Copyful Streaming String Transducers

Emmanuel Filiot, Pierre-Alain Reynier

Copyful Streaming String Transducers [⋆]

Emmanuel Filiot[1] and Pierre-Alain Reynier[2]

[1] Computer Science department, Université Libre de Bruxelles, Belgium
[2] Aix-Marseille Univ, LIF, CNRS, Marseille, France

Abstract. Copyless streaming string transducers (copyless SST) have been introduced by R. Alur and P. Černý in 2010 as a one-way deterministic automata model to define transductions of finite strings. Copyless SST extend deterministic finite state automata with a set of variables in which to store intermediate output strings, and those variables can be combined and updated all along the run, in a linear manner, i.e., no variable content can be copied on transitions. It is known that copyless SST capture exactly the class of MSO-definable string-to-string transductions, and are as expressive as deterministic two-way transducers. They enjoy good algorithmic properties. Most notably, they have decidable equivalence problem (in PSpace).

On the other hand, HDT0L systems have been introduced for a while, the most prominent result being the decidability of the equivalence problem. In this paper, we propose a semantics of HDT0L systems in terms of transductions, and use it to study the class of deterministic copyful SST. Our contributions are as follows:

(*i*) HDT0L systems and total deterministic copyful SST have the same expressive power,
(*ii*) the equivalence problem for deterministic copyful SST and the equivalence problem for HDT0L systems are inter-reducible, in linear time. As a consequence, equivalence of deterministic SST is decidable,
(*iii*) the functionality of non-deterministic copyful SST is decidable,
(*iv*) determining whether a deterministic copyful SST can be transformed into an equivalent deterministic copyless SST is decidable in polynomial time.

1 Introduction

The theory of languages is extremely rich and important automata-logic correspondances have been shown for various classes of logics, automata, and structures. There are less known automata-logic connections in the theory of transductions. Nevertheless, important results have been obtained for the class of MSO-definable transductions, as defined by Courcelle. Most notably, it has been

⋆ The work of Emmanuel Filiot was funded by the ARC project Transform (Federation Wallonia-Brussels) and the Belgian FNRS CDR project Flare. Emmanuel Filiot is research associate at F.R.S.-FNRS. The work of Pierre-Alain Reynier was funded by the DeLTA project (ANR-16-CE40-0007).

shown by J. Engelfriet and H.J. Hoogeboom that MSO-definable (finite) string to string transductions are exactly those transductions defined by deterministic two-way transducers [11]. This result has then been extended to ordered ranked trees by J. Engelfriet and S. Maneth, for the class of linear-size increase macro tree transducers [12] and recently to nested words-to-words transductions [9]. MSO-definable transductions of finite strings have also been characterized by a new automata model, that of (copyless) streaming string transducers, by R. Alur and P. Černý [2].

Copyless streaming string transducers (SST) extend deterministic finite state automata with a finite set of string variables $X, Y, \dots$. Each variable stores an intermediate string output and can be combined and updated with other variables. Along the transitions, a finite string can be appended or prepended to a variable, and variables can be reset or concatenated. The variable updates along the transitions are formally defined by variable substitutions and the copyless restriction is defined by considering only linear substitutions. Therefore, variable update such as $X := XX$ are forbidden by the copyless restriction. The SST model has then been extended to other structures such as infinite strings [7], trees [4], and quantitative languages [5].

Two examples of SST are depicted in Figure 1:

– The SST T_0 depicted on the left realizes the function f_0 mapping any input word $u \in \Sigma^*$ to the word $u\overline{u}$, where $\overline{u}$ is the mirror image of the word u. Indeed, when the input word u has been read by the automaton, the variable X contains the word u, while the variable Y contains the word $\overline{u}$. Hence, the final output, defined as XY, is equal to the concatenation $u\overline{u}$. It is worth noting that this SST is copyless.
– The SST T_1 depicted on the right realizes the function f_1 mapping any input word $u = a^n$, with $n \geq 1$, to the output word a^{2^n}. This SST is copyful.

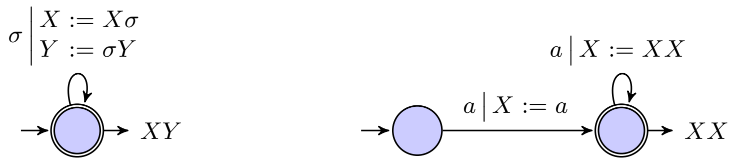

Fig. 1. Two streaming string transducers T_0 (left) and T_1 (right).

