One-way definability of sweeping transducers

Félix Baschenis, Olivier Gauwin, Anca Muscholl, Gabriele Puppis

One-way definability of sweeping transducers*

Félix Baschenis, Olivier Gauwin, Anca Muscholl, Gabriele Puppis

Université de Bordeaux, LaBRI & CNRS
{fbaschen,ogauwin,anca,gpuppis}@labri.fr

Abstract

Two-way finite-state transducers on words are strictly more expressive than one-way transducers. It has been shown recently how to decide if a two-way functional transducer has an equivalent one-way transducer, and the complexity of the algorithm is non-elementary. We propose an alternative and simpler characterization for sweeping functional transducers, namely, for transducers that can only reverse their head direction at the extremities of the input. Our algorithm works in 2ExpSpace and, in the positive case, produces an equivalent one-way transducer of doubly exponential size. We also show that the bound on the size of the transducer is tight, and that the one-way definability problem is undecidable for (sweeping) non-functional transducers.

1998 ACM Subject Classification Theory of computation – F.4.3 Formal Languages

Keywords and phrases Regular word transductions, sweeping transducers, one-way definability

1 Introduction

Regular word languages form the best understood class of languages. They enjoy several characterizations, in particular by different kinds of finite-state automata. For instance, two-way finite-state automata have the same expressive power as one-way automata. This result has been established independently by Rabin and Scott [10] and Shepherdson [11]. Besides automata, regular languages have logical and algebraic characterizations, namely through monadic second-order logic and congruences of finite index.

Transducers extend automata by producing outputs with each transition. A run generates an output word by concatenating the words produced by its transitions. A transducer thus defines a relation over words. It is called functional when this relation is a function. For finite-state transducers, expressiveness is different than for finite-state automata. As an example, two-way transducers are strictly more expressive than one-way transducers. For instance, the function that maps a word to its mirror image can be done by a back-and-forth pass over the input, but no one-way transducer can do it.

As seen above, we lose some robustness when going from automata to transducers. On the other hand, some of the classical characterizations of regular languages generalize well to transducers. An important result is the equivalence of functional two-way transducers and Ehrenfeucht-Courcelle's monadic-second order transductions [5] over words. Another characterization of two-way transducers was provided through a new model called streaming string transducers [1, 2], that process the input one-way and store the output in write-only registers. Finally, first-order transductions are known to be equivalent to aperiodic streaming transducers [7] and to aperiodic two-way transducers [4].

The question whether a functional two-way word transducer is equivalent to a one-way transducer has been solved recently in [6]. The algorithm proposed by [6] takes a two-

* This work was partially supported by the ExStream project (ANR-13-JS02-0010) and the Technische Universität München – Institute for Advanced Study, funded by the German Excellence Initiative and the European Union Seventh Framework Programme under grant agreement n° 291763.

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

way transducer $\mathcal{S}$ and builds a one-way transducer $\mathcal{T}$ that is "maximal" in the following sense: (1) all accepting runs of $\mathcal{T}$ produce correct outputs, and (2) all runs of $\mathcal{S}$ that can be performed one-way are realized by $\mathcal{T}$. As a consequence, the two-way transducer $\mathcal{S}$ has an equivalent one-way transducer if and only if the constructed one-way transducer $\mathcal{T}$ has the same domain as $\mathcal{S}$, which is a decidable problem. The problem is that the upper bound on both the decision procedure and the size of the constructed one-way transducer is non-elementary.

The main contribution of this paper is an elementary procedure for deciding whether a functional two-way word transducer is equivalent to a one-way transducer, for the particular class of sweeping transducers. While two-way transducers can reverse their head direction at any position of the input, sweeping transducers can only reverse it at the first and last position. Unsurprisingly, sweeping transducers are strictly less expressive than two-way transducers, and the following example shows the difference: on input $u_1\ a\ u_2\ a\ \dots\ a\ u_{n-1}\ a\ u_n$, where the words u_i contain no occurrence of a, the two-way transducer produces as output $u_n\ a\ u_{n-1}\ a\ \dots\ a\ u_2\ a\ u_1$ (we assume that the alphabet contains at least two letters).

Our decision procedure works in doubly exponential space and, when it succeeds, it produces an equivalent one-way transducer of doubly exponential size. We show that the bound on the size of the transducer is tight for any decision algorithm producing an equivalent one-way transducer from a sweeping transducer. This improves the PSPACE lower bound from [6]. The non-elementary procedure described in [6] relies on Rabin-Scott's construction for automata, and works by eliminating basic zigzags in runs. Our procedure is closer to the textbook approach (due to Shepherdson) and uses crossing sequences. This requires a decomposition of runs which is incomparable with the zigzag decomposition of [6]. Finally, we show that the one-way definability problem becomes undecidable for non-functional transducers.

Overview

Section 2 defines transducers and related concepts. Section 3 defines decomposition of runs and gives the construction of a one-way transducer based on such decompositions. In Section 4 we show that all one-way-definable runs admit a decomposition. Section 5 provides the lower bound and the undecidability result. Some proofs are deferred to the appendix.

2 Preliminaries

Transducers

A *two-way transducer* is a tuple $(\Sigma, \Delta, Q, I, F, \delta)$, where Σ (resp., Δ) is a finite input (resp., output) alphabet, Q is a finite set of states, I (resp., F) is a subset of Q representing the initial (resp., final) states, and $\delta \subseteq Q \times \Sigma \times \Delta^\star \times Q \times \{\mathsf{left}, \mathsf{right}\}$ is a finite set of transition rules describing, for each state and input symbol, the possible output string, target state, and direction of movement. We talk of a *one-way* transducer whenever $\delta \subseteq Q \times \Sigma \times \Delta^\star \times Q \times \{\mathsf{right}\}$. The *size* of a transducer is its number of states.

According to standard practice, the states of *one-way* automata and transducers are usually located between the letters of the input word $u = a_1 \dots a_n$. For this it is convenient to introduce $n+1$ *positions* $0, 1, \dots, n$ and think of each position $i > 0$ (resp., 0) as a placeholder between the i-th and the $i+1$-th symbols (resp., just before the first symbol a_1). Moreover, since here we deal with *two-way* devices, a single position can be visited several times along a run. Thus, to describe a run of a two-way transducer on input $u = a_1 \dots a_n$,

we will associate states with *locations*, namely, with pairs (x, y) where x is a position among $0, 1, \ldots, n$ and y is an integer representing the number of reversals performed up to a certain point – for short, we call this number y the *level* of the location.

A run is a sequence of locations, labelled by states and connected by edges, called *transitions*. The state at location $\ell = (x, y)$ of a run ρ is denoted $\rho(\ell)$. The transitions must connect pairs of locations that are either at adjacent positions and on the same level, or at the same position and on adjacent levels. In addition, each transition is labelled with a pair a/v consisting of an input symbol a and an output v. There are four types of transitions:

$$(i-1, 2y+1) \xleftarrow{a/v} (i, 2y+1) \qquad (i-1, 2y) \xrightarrow{a/v} (i, 2y)$$

$$a/v \circlearrowright \begin{matrix}(i, 2y+2)\\(i, 2y+1)\end{matrix} \qquad \begin{matrix}(i-1, 2y+1)\\(i-1, 2y)\end{matrix} \circlearrowleft a/v$$

Note that the transitions between locations at even levels are all directed from left to right, while the transitions at odd levels are directed from right to left. More precisely, the upper left (resp., upper right) transition may occur in a run ρ on $u = a_1 \ldots a_n$ if $\big(\rho(i, 2y+1), a, v, \rho(i-1, 2y+1), \mathsf{left}\big)$ (resp., $\big(\rho(i-1, 2y), a, v, \rho(i, 2y), \mathsf{right}\big)$) is a valid transition rule of $\mathcal{T}$ and $a = a_i$. Similarly, the lower left (resp., lower right) transition may occur if $\big(\rho(i, 2y+1), a, v, \rho(i, 2y+2), \mathsf{right}\big)$ (resp., $\big(\rho(i-1, 2y), a, v, \rho(i-1, 2y+1), \mathsf{left}\big)$) is a valid transition rule of $\mathcal{T}$ and $a = a_i$. For technical reasons (namely, to enable distinguished transitions at the extremities of the input word), we will introduce the special fresh symbols $\triangleright$ and $\triangleleft$ and allow the lower left (resp., lower right) transition also when $i = 0$ and $a = \triangleright$ (resp., when $i = |u|$ and $a = \triangleleft$).

Given a sequence $x_1, \ldots, x_n$, a *factor* denotes any contiguous subsequence $x_i, \ldots, x_j$, for $1 \leq i \leq j \leq n$. A run on the input $u = a_1 \ldots a_n$ is said to be *successful* if it starts at the lower left location $(0, 0)$ with an initial state of $\mathcal{T}$ and ends at the upper right location $(|u|, y_{\mathsf{max}})$ in a final state of $\mathcal{T}$. The output produced by a run is the concatenation of the outputs of its transitions, and it is denoted by $\mathsf{out}(\rho)$. We denote by $\mathsf{dom}(\mathcal{T})$ the language of all words u that admit a successful run of $\mathcal{T}$. We order the locations along a run ρ by letting $\ell_1 < \ell_2$ if ℓ_2 is reachable from ℓ_1 following the transitions in ρ. Given two locations $\ell_1 < \ell_2$ of a run ρ, we denote by $\rho[\ell_1, \ell_2]$ the factor of the run that starts in ℓ_1 and ends in ℓ_2. Note that $\rho[\ell_1, \ell_2]$ is also a run, hence the notation $\mathsf{out}\big(\rho[\ell_1, \ell_2]\big)$ is consistent.

Further assumptions

We will mostly work with two-way transducers that are *sweeping*. This means that on every successful run, the head can change direction only at the extremities of the input. In other words, the lower right (resp., lower left) transition is possible only if $a = \triangleleft$ (resp., $a = \triangleright$).

A transducer $\mathcal{T}$ is *functional* if, for each input word u, all successful runs on u produce the same output. In this case $\mathcal{T}(u)$ denotes the unique output produced on input u.

Unless otherwise stated, we will assume that all transducers are sweeping and functional. Note that functionality is a decidable property, as stated below. The proof is similar to the decidability proof of equivalence of streaming string transducers [1] and reduces the problem to the reachability of a 1-counter automaton of exponential size.

▶ **Proposition 1.** Functionality of two-way transducers can be decided in polynomial space. This problem is PSPACE-hard even for sweeping transducers.

Without loss of generality, we can also assume that the successful runs of a functional transducer are *normalized*, namely, they never visit two locations with the same position,

the same state and both either at an even level or at an odd level. Indeed, if this were not the case, say if a successful run ρ visits two locations $\ell_1 = (x, y_1)$ and $\ell_2 = (x, y_2)$ such that $\rho(\ell_1) = \rho(\ell_2)$ and y_1, y_2 are both even or both odd, then the output produced by ρ between ℓ_1 and ℓ_2 is either empty – in which case we could remove $\rho[\ell_1, \ell_2]$ and obtain an equivalent successful run – or is non-empty – in which case, by repeating $\rho[\ell_1, \ell_2]$, we could obtain successful runs that produces different outputs on the same input, thus contradicting the assumption that the transducer is functional.

Crossing sequences

Consider a run ρ of a transducer on input $u = a_1 \dots a_n$. For each position $x \in \{0, 1 \dots, n\}$, we are interested in the sequence of states labelling the locations at position x. Formally, we define the *crossing sequence of* ρ *at* x as the tuple $\rho|x = \big(\rho(x, y_0), \dots, \rho(x, y_h)\big)$, where $y_0 < \dots < y_h$ are exactly the levels of the locations of ρ of the form (x, y), with $y \in \mathbb{N}$ (if the transducer is sweeping, we simply have $y_i = i$). If the considered run ρ is successful, then the bottom and top locations at position x have even levels, and the outgoing transitions move rightward. In particular, every crossing sequence of a successful run has odd length. Moreover, if we assume that the successful run is normalized, then every crossing sequence has length at most $2|Q| - 1$. The *crossing number* of a run is the maximal length of a crossing sequence of that run. The *crossing number* of a transducer is the maximal crossing number of any of its normalized runs – note that this value is bounded by $2|Q| - 1$.

Intercepted factors

An *interval* of positions has the form $I = [x_1, x_2]$, with $x_1 < x_2$. We say that an interval $I = [x_1, x_2]$ *contains* (resp., *strongly contains*) another interval $I' = [x'_1, x'_2]$ if $x_1 \leq x'_1 \leq x'_2 \leq x_2$ (resp., $x_1 < x'_1 \leq x'_2 < x_2$). We say that a factor of a run ρ is *intercepted* by an interval $I = [x_1, x_2]$ if it is maximal among the factors of ρ that visit only positions in I and that never make a reversal (recall that reversals in sweeping transducers can only occur at the extremities of the input word).

It is easy to see that distinct factors intercepted by the same interval I visit disjoint sets of locations. This means that a factor intercepted by I can be uniquely identified by specifying a location ℓ in it, e.g., the first or the last one. Accordingly, we will denote by $\rho \mid I/\ell$ the factor intercepted by I that visits the location ℓ (if this factor does not exist, we simply let $\rho \mid I/\ell = \varepsilon$).

A *loop* of a run ρ is an interval $L = [x_1, x_2]$ such that the crossing sequences at positions x_1 and x_2 are equal, that is, $\rho|x_1 = \rho|x_2$. Loops can be used to pump parts of runs, as explained below.

