On Equivalence and Uniformisation Problems for Finite Transducers*

Emmanuel Filiot[†1], Ismaël Jecker[2], Christof Löding[3], and Sarah Winter[4]

1 **Université Libre de Bruxelles – F.R.S.-FNRS, Brussels, Belgium**
2 **Université Libre de Bruxelles – F.R.S.-FNRS, Brussels, Belgium**
3 **RWTH Aachen, Aachen, Germany**
4 **RWTH Aachen, Aachen, Germany**

Abstract

Transductions are binary relations of finite words. For rational transductions, i.e., transductions defined by finite transducers, the inclusion, equivalence and sequential uniformisation problems are known to be undecidable. In this paper, we investigate stronger variants of inclusion, equivalence and sequential uniformisation, based on a general notion of transducer resynchronisation, and show their decidability. We also investigate the classes of finite-valued rational transductions and deterministic rational transductions, which are known to have a decidable equivalence problem. We show that sequential uniformisation is also decidable for them.

1998 ACM Subject Classification F.4.3 Formal Languages

Keywords and phrases Transducers, Equivalence, Uniformisation

Digital Object Identifier 10.4230/LIPIcs.ICALP.2016.125

1 Introduction

Transductions generalise finite word languages to binary relations of finite words. The notion of rationality for languages, and its correspondence with finite automata, has been extended to transductions and finite automata over pairs of words, called *finite transducers* [2]. In this paper, we study decision problems for finite transducers and prove new decidability results.

Finite transducers. (Finite) transducers are nondeterministic finite automata whose transitions are labelled by pairs of words. The (rational) transduction $\mathcal{R}_T$ defined by a transducer T consists of all the pairs of words (u, v) obtained by concatenating the pairs occurring on transitions of its successful computations. In this paper, we follow a dynamic vision of transducers, as a machine that processes input words u and produces output words v. Therefore, we may speak of the domain of a transduction, as the language of input words that admit at least one output word.

Equivalence problem. Unlike finite automata, finite transducers have undecidable inclusion and equivalence problems [17, 14], even when restricted to unary alphabets [19]. The largest known classes with decidable equivalence problem are those of finite-valued transducers and

* A full version of this paper can be found in [12].

† Emmanuel Filiot is research associate at FNRS. This work is supported by the ARC project *Transform* (French speaking community of Belgium), the FNRS PDR project *Flare*, and the French ANR project ExtStream.

43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016).
Editors: Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi;
Article No. 125; pp. 125:1–125:14

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

deterministic transducers. A transducer is finite-valued if it produces at most k outputs per input, for a bound k that only depends on the transducer. It is decidable whether a transducer is k-valued for a given k [18], and whether there exists k such that it is k-valued [33]. Any finite-valued transducer is known to be (effectively) equivalent to a finite union of unambiguous transducers [32], and thus to a finitely ambiguous transducer. Equivalence of k-ambiguous transducers was shown to be decidable in [18], and equivalence of k-valued transducers was first shown to be decidable in [20, 32]. Other algorithms with better complexities appeared later, and the best known algorithm runs in exponential time, for a fixed k [8].

A transducer is deterministic if the transitions are deterministic in the classical sense, and furthermore each state processes either only input symbols or only output symbols. The class of deterministic rational transductions is also referred to as DRat, and it strictly extends the class of synchronous rational transductions (also called automatic relations), see e.g. [7] for an overview of these sub-classes of rational transductions. As opposed to the class of finite-valued transducers, it is undecidable whether a transduction is equivalent to a deterministic transduction [14]. However, the equivalence problem for DRat is known to be decidable [3], even in polynomial time [15]. This makes this class an interesting candidate for further investigations of decision problems.

Uniformisation problem. Two classes of interest are the rational and sequential functions, which are respectively defined by 1-valued transducers and sequential transducers. The latter read input words in a deterministic manner, and therefore produce a unique output word for each input. There are rational functions that are not sequential, but it is decidable in PTime whether a transducer defines a sequential function [34]. Since rational transductions do not define, in general, functions, an interesting question is whether a unique output word can be picked for each input word of a rational transduction R, in a regular way, thus defining a function $f \subseteq R$ with the same domain as R. Such a function f is called a *uniformiser* of R. It is known that any rational transduction admits a rational uniformiser [23, 10] and, in the case of DRat, even a *lexicographic* uniformiser that picks the smallest output words according to a lexicographic order, making the uniformiser only depend on the transduction [21, 27]. In this paper, we are interested in sequential uniformisers. Even rational functions do not admit sequential uniformisers in general, and therefore this gives rise to a decision problem: Given a finite transducer, does it admit a sequential uniformiser? It is worth noting that even if any rational transduction R can be uniformised by a rational uniformiser U, the sequential uniformisability of R does not imply, in general, that *any* of the uniformisers U is equivalent to a sequential transducer. As a matter of fact, it is known that the sequential uniformisation problem is undecidable for rational transductions [6].

The sequential uniformisation problem echoes a similar problem introduced by Church, the *synthesis problem*, which currently receives a lot of attention from the computer-aided verification community in the context of open reactive systems (see [24, 13, 4] for some work on this subject from the last decade). This problem asks whether given a logical specification of a system, there exists an implementation that satisfies it. In this context, reactive systems are non-terminating systems that react to some unpredictable environment stimuli in a *synchronised* fashion: for each environment input, they produce an output in a deterministic manner, such that the specification is met in the limit. Their executions are modelled by infinite words over a product alphabet, and the interaction with the environment makes game theory a powerful tool in this context. A seminal result due to Büchi and Landweber shows that the synthesis problem is decidable for MSO specifications [22] (see [31] for a modern presentation and an overview).

Restricted to finite words, the sequential uniformisation problem extends Church's problem to more general (asynchronous) classes of specifications and implementations, where the transduction R is the specification and the sequential uniformiser f the implementation.