One of the most important and fundamental problem in the theory of transducers is the *equivalence problem*, which asks, given two transducers, whether they realize the same transduction. This problem is well known to be decidable for rational functions, and more generally for MSO-definable transductions [13], and hence for copyless SST (in PSpace, as shown in [1]). The problem gets undecidable when the transducers define binary relations instead of functions: it is already undecidable for rational transducers [16], a strict subclass of non-deterministic SST. However, it was unknown whether, in the functional case, decidability still holds without the copyless restriction, as mentioned in [3], an extended version of [4].

HDT0L systems allow to define languages by means of morphism iteration: a sequence of indices $i_1, \dots, i_k$ induces a composition of morphisms (one morphism for each index i_j) which, applied on an initial and fixed word, produces a new word. An important result related to HDT0L systems has been obtained in the 80's, see [17]. It states that the equivalence of finite-valued transducers over HDT0L languages is decidable, with unknown complexity.

In this paper, we build a tight connection between HDT0L systems and streaming string transducers. To this end, we introduce a new semantics of HDT0L systems, viewed as transducers. This allows us to prove that (total) copyful SST and HDT0L systems (seen as transducers) have the same expressive power, with back and forth transformations of linear complexity. As a corollary of this result, we obtain that the equivalence problem of copyful SST and the equivalence problem of HDT0L systems are inter-reducible, in linear time. This result has two consequences:

- first, the decidability of SST equivalence directly follows from [17],
- second, the functionality problem for non-deterministic (copyful) SST is decidable.

Note that the decidability of SST equivalence also follows from the two recent results [20] and [8].

Last, we study the following subclass definability problem: given a (copyful) SST, does there exist an equivalent copyless SST? We show that this problem is decidable in polynomial time, using a reduction to a boundedness problem for products of matrices studied by Mandel and Simon [19], and show that when possible, we can build an equivalent copyless SST.

Organization of the paper. We introduce the models of streaming string transducers and HDT0L systems in Section 2. In Section 3, we prove that these two models are equi-expressive. We apply this result to prove the decidability of the equivalence of copyful SST and of the functionality of non-deterministic SST in Section 4. Last, in Section 5, we study the subclass definability problem for copyless SST.

2 Preliminaries

For all finite alphabets Σ, we denote by Σ^* the set of finite words over Σ, and by ϵ the empty word. Given two alphabets Σ and Γ, a *transduction* R from Σ^* to Γ^* is a subset of $\Sigma^* \times \Gamma^*$. It is *functional* if it defines a function, i.e. for all $w \in \Sigma^*$, there exists at most one $v \in \Gamma^*$ such that $(w, v) \in R$. The domain of R, denoted by $Dom(R)$, is the set $Dom(R) = \{w \in \Sigma^* \mid \exists v \in \Gamma^*, (w, v) \in R\}$. A transduction is *total* if $Dom(R) = \Sigma^*$. Given two finite alphabets Σ, Γ, a morphim from Σ^* to Γ^* is a mapping $h : \Sigma^* \to \Gamma^*$ such that $h(uv) = h(u)h(v)$ for any two words $u, v \in \Sigma^*$.

2.1 Streaming String Transducers

Let $\mathcal{X}$ be a finite set of variables denoted by $X, Y, \dots$ and Γ be a finite alphabet. A substitution s is defined as a mapping $s : \mathcal{X} \to (\Gamma \cup \mathcal{X})^*$. Let $\mathcal{S}_{\mathcal{X},\Gamma}$ be the set of all substitutions. Any substitution s can be extended to $\hat{s} : (\Gamma \cup \mathcal{X})^* \to (\Gamma \cup \mathcal{X})^*$ in a straightforward manner. The composition $s_1 \circ s_2$ (or $s_1 s_2$ for short) of two substitutions $s_1, s_2 \in \mathcal{S}_{\mathcal{X},\Gamma}$ is defined as the standard function composition $\hat{s_1} \circ s_2$, i.e. $(s_1 s_2)(X) = (\hat{s_1} s_2)(X) = \hat{s_1}(s_2(X))$ for all $X \in \mathcal{X}$.

Definition 1. *A non-deterministic streaming string transducer (*NSST *for short) is a tuple $T = (\Sigma, \Gamma, Q, Q_0, Q_f, \Delta, \mathcal{X}, \rho, s_0, s_f)$ where:*

- *Σ and Γ are finite alphabets of input and output symbols,*
- *Q is a finite set of states,*
- *$Q_0 \subseteq Q$ is a set of initial states,*
- *$Q_f \subseteq Q$ is a set of final states,*
- *$\Delta \subseteq Q \times \Sigma \times Q$ is a transition relation,*
- *$\mathcal{X}$ is a finite set of variables,*
- *$\rho : \Delta \to 2^{\mathcal{S}_{\mathcal{X},\Gamma}}$ is a variable update function such that $\rho(t)$ is finite, for all $t \in \Delta$,*
- *$s_0 : \mathcal{X} \to \Gamma^*$ is the initial function that gives the initial content of the variables,*
- *$s_f : Q_f \to (\mathcal{X} \cup \Gamma)^*$ is the final output function, which gives what is output for each final state.*