Pumping

Given a loop $L = [x_1, x_2]$ of a run ρ on u and a number $m \in \mathbb{N}$, we can replicate m times the factor $u[x_1, x_2]$ of the input and simultaneously, on the run, we replicate m times the loop L. Formally, with $\beta_0, \dots, \beta_h$ denoting the factors intercepted by L, we define the run obtained by replicating L as a sequence of the form

$$\mathsf{pump}^m_L(\rho) \;=\; \underbrace{\alpha_0\, \boldsymbol{\beta_0^m}\, \gamma_0}_{\text{forward}}\; \underbrace{\gamma_1\, \boldsymbol{\beta_1^m}\, \alpha_1}_{\text{backward}}\; \underbrace{\alpha_2\, \boldsymbol{\beta_2^m}\, \gamma_2}_{\text{forward}}\; \cdots\; \underbrace{\alpha_{y_{\max}}\, \boldsymbol{\beta_h^m}\, \gamma_{y_{\max}}}_{\text{forward}} \tag{1}$$

where $h < 2|Q| - 1$ is the maximum level visited by ρ, and α_y (resp., β_y, γ_y) is the factor of ρ at level y that is intercepted by the interval $[0, x_1]$ (resp., $L = [x_1, x_2]$, $[x_2, |u|]$). Note

that each $\alpha_y\,\beta_y\,\gamma_y$ (resp., $\gamma_y\,\beta_y\,\alpha_y$) is a maximal factor of the run ρ that is forward-oriented (resp., backward-oriented). We also define

$$\mathsf{pump}_L^m(u) \;=\; u[1,x_1]\cdot\big(\boldsymbol{u[x_1+1,x_2]}\big)^{\boldsymbol{m}}\cdot u[x_2+1,|u|]$$

and we observe that $\mathsf{pump}_L^m(\rho)$ is a valid run on $\mathsf{pump}_L^m(u)$. We also remark that the above definition of pumped run works only for sweeping transducers, as for arbitrary two-way transducers we would need to take into account the possible reversals within a loop L and combine the intercepted factors in a more complex way.

3 Decompositions of one-way definable runs

The problem we consider in this paper is the *one-way definability* of functional, sweeping transducers: given a transducer $\mathcal{S}$, we ask if there exists an equivalent one-way transducer $\mathcal{T}$, namely, one such that $\mathcal{T}(u)=\mathcal{S}(u)$ for all $u\in\Sigma^\star$. In the answer is "yes", then we also want to compute an equivalent one-way transducer. Of course, there are sweeping transducers $\mathcal{S}$, like $\mathcal{S}(u)=u\cdot u$, that are not equivalent to any one-way transducer (assuming that the alphabet Σ is not unary).

Before introducing some technical concepts, let us consider an example that highlights the main idea of the proof. Fix a regular language R (not containing the empty word) and consider the transduction that maps a word on the mirror of the rightmost maximal factor belonging to R. That is, $f(u\,v\,w)=\mathsf{mirror}(v)$ whenever (1) $v\in R$, (2) there is no $v'\in R$ such that v' is prefix of vw, and (3) w has no factor in R. It can be easily seen that f can be realized by a two-way transducer, but not by a one-way transducer. However, f *can* be realized by a one-way transducer for particular regular languages R, like the periodic language $R=(ab)^+$: we simply guess v and output $\mathsf{mirror}(v)\in(ba)^+$ from left to right, then we check w. This example shows that periodicities play an important role in deciding whether a given transduction can be realized by a one-way transducer.

We introduce in the following some notations and concepts that will help us to state a sufficient condition for the one-way definability of sweeping transducers. For simplicity, we fix for the remaining of the paper a functional, sweeping transducer $\mathcal{S}$ as input. We first introduce some constants: $h_{\mathcal{S}}$ is the maximum number of levels visited by the normalized runs of $\mathcal{S}$, $c_{\mathcal{S}}$ is the maximum number of symbols produced by a single transition of $\mathcal{S}$, and $e_{\mathcal{S}}=c_{\mathcal{S}}\cdot|Q|^{2|Q|}$, where Q is the state space of $\mathcal{S}$. The constant $e_{\mathcal{S}}$ will be used to bound the lengths or the periods of certain parts of the output produced by $\mathcal{S}$, and is related to the number of crossing sequences of $\mathcal{S}$ (see also Lemma 6 in Section 4).

A word v is said to have *period* p if $v\in w^*\,w'$ for some word w of length p and some prefix w' of w. For example, $v=abc\,abc\,ab$ has period $p=3$. Similarly, we say that v is *almost periodic with bound* p if $v=w_0\,w_1\,w_2$ for some words w_0,w_2 of length at most p and some non-empty word w_1 of period at most p.

We will also need to identify sub-sequences of a run ρ of $\mathcal{S}$ that are induced by particular sets of locations. Recall that $\rho[\ell_1,\ell_2]$ denotes the factor of ρ delimited by two locations ℓ_1 and ℓ_2. Similarly, we denote by $\rho\mid Z$ the sub-sequence of ρ induced by a set Z of locations – note that Z does not need to be an interval and, even though $\rho\mid Z$ might not be a valid run of $\mathcal{S}$, we can still refer to the number of transitions and the size of the output.

▶ **Definition 2.** Let ρ be a run of $\mathcal{S}$ on u. We define two types of pairs of locations of ρ:

- A *floor* is a pair of locations (ℓ_1,ℓ_2) such that $\ell_1\le\ell_2$ are on the same *even* level.

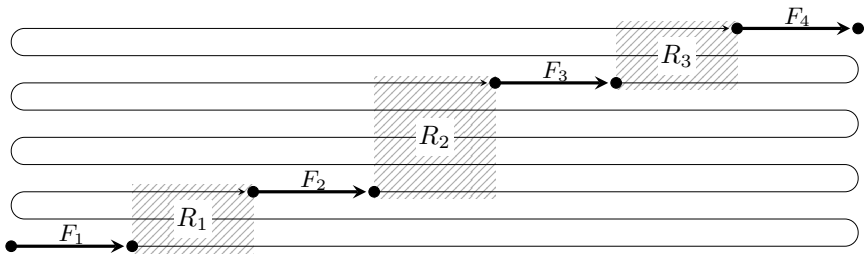
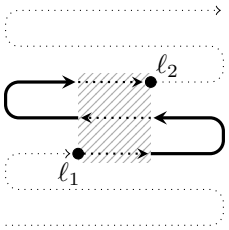


Figure 1 Decomposition of a run into floors and ramps.

- A *ramp* is a pair of locations (ℓ_1, ℓ_2), with $\ell_1 = (x_1, y_1)$ and $\ell_2 = (x_2, y_2)$, such that (i) $x_1 \leq x_2$, (ii) $y_1 < y_2$, (iii) both y_1 and y_2 are even, (iv) the output produced by $\rho[\ell_1, \ell_2]$ has length at most $(y_2 - y_1 + 1) \cdot e_{\mathcal{S}}$ or it is almost periodic with bound $2 \cdot e_{\mathcal{S}}$, and (v) the output produced by the sub-sequence $\rho \mid Z$, where $Z = [\ell_1, \ell_2] \setminus [x_1, x_2] \times [y_1, y_2]$, has length at most $2(y_2 - y_1) \cdot e_{\mathcal{S}}$.

Before discussing how the above definitions are used, we give some intuition. The simplest concept is that of floor, which is essentially a forward-oriented factor of a run. Ramps connect consecutive floors. An important constraint in the definition of a ramp (ℓ_1, ℓ_2) is that the output of $\rho[\ell_1, \ell_2]$ is bounded or almost periodic with small bound. We will see later how this constraint eases the production of the output of $\rho[\ell_1, \ell_2]$ by a one-way transducer. The last constraint on a ramp (ℓ_1, ℓ_2) bounds the length of the output produced by the sub-sequence $\rho \mid Z$, where $Z = [\ell_1, \ell_2] \setminus [x_1, x_2] \times [y_1, y_2]$. As shown by the figure to the right, this sub-sequence (represented by bold arrows) can be obtained from the factor $\rho[\ell_1, \ell_2]$ by removing the factors intercepted by $[x_1, x_2]$ (represented by the hatched area). The constraint is used for those parts of the run that are not covered by floors or ramps. In particular, it guarantees that the size of the output *above* each floor is bounded by $2h_{\mathcal{S}} \cdot e_{\mathcal{S}}$.



The general idea for turning $\mathcal{S}$ into an equivalent one-way transducer will be to guess (and check) a decomposition of the run of $\mathcal{S}$ into factors that are floors or ramps.

▶ **Definition 3.** A *decomposition* of a run ρ is a factorization into floors and ramps.

Figure 1 gives an example of a decomposition. Note that, thanks to Definition 2, the number of symbols produced outside the segments $F_1, F_2, \ldots$ and the rectangles $R_1, R_2, \ldots$ is small (indeed, bounded by $2h_{\mathcal{S}} \cdot e_{\mathcal{S}}$); so most of the output is produced inside these segments and rectangles. We can now state our main result:

▶ **Theorem 4.** *A sweeping functional transducer $\mathcal{S}$ is one-way definable if and only if every input word has some successful run of $\mathcal{S}$ that admits a decomposition.*
Moreover, we can construct from $\mathcal{S}$ a one-way transducer $\mathcal{T}$ that maps u to v whenever there is a successful run of $\mathcal{S}$ on u that outputs v and admits a decomposition. The construction of $\mathcal{T}$ takes doubly exponential time in $|\mathcal{S}|$.
In particular, $\mathcal{S}$ is one-way definable if and only if $\mathsf{dom}(\mathcal{T}) = \mathsf{dom}(\mathcal{S})$. The latter condition can be tested in polynomial space in $|\mathcal{S}|$ and $|\mathcal{T}|$, so in doubly exponential space in $|\mathcal{S}|$.

The first claim of the theorem gives the main characterization, namely, it shows that the existence of decompositions of successful runs, for all possible inputs in the domain of $\mathcal{S}$, is a

sufficient and *necessary* condition for the transduction to be one-way definable. The second claim shows a property that is slightly more general than sufficiency: it allows to compute a one-way transducer $\mathcal{T}$ that is somehow the "best one-way approximation" of $\mathcal{S}$, in the sense that the transduction computed by $\mathcal{T}$ is always contained in the transduction computed by $\mathcal{S}$ and it is equal when $\mathcal{S}$ is one-way definable. The last claim deals with the *effectiveness* of the characterization, showing that one-way definability is decidable in 2ExpSpace. For the sake of presentation, we divide the proof of the theorem into two parts. The first part, given below, deals with the *sufficiency* and the *effectiveness* of the characterization (i.e. the second and third claims of the theorem). The second part, which is the most technical one and is deferred to Section 4, deals with the *necessary* part of the characterization.

Proof of Theorem 4 (sufficiency and effectiveness). We build from $\mathcal{S}$ a one-way transducer $\mathcal{T}$ that simulates all successful runs of $\mathcal{S}$ that admit a decomposition. Consider one such run ρ. We begin by observing that a decomposition of ρ can be described by a sequence of locations $\ell_0 < \ell_1 < \ldots < \ell_t$, where $\ell_0 = (0,0)$, $\ell_t = (x_{\mathsf{max}}, y_{\mathsf{max}})$, and, for all $0 \leq i < t$, (ℓ_i, ℓ_{i+1}) is a floor or a ramp. In particular, the one-way transducer $\mathcal{T}$ will guess the crossing sequences of ρ, together with a sequence of locations $(\ell_i)_{i \leq t}$, which are intended to represent a decomposition of ρ. Below, we show how to check that the guessed sequence of locations represents a valid decomposition, and how to produce the corresponding output.

Traversing the floors of the decomposition does not pose particular problems, as these are forward-oriented factors of the run ρ, which can be directly simulated by $\mathcal{T}$ without reversing the head. Of course, we need to store the bounded output on the levels above the floor, and check that the output on the levels below the floor matches some stored output words. The interesting case happens when $\mathcal{T}$ simulates a ramp (ℓ_i, ℓ_{i+1}), with $\ell_i = (x_i, y_i)$ and $\ell_{i+1} = (x_{i+1}, y_{i+1})$. First of all, it is easy for $\mathcal{T}$ to verify the first three conditions of the definition of ramp, namely, that $x_i \leq x_{i+1}$, $y_i < y_{i+1}$, and both y_i and y_{i+1} are even. Checking the remaining conditions is more difficult and requires storing some words for a total length that does not exceed $8h_{\mathcal{S}} \cdot e_{\mathcal{S}}$ – in particular, this explains the doubly exponential blowup of the state space of $\mathcal{T}$. More precisely, at the beginning of the computation, $\mathcal{T}$ guesses, for each ramp (ℓ_i, ℓ_{i+1}), with $\ell_i = (x_i, y_i)$ and $\ell_{i+1} = (x_{i+1}, y_{i+1})$:

- a word v_i that has length at most $(y_{i+1} - y_i + 1) \cdot e_{\mathcal{S}}$ or is almost periodic with bound $2 \cdot e_{\mathcal{S}}$ (in the latter case, in fact, the word is described by a prefix, a suffix, and a repeating pattern, each one of length at most $2 \cdot e_{\mathcal{S}}$),
- some words $\overleftarrow{v}_y$ and $\overrightarrow{v}_y$, that is, two words for each level $y \in \{y_i + 1, \ldots, y_{i+1}\}$, whose lengths sum up to at most $2h_{\mathcal{S}} \cdot e_{\mathcal{S}}$.