Resynchronisers. One of the main difficulty of transducers is that two equivalent transducers may produce their outputs very differently: One transducer may go fast and be ahead of the other. By tagging symbols with two colours (for input and output), transductions can be seen as languages, called *synchronisation languages*. It is known by Nivat's theorem that rational transductions are synchronised by regular languages [26], and any transducer defines a regular synchronisation language. Other correspondences between classes of synchronisation languages and classes of rational transductions have been established in [11]. However in general, there is an infinite number of synchronisation languages for a single transduction, making problems such as equivalence and sequential uniformisation undecidable. To overcome this difficulty, Bojanczyk has introduced *transductions with origin information*, which amounts to adding the synchronisation information into the semantics of transducers, via an origin function mapping output positions to their originating input positions [5]. The main result of [5] is a machine-independent characterisation of transductions (with origin information) defined by two-way transducers. With respect to the equivalence problem, considering the origin information makes the problem easy: two transducers define the same transduction with same origin mappings if they have the same synchronisation language. In this paper, we generalise this idea and propose decision problems modulo resynchronisation. A *resynchroniser* $\mathbb{S}$ is a transduction, mapping a synchronisation language to another one. Then, we consider related equivalence and sequential uniformiser problems: for instance, given two transducers, are their synchronisation languages equal modulo $\mathbb{S}$? For the identity resynchroniser, it is the same as origin-equivalence.

Contributions. As a first contribution, we show that inclusion, equivalence and sequential uniformisation are decidable modulo *rational* resynchronisers. For equivalence, it easily reduces to an automata equivalence problem. For sequential uniformisation, it boils down to solving a two-player safety game. We then consider a particular class of resynchronisers, the *k-delay resynchronisers*, that can apply a fixed delay k to a synchronisation language, where the delay is a measure of how ahead an output word is from another one [1]. The k-delay resynchroniser is rational for each k, which implies the decidability of the corresponding decision problem. Interestingly, we show that for the class of real-time transducers (reading at least one input symbol in each transition), k-delay resynchronisers encompass all the power of rational synchronisers with respect to the decision problems we consider.

Our second main contribution is to show that equivalence and sequential uniformisation modulo k-delay resynchronisers are complete for finite-valued transducers. Given two finite-valued transducers, if they are equivalent, then some k can be computed such that they are k-delay equivalent. This yields another, delay-based, proof of the decidability of finite-valued transducer equivalence. We show a similar result for sequential uniformisation, by a pumping argument based on an analysis of the idempotent elements in the transition monoid of finitely-ambiguous transducers. This implies the following new result:

▶ **Theorem 16.** *The sequential uniformisation problem for finite-valued transducers is decidable.*

Finally, we consider deterministic rational transductions, i.e. transductions defined by deterministic transducers. A deterministic transducer is a deterministic automaton whose

states are partitioned into input and output states. They process pairs of words (u, v) as follows: two reading heads placed on u and v respectively process sequentially symbols from u and v. Whenever the current state is an input state, a symbol from u is read and the (input) head moves one step forward, and symmetrically on v when the current state is an output state. The equivalence problem for deterministic transducers is decidable [3], unlike the inclusion problem [14]. Our third main contribution is a decidability proof for the sequential uniformisation problem for deterministic rational transductions, extending a corresponding result for automatic relations from [6].

▶ **Theorem 18.** *The sequential uniformisation problem for deterministic transducers is decidable.*

Structure of the paper. In Section 2, we introduce automata, transducers and decision problems for them. In Section 3, we define the notion of resynchronisers for transductions and study their associated decision problems. We also introduce the particular class of bounded delay resynchronisers. In Section 4, we study the class of finite-valued rational transductions and prove decidability of their sequential uniformisation. Finally in Section 5, we prove decidability of sequential uniformisation for deterministic rational transductions.

2 Automata and Transducers

Let $\mathbb{N}$ denote the set of non-negative integers $\{0, 1, \dots\}$, and for every $n \in \mathbb{N}$, let $[n]$ denote the set $\{1, \dots, n\}$. Given a finite set A, let $|A|$ denote its cardinality.

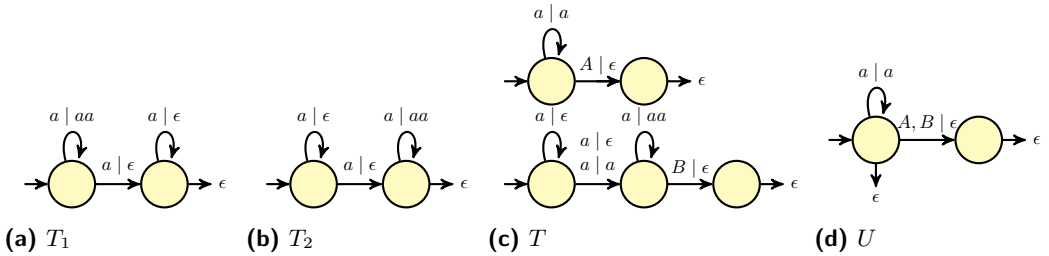
Languages and Transductions of Words. An alphabet Σ is a finite set of symbols. The elements of the free monoid Σ^* are called *words* over Σ. The length of a word w is the number of its symbols. It is written $|w|$. The empty word (of length 0) is denoted by ϵ, and $\Sigma^+ = \Sigma^* \setminus \{\epsilon\}$. The set Σ^* can be partially ordered by the word prefix relation $\preceq$.

We denote by Σ^{-1} the set of symbols σ^{-1} for all $\sigma \in \Sigma$. Any word $u \in (\Sigma \cup \Sigma^{-1})^*$ can be reduced into a unique irreducible word $\overline{u}$ by the equations $\sigma\sigma^{-1} = \sigma^{-1}\sigma = \epsilon$ for all $\sigma \in \Sigma$. Let G_Σ be the set of irreducible words over $\Sigma \cup \Sigma^{-1}$. The set G_Σ equipped with concatenation $u.v = \overline{uv}$ is a group, called the free group over Σ. We denote by u^{-1} the inverse of u. E.g. $(a^{-1}bc)^{-1} = c^{-1}b^{-1}a$. For $u \in G_\Sigma$, we denote by $|u|$ its number of symbols. E.g., $|a^{-1}b^{-1}| = 2$, $|a^{-1}bc^{-1}| = 3$.

A *language* L over Σ is a subset of Σ^*. A *transduction* R over Σ is a subset of $\Sigma^* \times \Sigma^*$. The *domain* of R is the set $\mathrm{dom}(R) = \{u \mid \exists v \in \Sigma^* \cdot (u, v) \in R\}$. For a word $u \in \Sigma^*$, we denote by $R(u)$ the set $\{v \mid (u, v) \in R\}$, and extend this notation to languages L by $R(L) = \bigcup_{u \in L} R(u)$. When R is a function, we simply write $R(u) = v$ instead of $R(u) = \{v\}$. Finally, we denote by id_{Σ^*} the identity relation on Σ^*.