The concept of a run of an NSST is defined in an analogous manner to that of a finite state automaton: it is a finite sequence $r \in (Q\Sigma)^*Q$, denoted $r = q_0 \xrightarrow{\sigma_1} q_1 \xrightarrow{\sigma_2} q_2 \dots q_{n-1} \xrightarrow{\sigma_n} q_n$, such that $(q_i, \sigma_{i+1}, q_{i+1}) \in \Delta$ for all $0 \le i < n$. It is accepting if $q_0 \in Q_0$ and $q_n \in Q_f$. A sequence of substitutions $\overline{s} = \langle s_i \rangle_{1 \le i \le n}$ in $\mathcal{S}_{\mathcal{X},\Gamma}$ is *compatible* with r if for all $1 \le i \le n$, $s_i \in \rho(q_{i-1}, \sigma_i, q_i)$.

If r is accepting, the output of r, denoted by $Out(r) \subseteq \Gamma^*$, is defined as

$$Out(r) = \{s_0 s_1 \dots s_n s_f(q_n) \mid \langle s_i \rangle_{1 \le i \le n} \in (\mathcal{S}_{\mathcal{X},\Gamma})^* \text{ is compatible with } r\}.$$

For all words $w \in \Sigma^*$, the output of w by T, denoted by $T(w)$, is $T(w) = \{Out(r) \mid r \text{ is an accepting run on } w\}$. The *domain* of T, denoted by $Dom(T)$, is defined as the set of words w such that $T(w) \neq \emptyset$. The transduction $[\![T]\!]$ defined by T is the relation from Σ^* to Γ^* given by the set of pairs (w, v) such that $v \in T(w)$.

Deterministic and functional SST A deterministic SST (SST for short) is an NSST such that $|Q_0| = 1$ and for all $p \in Q, \sigma \in \Sigma$, there exists at most one $q \in Q$ such that $(p, \sigma, q) \in \Delta$, and such that for all $t \in \Delta$, $|\rho(t)| = 1$. When an SST is deterministic, we identify Q_0 with q_0, and given $t \in \Delta$, we write $\rho(t) = s$ instead of $\rho(t) = \{s\}$.

In the following, the streaming string transducers we consider are deterministic, unless they are explicitly stated to be non deterministic.

In [2,6], the variable updates are required to be *copyless*, i.e. for every variable $X \in \mathcal{X}$, and for every transition $t \in \Delta$, X occurs at most once in $\rho(t)(X_1), \dots, \rho(t)(X_n)$ where $\{X_1, \dots, X_n\} = \mathcal{X}$. One of the main result of [2] is to show that this restriction, as well as determinism, allows one to capture exactly the class of MSO-definable transductions.

It is worth noting that any SST, since it is deterministic, defines a functional transduction. More generally, we say that an NSST T is *functional* if $[\![T]\!]$ is functional. It is known that functional NSST with copyless update are no more expressive than (deterministic) SST with copyless update [6]. We show a similar result for (copyful) SST:

Proposition 1. *Functional* NSST *and* SST *are equi-expressive.*

Proof. Let $T = (\Sigma, \Gamma, Q, Q_0, Q_f, \Delta, \mathcal{X}, \rho, s_0, s_f)$ be a functional NSST. Without loss of generality, we assume that T is trim, *i.e.* every state of T is reachable from some initial state, and co-reachable from an accepting state. Any SST can be made trim by filtering out the states that do not have this property (which is decidable in PTime).

The main idea is to realize a subset construction on T (a similar construction was given in [7]). On states, the subset construction is just as the subset construction for NFA. On variables, one needs to duplicate each variable as many times as the number of states. The invariant property is the following: after reading a word w, if there exists a run ρ of T on w leading to q, then for all $X \in \mathcal{X}$ such that there exists an accepting continuation w' of w (*i.e.* $ww' \in Dom(T)$) whose output uses the content of X after ρ, then X_q and X have the same content after reading w. There might be several runs of T leading to q, but since T is trim and functional, then content of F does not depend on the chosen run. Hence the invariant is well-defined.