The idea is that each word v_i represents the output produced by the factor $\rho[\ell_i, \ell_{i+1}]$ (this output is bounded or is almost periodic, thanks to the fourth condition of the definition of ramp). Note that, by construction, there can be at most $h_{\mathcal{S}}$ ramps in the decomposition, and hence the sum of the lengths of the words used to represent the v_i's does not exceed $6 \cdot h_{\mathcal{S}} \cdot e_{\mathcal{S}}$. Similarly, each word $\overleftarrow{v}_y$ (resp., $\overrightarrow{v}_y$) represents the output produced by the factor of the run ρ that is at level y and to the left of the position x_i (resp., right of x_{i+1}), where i is the unique index such that $y_i < y \leq y_{i+1}$ (resp., $y_i \leq y < y_{i+1}$). The total length of these words is at most $2h_{\mathcal{S}} \cdot e_{\mathcal{S}}$. Overall, the sum of the lengths of all the words guessed by $\mathcal{T}$ never exceeds $8h_{\mathcal{S}} \cdot e_{\mathcal{S}}$.

Using the words v_i, $\overleftarrow{v}_y$, $\overrightarrow{v}_y$ and some additional pointers, the transducer $\mathcal{T}$ can verify that the guessed ramps satisfy the required conditions and that the decomposition is thus valid. In the same way, the words v_i, $\overleftarrow{v}_y$, $\overrightarrow{v}_y$ can be used to produce the output $\mathsf{out}(\rho[\ell_i, \ell_{i+1}])$ associated with each ramp (ℓ_i, ℓ_{i+1}). For this, it is sufficient to visit the positions of the

ramp (ℓ_i, ℓ_{i+1}) and, at the same time, fetch blocks of symbols of appropriate length in the word v_i, so as to eventually match the length of the desired output $\mathsf{out}(\rho[\ell_i, \ell_{i+1}])$ – note that this requires taking into account also the words $\overleftarrow{v}_y$ and $\overrightarrow{v}_y$.

We just described informally a one-way transducer $\mathcal{T}$ that simulates any run ρ of $\mathcal{S}$ that admits a decomposition. The size of $\mathcal{T}$ is doubly exponential in the size of $\mathcal{S}$ and this proves the second claim of the theorem.

Assuming that the existence of decompositions of successful runs is also a necessary condition for the one-way definability of $\mathcal{S}$ (this will be proved in the next section), we can easily derive from the above constructions a 2ExpSpace decision procedure for testing one-way definability. More precisely, given a sweeping transducer $\mathcal{S}$, one constructs $\mathcal{T}$ as above in doubly exponential time, and then tests whether $\mathsf{dom}(\mathcal{S}) \subseteq \mathsf{dom}(\mathcal{T})$. The latter problem can be seen as a containment problem between a two-way non-deterministic finite-state automaton $\mathcal{A}$ and a one-way non-deterministic finite-state automaton $\mathcal{B}$. Using standard constructions, one can turn $\mathcal{A}$ into an equivalent one-way non-deterministic finite-state automaton $\mathcal{A}'$ (which is exponential in $\mathcal{S}$), and finally decide the containment $\mathcal{A}' \subseteq \mathcal{B}$ in polynomial space in $\mathcal{A}'$ and $\mathcal{B}$, that is, in doubly exponential space in $\mathcal{S}$.

◀

4 Characterizing one-way definability

In this section we prove the harder direction of the first claim of Theorem 4: if a sweeping functional transducer is one-way definable, then every accepted input has a successful run that can be decomposed into floors and ramps.

We begin by identifying some phenomena that prevent a transducer to be one-way definable. A first example is the mirror transduction, where a large number of symbols need to be generated from right to left. Another example is the doubling transduction $\mathcal{S}(u) = u \cdot u$. Here, we have an *inversion*, namely large parts of the input must be generated before other large parts that are located to their left. We give a formal definition of inversions below.

Fix a successful run ρ of $\mathcal{S}$ and consider a loop $L = [x_1, x_2]$ of ρ. A location ℓ_1 of ρ is called *entry point of* L if ℓ_1 is the first location of the factor intercepted by L at level y, for some y. Similarly, a location ℓ_2 is called an *exit point of* L if ℓ_2 is the last location of the factor intercepted by L at level k, for some k. Note that ℓ_1 belongs to $\{x_1\} \times (2\mathbb{N}) \cup \{x_2\} \times (2\mathbb{N}+1)$, and ℓ_2 belongs to $\{x_2\} \times (2\mathbb{N}) \cup \{x_1\} \times (2\mathbb{N}+1)$. Finally, we say that an intercepted factor $\rho \mid I/\ell$ is *captured* by a loop L if I contains L (possibly, $I = L$) and the subfactor intercepted by L on the same level as ℓ, has non-empty output.

▶ **Definition 5.** An *inversion* of the run ρ is a pair of locations ℓ_1 and ℓ_2 for which there exist two loops $L_1 = (x_1, x_1')$ and $L_2 = (x_2, x_2')$ such that:

- ℓ_1 is an entry point of L_1 and ℓ_2 is an exit point of L_2,
- $\ell_1 < \ell_2$ and $x_1 \geq x_2$ (namely, ℓ_2 strictly follows ℓ_1 along the run, but the left endpoint of L_2 precedes the left endpoint of L_1),
- for both $i = 1$ and $i = 2$, the intercepted factor $\rho \mid L_i/\ell_i$ is captured by the loop L_i, but it is not captured by any other loop strongly contained in L_i.

We say that the above loops L_1 and L_2 are *witnessing* the inversion (ℓ_1, ℓ_2).

The left-hand side of Figure 2 gives an example of an inversion, where the entry point ℓ_1 of L_1 and the exit point ℓ_2 of are represented by white circles.

The first lemma (proved in the appendix) can be used to bound the lengths of the outputs produced by the factors $\rho \mid L_1/\ell_1$ and $\rho \mid L_2/\ell_2$, where ℓ_1, ℓ_2, L_1, L_2 are as in Definition 5:

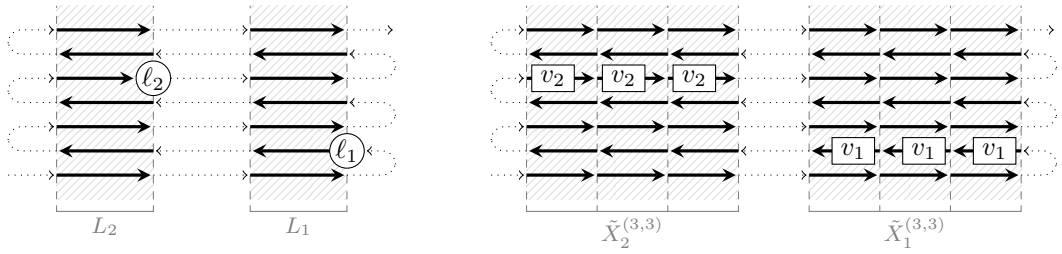


Figure 2 To the left: an entry point ℓ_1 of L_1 and an exit point ℓ_2 of L_2 forming an inversion. To the right: the run obtained by pumping the loops L_1 and L_2.

▶ **Lemma 6.** *If an intercepted factor $\rho \mid I/\ell$ is not captured by any loop L strongly contained in I, then the length of its output is at most $e_{\mathcal{S}}$.*
In particular, for every inversion (ℓ_1, ℓ_2) witnessed by some loops L_1 and L_2, we have $1 \leq \big|\mathsf{out}(\rho \mid L_i/\ell_i)\big| \leq e_{\mathcal{S}}$, for both $i = 1$ and $i = 2$.

The next proposition gives the crucial property for characterizing one-way definability, as it shows that the transducer $\mathcal{S}$ is one-way definable only if for every inversion (ℓ_1, ℓ_2), the output of $\rho[\ell_1, \ell_2]$ is periodic.

▶ **Proposition 7.** Suppose that the sweeping transducer $\mathcal{S}$ is one-way definable. Then, for all inversions (ℓ_1, ℓ_2) of the run ρ witnessed by loops L_1, L_2 and for both $i = 1$ and $i = 2$, the output of $\rho[\ell_1, \ell_2]$ has period $\big|\mathsf{out}(\rho \mid L_i/\ell_i)\big|$, hence, in particular, at most $e_{\mathcal{S}}$.

A key ingredient for the proof of the above proposition is Fine and Wilf's theorem [9]. In short, this theorem says that, whenever two periodic words w_1, w_2 share a sufficiently long factor, then they have the same periods. Below, we state a slightly stronger variant of Fine and Wilf's theorem, which contains an additional claim that shows how to align a common factor of the two words w_1, w_2 so as to form a third word containing a prefix of w_1 and a suffix of w_2. The additional claim will be exploited in the proof of Proposition 7 and Lemma 11.

▶ **Lemma 8** (Fine and Wilf)**.** *If w_1 is a word with period p_1, w_2 is a word with period p_2, and w_1 and w_2 have a common factor of length at least $p_1 + p_2 - \gcd(p_1, p_2)$, then w_1 and w_2 have also period $\gcd(p_1, p_2)$. If in addition we have $w_1 = u_1\, w\, v_1$, $w_2 = u_2\, w\, v_2$, and $|w| \geq \gcd(p_1, p_2)$, then $w_3 = u_1\, w\, v_2$ has also period $\gcd(p_1, p_2)$.*

Proof of Proposition 7. Let $L_1 = (x_1, x_1')$, $L_2 = (x_2, x_2')$ be the loops witnessing the inversion (ℓ_1, ℓ_2). Note that the two loops L_1 and L_2 might not be ordered exactly as shown in Figure 2. In fact, two cases can arise: either $x_2 < x_2' \leq x_1 < x_1'$ (that is, L_1 and L_2 are disjoint and L_1 is to the right of L_2) or $x_2 \leq x_1 < x_2' \leq x_1'$ (that is, L_1 overlaps to the right with L_2).

We begin by pumping the loops L_1 and L_2 (see the right-hand side of Figure 2). Formally, for all positive numbers m_1, m_2, we define

$$\rho^{(m_1,m_2)} \;=\; \mathsf{pump}_{L_2}^{m_2}(\mathsf{pump}_{L_1}^{m_1}(\rho)) \qquad \text{and} \qquad u^{(m_1,m_2)} \;=\; \mathsf{pump}_{L_2}^{m_2}(\mathsf{pump}_{L_1}^{m_1}(u))\ .$$

We identify the positions of $u^{(m_1,m_2)}$ that mark the endpoints of the copies of the loops L_1 and L_2 in the pumped run $\rho^{(m_1,m_2)}$. Because L_2 precedes L_1 with respect to the ordering of positions, it is easier to define first the set of endpoints of the copies of L_2:

$$\tilde{X}_2^{(m_1,m_2)} \;=\; \{x_2 + i\Delta_2 \;\mid\; 0 \leq i \leq m_2\} \qquad\qquad \text{where} \quad \Delta_2 = x_2' - x_2\ .$$

The set of endpoints of the copies of L_1 is defined as

$$\tilde{X}_1^{(m_1,m_2)} \;=\; \{x_1 + i\Delta_1 + m_2\Delta_2 \;\mid\; 0 \le i \le m_1\} \qquad \text{where } \Delta_1 = x_1' - x_1 \;.$$

We then exploit the hypothesis that $\mathcal{S}$ is one-way definable and assume that the one-way transducer $\mathcal{T}$ is equivalent to $\mathcal{S}$. In particular, $\mathcal{T}$ produces the same output as $\mathcal{S}$ on every input $u^{(m_1,m_2)}$. Let $\sigma^{(m_1,m_2)}$ be a successful run of $\mathcal{T}$ on $u^{(m_1,m_2)}$. Since $\mathcal{T}$ has finitely many states, we can find a large enough number $h > 0$ and two positions $\tilde{x}_1 < \tilde{x}_1' \in \tilde{X}_1^{(h,h)}$ such that the crossing sequences of $\sigma^{(h,h)}$ at $\tilde{x}_1$ and $\tilde{x}_1'$ are the same. Similarly, we can find two positions $\tilde{x}_2 < \tilde{x}_2' \in \tilde{X}_2^{(h,h)}$ such that the crossing sequences of $\sigma^{(h,h)}$ at $\tilde{x}_2$ and $\tilde{x}_2'$ are the same. This means that $\tilde{L}_1 = (\tilde{x}_1, \tilde{x}_1')$ and $\tilde{L}_2 = (\tilde{x}_2, \tilde{x}_2')$ can be equally seen as loops of $\rho^{(h,h)}$ or as loops of $\sigma^{(h,h)}$. In particular, there are constants $k_1, k_2 > 0$, $0 \le h_1 < k_1$, and $0 \le h_2 < k_2$ such that, for all positive numbers m_1, m_2:

$$u^{(k_1\cdot m_1+h_1,\, k_2\cdot m_2+h_2)} \;=\; \mathsf{pump}_{\tilde{L}_2}^{m_2}(\mathsf{pump}_{\tilde{L}_1}^{m_1}(u^{(h,h)}))$$

and the above word has a successful run in $\mathcal{T}$ of the form $\mathsf{pump}_{\tilde{L}_2}^{m_2}(\mathsf{pump}_{\tilde{L}_1}^{m_1}(\sigma^{(h,h)}))$. Consider the outputs v_1 and v_2 produced by the intercepted factors $\rho \mid L_1/\ell_1$ and $\rho \mid L_2/\ell_2$, respectively. By Lemma 6, both v_1 and v_2 are non-empty. Moreover, by definition of pumped run, the output produced by $\rho^{(k_1\cdot m_1+h_1,\, k_2\cdot m_2+h_2)}$ contains $k_1 \cdot m_1 + h_1$ consecutive occurrences of v_1 followed by $k_2 \cdot m_2 + h_2$ consecutive occurrences of v_2 (see again the right-hand side Figure 2). Formally, we can write

$$\mathsf{out}(\rho^{(k_1\cdot m_1+h_1,\, k_2\cdot m_2+h_2)}) \;=\; v_0(m_1,m_2)\cdot \boldsymbol{v_1^{k_1\cdot m_1+h_1}} \cdot v_3(m_1,m_2) \cdot \boldsymbol{v_2^{k_2\cdot m_2+h_2}} \cdot v_4(m_1,m_2) \quad (2)$$

for some words $v_0(m_1,m_2)$, $v_3(m_1,m_2)$, and $v_4(m_1,m_2)$ that may depend on m_1 and m_2 (we highlighted in bold the repeated occurrences of v_1 and v_2 and we observe that v_1 precedes v_2).