Automata. A (finite state) *automaton* over an alphabet Σ is a tuple $A = (Q, I, F, \Delta)$, where Q is the finite set of states, $I \subseteq Q$ is the set of initial states, $F \subseteq Q$ is the set of final states, and $\Delta \subseteq Q \times \Sigma^* \times Q$ is the finite transition relation. Given a transition $(q, w, q') \in \Delta$, q is called its source, q' its target, and w its label. An automaton is called *deterministic* if each of its transition is labelled by a single letter, and it admits no pair of transitions that have same source, same label, and different targets.

A *run* of A on a word $u \in \Sigma^*$ from state q to state p is either a single state $q \in Q$ if $u = \epsilon$ and $q = p$, or a word $r = (q_1, u_1, p_1)(q_2, u_2, p_2) \dots (q_n, u_n, p_n) \in \Delta^+$ if $u \in \Sigma^+$, where $u = u_1 \dots u_n$, $q_1 = q$ and $p_n = p$, and for all $i \in \{1, \dots, n-1\}$, $p_i = q_{i+1}$. We write $q_1 \xrightarrow{u}_A p_n$



(a) T_1 **(b)** T_2 **(c)** T **(d)** U

Figure 1 Transducers such that $T_1 \equiv T_2$ and T is seq-uniformisable by U.

(or simply $q_1 \xrightarrow{u} p_n$) if such a run exists. A run r from a state q to a state p is *accepting* if q is initial and p is final. The *language* recognised by A, written $\mathcal{L}_A$, is the set of words $w \in \Sigma^*$ such that there exists an accepting run of A on w. If B is an automaton, we write $A \subseteq B$ (resp. $A \equiv B$) whenever $\mathcal{L}_A \subseteq \mathcal{L}_B$ (resp. $\mathcal{L}_A = \mathcal{L}_B$).

Transducers. A (finite state) *transducer* over an alphabet Σ is a tuple $T = (Q, I, F, \Delta, f)$, where Q is the finite set of states, $I \subseteq Q$ the set of initial states, $F \subseteq Q$ the set of final states, $\Delta \subseteq Q \times \Sigma^* \times \Sigma^* \times Q$ the transition relation, and $f : F \to \Sigma^*$ the final output function[1]

As for automata, a *run* of a transducer is either a single state or a sequence of transitions. The *input* (resp. *output*) of a run $r = (q_1, u_1, v_1, p_1) \dots (q_n, u_n, v_n, p_n) \in \Delta^*$ is $\text{in}(r) = u_1 \dots u_n$ (resp. $\text{out}(r) = v_1 \dots v_n$). If r is reduced to a single state, its input and output are both ϵ. We say that r is a run of T on $u_1 \dots u_n$. We write $q \xrightarrow{u|v} p$ to mean that there exists a run on input $u \in \Sigma^*$ whose output is $v \in \Sigma^*$. In particular, $q \xrightarrow{\epsilon|\epsilon} q$ for all $q \in Q$. The notion of accepting run of automata carries over to transducers. The *transduction* recognised by T, written $\mathcal{R}_T$ is the set of pairs $(u, vf(p)) \in \Sigma^* \times \Sigma^*$ such that there exists an accepting run of T on u from a state q to a state p whose output is v. We define $\text{dom}(T)$ as $\text{dom}(\mathcal{R}_T)$. The class of *rational transductions* is the class of relations definable by finite state transducers.

The *input automaton* of T is the automaton $A = (Q, I, F, \Delta')$ over the alphabet Σ, where $\Delta' = \{(q, u, q') | (q, u, v, q') \in \Delta\}$. A transducer is called *real time* if each of its transition is labelled by a pair (a, v), where $a \in \Sigma$ and $v \in \Sigma^*$. A transducer is called *sequential* if its input automaton is deterministic[2]. Sequential transducers define *sequential transductions*. A transducer is *trim* if all its accessible states are co-accessible, i.e. for all $q \in Q$, $q_0 \in I$, $u, v \in \Sigma^*$, if $q_0 \xrightarrow{u|v} q$, then there exist $u', v' \in \Sigma^*$ and $q_f \in F$ such that $q \xrightarrow{u'|v'} q_f$.

Decision Problems for Transducers. Let T_1, T_2 be two transducers over an alphabet Σ. We write $T_1 \subseteq T_2$ whenever $\mathcal{R}_{T_1} \subseteq \mathcal{R}_{T_2}$. The *inclusion problem* asks, given T_1, T_2, whether $T_1 \subseteq T_2$. Similarly, we define the equivalence problem by asking whether $\mathcal{R}_{T_1} = \mathcal{R}_{T_2}$, denoted $T_1 \equiv T_2$. Let T be a transducer over an alphabet Σ. A uniformiser of T is a transducer U such that $U \subseteq T$ and $\text{dom}(U) = \text{dom}(T)$. We sometimes write seq-uniformiser for sequential uniformiser. The *sequential uniformisation problem* (seq-uniformisation problem) asks, given a transducer T over Σ, whether T admits a seq-uniformiser.

[1] Allowing final output functions does not increase the expressiveness of the general model, but is required to define the notion of sequentiality

[2] The term subsequential was originally used in the literature. We follow the terminology of [25].

▶ **Example 1.** The transducers T_1 and T_2 of Fig. 1 both define the transduction $\{(a^n, a^{2i}) \mid n \geq 1, 0 \leq i \leq n-1\}$, thus are equivalent. The transducer T is over the alphabet $\{a, A, B\}$ and defines the transduction $\{(a^nA, a^n) \mid n \geq 0\} \cup \{(a^nB, a^i) \mid n \geq 1,\ 0 \leq i \leq 2n-1\}$. It is uniformisable by the sequential transducer U with $\mathcal{R}_U = \{(a^n\alpha, a^n) \mid n \geq 0, \alpha \in \{A, B\}\}$.