Formally, we define an equivalent SST $T' = (\Sigma, \Gamma, Q', q_0', Q_f', \Delta', \mathcal{X}', \rho', s_0', s_f')$ such that $(\Sigma, \Gamma, Q', q_0', Q_f', \Delta')$ is the DFA resulting from the classical subset construction (in particular $Q' = 2^Q$) and such that:

- $\mathcal{X}' = \mathcal{X} \times Q$ (each variable is denoted by X_q)
- $\forall t' = (Q_1, \sigma, Q_2) \in \Delta', \forall q_2 \in Q_2, \forall X \in \mathcal{X}$, $\rho'(t')(X_{q_2}) = \text{rename}_{q_1}(\rho(t)(X))$ for some $q_1 \in Q_1$ such that $t = (q_1, \sigma, q_2) \in \Delta$, where rename_{q_1} is the identity morphism on Σ and replaces any $Y \in \mathcal{X}$ by Y_{q_1}. As explained before, the functionality of T entails that the choice of q_1 is not important (a different choice would give the same value to X_{q_2}). This choice can be made canonical by using some order on the states of T.
- $\forall P \in Q_f'$, $\rho'(P) = \text{rename}_q(\rho(q))$ for some $q \in P \cap Q_f$. Once again, by functionality of T, the choice of q does not matter and can be made canonical. □

We say that a SST T is *total* if $[\![T]\!]$ is total. We also show that regarding the equivalence problem, considering total SST is harmless, as one can modify a SST in linear time in order to make it total.

Proposition 2. *Given two SST T, T', one can build in linear time two total SST T_{tot} and T'_{tot} such that $[\![T]\!] = [\![T']\!]$ iff $[\![T_{tot}]\!] = [\![T'_{tot}]\!]$.*

Proof. Indeed, let $\# \notin \Gamma$. Any (partial) SST T can be transformed into an SST T_{tot} that defines the following transduction: $[\![T_{tot}]\!](u) = [\![T]\!](u)$ if $u \in Dom(T)$, and $[\![T_{tot}]\!](u) = \#$ otherwise. This is achieved using a new variable $X_\#$ whose content is always $\#$, and by completing the rules of T by adding an accepting sink state q_{sink}. We also modify final states and the final output function: states that were not final are declared final, and the final output function associates with these states the variable $X_\#$. □

2.2 HDT0L Systems

Lindenmayer introduced in the sixties a formal grammar in order to model the developement process of some biological systems [18]. We consider here a particular class of these systems, called HDT0L systems (HDT0L stands for Deterministic 0-context Lindenmayer systems with Tables and with an additional Homomorphism).

Definition 2 (HDT0L System). *An* HDT0L *system over Σ and Γ is defined as a tuple $H = (\Sigma, A, \Gamma, v, h, (h_\sigma)_{\sigma\in\Sigma})$ where:*

- *Σ, A and Γ are finite alphabets,*
- *$v \in A^*$ is the initial word,*
- *h is a morphism from A^* to Γ^*,*
- *for each $\sigma \in \Sigma$, h_σ is a morphism from A^* to A^*.*

The equivalence problem for HDT0L systems asks, given two such systems $H = (\Sigma, A, \Gamma, v, h, (h_\sigma)_{\sigma\in\Sigma})$ and $G = (\Sigma, A, \Gamma, w, g, (g_\sigma)_{\sigma\in\Sigma})$, whether, for every $\sigma_1 \dots \sigma_k \in \Sigma^*$, we have $h(h_{\sigma_1} \dots h_{\sigma_k}(v)) = g(g_{\sigma_1} \dots g_{\sigma_k}(w))$. This problem is known to be decidable [17], with unknown complexity. The original proof of [17] is based on Ehrenfeucht's conjecture and Makanin's algorithm. Honkala provided a simpler proof in [**?**], based on Hilbert's Basis Theorem.

In order to transfer this decidability result to SST, we introduce a semantics of HDT0L systems in terms of transductions.

Definition 3 (Transduction realized by an HDT0L system). *Let $H = (\Sigma, A, \Gamma, v, h, (h_\sigma)_{\sigma\in\Sigma})$ be an HDT0L system. We define $[\![H]\!]$ as a (total) transduction from Σ^* to Γ^* defined by $[\![H]\!](\sigma_1 \dots \sigma_k) = h(h_{\sigma_1} \dots h_{\sigma_k}(v))$.*

Example 1. Let us consider the function f_0 introduced in the introduction. We define an HDT0L $H_0 = (\Sigma, A, \Sigma, v_0, h, (h_\sigma)_{\sigma\in\Sigma})$ such that $[\![H_0]\!] = f_0$, with $A = \{\$_1, \$_2, a, b\}$, $\Sigma = \{a, b\}$, $v_0 = \$_1\$_2$, and the morphisms are defined as follows:

$$\begin{array}{rrcl@{\qquad}rrcl@{\qquad}rrcl}
h: & a & \to & a & h_a: & a & \to & a & h_b: & a & \to & a \\
 & b & \to & b & & b & \to & b & & b & \to & b \\
 & \$_1 & \to & \epsilon & & \$_1 & \to & \$_1 a & & \$_1 & \to & \$_1 b \\
 & \$_2 & \to & \epsilon & & \$_2 & \to & a\$_2 & & \$_2 & \to & b\$_2
\end{array}$$