In a similar way, because the same output is also produced by the the one-way transducer $\mathcal{T}$, i.e. by the run $\mathsf{pump}_{\tilde{L}_2}^{m_2}(\mathsf{pump}_{\tilde{L}_1}^{m_1}(\sigma^{(h,h)}))$, and because the loop L_2 precedes the loop L_1 according to the natural ordering of positions, we have

$$\mathsf{out}(\rho^{(k_1\cdot m_1+h_1,\, k_2\cdot m_2+h_2)}) \;=\; w_0 \cdot \boldsymbol{w_2^{m_2}} \cdot w_3 \cdot \boldsymbol{w_1^{m_1}} \cdot w_4 \quad (3)$$

where w_1 (resp., w_2) is the output produced by the unique factor of $\sigma^{(h,h)}$ intercepted by $\tilde{L}_1$ (resp., $\tilde{L}_2$), and w_0, w_3, w_4 are the remaining parts of the output. Note that, differently from the previous equation, here the first repetition is produced during the loop $\tilde{L}_2$ and the remaining parts w_0, w_3, w_4 do not depend on m_1, m_2. We now consider the factor

$$v(m_1,m_2) \;=\; \boldsymbol{v_1^{k_1\cdot m_1+h_1}} \cdot v_3(m_1,m_2) \cdot \boldsymbol{v_2^{k_2\cdot m_2+h_2}}$$

of the output produced by $\mathcal{S}$. The following claim shows that this factor is periodic, with a small period that only depends on $\mathcal{S}$ (in particular, it does not depend on any of the indices $h, k_1, k_2, h_1, h_2, m_1, m_2$).

▶ **Claim.** *For all numbers $m_1, m_2 > 0$, the word $v(m_1,m_2) = \boldsymbol{v_1^{k_1\cdot m_1+h_1}} \cdot v_3(m_1,m_2) \cdot \boldsymbol{v_2^{k_2\cdot m_2+h_2}}$ is periodic with period* $\gcd(|v_1|, |v_2|)$.

The idea for the proof of the above claim, detailed in the appendix, is to let m_1 and m_2 grow independently. We exploit Equations (2) and (3) to show that $\boldsymbol{v_1^{k_1\cdot m_1+h_1}} \cdot v_3(m_1,m_2) \cdot \boldsymbol{v_2}$ has period $\gcd(|v_1|, |w_1|)$, and that $\boldsymbol{v_1} \cdot v_3(m_1,m_2) \cdot \boldsymbol{v_2^{k_2\cdot m_2+h_2}}$ has period $\gcd(|v_2|, |w_2|)$. A last application of Fine and Wilf's theorem (Lemma 8) gives the desired periodicity.

Recall that we aim at proving the periodicity of the output $\mathsf{out}(\rho[\ell_1, \ell_2])$ of the *original* run ρ of $\mathcal{S}$ between the locations ℓ_1 and ℓ_2. The previous arguments, however, concern the outputs $v(m_1, m_2)$, which are produced by factors of the *pumped* runs $\rho^{(k_1 \cdot m_1 + h_1, k_2 \cdot m_2 + h_2)}$. By Equation (1) in Section 2, $\mathsf{out}(\rho[\ell_1, \ell_2])$ can be obtained from any $v(m_1, m_2)$ by deleting some occurrences of non-empty words produced by factors intercepted by L_1 or L_2. More precisely, the words that need to be deleted in $v(m_1, m_2)$ to obtain $\mathsf{out}(\rho[\ell_1, \ell_2])$ are non-empty and of the form $\mathsf{out}(\rho \mid L_i/\ell_i')$, for some $i \in \{1, 2\}$ and some location ℓ_i' such that $\ell_1 \leq \ell_i' \leq \ell_2$. Let us denote by $v_1', \ldots, v_m'$ these words. Note that, as m_1 and m_2 get larger, $v(m_1, m_2)$ contains arbitrarily long repetitions of each word v_i', and hence long factors of period $|v_i'|$, for $i = 1, \ldots, m$. Thus, by applying Lemma 8, we get that $v(m_1, m_2)$ has period $p = \gcd(|v_1|, |v_2|, |v_1'|, \ldots, |v_m'|)$.

Towards a conclusion, we know that $\mathsf{out}(\rho[\ell_1, \ell_2])$ is obtained from $v(m_1, m_2)$ by removing occurrences of the words $v_1', \ldots, v_m'$ whose lengths are multiple of the period p of $v(m_1, m_2)$. This implies that $\mathsf{out}(\rho[\ell_1, \ell_2])$ is also periodic with period p, which divides $\left|\mathsf{out}(\rho \mid L_i/\ell_i)\right|$ for both $i = 1$ and $i = 2$. ◀

Recall that the proof of the remaining part of Theorem 4 (necessity of the condition characterizing one-way definability) amounts at constructing a decomposition of the successful run ρ under the assumption that $\mathcal{S}$ is one-way definable. We begin to construct a decomposition of ρ by identifying some ramps in it. Intuitively, such ramps are obtained by considering the classes of a suitable equivalence relation:

▶ **Definition 9.** Let $\leftrightharpoons$ be the relation that pairs every two locations ℓ, ℓ' along the run ρ whenever there is an inversion (ℓ_1, ℓ_2) of ρ such that $\ell_1 \leq \ell, \ell' \leq \ell_2$, namely, whenever ℓ and ℓ' occur within the same inversion. Let $\leftrightharpoons^\star$ be the reflexive and transitive closure of $\leftrightharpoons$.

It is easy to see that every equivalence class of $\leftrightharpoons^\star$ is a convex subset with respect to the natural ordering of locations of ρ. The following lemma shows that every *non-singleton* equivalence class of $\leftrightharpoons^\star$ is a union of a series of inversions that are two-by-two overlapping.

▶ **Lemma 10.** *If two locations $\ell \leq \ell'$ of ρ belong to the same* non-singleton *equivalence class of $\leftrightharpoons^\star$, then there is a sequence of locations $\ell_1 \leq \ell_3 \leq \ell_4 \leq \ldots \leq \ell_{n-3} \leq \ell_{n-2} \leq \ell_n$, for some even number $n \geq 4$, such that*

- *$\ell_1 \leq \ell \leq \ell_4$ and $\ell_{n-3} \leq \ell' \leq \ell_n$,*
- *(ℓ_1, ℓ_4), (ℓ_3, ℓ_6), (ℓ_5, ℓ_8), $\ldots$, (ℓ_{n-5}, ℓ_{n-2}), (ℓ_{n-3}, ℓ_n) are inversions.*

The next result uses Lemma 6, Proposition 7, and Lemma 10 to show that the output produced inside a $\leftrightharpoons^\star$-equivalence class is also periodic with small period, provided that $\mathcal{S}$ is one-way definable.

▶ **Lemma 11.** *If $\mathcal{S}$ is one-way definable and $\ell \leq \ell'$ are two locations of the run ρ such that $\ell \leftrightharpoons^\star \ell'$, then the output $\mathsf{out}(\rho[\ell, \ell'])$ produced between ℓ and ℓ' has period at most $e_{\mathcal{S}}$.*

Below, we introduce some "bounding boxes" of non-singleton $\leftrightharpoons^\star$-equivalence classes. Intuitively, these bounding boxes are the smallest possible rectangles that start and end at some even levels and that cover all the locations forming an inversion inside a non-singleton $\leftrightharpoons^\star$-equivalence class. Subsequently, in Lemma 13 we show that these bounding boxes can be given the status of ramps in a suitable decomposition of ρ.

▶ **Definition 12.** Let K be a non-singleton $\leftrightharpoons^\star$-equivalence class and let H be the subset of K that contains all the locations $\ell, \ell' \in K$ forming an inversion (ℓ, ℓ').
We define $[K]$ to be the pair of locations $\ell_1 = (x_1, y_1)$ and $\ell_2 = (x_2, y_2)$ such that

- x_1 (resp., x_2) is the position of the leftmost (resp., rightmost) location $\ell \in H$,
- y_1 (resp., y_2) is the highest (resp., lowest) even level such that $y_1 \leq y$ (resp., $y_2 \geq y$) for all locations $\ell = (x, y) \in H$.

▶ **Lemma 13.** *If K is a non-singleton $\leftrightharpoons^\star$-equivalence class, then $[K]$ is a ramp.*

For the sake of brevity, we call $\leftrightharpoons^\star$-*ramp* any ramp of the form $[K]$, where K is a non-singleton $\leftrightharpoons^\star$-equivalence class. The results obtained so far imply that every location of the run ρ covered by an inversion is also covered by a $\leftrightharpoons^\star$-ramp. To complete the decomposition of ρ, we need to consider the locations that are not strictly covered by $\leftrightharpoons^\star$-ramps, formally, the set $B = \{\ell \mid \nexists (\ell_1, \ell_2) \leftrightharpoons^\star\text{-ramp s.t. } \ell_1 < \ell < \ell_2\}$. We equip B with the natural ordering of locations induced by ρ. We now consider some maximal convex subset C of B. Note that the left/right endpoint of C coincides with the first/last location of the run ρ or with the right/left endpoint of some $\leftrightharpoons^\star$-ramp. Below, we show how to decompose the sub-run $\rho \mid C$ into a series of floors and ramps. After this, we will be able to get a full decomposition of ρ by interleaving the $\leftrightharpoons^\star$-ramps that we defined above with the floors and the ramps that decompose each sub-run $\rho \mid C$.

Let D_C be the set of locations $\ell = (x, y)$ of C such that there is some loop $L = [x, x']$, with $x' \geq x$, whose intercepted factor $\rho \mid L/\ell$ lies entirely inside C and produces non-empty output. We remark that the set D_C may be non-empty. To see this, one can imagine the existence of two consecutive $\leftrightharpoons^\star$-ramps (e.g. R_1 and R_2 in Figure 1) and a loop between them that produces non-empty output (e.g. the factor F_2). In a more general scenario, one can find several loops between two consecutive $\leftrightharpoons^\star$-ramps that span across different levels. We can observe however that all the locations in D_C are on even levels. Indeed, if this were not the case for some $\ell = (x, y) \in D_C$, then we could select a minimal loop $L = [x_1, x_2]$ such that $x_1 \leq x \leq x_2$ and $\mathsf{out}(\rho \mid L/\ell) \neq \varepsilon$. Since y is odd, $\ell_1 = (x_2, y)$ is an entry point of L and $\ell_2 = (x_1, y)$ is an exit point of L, and hence (ℓ_1, ℓ_2) is an inversion. Since $\ell_1 \leq \ell \leq \ell_2$ and all inversions are covered by $\leftrightharpoons^\star$-ramps, there is a $\leftrightharpoons^\star$-ramp (ℓ'_1, ℓ'_2) such that $\ell'_1 \leq \ell \leq \ell'_2$. However, as ℓ'_1 and ℓ'_2 are at even levels, ℓ must be different from these two locations, and this would contradict the definition of D_C. Using similar arguments, one can also show that the locations in D_C are arranged along a "rising diagonal", from lower left to upper right.

The above properties suggest that the locations in D_C identify some floors and ramps that form a decomposition of $\rho \mid C$. The following lemma shows that this is indeed the case, namely, that any two consecutive locations in D_C form a floor or a ramp.

▶ **Lemma 14.** *Let $\ell_1 = (x_1, y_1)$ and $\ell_2 = (x_2, y_2)$ be two consecutive locations of D_C. Then, $x_1 \leq x_2$ and $y_1 \leq y_2$ and the pair (ℓ_1, ℓ_2) is a floor or a ramp, depending on whether $y_1 = y_2$ or $y_1 < y_2$.*

We have just shown how to construct a decomposition of the entire run ρ, assuming that the sweeping transducer $\mathcal{S}$ is one-way definable. This completes the proof of the only-if direction of the first claim of Theorem 4.

5 Lower bound and undecidability

We show now that the doubly exponential blow-up in size stated by Theorem 4, cannot be avoided.

▶ **Proposition 15.** There is a family $(f_n)_n$ of functions from $\{0,1\}^*$ to $\{0,1\}^*$ such that:

- f_n can be computed by a sweeping transducer of size quadratic in n,

- f_n can be computed by a one-way transducer,
- any one-way transducer computing f_n has at least $2^{2^{n-1}}$ states.