▶ **Theorem 2** ([17, 6]). *The inclusion, equivalence and sequential uniformisation problems for rational transductions are undecidable.*

3 Decision Problems Modulo Resynchronisers

A pair $(u, v) \in \Sigma^* \times \Sigma^*$ can be represented by a coloured word over $\Sigma \times \{\mathtt{i}, \mathtt{o}\}$, where the colours indicate whether a symbol in Σ is an input or an output symbol. Such a coloured word is called a synchronisation of (u, v). More generally, any language over the alphabet $\Sigma \times \{\mathtt{i}, \mathtt{o}\}$ represents a transduction $R \subseteq \Sigma^* \times \Sigma^*$, and is called a synchronisation language for R. This way of representing transductions is analysed in [11]. What we call a resynchroniser below, is a transduction of synchronisations, that is, over words in $(\Sigma \times \{\mathtt{i}, \mathtt{o}\})^*$, that preserves the represented pairs. In this section, we study stronger notion of inclusion, equivalence and sequential uniformisation, parametrised by such a resynchroniser. We show their decidability for rational resynchronisers and introduce the class of bounded delay resynchronisers, and show that it has appealing properties.

Synchronisations and resynchronisers. Given an alphabet Σ, we let $\Sigma_{\mathtt{io}} = \Sigma \times \{\mathtt{i}, \mathtt{o}\}$, $\Sigma_{\mathtt{i}} = \Sigma \times \{\mathtt{i}\}$ and $\Sigma_{\mathtt{o}} = \Sigma \times \{\mathtt{o}\}$. For $c \in \{\mathtt{i}, \mathtt{o}\}$, we write σ^c instead of (σ, c). The colouring c can be seen as a morphism $.^c : \Sigma^* \to \Sigma_c^*$ and we write u^c its application on a word $u \in \Sigma^*$. Conversely, for $c \in \{\mathtt{i}, \mathtt{o}\}$, we define two morphisms $\pi_c : \Sigma_{\mathtt{io}}^* \to \Sigma$ that extract the input and output words, by $\pi_c(\sigma^c) = \sigma$ and $\pi_c(\sigma^d) = \epsilon$, for all $\sigma \in \Sigma$, and $d \neq c$. Two words $u, v \in (\Sigma_{\mathtt{io}})^*$ are said to be *equivalent*, denoted by $u \sim_{\mathtt{io}} v$, if $\pi_c(u) = \pi_c(v)$ for all $c \in \{\mathtt{i}, \mathtt{o}\}$. For example, $a^{\mathtt{i}}b^{\mathtt{i}}a^{\mathtt{o}}$ and $a^{\mathtt{i}}a^{\mathtt{o}}b^{\mathtt{i}}$ are equivalent, and both are synchronisations of (ab, a). Any language $L \subseteq \Sigma_{\mathtt{io}}^*$ defines a transduction over Σ defined by $\mathcal{R}_L = \{(\pi_{\mathtt{i}}(w), \pi_{\mathtt{o}}(w)) \in \Sigma^* \times \Sigma^* \mid w \in L\}$, and L is called a *synchronisation* of a transduction $R \subseteq \Sigma^* \times \Sigma^*$ if $\mathcal{R}_L = R$. We also say that L synchronises R. Note that two different languages may synchronise the same transduction.

Mapping a synchronisation to another one is done through the notion of resynchroniser. A *resynchroniser* is a transduction $\mathbb{S} \subseteq \Sigma_{\mathtt{io}}^* \times \Sigma_{\mathtt{io}}^*$, such that (i) $\mathrm{id}_{\Sigma_{\mathtt{io}}^*} \subseteq \mathbb{S}$ and (ii) for all $(w, w') \in \mathbb{S}$, it holds $w \sim_{\mathtt{io}} w'$. For instance, the identity relation $\mathrm{id}_{(\Sigma_{\mathtt{io}})^*}$ is a resynchroniser that we shall denote by $\mathbb{I}$, as well as the relation $\mathbb{U}_{\Sigma_{\mathtt{io}}} = \{(w, w') \in \Sigma_{\mathtt{io}}^* \times \Sigma_{\mathtt{io}}^* \mid w \sim_{\mathtt{io}} w'\}$, called the *universal resynchroniser* over $\Sigma_{\mathtt{io}}$. We write $\mathbb{U}$ instead of $\mathbb{U}_{\Sigma_{\mathtt{io}}}$ when it is clear from the context. Note that for any resynchroniser $\mathbb{S}$, we have $\mathrm{id}_{\Sigma_{\mathtt{io}}^*} \subseteq \mathbb{S} \subseteq \mathbb{U}$. The properties (i) and (ii) of resynchronisers are chosen such that they preserve the represented transductions, as stated in the proposition below.

▶ **Proposition 3.** *For all $L \subseteq \Sigma_{\mathtt{io}}^*$ and all resynchronisers $\mathbb{S} \subseteq \Sigma_{\mathtt{io}}^* \times \Sigma_{\mathtt{io}}^*$, $\mathcal{R}_L = \mathcal{R}_{\mathbb{S}(L)}$.*

Classes of synchronisation languages and their correspondence with the classes of rational relations they synchronise have been studied in [11]. We can formulate in this framework a result known as Nivat's theorem [26] as follows.

▶ **Theorem 4.** *[26] A transduction R is rational iff it is synchronised by a regular language.*

A regular language synchronising a rational transduction can be obtained as follows. Any transducer $T = (Q, I, F, \Delta, f)$ naturally defines a regular synchronisation for $\mathcal{R}_T$ by its

underlying automaton, which is the automaton obtained by concatenating the pairs of input and output words on the transitions and marking them with the respective symbol from $\{\mathtt{i}, \mathtt{o}\}$. Formally, it is the automaton $A = (Q \cup \{q_\dashv\}, I, \{q_\dashv\}, \Delta')$ over $\Sigma_{\mathtt{io}}$, where $\Delta' = \{(q, v^{\mathtt{i}}w^{\mathtt{o}}, q') | (q, v, w, q') \in \Delta\} \cup \{(q, f(q)^{\mathtt{o}}, q_\dashv) | q \in F\}$. The *language recognised by* T is the language recognised by its underlying automaton, denoted by $\mathcal{L}_T$, i.e. $\mathcal{L}_T = \mathcal{L}_A$. Obviously, $\mathcal{L}_T$ is a synchronisation for the relation $\mathcal{R}_T$ (proving one direction of Thm 4).