For instance, we have the following derivation:

$$\begin{aligned}\llbracket H_0 \rrbracket(abb) &= hh_{abb}(\$_1\$_2) = hh_{ab} \circ h_b(\$_1\$_2) = hh_{ab}(\$_1 bb\$_2) = hh_a(\$_1 bbbb\$_2) \\ &= h(\$_1 abbbba\$_2) = abbbba\end{aligned}$$

We can now rephrase the result of [17] as follows:

Theorem 1. *[17] Given two* HDT0L *systems H_1, H_2 over Σ and Γ, it is decidable whether $\llbracket H_1 \rrbracket = \llbracket H_2 \rrbracket$.*

In the next section, we show that HDT0L systems and SST define the same class of transductions.

3 SST and HDT0L systems are equi-expressive

Let Σ and Γ two alphabets. In this section, we always consider that SST and HDT0L systems are over Σ and Γ. We prove the following theorem:

Theorem 2. HDT0L *systems over Σ and Γ and total SST define the same class of transductions. Moreover, the constructions are effective in both directions, in linear-time.*

A direct consequence of this result is:

Corollary 1. *The equivalence problems for* HDT0L *systems and for (copyful) streaming string transducers are inter-reducible in linear time.*

We prove successively the two directions of Theorem 2.

Lemma 1. *For all HDT0L systems H, there exists an equivalent (total) SST T with only one state.*

Proof. Let $H = (\Sigma, A, \Gamma, v, h, (h_\sigma)_{\sigma \in \Sigma})$ be an HDT0L system. We construct a total SST T over Σ and Γ such that $\llbracket T \rrbracket = \llbracket H \rrbracket$. The SST has one state q, both initial and accepting. Its set of variables is the set $\mathcal{X} = \{X_a \mid a \in A\}$. Its transitions are defined by $q \xrightarrow{\sigma} q$ for all $\sigma \in \Sigma$.

To define the update functions, we first introduce the morphism $\mathrm{rename}_X : A^* \to \mathcal{X}^*$ defined for all $a \in A$ by $\mathrm{rename}_X(a) = X_a$. Then, the update function ρ is defined, for all $\sigma \in \Sigma$ and $a \in A$ by $\rho((q, \sigma, q))(X_a) = \mathrm{rename}_X(h_\sigma(a))$.

Finally, the initial function is defined by $s_0(X_a) = h(a)$ for all $a \in A$, and the final function s_f by $s_f(q) = \mathrm{rename}_X(v)$. $\square$

Example 2. For the HDT0L system H_0 of Example 1, we obtain the SST T_2 depicted in Figure 2.

We now prove the converse.

Lemma 2. *For all total SST T, there exists an equivalent HDT0L system H.*

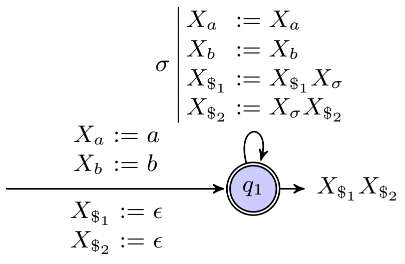

Fig. 2. A SST T_2.

Proof. Let $T = (\Sigma, \Gamma, Q, q_0, Q_f, \Delta, \mathcal{X}, \rho, s_0, s_f)$ be a total SST (remember that by default an SST is deterministic). We define an equivalent HDT0L H as follows.

We consider the finite alphabet $A = \{\alpha_q \mid \alpha \in \Gamma \cup \mathcal{X}, q \in Q\}$. For every $q \in Q$, we consider the morphism $\text{subscript}_q : (\Gamma \cup \mathcal{X})^* \to A^*$ defined for all $\alpha \in \Gamma \cup \mathcal{X}$ by $\text{subscript}_q(\alpha) = \alpha_q$.

As T is total, we have that $Q_f = Q$. We consider an enumeration $q_1, \ldots, q_n$ of Q. We define the initial word v as follows:

$$v = \text{subscript}_{q_1}(s_f(q_1)) \ldots \text{subscript}_{q_n}(s_f(q_n))$$

We define the morphim $h : A^* \to \Gamma^*$ as follows:

$$\begin{aligned} h : \gamma_{q_0} &\to \gamma && \text{with } \gamma \in \Gamma \\ X_{q_0} &\to s_0(X) && \text{with } X \in \mathcal{X} \\ \alpha_q &\to \epsilon && \text{with } q \neq q_0 \text{ and } \alpha \in \Gamma \cup \mathcal{X} \end{aligned}$$

Last, given $\sigma \in \Sigma$ we define the morphism $h_\sigma : A^* \to A^*$ as follows. Given a state q, we define the set $Pre_q^\sigma \subseteq Q$ as the set of states p such that $(p, \sigma, q) \in \Delta$.

