1 Deterministic Top-down Finite-state Tree Transducers

1.1 Orientation

This section is about deterministic bottom-up finite-state tree transducers. The term finite-state means that the amount of memory needed in the course of computation is independent of the size of the input. The term deterministic means there is single course of action the machine follows to compute its output. The term transducer means this machine solves transformation problem: given an input object \( x \), what object \( y \) is \( x \) transformed into? The term tree means we are considering the transformation problem from trees to trees. The term top-down means that for each node \( a \) in a tree, the computation transforms the node before transforming its children. This contrasts with bottom-up transducers which transform the children before transforming their parent. Visually, these terms make sense provided the root of the tree is at the top and branches of the tree move downward.

A definitive reference for finite-state automata for trees is freely available online. It is “Tree Automata Techniques and Applications” (TATA) (Comon et al., 2007). The presentation here differs from the one there, as mentioned below.

1.2 Definitions

As before, we use variably leafed trees \( \Sigma^T[X] \).

**Definition 1 (DTFTT).** A Deterministic Top-down Finite-state Acceptor (DTFTT) is a tuple \((Q, \Sigma_r, q_0, \delta)\) where
- \( Q \) is a finite set of states;
- \( \Sigma \) is a finite alphabet;
- \( q_0 \in Q \) is the initial state; and
- \( \delta : Q \times \Sigma \times \mathbb{N} \rightarrow Q^* \) is the transition function.
- \( \Omega \) is a function with domain \( Q \times \Sigma \times \mathbb{N} \) and co-domain \( \Sigma^T[X] \).

Generally, the pre-images of \( \delta \) and \( \Omega \) should coincide.

We also define a new function “process” \( \pi : Q \times \Sigma^T \rightarrow \Sigma^T \) which will process the tree and produce its output. It is defined as follows.

\[
\begin{align*}
\pi(q, a[ ]) &= \Omega(q, a, 0) \\
\pi(q, a[t_1 \cdots t_n]) &= \Omega(q, a, n)\langle \pi(q_1, t_1) \cdots \pi(q_n, t_n) \rangle \\
&\text{where } q_1 \cdots q_n = \delta(q, a, n)
\end{align*}
\]

Intuitively, \( \Omega \) transforms the root node into a variably leafed tree. The variables are replaced with the children of the root node. These children are also trees with states assigned by \( \delta \). Then \( \pi \) transforms each tree-child as well.
Definition 2 (Tree-to-tree function of a DTFTT). The function defined by the transducer $M$ is $\{(t, s) \mid t, s \in \Sigma^T, \pi(q_0, t) = s\}$. If $(t, s)$ belongs to this set, we say $M$ transduces $t$ to $s$ and write $M(t) = s$.

Example 1. Consider the transducer $M$ constructed as follows.

- $Q = \{q, qs\}$
- $\Sigma = \{a, b, S\}$
- $q_0 = qs$
- $\delta(q, a, 0) = \lambda$
- $\delta(q, b, 0) = \lambda$
- $\delta(qs, S, 3) = qqsq$
- $\delta(qs, S, 2) = qq$
- $\Omega(q, a, 0) = a[ ]$
- $\Omega(q, b, 0) = b[ ]$
- $\Omega(q, S, 3) = S[x_3x_2x_1]$
- $\Omega(q, S, 2) = S[x_2x_1]$

Let see how $M$ transforms the tree below.

Exercise 1. Recall the “wh-movement” example from before. Explain why this transformation cannot be computed by a deterministic top-down tree transducer.

2 Theorems about Deterministic Tree Transducers

Theorem 1 (composition closure). The class of deterministic bottom-up transductions is closed under composition, but the class of top-down deterministic transductions is not.

Theorem 2 (Incomparable). The class of deterministic bottom-up transductions is incomparable with the class of top-down deterministic transductions.

This theorem is based on the same kind of examples which separated the left and right sequential functions. Let relations $U = (f^na, f^na) \mid n \in \mathbb{N} \cup (f^nb, g^nb) \mid n \in \mathbb{N}$ and $D = (ff^na, ff^na) \mid n \in \mathbb{N} \cup (gf^na, gf^nb) \mid n \in \mathbb{N}$.

References