We exhibit such a family of function by defining the domain of f_n to be the set of words of the form

$$a_0\ \mathsf{bin}_0\ a_1\ \mathsf{bin}_1\ a_2\ \dots\ a_{2^n-1}\ \mathsf{bin}_{2^n-1}\ a_{2^n}$$

where $a_i \in \{0,1\}$ for all $i \in \{0,\dots,2^n\}$. On those words, we define f_n as follows:

$$f_n\big(a_0\ \mathsf{bin}_0\ a_1\ \mathsf{bin}_1\ a_2\ \dots\ a_{2^n-1}\ \mathsf{bin}_{2^n-1}\ a_{2^n}\big) \;=\; w \cdot w \quad \text{where } w \;=\; a_0\, a_1\, \dots\, a_{2^n}\ .$$

We conclude the section by showing that the one-way definability problem becomes undecidable for relations computed by sweeping *non-functional* transducers. Note that ε-transitions are needed in order to capture the class of one-way definable relations. On the other hand, for one-way definable functions, ε-transitions can be excluded.

▶ **Proposition 16.** The problem of testing whether a sweeping *non-functional* transducer is one-way definable is undecidable.

6 Conclusion

In this paper we proposed a new algorithm that decides whether a sweeping transducer is equivalent to a one-way transducer. Our decision algorithm works in doubly exponential space and produces one-way transducers of doubly exponential size. The latter bound is shown to be optimal. An open question is whether the decision problem has lower complexity if we do not build the one-way transducer.

The main open question is whether our algorithm can be extended to two-way functional transducers that are not necessarily sweeping. We conjecture that this is the case and that a similar characterization based on decompositions of runs into floors and ramps can be obtained. The main difficulty is that (de)pumping loops is more complicated because of permutations.

One-way definability is also a special case of the following open problem: given an integer k and a two-way transducer, decide if there is an equivalent k-crossing two-way transducer. Finally, note that the problem that we considered here becomes much simpler in the origin semantics of [3]: there, the output of a transducer also includes the origin of each symbol, i.e., the input position where the symbol was generated. In the origin semantics, the one-way definability problem is PSpace-complete, and an equivalent one-way transducer has exponential size.

References

1 R. Alur and P. Cerný. Streaming transducers for algorithmic verification of single-pass list-processing programs. In *Proceedings of the 38th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, pages 599–610. ACM, 2011.

2 R. Alur, A. Durand-Gasselin, and A. Trivedi. From monadic second-order definable string transformations to transducers. In *28th Annual ACM/IEEE Symposium on Logic in Computer Science, (LICS)*, pages 458–467. IEEE Computer Society, 2013.

3 M. Bojańczyk. Transducers with origin information. In *Proceedings of the 41st International Colloquium, ICALP 2014*, LNCS, pages 26–37. Springer, 2014.

4 O. Carton and L. Dartois. Aperiodic two-way transducers and FO-transductions. In *24th EACSL Annual Conference on Computer Science Logic (CSL)*, LIPIcs, pages 160–174. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015.

5 J. Engelfriet and H. J. Hoogeboom. MSO definable string transductions and two-way finite-state transducers. *ACM Trans. Comput. Logic*, 2(2):216–254, Apr. 2001.

6 E. Filiot, O. Gauwin, P. Reynier, and F. Servais. From two-way to one-way finite state transducers. In *28th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 468–477. IEEE Computer Society, 2013.

7 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)*, volume 29 of *LIPIcs*, pages 147–159, 2014.

8 O. H. Ibarra. The unsolvability of the equivalence problem for epsilon-free NGSM's with unary input (output) alphabet and applications. *SIAM J. Comput.*, 7(4):524–532, 1978.

9 M. Lothaire. *Combinatorics on words*. Cambridge University Press, 1997.

10 M. O. Rabin and D. Scott. Finite automata and their decision problems. *IBM Journal of Research and Development*, 3(2):114–125, 1959.

11 J. C. Shepherdson. The reduction of two-way automata to one-way automata. *IBM Journal of Research and Development*, 3(2):198–200, 1959.

A Appendix

▶ **Proposition 1.** Functionality of two-way transducers can be decided in polynomial space. This problem is PSPACE-hard even for sweeping transducers.

Proof. Showing that the problem is PSPACE-hard for sweeping transducers is easy: we reduce from the emptiness problem of the intersection of NFAs.

Given n NFAs $\mathcal{A}_1, \dots, \mathcal{A}_n$ we build a sweeping transducer $\mathcal{S}$ which simulates the automaton $\mathcal{A}_i$ on its i-th pass. At the end of the i-th pass, if the simulation of $\mathcal{A}_i$ is in an accepting state, we choose non-deterministically either to stop the computation and output the number i, or to continue testing if the input is in $L(\mathcal{A}_{i+1})$. That way, a pair (u, i) is in the transduction defined by $\mathcal{S}$ if and only if $\mathcal{A}_i$ accepts the input u. This yields exactly to the desired property of $\mathcal{S}$: it is functional if and only if $\bigcap_{i \leq n} L(\mathcal{A}_i) = \emptyset$.

To show that the functionality problem is in PSPACE for two-way transducers, we use a reduction similar to [1], to the reachability of 1-counter machines. The latter problem belongs to NLOGSPACE and we will obtain an exponential-sized 1-counter machine, that can be simulated on-the-fly. Altogether this yields PSPACE complexity. We recall that $N = 2|Q|$ is the maximal crossing number of a normalized run in a functional transducer. We first show the following claim:

▶ **Claim.** If $\mathcal{S}$ is not functional, then either there exists (1) a successful run with crossing number at most $2N$ and non-empty output on some repetition in a crossing sequence, or (2) two normalized runs on the same input, each with crossing number at most N, and with different outputs.

Proof. Assume there is a successful run with non-empty output on some repetition in a crossing sequence. That is, the crossing sequence contains two locations with the same state and both on even/odd level, such that the output on the factor run delimited by these locations is non-empty. We can first remove from this run all repetitions that produce empty outputs. Then, we can also remove the repetitions with non-empty outputs, obtaining other successful runs, until there remains only one non-empty repetition. The run produced is an instance of case (1).

On the other hand, if all the repetitions in the crossing sequences of any successful run produce an empty output, we can find a pair of runs satisfying (2): We know by non functionality that there is pair (u, v) of outputs produced by different runs on the same input, such that $u \neq v$. As we know there is no repetition producing a non-empty output, we can remove the repetitions of the runs of u and v and obtain new successful runs producing the same u and v. As those runs do not have any repetition in their crossing sequence, they have a crossing number $\leq N$.

◀

To determine non-functionality, the 1-counter machine will guess between the two cases of the lemma: in (1) it will guess a run of $\mathcal{S}$ of crossing number smaller than $2N$, and in (2) it will guess two runs of $\mathcal{S}$ on the same input of crossing number smaller than N. In the second case, we define u and v to be the two outputs of the runs, and $\mathcal{S}$ decides if it will check $|u| \neq |v|$ or the existence of a position i such that $u_i \neq v_i$.

In each case, $\mathcal{M}$ guesses two locations in the runs and marks locations on the crossing sequences that will help to identify certain factors of runs.

In the first case $\mathcal{M}$ guesses two locations in the run at the same crossing sequence, that have the same state, and the same movement of the input head, and $\mathcal{M}$ marks all

locations between those two locations. It checks then that the output produced by the factor containing marked locations is non-empty.

In the second case, when verifying that $|u| \neq |v|$, it guesses one location in the second run, and marks the locations before it in the corresponding crossing sequences. The counter is incremented by the number of letters produced in the first run, and decremented by those produced by marked locations of the second. The guessed location represents the moment in the second run where it has produced an output longer than $|u|$.

In the second case, when checking the existence of a position such that $u_i \neq v_i$, it guesses one location in each run, and marks the locations before those in the corresponding crossing sequences. The counter is incremented (resp. decremented) by the number of letters produced by marked locations of the first (resp. second) run. The machine also checks that the letters produced at the guessed locations are different, and the counter's value is 0 at the end of the run. This guarantees that the letters were produced at the same place. ◀

▶ **Lemma 6.** If an intercepted factor $\rho \mid I/\ell$ is not captured by any loop L strongly contained in I, then the length of its output is at most $e_{\mathcal{S}}$.
In particular, for every inversion (ℓ_1, ℓ_2) witnessed by some loops L_1 and L_2, we have $1 \leq \left|\mathsf{out}(\rho \mid L_i/\ell_i)\right| \leq e_{\mathcal{S}}$, for both $i = 1$ and $i = 2$.

Proof. Let $I = [x_1, x_2]$ be an interval, $\ell = (x, y)$ a location, with $x_1 \leq x \leq x_2$, and $\alpha = \rho \mid I/\ell$ the intercepted factor. Suppose that there is no loop L strongly contained in I that captures α. We have to show that $|\mathsf{out}(\alpha)| \leq e_{\mathcal{S}}$.

We first claim that the number of transitions of the factor α that produce non-empty output does not exceed the number of distinct crossing sequences of the form $\rho|x$, with $x_1 \leq x < x_2$. Indeed, if this were not the case, then there would exist two locations $\ell' = (x', y)$ and $\ell'' = (x'', y)$, with $x_1 < x' < x'' < x_2$, such that

- the crossing sequences at x' and x'' are the same,
- depending on whether y is even or odd, ℓ' and ℓ'' are either targets or sources of two distinct transitions that produce non-empty output.

From the above, we derive the existence of a loop $L' = [x', x'']$ that is strongly contained in I and intercepts a factor on level y producing non-empty output, i.e. a contradiction to our hypotheses on α.

Now, recall that the run ρ is normalized, and hence there are at most $|Q|^{2|Q|}$ distinct crossing sequences of the form $\rho|x$, with $x_1 \leq x < x_2$. Together with the previous claim, this implies that the factor $\alpha = \rho \mid I/\ell$ contains at most $|Q|^{2|Q|}$ transitions that produce non-empty output. To prove the first result stated in the lemma it is thus sufficient to recall that each of those transitions can produce at most $c_{\mathcal{S}}$ symbols and $e_{\mathcal{S}} = c_{\mathcal{S}} \cdot |Q|^{2|Q|}$.

We conclude the proof by showing the last claim of the lemma. Consider an inversion (ℓ_1, ℓ_2) witnessed by some loop L_1 and L_2. By definition, we have that, for both $i = 1$ and $i = 2$, the intercepted factor $\rho \mid L_i/\ell_i$ is captured by L_i, but is not captured by any loop strongly contained in L_i. Together with the previous claims, this implies that $1 \leq \left|\mathsf{out}(\rho \mid L_i/\ell_i)\right| \leq e_{\mathcal{S}}$. ◀

▶ **Claim (in the proof of Proposition 7).** *For all numbers $m_1, m_2 > 0$, the word $v(m_1, m_2) = \boldsymbol{v_1^{k_1 \cdot m_1 + h_1}} \cdot v_3(m_1, m_2) \cdot \boldsymbol{v_2^{k_2 \cdot m_2 + h_2}}$ is periodic with period* $\gcd(|v_1|, |v_2|)$.

Proof. The proof of this claim is based on Equations (2) and (3), which we recall here:

$$\mathsf{out}(\rho^{(k_1\cdot m_1+h_1,k_2\cdot m_2+h_2)}) \;=\; v_0(m_1,m_2)\cdot \boldsymbol{v_1}^{\boldsymbol{k_1\cdot m_1+h_1}}\cdot v_3(m_1,m_2)\cdot \boldsymbol{v_2}^{\boldsymbol{k_2\cdot m_2+h_2}}\cdot v_4(m_1,m_2) \tag{2}$$

$$\mathsf{out}(\rho^{(k_1\cdot m_1+h_1,k_2\cdot m_2+h_2)}) \;=\; w_0\cdot \boldsymbol{w_2}^{\boldsymbol{m_2}}\cdot w_3\cdot \boldsymbol{w_1}^{\boldsymbol{m_1}}\cdot w_4 \tag{3}$$

We are going to use the above equations for varying parameters m_1, m_2. We first fix m_1 large enough so that the prefix $v_0(m_1,m_2)\cdot \boldsymbol{v_1}^{\boldsymbol{k_1\cdot m_1+h_1}}$ is longer than $|w_0|+|v_1|$. Then, by letting m_2 grow arbitrarily large and by using the above equations, we get that the periodic word $\boldsymbol{w_2}^{\boldsymbol{m_2}}$ covers an arbitrarily long prefix of $\boldsymbol{v_1}\cdot v_3(m_1,m_2)\cdot \boldsymbol{v_2}^{\boldsymbol{k_2\cdot m_2+h_2}}$, and in particular an arbitrarily long infix of $\boldsymbol{v_2}^{\boldsymbol{k_2\cdot m_2+h_2}}$. Hence, by Fine-Wilf's Theorem (first claim of Lemma 8), the words $\boldsymbol{w_2}^{\boldsymbol{m_2}}$ and $\boldsymbol{v_2}^{\boldsymbol{k_2\cdot m_2+h_2}}$ are periodic with period $p_2=\gcd(|v_2|,|w_2|)$. Moreover, because an arbitrary long prefix of $\boldsymbol{v_1}\cdot v_3^{(m_1,m_2)}\cdot \boldsymbol{v_2}^{\boldsymbol{k_2\cdot m_2+h_2}}$ is covered by $\boldsymbol{w_2}^{\boldsymbol{m_2}}$, we have that the word $\boldsymbol{v_1}\cdot v_3(m_1,m_2)\cdot \boldsymbol{v_2}^{\boldsymbol{k_2\cdot m_2+h_2}}$ is also periodic with period $p_2=\gcd(|v_2|,|w_2|)$. Symmetrically, by fixing m_2 large enough and letting m_1 grow, we get that the word $\boldsymbol{v_1}^{\boldsymbol{h_1\cdot m_1+h_1}}\cdot v_3(m_1,m_2)\cdot \boldsymbol{v_2}$ is periodic with period $p_1=\gcd(|v_1|,|w_1|)$. We can summarize the previous results as follows:

$$v(m_1,m_2) \;=\; \underbrace{\boldsymbol{v_1}^{\boldsymbol{k_1\cdot m_1+h_1-1}}\cdot \overbrace{\boldsymbol{v_1}\;\cdot\; v_3(m_1,m_2)\;\cdot\; \boldsymbol{v_2}}^{\text{period } p_2}\cdot \boldsymbol{v_2}^{\boldsymbol{k_2\cdot m_2+h_2-1}}}_{\text{period } p_1}\,.$$

Finally, we observe that the words $\boldsymbol{v_1}^{\boldsymbol{h_1\cdot m_1+h_1}}\cdot v_3(m_1,m_2)\cdot \boldsymbol{v_2}$ and $\boldsymbol{v_1}\cdot v_3(m_1,m_2)\cdot \boldsymbol{v_2}^{\boldsymbol{k_2\cdot m_2+h_2}}$ share the factor $\boldsymbol{v_1}\cdot v_3(m_1,m_2)\cdot \boldsymbol{v_2}$ of length at least $|v_1|+|v_2|$. Hence, by applying the second claim of Lemma 8, we conclude that the word $v(m_1,m_2)$ is periodic with period $\gcd(|v_1|,|v_2|)$. ◀

▶ **Lemma 10.** If two locations $\ell \le \ell'$ of ρ belong to the same *non-singleton* equivalence class of $\hookrightarrow^\star$, then there is a sequence of locations $\ell_1 \le \ell_3 \le \ell_4 \le \ldots \le \ell_{n-3} \le \ell_{n-2} \le \ell_n$, for some even number $n \ge 4$, such that

- $\ell_1 \le \ell \le \ell_4$ and $\ell_{n-3} \le \ell' \le \ell_n$,
- (ℓ_1,ℓ_4), (ℓ_3,ℓ_6), (ℓ_5,ℓ_8), $\ldots$, (ℓ_{n-5},ℓ_{n-2}), (ℓ_{n-3},ℓ_n) are inversions.