We define: (by convention, the product over the empty set gives the empty word)

$$\begin{aligned} \forall \gamma \in \Gamma, \quad h_\sigma(\gamma_q) &= \Pi_{p \in Pre_q^\sigma} \text{subscript}_p(\gamma) \\ \forall X \in \mathcal{X}, \, h_\sigma(X_q) &= \Pi_{p \in Pre_q^\sigma} \text{subscript}_p(\rho(p, \sigma, q)(X)) \end{aligned}$$

Intuitively, the HDT0L system simulates the computations of the SST in a backward manner, starting from the final states. These computations are encoded using the labelling of symbols by states. One can easily prove by induction on the length of some input word w that after reading w, for every state q, the projection of $h_w(v)$ on the subalphabet $\text{subscript}_q(\Gamma \cup \mathcal{X})$ encodes the run of the SST on w starting in state q, which is unique since T is deterministic. The morphism h then simply erases parts of the computations that did not reach the initial state q_0. $\square$

We point out the following result, which follows from Lemma 1 and Lemma 2:

Corollary 2. *For all SST T, one can construct in polynomial time an equivalent SST T' such that the underlying input DFA of T' is the minimal complete DFA recognizing $Dom(T)$.*

Proof. If T is total, then the result is a direct consequence of the successive application of Lemma 2 and Lemma 1. Note that in this case, T' has only one state.

If T is not total, we make it total as in the proof of Proposition 2, and obtain a total SST S which can be converted into a single state SST S'. Then, we minimize the underlying input DFA of T (which recognizes $Dom(T)$) into a minimal complete DFA A_{min}. Finally, T' is defined as a kind of product of S' and A_{min}: if $A_{min} = (\Sigma, P, p_0, P_f, \delta)$ and $S' = (\Sigma, \Gamma, \{q\}, q, \{q\}, \{(q, \sigma, q) \mid \sigma \in \Sigma\}, \mathcal{X}, \rho, s_0, s_f)$, then we let $T' = (\Sigma, \Gamma, P, p_0, P_f, \delta, \mathcal{X}, \rho', s_0, s'_f)$ where:

- $\rho'(p, \sigma, p') = \rho(q, \sigma, q)$ for all $(p, \sigma, p') \in \delta$,
- $s'_f(p_f) = s_f(q)$ for all $p_f \in P_f$. □

Remark 1. By this corollary, any total SST is equivalent to some single state SST.

4 Applications: SST Equivalence and Functionality of NSST

Based on the correspondence between SST and HDT0L systems, and the fact that the HDT0L system equivalence problem is decidable, we show that that the SST equivalence and functionality problems are decidable.

Theorem 3. *1. Given two SST T and T', it is decidable whether they are equivalent, i.e. $[\![T]\!] = [\![T']\!]$.*
2. Given an NSST T, it is decidable whether T is functional.

Proof. The first statement is straightforward by Theorem 2, Theorem 1 and Proposition 2.

To prove the second statement, we reduce the functionality problem to the equivalence of two (deterministic) SST T_1 and T_2. Let $T = (\Sigma, \Gamma, Q, Q_0, Q_f, \Delta, \mathcal{X}, \rho, s_0, s_f)$ be an NSST. We extend the alphabet Σ with pairs of rules of T as follows: $\Sigma' = \Sigma \times \Delta^2$. Now, T_1 and T_2 are defined as the square of T: they run on words w' over Σ', and make sure that the sequence of transitions are valid runs of T on the Σ-projections of w'. In addition, T_i simulates T on the $(i+1)$-th component, for all $i = 1, 2$, by following the transitions defined on the input letters. Clearly, T_1 and T_2 have the same domain, are deterministic, and are equivalent iff all pairs of accepting runs of T on the same input word produce the same output, i.e., iff T is functional. The conclusion follows from statement 1. □

5 Deciding the subclass of copyless SST

The subclass of copyless SST is of great interest, as it exactly corresponds to the class of regular functions which enjoys multiple characterizations (MSO transductions and deterministic two-way transducers for instance) and has been widely studied in the literature (see for instance [15] for a survey). Given a copyful SST, it is thus a natural question whether there exists an equivalent copyless SST and if so, whether one can actually compute such an equivalent machine. In this section, we answer these questions positively:

Theorem 4. *Given an* SST *T, it is decidable in* PTime *whether there exists an equivalent copyless* SST*. If this is the case, one can build an equivalent copyless* SST*.*

It is well-known that copyless SST define transductions f that are linear-size increase (LSI for short), *i.e.* there exists some constant M such that for every w, we have $|f(w)| \leq M|w|$ (this can be observed for instance using the MSOT presentation). We will use this semantical condition in order to solve the above problem.