Decision problems for transducers modulo resynchronisers. Let Σ be an alphabet, $\mathbb{S}$ be a resynchroniser over $(\Sigma_{\mathtt{io}})^*$, and T_1, T_2 be two transducers over Σ. We say that T_1 is *included in* T_2 *modulo* $\mathbb{S}$ (or $\mathbb{S}$-*included*), denoted by $T_1 \subseteq_{\mathbb{S}} T_2$, if $\mathcal{L}_{T_1} \subseteq \mathbb{S}(\mathcal{L}_{T_2})$. We say that T_1 is *equivalent to* T_2 *modulo* $\mathbb{S}$ (or $\mathbb{S}$-*equivalent*), denoted by $T_1 \equiv_{\mathbb{S}} T_2$, if $T_1 \subseteq_{\mathbb{S}} T_2$ and $T_2 \subseteq_{\mathbb{S}} T_1$. For a fixed synchroniser $\mathbb{S}$, the $\mathbb{S}$-*inclusion (resp.* $\mathbb{S}$-*equivalence) problem* asks, given two transducers T_1, T_2 over Σ, whether $T_1 \subseteq_{\mathbb{S}} T_2$ (resp. $T_1 \equiv_{\mathbb{S}} T_2$). We say that T_1 is *sequentially* $\mathbb{S}$-*uniformisable* if it admits a sequential uniformiser U such that $U \subseteq_{\mathbb{S}} T_1$, and in that case U is called a sequential $\mathbb{S}$-uniformiser of T_1 (seq-$\mathbb{S}$-uniformiser for short). The *sequential* $\mathbb{S}$-*uniformisation problem* asks whether a given transducer is seq-$\mathbb{S}$-uniformisable.

It should be clear from the definition that $\mathbb{I}$-inclusion implies $\mathbb{S}$-inclusion for any resynchroniser $\mathbb{S}$, which in turn implies $\mathbb{U}$-inclusion. As a matter of fact, it is easy to see that $\mathbb{U}$-inclusion is equivalent to classical inclusion. The same remarks can be made for equivalence and sequential uniformisation, and as a consequence of Theorem 2, we get:

▶ **Theorem 5.** *The* $\mathbb{U}$*-inclusion,* $\mathbb{U}$*-equivalence, sequential* $\mathbb{U}$*-uniformisation problems for rational transductions are undecidable.*

Decision problems for transducers modulo rational resynchronisers. The $\mathbb{U}$-decision problems are undecidable, this raises the question whether there is an interesting class of resynchronisers for which we can recover decidability. It turns out that $\mathbb{U}$ is not rational. In contrast, we show that, as long as $\mathbb{S}$ is rational, the $\mathbb{S}$-decision problems are reducible to the $\mathbb{I}$-decision problems, which in turn can be solved by reduction to decidable problems of automata and two-player games.

▶ **Proposition 6.** *The* $\mathbb{I}$*-inclusion and* $\mathbb{I}$*-equivalence problems are* PSpace-complete*. The sequential* $\mathbb{I}$*-uniformisation problem is* ExpTime-complete*.*

Proof. First, note that $T_1 \subseteq_{\mathbb{I}} T_2$ iff $\mathcal{L}_{T_1} \subseteq \mathcal{L}_{T_2}$ iff $A_1 \subseteq A_2$, where A_1, A_2 are the underlying automata of T_1, T_2 respectively. Automata inclusion and equivalence problems are PSpace-complete, and they easily reduce (by putting ϵ outputs) to $\mathbb{I}$-inclusion and $\mathbb{I}$-equivalence.

To get ExpTime membership of seq-$\mathbb{I}$-uniformisation, for a transducer T, we construct a two-player safety game $G_T = (V = V_{\mathsf{In}} \uplus V_{\mathsf{Out}}, v_0, E)$ between an adversary (Player In) who picks input symbols and controls positions in V_{In}, and a protagonist (Player Out) who picks sequences of output symbols and controls positions in V_{Out}. Wlog we assume that T has no final output function, by adding an endmarker $\dashv$ to words of its domain. Let $A = (Q, q_0, F, \delta)$ be a complete DFA equivalent to the the underlying automaton of T (whose size is at most exponential in the size of T). Player positions have three components: a residual language[3] of $\mathrm{dom}(T)$ controlling the possible continuations of the input word chosen so far by Player In, a state of A and a round $r \in \{\mathsf{In}, \mathsf{Out}\}$. Let $\mathcal{D} = \{u^{-1}\mathrm{dom}(T) \mid u \in \Sigma^*\}$ be the set of residuals of $\mathrm{dom}(T)$ (represented by the states of the minimal DFA for $\mathrm{dom}(T)$, computed in

[3] A residual of a language L over some alphabet Σ is a language $u^{-1}L = \{v \mid uv \in L\}$ for $u \in \Sigma^*$.

exponential time in the size of T). Then, $V_{\mathsf{In}} = \mathcal{D} \times Q \times \{\mathsf{In}\}$ and $V_{\mathsf{Out}} = \mathcal{D} \times Q \times \{\mathsf{Out}\}$. The initial position is $v_0 = (\mathrm{dom}(T), q_0, \mathsf{In})$ and the edge relation E as follows: from a position (D, q, In), there are outgoing edges to all states $(\sigma^{-1}D, \delta(q, \sigma^{\mathtt{i}}), \mathsf{Out})$ for all $\sigma \in \Sigma$. From a position (D, q, Out), Player Out can pick any state $q' \in Q$ such that there exists a sequence $v \in \Sigma_{\mathtt{o}}^*$ such that $q \xrightarrow{v}_A q'$, and in that case an outgoing edge to (D, q', In) is added to E. The *unsafe positions* for Player Out are all positions (D, q, In) such that $\epsilon \in D$ and $q \notin F$: at such positions, Player In could choose to terminate the sequence of input symbols (while staying in $\mathrm{dom}(T)$ since $\epsilon \in D$) and the sequence of output symbols chosen by Player Out, mixed with the input symbols chosen by Player In, does not belong to $L(A)$ (since $q \notin F$).