For the sake of presentation, in Figure 3 we give an intuitive account of the structure of the series of inversions claimed in the lemma: the thick arrows represent the transitions within the equivalence class and the hatched areas mark the endpoints of the overlapping inversions (for example, (ℓ_1,ℓ_4) overlaps with (ℓ_3,ℓ_6)).

Proof. We will say that an inversion (ℓ_1,ℓ_2) *covers* a location ℓ when $\ell_1 \le \ell \le \ell_2$. We prove the lemma by induction on the distance between the two locations ℓ and ℓ'. The basic case is when $\ell = \ell'$. Because ℓ belongs to a non-singleton $\hookrightarrow^\star$-equivalence class, we know that there is an inversion (ℓ_1,ℓ_4) that covers ℓ, namely, such that $\ell_1 \le \ell \le \ell_4$. Thus, the claim of the lemma holds by simply letting $n=4$.

To prove the induction step, we consider two locations ℓ,ℓ' satisfying the hypothesis of the lemma and being at distance $t>0$, and we assume that the claim holds for analogous pairs of locations at distance strictly less than t. As before, because ℓ belongs to a non-singleton $\hookrightarrow^\star$-equivalence class, we know that there is an inversion (ℓ_1,ℓ_4) that covers ℓ. Without loss of generality we can assume that ℓ_4 is the greatest location that forms an

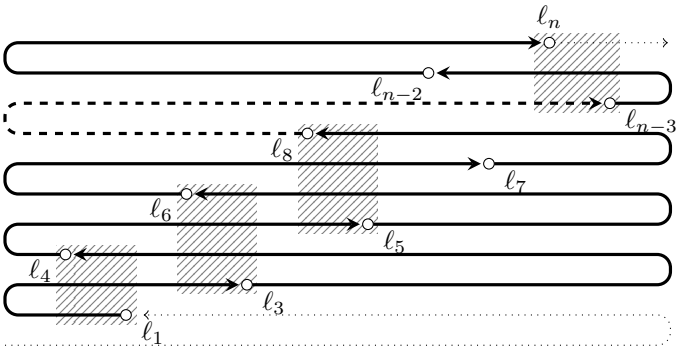


Figure 3 A $\rightharpoonup^\star$-equivalence class seen as a series of overlapping inversions.

inversion covering ℓ. In particular, we have that $\ell < \ell_4$, as otherwise we would have $\ell = \ell_4$: the latter equality however is impossible, as all the locations in the equivalence class of ℓ would then occur before ℓ, thus contradicting $\ell' > \ell$ (recall that ℓ and ℓ' are by hypothesis in the same equivalence class of $\rightharpoonup^\star$).

If ℓ' is also covered by (ℓ_1, ℓ_4), then we are done, namely, the claim holds for $n = 4$. Otherwise, we let $\ell'' = \ell_4$ and we observe that $\ell'' < \ell'$ is a pair of locations satisfying the hypothesis of the lemma and being at distance strictly less than t. By applying the inductive hypothesis, we derive the existence of a series of inversions of the form (ℓ_3, ℓ_6), (ℓ_5, ℓ_8), $\ldots$, (ℓ_{n-5}, ℓ_{n-2}), (ℓ_{n-3}, ℓ_n), for some $n \geq 4$ and some locations $\ell_3 \leq \ell_5 \leq \ldots \leq \ell_{n-3} \leq \ell_{n-2} \leq \ell_n$. By prepending the pair (ℓ_1, ℓ_4) to this sequence we get the desired claim. ◀

▶ **Lemma 11.** If $\mathcal{S}$ is one-way definable and $\ell \leq \ell'$ are two locations of the run ρ such that $\ell \rightharpoonup^\star \ell'$, then the output $\mathsf{out}(\rho[\ell, \ell'])$ produced between ℓ and ℓ' has period at most $e_{\mathcal{S}}$.

Proof. The claim for $\ell = \ell'$ holds trivially, so for the rest of the proof we focus only on the case $\ell < \ell'$. By Lemma 10, we know that there is a series of inversions

$$(\ell_1, \ell_4) \quad (\ell_3, \ell_6) \quad (\ell_5, \ell_8) \quad \ldots \quad (\ell_{n-5}, \ell_{n-2}) \quad (\ell_{n-3}, \ell_n)$$

for some locations $\ell_1 \leq \ell_3 \leq \ell_4 \leq \ldots \leq \ell_{n-3} \leq \ell_{n-2} \leq \ell_n$, with $\ell_1 \leq \ell_4$ and $\ell_{n-3} \leq \ell' \leq \ell_n$.

For each of the above inversions (ℓ_{i-3}, ℓ_i), we denote by L_{i-3} and L_i the two witnessing loops and by p_{i-3} and p_i the lengths of the outputs produced by the intercepted factors $\rho \mid L_{i-3}/\ell_{i-3}$ and $\rho \mid L_i/\ell_i$, respectively. Recall that, by Lemma 6, $e_{\mathcal{S}}$ is an upper bound to the lengths $p_1, p_4, p_3, p_6, \ldots, p_{n-5}, p_{n-2}, p_{n-3}, p_n$. Therefore, in order to show that $\mathsf{out}\big(\rho[\ell, \ell']\big)$ is periodic with period at most $e_{\mathcal{S}}$, it suffices to prove the following claim by induction on i:

▶ **Claim.** For all $i \in \{4, 6, \ldots, n-2, n\}$, the output $\mathsf{out}(\rho[\ell_1, \ell_i])$ produced between ℓ_1 and ℓ_i is periodic with period $\gcd(p_{i-3}, p_i)$.

The base case $i = 4$ follows immediately from Proposition 7, since (ℓ_1, ℓ_4) is an inversion. For the inductive step, we assume that the claim holds for $i \leq n-2$ and we prove it for $i+2$. First of all, we decompose our word as follows:

$$\mathsf{out}\big(\rho[\ell_1, \ell_{i+2}]\big) \quad = \quad \mathsf{out}\big(\rho[\ell_1, \ell_{i-1}]\big)\ \mathsf{out}\big(\rho[\ell_{i-1}, \ell_i]\big)\ \mathsf{out}\big(\rho[\ell_i, \ell_{i+2}]\big)\ .$$

We then observe that, by the inductive hypothesis, the output produced between ℓ_1 and ℓ_i has period $\gcd(p_{i-3}, p_i)$. Similarly, because (ℓ_{i-1}, ℓ_{i+2}) is an inversion, we know from Proposition 7 that the output produced between ℓ_{i-1} and ℓ_{i+2} has period $\gcd(p_{i-1}, p_{i+2})$.

Now, we focus on the output $\mathsf{out}\big(\rho[\ell_{i-1}, \ell_i]\big)$. We first verify that (ℓ_{i-1}, ℓ_i) is an inversion. Indeed, we have $\ell_{i-1} \leq \ell_i$. Moreover, because (ℓ_{i-1}, ℓ_{i+2}) and (ℓ_{i-3}, ℓ_i) are inversions, we know that ℓ_{i-1} is an entry point of L_{i-1} and ℓ_i is an exit point of L_i. In particular, we have $\ell_{i-1} < \ell_i$ and the x-coordinate of ℓ_{i-1} is greater than that of ℓ_i. For the same reasons, the intercepted factor $\rho \mid L_{i-1}/\ell_{i-1}$ is captured by L_{i-1}, but is not captured by any other loop strongly contained in L_{i-1}. Analogous properties holds for the intercepted factor $\rho \mid L_i/\ell_i$ and the loop L_i. This proves that (ℓ_{i-1}, ℓ_i) is an inversion. Then, by applying Proposition 7 to this inversion, we get that the the output produced between ℓ_{i-1} and ℓ_i has period $\gcd(p_{i-1}, p_i)$.

We can summarize the previous results as follows:

$$\mathsf{out}\big(\rho[\ell_1, \ell_{i+2}]\big) \;=\; \underbrace{\mathsf{out}\big(\rho[\ell_1, \ell_{i-1}]\big)\; \overbrace{\mathsf{out}\big(\rho[\ell_{i-1}, \ell_i]\big)}^{\text{period } p_{i-1} \text{ and } p_i}\; \mathsf{out}\big(\rho[\ell_i, \ell_{i+2}]\big)}_{\text{period } p_{i-3} \text{ and } p_i} \;\overbrace{\qquad\qquad}^{\text{period } p_{i-1} \text{ and } p_{i+2}} .$$

To head towards the conclusion, we observe that the word $\mathsf{out}\big(\rho[\ell_{i-1}, \ell_i]\big)$ ends with the output produced by the intercepted factor $\rho \mid L_i/\ell_i$, and hence it has length at least p_i. Moreover, the same word occurs as a factor of $\mathsf{out}\big(\rho[\ell_1, \ell_{i-1}]\big)\,\mathsf{out}\big(\rho[\ell_{i-1}, \ell_i]\big)$. Hence, by Lemma 8, the two words $\mathsf{out}\big(\rho[\ell_{i-1}, \ell_i]\big)$ and $\mathsf{out}\big(\rho[\ell_1, \ell_{i-1}]\big)\,\mathsf{out}\big(\rho[\ell_{i-1}, \ell_i]\big)$ have period $\gcd(p_{i-3}, p_{i-1}, p_i)$. In a similar way, by observing that $\mathsf{out}\big(\rho[\ell_{i-1}, \ell_i]\big)$ has length at least p_{i-1} and occurs as a factor of $\mathsf{out}\big(\rho[\ell_{i-1}, \ell_i]\big)\,\mathsf{out}\big(\rho[\ell_i, \ell_{i+2}]\big)$, we derive that $\mathsf{out}\big(\rho[\ell_{i-1}, \ell_i]\big)$ and $\mathsf{out}\big(\rho[\ell_{i-1}, \ell_i]\big)\,\mathsf{out}\big(\rho[\ell_i, \ell_{i+2}]\big)$ have period $\gcd(p_{i-3}, p_{i-1}, p_i, p_{i+2})$. A third application of Lemma 8 allows us to conclude that the entire word $\mathsf{out}\big(\rho[\ell_1, \ell_{i+2}]\big)$ has period $\gcd(p_{i-3}, p_{i-1}, p_i, p_{i+2})$, which clearly divides $\gcd(p_{i-1}, p_{i+2})$. This proves the inductive step of the claim.

Finally, we recall that the lemma follows from the above claim and from Lemma 6 by letting $i = n$ (note that $\mathsf{out}\big(\rho[\ell, \ell']\big)$ is a factor of $\mathsf{out}\big(\rho[\ell_1, \ell_n]\big)$). ◀

▶ **Lemma 13.** If K is a non-singleton $\leftrightarrows^\star$-equivalence class, then $[K]$ is a ramp.

Proof. Let H be the subset of K that contains all the locations $\ell, \ell' \in K$ forming an inversion (ℓ, ℓ'). Recall from Definition 12 that we have $[K] = (\ell_1, \ell_2)$, where $\ell_1 = (x_1, y_1)$, $\ell_2 = (x_2, y_2)$, $x_1 = \min\{x \mid (x, y) \in H\}$, $x_2 = \mathsf{max}\{x \mid (x, y) \in H\}$, and y_1 (resp., y_2) is the even number among $\min\{y \mid (x, y) \in H\}$ and $\min\{y \mid (x, y) \in H\} - 1$ (resp., among $\mathsf{max}\{y \mid (x, y) \in H\}$ and $\mathsf{max}\{y \mid (x, y) \in H\} + 1$). We verify that $[K] = (\ell_1, \ell_2)$ satisfies all the conditions of the definition of ramp.