Proof. Let $T = (\Sigma, \Gamma, Q, q_0, Q_f, \Delta, \mathcal{X}, \rho, s_0, s_f)$ be an SST. Consider a run $q_0 \xrightarrow{\sigma_1} q_1 \dots q_{n-1} \xrightarrow{\sigma_n} q_n = p$ of T starting in the initial state. By definition of the semantics of T, after this run, one can associate, in state p, with variable X the content $\nu(X) \in \Gamma^*$ defined as $s_0 s_1 \dots s_n(X)$ where $s_i = \rho(q_{i-1}, \sigma_i, q_i)$. We use the notation $q_0 \xrightarrow{\sigma_1 \dots \sigma_n} (p, \nu)$ to describe this fact, and may remove the label $\sigma_1 \dots \sigma_n$ when it is useless.

We also say that a pair $(p, X) \in Q \times \mathcal{X}$ is *co-accessible* whenever there exists a run starting in state p reaching a final state q_f such that the final output $s_f(q_f)$ involves a variable whose content depends on the content of X at the beginning of the run. In other words, the content of X at configuration (p, X) flows into $s_f(q_f)$, *i.e.* X is "useful" for $s_f(q_f)$. This intuitive notion of variable flow is formally defined in [14].

We introduce the following objects:

- we let $val(p, X) = \{\nu(X) \in \Gamma^* \mid \text{ there exists a run } q_0 \to (p, \nu)\}$
- we let $INF = \{(p, X) \mid (p, X) \text{ is co-accessible and } val(p, X) \text{ is infinite}\}$
- we define $(p, X) \xrightarrow{u|n} (q, Y)$ if there exists a run from p to q on word u on which X flows n times in Y. We may omit u when it is useless.

Claim: $[\![T]\!]$ is definable by a copyless SST iff there exists $K \in \mathbb{N}$ such that for all $(p, X) \xrightarrow{n} (q, Y)$ with $(p, X), (q, Y) \in INF$, we have $n \leq K$.

Before proving the claim, we show that it implies decidability. First, the set INF is computable in polynomial time (fixpoint computation in the set of pairs $(p, X) \in Q \times \mathcal{X}$). Second, we define the following set $\mathcal{M}$ of square matrices indexed by elements of INF with coefficients in $\mathbb{N} \cup \{\perp\}$. We suppose that $\perp$ behaves as 0: for every integer $n \in \mathbb{N}$, we have $n.\perp = \perp.n = \perp$ and

$n+\bot = \bot+n = n$. The set $\mathcal{M}$ is defined as the set of finite products of matrices $\{M_a \mid a \in \Sigma\}$, where, for each letter $a \in \Sigma$, we define matrix M_a by:

$$M_a[(p,X),(q,Y)] = \begin{cases} n \text{ if } (p,a,q) \in \Delta \text{ and } |\rho(p,a,q)(Y)|_X = n \\ \bot \text{ if } (p,a,q) \notin \Delta \end{cases}$$

We then observe that the right property of the claim is satisfied iff the set $\mathcal{M}$ is finite. By Mandel and Simon [19], this last property is decidable in polynomial time.

We turn to the proof of the claim:

$\Rightarrow$ We prove the contraposition, by showing that if the right property of the claim is not satisfied, then $[\![T]\!]$ is not LSI, which implies that $[\![T]\!]$ is not definable by a copyless SST as copyless SST are LSI. We thus assume that the set of flow matrices defined previously is not bounded. By the characterization proven in Mandel and Simon of this property, two cases may occur:

1. there exists $(p,X) \in INF$ such that $(p,X) \xrightarrow{n} (p,X)$ with $n \geq 2$,
2. there exist $(p,X),(q,Y) \in INF$ such that $(p,X) \xrightarrow{u|n_1} (p,X)$, $(p,X) \xrightarrow{u|n_2} (q,Y)$ and $(q,Y) \xrightarrow{u|n_3} (q,Y)$, with $n_1,n_2,n_3 \geq 1$, for some word u.

In the first case, using the fact that $(p,X) \in INF$, one can prove that $[\![T]\!]$ is not LSI.

In the second case, for every $n \geq 1$, one has $(p,X) \xrightarrow{u^n|m} (q,Y)$ for some $m \geq n$. Again, one can use this property to show that $[\![T]\!]$ is not LSI.