It can be shown that Player Out has a strategy to avoid the unsafe positions in G_T iff there exists a seq-$\mathbb{I}$-uniformiser of T. We briefly explain how to extract a uniformiser from a memoryless winning strategy. A memoryless winning strategy of Player Out can be represented by a function $\lambda : V_{\mathsf{Out}} \to V_{\mathsf{In}}$. The uniformiser U_λ has V_{In} as state set. Let v be a state of U_λ where $v = (D, q, \mathsf{In}) \in V_{\mathsf{In}}$ and $\sigma \in \Sigma$. Let $v' = (\sigma^{-1}D, \delta(q, \sigma^{\mathtt{i}}), \mathsf{Out})$ and $v'' = \lambda(v')$. By definition of G_T, $v'' = (\sigma^{-1}D, q', \mathsf{In})$ such that $\delta(q, \sigma^{\mathtt{i}}) \xrightarrow{w}_A q'$ for some word $w \in \Sigma^*$. We then add the transition (v, σ, w, v'') to U_λ. The word w can be uniquely chosen by taking the minimal word for some lexicographic order, making U_λ sequential. Accepting states are states $v = (D, q, \mathsf{In})$ with $\epsilon \in D$, thus ensuring $\mathrm{dom}(U_\lambda) = \mathrm{dom}(T)$. Since λ is winning, we then necessarily have $q \in F$, ensuring that the sequence of input and output symbols read and produced by U_λ belongs to $\mathcal{L}_A$, making U_λ an $\mathbb{I}$-uniformiser.

Since safety games can be solved in polynomial time and G_T has exponential size, we get the result. The results on safety games that we use here can be found, e.g., in [16].

For the ExpTime lower bound, we note that in our formalism we can model the synchronous uniformisation (or synthesis) problem, as considered in [28] for infinite words, by taking synchronisations that strictly alternate between input and output. It seems to be common knowledge in the synthesis community that the synchronous uniformisation problem is ExpTime-complete if the relation is given by a nondeterministic automaton. However, we were not able to find a reference for this result. We thus give a reduction from the acceptance problem for alternating Pspace Turing machines in a long version. ◀

For all transducers T and resynchronisers $\mathbb{S}$, $\mathbb{S}(\mathcal{L}_T)$ is a regular synchronisation language and by Nivat's theorem (Theorem 4), there exists a transducer $T^{\mathbb{S}}$ such that $\mathcal{L}_{T^{\mathbb{S}}} = \mathbb{S}(\mathcal{L}_T)$. It implies that the seq-$\mathbb{S}$-uniformisation of T reduces to the seq-$\mathbb{I}$-uniformisation of $T^{\mathbb{S}}$. Similar arguments apply for inclusion and equivalence and from Proposition 6 we obtain:

▶ **Theorem 7.** *Let $\mathbb{S}$ be a rational resynchroniser, given as a transducer. The $\mathbb{S}$-inclusion and $\mathbb{S}$-equivalence problems are* PSpace-complete. *The sequential $\mathbb{S}$-uniformisation problem is* ExpTime-complete.

Bounded delay resynchronisers. The notion of *delay* between outputs of transducers is a powerful way of comparing transducers, which has been used, for instance, to characterise sequential functions [2]. Intuitively, the delay between two runs on the same input is a parameter that measures how a run is ahead of the other, and the lag is the maximal delay over prefixes of the two runs. We adapt the notion of delay and lag to coloured words and define delay resynchronisers as resynchronisers that apply a fixed delay to words in $\Sigma_{\mathtt{io}}^*$ (our notion of lag is not related to the one from [11]). Our results show that delay resynchronisers form a fundamental class of resynchronisers.

The *delay* between two words u and v over an alphabet Σ is the element from the free group G_Σ defined by $\mathrm{delay}(u, v) = u^{-1}v$. E.g., $\mathrm{delay}(ab, acd) = b^{-1}cd$. Note that $\mathrm{delay}(u, v) \in \Sigma^*$ iff

$u \preceq v$, and $\text{delay}(u,v) \in (\Sigma^{-1})^*$ iff $v \preceq u$. The *lag mapping* $\text{lag} : (\Sigma_{\mathtt{i}\mathtt{o}})^* \times (\Sigma_{\mathtt{i}\mathtt{o}})^* \to \mathbb{N} \cup \{+\infty\}$ gives the maximal length of the delay between the output part of two words in $(\Sigma_{\mathtt{i}\mathtt{o}})^*$ that have the same input. It is the metric defined by $\text{lag}(u,v) = +\infty$ if $\pi_{\mathtt{i}}(u) \neq \pi_{\mathtt{i}}(v)$. If $\pi_{\mathtt{i}}(u) = \pi_{\mathtt{i}}(v)$, then u and v can be decomposed into $u = u_0 a_1 u_1 \dots u_{n-1} a_n u_n$ and $v = v_0 a_1 v_1 \dots a_{n-1} b_n v_n$, such that $a_1, \dots, a_n \in \Sigma_{\mathtt{i}}$, $u_0, v_0, \dots, u_n, v_n \in (\Sigma_{\mathtt{o}})^*$. Then $\text{lag}(u,v) = \max_{0 \leq i \leq n} |\text{delay}(u_0 \dots u_i, v_0 \dots v_i)|$. As an example, for $n \geq 1$, take $u_n = a^{\mathtt{i}} a^{\mathtt{o}} (a^{\mathtt{i}})^n$ and $v_n = (a^{\mathtt{i}})^n a^{\mathtt{i}} a^{\mathtt{o}}$. Then for all $n \geq 1$, $\text{lag}(u_n, v_n)$=1. Note that the occurrence of $a^{\mathtt{o}}$ in u_n is arbitrary far from that of $a^{\mathtt{o}}$ in v_n.

