_\tau)_{\tau\in\mathcal{T}(A)}, \text{person}, \text{love}\}$, and typing $t(\bot) = o$, $t(\supset) = o \to o \to o$, $t(\forall_\tau) = (\tau \to o) \to o$, $t(\text{person}) = \iota \to o$, and $t(\text{love}) = \iota \to \iota \to o$. After β-reduction, we hope to see at least the following λ-term on its semantic side

$$\forall_\iota x.\text{person}\, x \supset \exists_\iota y.\text{person}\, y \wedge \text{love}\, x\, y\ .$$

[5] 1. Define a lexicalised STSG that defines a *compositional* syntax/semantics interface for the example sentence "every one loves some one". (For the semantic side, write trees with λ-terms as leaves; write "@" on the internal nodes to denote applications.)