$\Leftarrow$ The constraint expressed by the right property of the claim precisely states that, with respect to pairs $(p,X) \in INF$, the SST is bounded copy. One can then easily remove variables (p,X) that do not belong to INF, so as to obtain a bounded copy SST, and it is known that every bounded copy SST can be turned into an equivalent copyless SST [7, 10]. □

6 Conclusion

Our results establish a bridge between the theory of SST and the theory of systems of iterated morphisms. It allows to solve an interesting open problem for copyful streaming string transducers, namely the decidability of the equivalence problem. We have also proven the decidability of functionality for non-deterministic SST, and that of the subclass of copyless SST, using a reduction to a boundedness problem.

We hope that these positive decidability results will pave the way to a further study of the class of copyful SST. As future work, we want to investigate what the theory of iterated morphisms can bring to the theory of SST, and conversely, in terms of tight complexity results. For instance, the class of copyless SST, for which equivalence is PSpace-complete, could have an interesting interpretation in terms of HDT0L systems.

References

1. A. Alur and P. Černý. Streaming transducers for algorithmic verification of single-pass list-processing programs. In *POPL*, pages 599–610, 2011.
2. R. Alur and P. Černý. Expressiveness of streaming string transducers. In *FSTTCS*, volume 8, pages 1–12, 2010.
3. R. Alur and L. D'Antoni. Streaming tree transducers. *CoRR*, abs/1104.2599, 2011.
4. R. Alur and L. D'Antoni. Streaming tree transducers. In *ICALP (2)*, volume 7392 of *LNCS*, pages 42–53. Springer, 2012.
5. R. Alur, L. D'Antoni, J. V. Deshmukh, M. Raghothaman, and Y. Yuan. Regular functions and cost register automata. In *28th Annual ACM/IEEE Symp. on Logic in Computer Science, LICS 2013*, pages 13–22. IEEE Computer Society, 2013.
6. R. Alur and J. V. Deshmukh. Nondeterministic streaming string transducers. In *ICALP*, volume 6756 of *LNCS*, pages 1–20. Springer, 2011.
7. R. Alur, E. Filiot, and A. Trivedi. Regular transformations of infinite strings. In *LICS*, pages 65–74. IEEE, 2012.
8. M. Benedikt, T. Duff, A. Sharad, and J. Worrell. Polynomial automata: zeroness and applications. In *Proceedings of the 32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '17*. ACM, 2017. To appear.
9. L. Dartois, E. Filiot, P.-A. Reynier, and J.-M. Talbot. Two-way visibly pushdown automata and transducers. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16*, pages 217–226. ACM, 2016.
10. L. Dartois, I. Jecker, and P. Reynier. Aperiodic string transducers. In *Developments in Language Theory - 20th International Conference, DLT 2016*, volume 9840 of *Lecture Notes in Computer Science*, pages 125–137. Springer, 2016.
11. J. Engelfriet and H. J. Hoogeboom. MSO definable string transductions and two-way finite-state transducers. *ACM Trans. Comput. Logic*, 2:216–254, 2001.
12. J. Engelfriet and S. Maneth. Macro tree transducers, attribute grammars, and mso definable tree translations. *Information and Computation*, 154(1):34–91, 1999.
13. J. Engelfriet and S. Maneth. The equivalence problem for deterministic MSO tree transducers is decidable. *Inf. Process. Lett.*, 100(5):206–212, 2006.
14. E. Filiot, S. N. Krishna, and A. Trivedi. First-order definable string transformations. In *34th International Conference on Foundation of Software Technology and Theoretical Computer Science, FSTTCS 2014*, volume 29 of *LIPIcs*, pages 147–159. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2014.
15. E. Filiot and P.-A. Reynier. Transducers, logic and algebra for functions of finite words. *SIGLOG News*, 3(3):4–19, 2016.
16. T. V. Griffiths. The unsolvability of the equivalence problem for lambda-free nondeterministic generalized machines. *J. ACM*, 15(3):409–413, 1968.
17. K. C. II and J. Karhumäki. The equivalence of finite valued transducers (on HDT0L languages) is decidable. *Theor. Comput. Sci.*, 47(3):71–84, 1986.
18. A. Lindenmayer. Mathematical models for cellular interaction in development. *Journal of Theoretical Biology*, 18:280–315, 1968.
19. A. Mandel and I. Simon. On Finite Semigroups of Matrices. *Theor. Comput. Sci.*, 5(2):101–111, 1977.
20. H. Seidl, S. Maneth, and G. Kemper. Equivalence of deterministic top-down tree-to-string transducers is decidable. In *IEEE 56th Annual Symp. on Foundations of Computer Science, FOCS 2015*, pages 943–962. IEEE Computer Society, 2015.