Clearly, we have $x_1 \leq x_2$, $y_1 < y_2$, and both y_1 and y_2 are even. We now prove that the output produce by the sub-sequence $\rho \mid Z$, where $Z = [\ell_1, \ell_2] \setminus [x_1, x_2] \times [y_1, y_2]$, has length at most $2(y_2 - y_1) \cdot e_{\mathcal{S}}$.

Suppose, by way of contradiction, that this is not the case, and that $\big|\mathsf{out}(\rho \mid Z)\big| > 2(y_2 - y_1) \cdot e_{\mathcal{S}}$. Since the set Z spans across at most $y_2 - y_1$ levels and does not cover the locations inside the "bounding box" $[x_1, x_2] \times [y_1, y_2]$, we could find a factor α that produces an output longer than $e_{\mathcal{S}}$, that lies on a single level y', and either to the left of x_1 or to the right of x_2.

Suppose that this output is on the left (the right case is symmetric), and that α is a factor on some level $y' \in \{y_1 + 1, \ldots, y_2\}$ that is intercepted by the interval $I = [1, x_1]$ and that produces an output of length strictly greater than $e_{\mathcal{S}}$. By the contrapositive of Lemma

6, the factor α is captured by some loop $L' = [x'_1, x'_2]$ that is *strongly contained* in $I = [1, x_1]$, namely, such that $1 \leq x'_1 < x'_2 < x_1$. Let $\ell' = (x', y')$ be the exit point of L', where $x' = x'_1$ or $x' = x'_2$ depending on whether y' is odd or even. Recall that, by the definition of $[K]$, there is also a location $\ell = (x, y)$ that strictly precedes ℓ' along the run and that belongs to H. This implies that ℓ forms an inversion with another location and, without loss of generality, we can assume that ℓ is the first element of this inversion. Moreover, recall that $x'_1 \leq x' \leq x'_2 < x_1$ and $x_1 \leq x \leq x_2$, namely, ℓ' is *strictly to the left* of ℓ according to the ordering of the x-coordinates. This shows that (ℓ, ℓ') is also an inversion, and hence ℓ' also belongs to H. However, this contradicts the definition of x_1 as the minimum of the x-coordinates of the locations in H. A similar contradiction can be obtained in the case where the factor that produces an output of length strictly greater than $e_{\mathcal{S}}$ lies at the right of x_2.

By the above arguments, we know that $\big|\mathsf{out}(\rho \mid Z)\big| \leq 2(y_2 - y_1) \cdot e_{\mathcal{S}}$, where $Z = [\ell_1, \ell_2] \setminus [x_1, x_2] \times [y_1, y_2]$, namely, that the fourth condition of the definition of ramp is satisfied.

It remains to verify the five condition, that the output produced by $\rho[\ell_1, \ell_2]$ is almost periodic with bound $2 \cdot e_{\mathcal{S}}$. Let ℓ'_1 and ℓ'_2 be the first and the last locations of the $\hookrightarrow^\star$-equivalence class K (note that $\ell_1 \leq \ell'_1 \leq \ell'_2 \leq \ell_2$). Recall that Lemma 11 already shows that the output produced between ℓ'_1 and ℓ'_2 is periodic with period at most $e_{\mathcal{S}}$. Thus, we just need to show that the prefix $\mathsf{out}(\rho[\ell_1, \ell'_1])$ and the suffix $\mathsf{out}(\rho[\ell'_2, \ell_2])$ have length at most $2 \cdot e_{\mathcal{S}}$.

Suppose that the length of $\mathsf{out}(\rho[\ell_1, \ell'_1])$ exceeds $2 \cdot e_{\mathcal{S}}$. By the definition of $[K]$, the two locations ℓ_1 and ℓ'_1 would be either on the same level, i.e. y_1, or on adjacent levels, i.e. y_1 and $y_1 + 1$. In the following, we show that none of these cases can happen, thus reaching a contradiction from the assumption $\big|\mathsf{out}(\rho[\ell_1, \ell'_1])\big| > 2 \cdot e_{\mathcal{S}}$. If ℓ_1 were on the same level as ℓ'_1, then clearly the factor $\mathsf{out}(\rho[\ell_1, \ell'_1])$ would lie on a single level and to the right of x_1, and would produce an output longer than $e_{\mathcal{S}}$. Then, by using the contrapositive of Lemma 6, we could find a loop $L' = [x'_1, x'_2]$ strongly contained in the interval $I = [x_1, n]$, where n is the rightmost position of the input, that captures the factor $\mathsf{out}(\rho[\ell_1, \ell'_1])$ with non-empty output. We recall that ℓ'_1 is the first location of the $\hookrightarrow^\star$-equivalence class K, and hence there is an inversion (ℓ'_1, ℓ''), for some location ℓ'' that follows ℓ'_1 along the run. We then define $\ell = (x'_1, y_1)$ and we observe that this location is an entry point of L' and it strictly precedes ℓ''. We thus get that (ℓ, ℓ'') is also an inversion. This, however, is a contradiction because the inversion (ℓ, ℓ'') intersects the $\hookrightarrow^\star$-equivalence class K, without being contained in it. Let us now consider the second case, where ℓ_1 is on level y_1 and ℓ'_1 is on level $y_1 + 1$. Since $\rho[\ell_1, \ell'_1]$ spans across two levels and produces an output longer than $2e_{\mathcal{S}}$, there is a factor α of $\mathsf{out}(\rho[\ell_1, \ell'_1])$ that lies entirely on a single level – either y_1 or $y_1 + 1$ – and to the right of x_1, and produces an output longer than $e_{\mathcal{S}}$. Then, by reasoning as in the previous case, we can get a contradiction by finding an inversion (ℓ, ℓ'') that intersects K without being contained in it.

We have just shown that $\big|\mathsf{out}(\rho[\ell_1, \ell'_1])\big| \leq 2 \cdot e_{\mathcal{S}}$. By using symmetric arguments, we can also prove that $\big|\mathsf{out}(\rho[\ell'_2, \ell_2])\big| \leq 2 \cdot e_{\mathcal{S}}$. Finally, we recall that, by Lemma 11, $\big|\mathsf{out}(\rho[\ell'_1, \ell'_2])\big| \leq e_{\mathcal{S}}$. All together, this shows that the output produced between the locations ℓ_1 and ℓ_2 is almost periodic with bound $2 \cdot e_{\mathcal{S}}$, and hence $[K] = (\ell_1, \ell_2)$ is a ramp. ◀

▶ **Lemma 14.** Let $\ell_1 = (x_1, y_1)$ and $\ell_2 = (x_2, y_2)$ be two consecutive locations of D_C. The pair (ℓ_1, ℓ_2) is a floor or a ramp, depending on whether $y_1 = y_2$ or $y_1 < y_2$..

Proof. Recall that all the locations in D_C are on even levels, so the lemma holds in particular for ℓ_1 and ℓ_2 If $y_1 = y_2$, then (ℓ_1, ℓ_2) is clearly a floor. So let us assume that $y_1 \neq y_1$. The fact that $x_1 \leq x_2$ and $y_1 < y_2$ holds follows from the arguments given in part of the body that just preceded the lemma. The first three conditions of the definition of ramp are also satisfied because ℓ_1 and ℓ_2 are consecutive locations in D_C.

Below, we verify that the output produced by the factor $\rho[\ell_1, \ell_2]$ has length at most $(y_2 - y_1 + 1) \cdot e_{\mathcal{S}}$. Note that this will prove both the fourth and the fifth conditions: for $Z = [\ell_1, \ell_2] \setminus [x_1, x_2] \times [y_1, y_2]$, the sub-sequence $\rho \mid Z$ produces an output that is clearly shorter than that of $\rho[\ell_1, \ell_2]$, and in addition we have $(y_2 - y_1 + 1) < 2(y_2 - y_1)$.

Suppose, by way of contradiction, that $\mathsf{out}(\rho[\ell_1, \ell_2]) > (y_2 - y_1 + 1) \cdot e_{\mathcal{S}}$. Since between ℓ_1 and ℓ_2 there are $y_2 - y_1 + 1$ levels, there is a factor α of $\rho[\ell_1, \ell_2]$ that produces an output longer than $e_{\mathcal{S}}$ and that lies on a single level y, for some $y_1 \leq y \leq y_2$. The contrapositive of Lemma 6 implies that the factor α is captured by some loop $L' = [x'_1, x'_2]$ such that $x'_1 > x_1$ if $y = y_1$, and $x'_2 < y_2$ if $y = y_2$. In particular, the location $\ell' = (x'_1, y)$ is an entry point of L and it belongs to D_C. We have just shown that there is $\ell' \in D_C$ such that $\ell_1 < \ell' < \ell_2$. However, this contradicts the hypothesis that ℓ_1 and ℓ_2 were consecutive locations in D_C. We must conclude that the output produced by the infix $\rho[\ell_1, \ell_2]$ has length at most $(y_2 - y_1 + 1) \cdot e_{\mathcal{S}}$, and hence (ℓ_1, ℓ_2) is a ramp. ◀

▶ **Proposition 15.** There is a family $(f_n)_n$ of functions from $\{0,1\}^*$ to $\{0,1\}^*$ such that:

- f_n can be computed by a sweeping transducer of size quadratic in n,
- f_n can be computed by a one-way transducer,
- any one-way transducer computing f_n has at least $2^{2^{n-1}}$ states.

Proof. In this proof we use the standard binary encodings $\mathsf{bin}_{i,n}$ of the natural numbers $i \in \{0, \dots, 2^n - 1\}$ (for simplicity, hereafter we write bin_i in place of $\mathsf{bin}_{i,n}$): $\mathsf{bin}_0 = 0^n$, $\mathsf{bin}_1 = 0^{n-1}1$, …, $\mathsf{bin}_{2^n-1} = 1^n$. Formally, bin_i is the unique word of length n such that $i = \sum_{j=0}^{n-1} \mathsf{bin}_i(n-j) \cdot 2^j$. Note that the least significant bit is in the rightmost position.

Let f_n be the function over the binary alphabet $\{0,1\}$ defined as follows. The domain of f_n is the set of words of the form

$$a_0\ \mathsf{bin}_0\ a_1\ \mathsf{bin}_1\ a_2\ \dots\ a_{2^n-1}\ \mathsf{bin}_{2^n-1}\ a_{2^n}$$

where $a_i \in \{0,1\}$ for all $i \in \{0, \dots, 2^n\}$. The function f_n extracts from the above word the sub-sequence $a_0 \dots a_{2^n}$ and copies it twice to form the output, namely,

$$f_n\big(a_0\ \mathsf{bin}_0\ a_1\ \mathsf{bin}_1\ a_2\ \dots\ a_{2^n-1}\ \mathsf{bin}_{2^n-1}\ a_{2^n}\big) \;=\; w \cdot w \quad \text{where } w \;=\; a_0\, a_1\, \dots\, a_{2^n}\ .$$

We first show that there exists a sweeping transducer $\mathcal{S}_n$ that computes f_n and has size quadratic in n. Then, we prove that there is a one-way transducer computing f_n. Finally, we show that every one-way transducer computing f_n has at least $2^{2^{n-1}}$ states.

Let us describe the sweeping transducer $\mathcal{S}_n$ that computes f_n. A first task of the transducer $\mathcal{S}_n$ is to check that the input contains a sub-sequence of the form $\mathsf{bin}_0\, \mathsf{bin}_1\, \dots\, \mathsf{bin}_{2^n-1}$. This can be done with n left-to-right passes, which are of course interleaved by $n-1$ right-to-left passes. During the j-th left-to-right pass, with $j \in \{1, \dots, n\}$, the transducer checks the j-th bit of each block bin_i, where $i \in \{0, \dots, 2^n - 1\}$. It does so by applying the following rule: if it reads 0 (resp., 1) at position j of bin_i and if all bits after position j in bin_i are set to 1, then it must read 1 (resp., 0) at position j of bin_{i+1}. Dually, if some bits after

position j in bin_i are set to 0, then the bit at position j of bin_i must be the same as the bit at position j of bin_{i+1}. Moreover, the first pass, where $j = 1$, verifies two additional constraints, namely, that $\mathsf{bin}_0 = 0^n$ and that, after reading 1^n ($= \mathsf{bin}_{2^n-1}$), the input contains exactly one letter (this is needed to avoid that the block bin_{2^n-1} is followed by a new block bin_0). The constraints of a single left-to-right pass can be verified with a number of states that is linear in n. As there are n left-to-right passes, all the checks can be performed with a quadratic number of states. The second task of the transducer aims at producing the correct output, namely, two copies of the sub-sequence $a_0\, a_1 \,\ldots\, a_{2^n}$. This can be done easily, for example, during the last two passes. The transducer that we just described is sweeping, input-deterministic, has $\mathcal{O}(n^2)$ states, and computes the function f_n.

Let us now prove the second item, namely that there exists a one-way transducer computing f_n. Consider the transducer that performs the two following tasks. The first task is to store the word $a_0 \ldots a_{2^n}$ (which size only depends on n) in order to output it twice at the end. The second task is to check that the sub-sequence $\mathsf{bin}_0 \ldots \mathsf{bin}_{2^n-1}$ is correct. This can be done by storing bin_{j-1} and bin_j when bin_j is being read and checking that they encode successive integers (this is regular). It requires to store two words of length n in the state. The transducer also checks that bin_0 follows a_0 and that a_{2^n} is the last letter (at that point bin_{2^n-1} is stored in the state). This transducer is one-way and computes f_n.