We now define the k-delay resynchroniser $\mathbb{D}_k$. Intuitively, it can shift output symbols of a word u to the left or to the right, as long as the lag between u and the new word obtained this way is bounded by k. Formally, the *k-delay resynchroniser* is defined by $\mathbb{D}_k = \{(u,v) \in (\Sigma_{\mathtt{i}\mathtt{o}})^2 \mid u \sim_{\mathtt{i}\mathtt{o}} v \wedge \text{lag}(u,v) \leq k\}$. We define the *$k$-inclusion*, *$k$-equivalence* and sequential *k-uniformisation* problems as the corresponding $\mathbb{D}_k$-decision problems, and write $\subseteq_k$ and $\equiv_k$ instead of $\subseteq_{\mathbb{D}_k}$ and $\equiv_{\mathbb{D}_k}$ respectively. We also say that a transduction is *seq-k-uniformisable* if it is seq-$\mathbb{D}_k$-uniformisable. An important property of $\mathbb{D}_k$ is:

▶ **Proposition 8.** *For all $k \geq 0$, $\mathbb{D}_k$ is rational.*

As a direct consequence of the latter proposition and Theorem 7, the k-delay decision problems are all decidable. We can be more precise:

▶ **Theorem 9.** *For all $k \geq 0$, the k-inclusion, k-equivalence and sequential k-uniformisation problems are decidable and* ExpSpace-hard *if k is part of the input. If k is fixed, then the k-inclusion and k-equivalence problems are* PSpace-complete*, and the sequential k-uniformisation problem is* ExpTime-complete*.*

Even if inclusion is undecidable while k-inclusion is decidable, it could be the case that inclusion reduces to k-inclusion, for some k that cannot be computed. We show that that it is not the case, by using the transducers of Fig. 1.

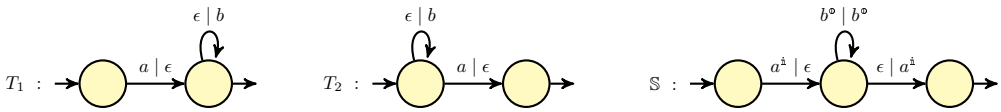
▶ **Proposition 10.** *There exist transducers T_1, T_2, T such that $T_1 \equiv T_2$, $T_1 \subseteq T_2$ and T is seq-uniformisable, but for all $k \geq 0$, $T_1 \not\equiv_k T_2$, $T_1 \not\subseteq_k T_2$, and T is not seq-k-uniformisable.*

Proof. Consider T_1, T_2 of Fig.1 and pairs of the form $(a^{n+1}, a^{2n}) \in \mathcal{R}_{T_1} = \mathcal{R}_{T_2}$. They both accept these pairs but T_2 will be arbitrarily late compared to T_1. Consider now the transducer T and its sequential uniformiser U. On inputs $a^n B$, U will be arbitrarily ahead of T, and one can show that is the case for *any* seq-uniformiser of T. ◀

Finally, we show that for real-time transductions, k-delay resynchronisers subsume any rational resynchroniser $\mathbb{S}$, in the sense that $\mathbb{S}$-inclusion implies k-inclusion, for some k that depends on $\mathbb{S}$. Similar results hold for equivalence and sequential uniformisation. The idea is that a rational synchroniser cannot advance or delay the production of outputs arbitrarily far away with a finite set of states.

▶ **Theorem 11.** *Let $\mathbb{S}$ be a rational synchroniser (given by a transducer). Let T_1, T_2, T be real-time transducers. There exists a computable integer $k \in \mathbb{N}$ such that:* (i) *if $T_1 \subseteq_{\mathbb{S}} T_2$, then $T_1 \subseteq_k T_2$,* (ii) *if $T_1 \equiv_{\mathbb{S}} T_2$, then $T_1 \equiv_k T_2$, and* (iii) *if T is seq-$\mathbb{S}$-uniformisable, then T is seq-k-uniformisable.*

One cannot drop the real-time assumption in the latter theorem. Indeed consider the following transducers $T_1, T_2, \mathbb{S}$, for which $T_1 \equiv_{\mathbb{S}} T_2$ but T_1, T_2 are not k-equivalent for any k:



4 Finite-valued transducers

Let $m \in \mathbb{N}$. We remind the reader that a transducer T is called *m-valued* if each input has at most m outputs, i.e. for all $w \in \mathrm{dom}(T)$, $|\mathcal{R}_T(w)| \leq m$. It is finite-valued if it is m-valued for some m. Finite-valuedness is decidable [33]. We prove that for the class of finite-valued transducers, k-inclusion and sequential k-uniformisation are complete. This yields, for finite-valued transducers, an alternative proof of the decidability of the inclusion problem, and a new result: The decidability of sequential uniformisation.

Let m be a natural number. An automaton A (resp. transducer T) is called *m-ambiguous* if it is real-time[4], and for any word $w \in \mathcal{L}_A$ (resp. $w \in \mathrm{dom}(T)$), there exist at most m accepting runs of A (resp. T) on w. An automaton (transducer) is called *finitely ambiguous* if there exists $m \in \mathbb{N}$ such that it is m-ambiguous, and *unambiguous* if it is 1-ambiguous. Our proofs uses the following known decomposition initially due to Weber:

▶ **Theorem 12.** *[32, 30] Any finite-valued transducer T is (effectively) equivalent to a union of unambiguous transducers.*

We first prove that, for the class of finitely ambiguous transducers, inclusion and equivalence reduces to k-inclusion and k-equivalence for some computable k. We state this result for inclusion, which immediately implies it for equivalence. The proof is based on similar pumping techniques than Lemma 2 in [9].

▶ **Theorem 13.** *Let T_1 and T_2 be two real-time transducers such that T_2 is m-ambiguous. Then there exists a computable integer k such that $T_1 \subseteq T_2 \implies T_1 \subseteq_k T_2$. Moreover, k can be chosen to be exponential in the size of T_2 and linear in the size of T_1.*

Since k-inclusion is decidable by Theorem 7, Theorem 13 implies that the inclusion and equivalence problems are decidable for finitely ambiguous transducers. From the decomposition of Theorem 12, we obtain an alternative proof of the decidability of equivalence of finite-valued transducers, which was proved in [20, 32].

▶ **Corollary 14** ([20, 32] Alternative proof)**.** *The inclusion and equivalence problems for finite-valued transducers are decidable.*

We now prove the two corresponding results for the sequential uniformisation problem.

▶ **Theorem 15.** *Let T be a real-time trim transducer given as a finite union of unambiguous transducers. Then there exists a computable integer N_T such that if T is sequentially uniformisable, then it is sequentially N_T-uniformisable.*

Sketch of proof. If T is seq-uniformisable, then there exists a sequential uniformiser U of T such that $\mathrm{dom}(U) = \mathrm{dom}(T)$ and $U \subseteq T$. The latter inclusion implies, by Theorem 13, that there exists an integer k such that $U \subseteq_k T$, and so U is a seq-k-uniformiser of T. However, k depends on the number of states of the hypothetical uniformiser U. We show how to construct, by simulating the behaviour of U, another seq-N_T-uniformiser U', where N_T only depends on T and can be computed.