We now prove the claim in the third item. Consider a one-way transducer $\mathcal{T}_n$ that computes the function f_n and suppose, by way of contradiction, that $\mathcal{T}_n$ has less than $2^{2^{n-1}}$ states. Let $w_1, \ldots, w_N$ be an enumeration of all the words over $\{0,1\}$ of length 2^n+1, where $N = 2^{2^n+1}$. For all $i \leq N$, let ρ_i be a successful run of $\mathcal{T}$ that produces $w_i \cdot w_i$ as output (the corresponding input can be obtained from w_i by inserting bin_j immediately before each position j). We define $\overleftarrow{\rho}_i$ to be the maximal prefix of ρ_i that produces a prefix of w_i as output. We also define (p_i, v_i, q_i) to be the transition that immediately follows $\overleftarrow{\rho}_i$ in the run ρ_i, and $\overrightarrow{\rho}_i$ to be the remaining part of the run. We finally denote by $\overleftarrow{w}_i$ (resp., $\overrightarrow{w}_i$) the output produced by $\overleftarrow{\rho}_i$ (resp., $\overrightarrow{\rho}_i$). To sum up, we have:

$$\begin{array}{rcl} \rho_i & = & \overleftarrow{\rho}_i \; (p_i, a_i, v_i, q_i) \; \overrightarrow{\rho}_i \\ w_i \cdot w_i & = & \overleftarrow{w}_i \quad \cdot \quad v_i \quad \cdot \quad \overrightarrow{w}_i \; . \end{array}$$

Now, consider the transitions (p_i, a_i, v_i, q_i) that we just defined. Note that the corresponding triples (p_i, a_i, q_i) range over the finite set $Q \times \{0,1\} \times Q$, which has size smaller than $2 \cdot \left(2^{2^{n-1}}\right)^2 = 2^{2^n+1} = N$. Thus, there must exist two distinct indices $i, j \leq N$ such that $(p_i, a_i, q_i) = (p_j, a_j, q_j)$.

Since $p_i = p_j$, the sequences $\overleftarrow{\rho}_i$ and $\overleftarrow{\rho}_j$ end with the same state and hence we can replace $\overleftarrow{\rho}_j$ with $\overleftarrow{\rho}_i$ inside the run ρ_j. This results in the successful run $\overleftarrow{\rho}_i \; (p_j, a_j, v_j, q_j) \; \overrightarrow{\rho}_j$ that produces the output $\overleftarrow{w}_i \cdot v_j \cdot \overrightarrow{w}_j$. Moreover, because every output produced by $\mathcal{T}_n$ must be of the form $w_k \cdot w_k$, for some $1 \leq k \leq N$, and because $v_j \cdot \overrightarrow{w}_j$ contains w_j as a suffix, we can write

$$\overleftarrow{w}_i \cdot v_j \cdot \overrightarrow{w}_j \;=\; w_j \cdot w_j \;=\; \overleftarrow{w}_j \cdot v_j \cdot \overrightarrow{w}_j \; .$$

From the above equation we immediately derive $\overleftarrow{w}_i = \overleftarrow{w}_j$.

As concerns the pieces v_i and v_j, we observe the following. By construction, $\overleftarrow{w}_i \cdot v_i$ (resp., $\overleftarrow{w}_j \cdot v_j$) contains w_i (resp., w_j) as a prefix. Thus, since $w_i \neq w_j$ and $\overleftarrow{w}_i = \overleftarrow{w}_j$, we have that $v_i \neq v_j$. We also know that $(p_i, a_i, q_i) = (p_j, a_j, q_j)$. In particular, substituting in the run ρ_i the transition (p_i, a_i, v_i, q_i) with the transition (p_j, a_j, v_j, q_j) would result in a new successful run on the same input $w_i \cdot w_i$ that produces the output $\overleftarrow{w}_i \cdot v_j \cdot \overrightarrow{w}_i$. However, since

the latter output is different from $w_i \cdot w_i$, this is in contradiction with the functionality of $\mathcal{T}_n$. We must conclude that every one-way transducer that computes f_n has at least $2^{2^{n-1}}$ states. ◀

▶ **Proposition 16.** The problem of testing whether a sweeping *non-functional* transducer is one-way definable is undecidable.

Proof. The proof uses some ideas and variants of constructions provided in [8], concerning the proof of undecidability of the equivalence problem for one-way non-functional transducers.

We show a reduction from the Post Correspondence Problem (PCP). A *PCP instance* is described by two finite alphabets Σ and Δ and two morphisms $f, g : \Sigma^* \to \Delta^*$. A *solution* of such an instance is any non-empty word $w \in \Sigma^+$ such that $f(w) = g(w)$. We recall that the problem of testing whether a PCP instance has a solution is undecidable.

Below, we fix a tuple $\tau = (\Sigma, \Delta, f, g)$ describing a PCP instance and we show how to reduce the problem of testing the *non-existence of solutions* of τ to the problem of deciding *one-way definability* of a relation computed by a sweeping transducer. Roughly, the idea is to construct a relation B_τ between words over a suitable alphabet Γ that encodes all the *non-solutions* to the PCP instance τ (this is simpler than encoding solutions because the presence of errors can be easily checked). The goal is to have a relation B_τ that (i) can be computed by a sweeping transducer and (ii) coincides with a trivial one-way definable relation when τ has no solution.

We begin by describing the encodings for the solutions of the PCP instance. We assume that the two alphabets of the PCP instance, Σ and Δ, are disjoint and we use a fresh symbol $\# \notin \Sigma \cup \Delta$. We define the new alphabet $\Gamma = \Sigma \cup \Delta \cup \{\#\}$ that will serve both as input alphabet and as output alphabet for the transduction. We call *encoding* any pair of words over Γ of the form $(w \cdot u, w \cdot v)$, where $w \in \Sigma^+$, $u \in \Delta^*$, and $v \in \{\#\}^*$. We will write the encodings as vectors to improve readability, e.g., as

$$\begin{pmatrix} w \cdot u \\ w \cdot v \end{pmatrix} .$$

We denote by E_τ the set of all encodings and we observe that E_τ is computable by a one-way transducer (note that this transducer needs ε-transitions). We then restrict our attention to the pairs in E_τ that are encodings of valid solutions of the PCP instance. Formally, we call *good encodings* the pairs in E_τ of the form

$$\begin{pmatrix} w \cdot u \\ w \cdot \#^{|u|} \end{pmatrix} \qquad \text{where } u = f(w) = g(w) .$$

All the other pairs in E_τ are called *bad encodings*. Of course, the relation that contains the good encodings is not computable by any transducer. On the other hand, we can show that the complement of this relation w.r.t. E_τ is computable by a sweeping transducer. Let B_τ be the set of all bad encodings. Consider $(w \cdot u, w \cdot \#^m) \in E_\tau$, with $w \in \Sigma^+$, $u \in \Delta^*$, and $m \in \mathbb{N}$, and we observe that this pair belongs to B_τ if and only if one of the following conditions is satisfied:

1. $m < |u|$,
2. $m > |u|$,

3. $u \neq f(w)$,
4. $u \neq g(w)$.

We explain how to construct a sweeping transducer $\mathcal{S}_\tau$ that computes B_τ. Essentially, $\mathcal{S}_\tau$ guesses which of the above conditions holds and processes the input accordingly. More precisely, if $\mathcal{S}_\tau$ guesses that the first condition holds, then it performs a single left-to-right pass, first copying the prefix w to the output and then producing a block of occurrences of the symbol # that is shorter than the suffix u. This task can be easily performed while reading u: it suffices to emit at most one occurrence of # for each position in u, and at the same time guarantee that, for at least one such position, no occurrence of # is emitted. The second condition can be dealt with by a similar strategy: first copy the prefix w, then output a block of # that is longer than the suffix u. To deal with the third condition, the transducer $\mathcal{S}_\tau$ has to perform two left-to-right passes, interleaved by a backward pass that brings the head back to the initial position. During the first left-to-right pass, $\mathcal{S}_\tau$ copies the prefix w to the output. During the second left-to-right pass, it reads again the prefix w, but this time he guesses a factorization of it of the form $w_1\, a\, w_2$. On reading w_1, $\mathcal{S}_\tau$ will output $\#^{|f(w_1)|}$. After reading w_1, $\mathcal{S}_\tau$ will store the symbol a and move to the position where the suffix u begins. From there, it will guess a factorization of u of the form $u_1\, u_2$, check that u_2 does not begin with $f(a)$, and emit one occurrence of # for each position in u_2. Note that the described behaviour does not immediately guarantee that $u \neq f(w)$. Indeed, it may still happen that $u = f(w)$, but in this case the length of u will not match the number m of occurrences of # produced in output. However, this case is already covered by the first and second condition, so the computation is still correct in the sense that it produces only bad encodings. On the other hand, if the number m of occurrences of # produced in output happens to be the same as u, then the computation of $\mathcal{S}_\tau$ guarantees that $u \neq f(w)$. A similar behaviour can be used to deal with the fourth condition.

We have just shown that there is a sweeping non-functional transducer $\mathcal{S}_\tau$ that computes the relation B_τ containing all the bad encodings. Note that, if the PCP instance τ admits no solution, then all encodings are bad, i.e., $B_\tau = E_\tau$, and hence B_τ is one-way definable. It remains to show that when τ has a solution, B_τ is not one-way definable. Suppose that τ has solution $w \in \Sigma^*$ and let $\big(w \cdot u,\ w \cdot \#^{|u|}\big)$ be the corresponding good encoding, where $u = f(w) = g(w)$. Note that every exact repetition of w is also a solution, and hence the pairs $\big(w^n \cdot u^n,\ w^n \cdot \#^{n\cdot|u|}\big)$ are also good encodings, for all $n \geq 1$.

Suppose, by way of contradiction, that there is a one-way transducer $\mathcal{T}$ that computes the relation B_τ. For every $n, m \in \mathbb{N}$, we define the encoding

$$\alpha_{n,m} \;=\; \begin{pmatrix} w^n \cdot u^m \\ w^n \cdot \#^{m\cdot|u|} \end{pmatrix}$$

and we observe that $\alpha_{n,m} \in B_\tau$ if and only if $n \neq m$ (recall that w is the solution of the PCP instance τ and $u = f(w) = g(w)$). Below, we consider bad encodings like the above ones, where the parameter n is supposed to be large enough. Formally, we define the set I of all pairs of indices $(n, m) \in \mathbb{N}^2$ such that (i) $n \neq m$ (this guarantees that $\alpha_{n,m} \in B_\tau$) and (ii) n is larger than the number $|Q|$ of states of $\mathcal{T}$.

We consider some pair $(n, m) \in I$ and we choose a successful run $\rho_{n,m}$ of $\mathcal{T}$ that witnesses the membership of $\alpha_{n,m}$ in B_τ, namely, that reads the input $w^n \cdot u^m$ and produces the output $w^n \cdot \#^{m\cdot|u|}$. We can split the run $\rho_{n,m}$ into a prefix $\overleftarrow{\rho}_{n,m}$ and a suffix $\overrightarrow{\rho}_{n,m}$ in such a way that $\overleftarrow{\rho}_{n,m}$ consumes the prefix w^n and $\overrightarrow{\rho}_{n,m}$ consumes the remaining suffix u^m. Since n is larger than the number of state of $\mathcal{T}$, we can find a factor $\hat{\rho}_{n,m}$ of $\overleftarrow{\rho}_{n,m}$ that starts and ends with the same state and consumes a non-empty exact repetition of w, say w^{n_1}, for some

$1 \leq n_1 \leq |Q|$. We claim that the output produced by the factor $\hat{\rho}_{n,m}$ must coincide with the consumed part w^{n_1} of the input. Indeed, if this were not the case, then deleting the factor $\hat{\rho}_{n,m}$ from $\rho_{n,m}$ would result in a new successful run that reads $w^{n-n_1} \cdot u^m$ and produces $w^{n-n_2} \cdot \#^{m \cdot |u|}$ as output, for some $n_2 \neq n_1$. This however would contradict the fact that, by definition of encoding, the possible outputs produced by $\mathcal{T}$ on input $w^{n-n_1} \cdot u^m$ must agree on the prefix w^{n-n_1}. We also remark that, even if we do not annotate this explicitly, the number n_1 depends on the choice of the pair $(n, m) \in I$. This number, however, range over the fixed finite set $J = \big[1, |Q|\big]$.

We can now pump the factor $\hat{\rho}_{n,m}$ of the run $\rho_{n,m}$ any arbitrary number of times. In this way, we obtain new successful runs of $\mathcal{T}$ that consume inputs of the form $w^{n+k \cdot n_1} \cdot u^m$ and produce outputs of the form $w^{n+k \cdot n_1} \cdot \#^m$, for all $k \in \mathbb{N}$. In particular, we know that B_τ contains all pairs of the form $\alpha_{n+k \cdot n_1, m}$. Summing up, we can claim the following:

▶ **Claim.** There is a function $h : I \to J$ such that, for all pairs $(n, m) \in I$,

$$\big\{(n + k \cdot h(n,m), m) \;\big|\; k \in \mathbb{N}\big\} \;\subseteq\; I \;.$$

We can now head towards a contradiction. Let $\tilde{n}$ be the maximum common multiple of the numbers $h(n, m)$, for all $(n, m) \in I$. Let $m = n + \tilde{n}$ and observe that $n \neq m$, whence $(n, m) \in I$. Since $\tilde{n}$ is a multiple of $h(n, m)$, we derive from the above claim that the pair $(n + \tilde{n}, m) = (m, m)$ also belongs to I. However, this contradicts the definition of I, since we observed earlier that $\alpha_{n,m}$ is a bad encoding if and only if $n \neq m$. We conclude that B_τ is not one-way definable when τ has a solution. ◀