More precisely, we define a function $\rho : \Sigma^* \to \Sigma^*$ and define U' such that on any input w, it simulates U on input $\rho(w)$. The function ρ iterates some well-chosen subwords of w to blow up the delay between the outputs of the runs of T on $\rho(w)$. On input $\rho(w)$, any

[4] For simplicity reasons, we put real-timeness in the definition, but it is known to be wlog.

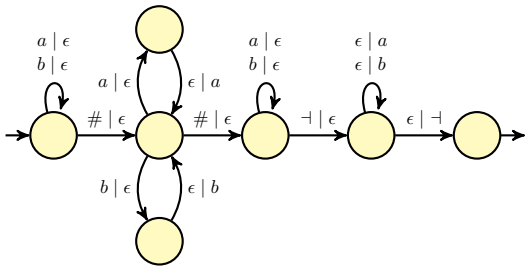


Figure 2 A deterministic transducer with endmarker for $\mathcal{R}_1$ from Ex. 17.

seq-k-uniformiser of T, and U in particular, is forced to make choices between possible outputs of T on $\rho(w)$, in order to decrease the delay. The main idea is that if, by making some good choice of output, U is able to react to a threat of exceeding delay k on $\rho(w)$, then by doing the same choice on w, U' can also react to a threat of exceeding delay N_T.

We identify several key properties that ρ must satisfy, in order to be able to construct U'. For instance, we require that $\rho(w)$ is a prefix of $\rho(wa)$ for all $w \in \Sigma^*$, $a \in \Sigma$, but also some property relating the delays between U and T on input w and on input $\rho(w)$.

The challenging part of the proof is to prove the existence of N_T and ρ. It is based on a study of the structural properties of the transition monoid of finitely ambiguous transducers (a monoid that captures the state behaviour of automata and transducers), and the effect of its elements on the delays. In particular, subwords of w that are iterated to define $\rho(w)$ correspond to idempotent elements in the transition monoid of T, and the bound N_T is obtained by an application of Ramsey's theorem. ◀

Since by Theorem 7 every finite-valued transducer is effectively equivalent to a finitely ambiguous transducer, the sequential uniformisation problem for finite-valued transducers reduces to sequential N-uniformisation, for computable integers N. Hence by Theorem 7 and the fact that any transducer can be trimmed in polynomial time, we get decidability of sequential uniformisation of finite-valued transducers, one of the main results of this paper.

▶ **Theorem 16.** *The sequential uniformisation problem for finite-valued transducers is decidable.*

5 Deterministic Rational Transductions

In this section we consider another subclass of rational transductions, namely the deterministic rational transductions, denoted by DRat. This class is defined in terms of specific deterministic transducers and some problems that are undecidable for general rational transductions are decidable in the case of DRat. For example, the equivalence problem is decidable [3] (while inclusion is easily seen to be undecidable [14]), and whether a given relation in DRat is recognisable [7] is also decidable. We obtain here another decidability result, namely that the sequential uniformisation problem is decidable for deterministic transducers.

For the definition of deterministic rational transducers, we work with endmarkers (this is the common way of doing it, see also [29]). The determinism includes the standard definition of unique successor states for each symbol and additionally a deterministic choice between input and output. This is enforced by a partition of the state space into states processing input symbols and states processing output symbols.

▶ **Example 17.** The transduction $\mathcal{R}_1 = \{(u\#v\#w, vx) \mid u, v, w, x \in \{a, b\}^*\}$ is in DRat since it is recognised by the deterministic transducer depicted in Fig. 2 over the alphabet $\{a, b, \#\}$ with endmarker $\dashv$. Note that for each state the outgoing transitions either all have ϵ as output component, or all have ϵ as input component. In the formal definition, this is captured by partitioning the states into input and output states.

Let Σ be an alphabet and $\dashv$ a fresh symbol used as endmarker. We let $\Sigma_\dashv := \Sigma \cup \{\dashv\}$. A deterministic transducer over the alphabet Σ with endmarker $\dashv$ is of the form $T = (Q^{\mathtt{i}}, Q^{\mathtt{o}}, F, q_0, \delta)$ with a set $Q^{\mathtt{i}}$ of input states, a set $Q^{\mathtt{o}}$ of output states (we write Q for the union of these two sets), a unique initial state q_0, a transition function $\delta : Q \times \Sigma_\dashv \to Q$, and a set $F \subseteq Q$ of accepting states. In the presence of endmarkers, the final output function is not required anymore.

For defining the semantics of such a deterministic transducer, one can transform it into a standard transducer. However, this transformation needs to take care of the endmarker only being allowed at the end of the word, which is not enforced in the definition of deterministic transducers. To avoid this, we rather define the semantics by extending the transition function to pairs of words (input and output word). For $(u, v) \in \Sigma_\dashv^* \times \Sigma_\dashv^*$ and $q \in Q$, we define $\delta^* : Q \times \Sigma_\dashv^* \times \Sigma_\dashv^* \to Q \times \Sigma_\dashv^* \times \Sigma_\dashv^*$ inductively as follows:

- If $q \in Q^{\mathtt{i}}$, then $\delta^*(q, \epsilon, v) = (q, \epsilon, v)$ and $\delta^*(q, au, v) = \delta^*(\delta(q, a), u, v)$.
- If $q \in Q^{\mathtt{o}}$, then $\delta^*(q, u, \epsilon) = (q, u, \epsilon)$ and $\delta^*(q, u, av) = \delta^*(\delta(q, a), u, v)$.

So δ^* applies δ to the next input letter from states in $Q^{\mathtt{i}}$ and to the next output letter from states in $Q^{\mathtt{o}}$ as long as possible. The transduction $\mathcal{R}_T$ defined by T is

$$\mathcal{R}_T = \{(u, v) \in \Sigma^* \times \Sigma^* \mid \delta^*(q_0, u \dashv, v \dashv) = (q, \epsilon, \epsilon) \text{ with } q \in F\}.$$

Recall from Section 4 that k-delay inclusion and equivalence are complete for finite-valued transducers, as stated in Theorem 13. We note that this is not the case for DRat.

▶ Remark. There are deterministic transducers T_1 and T_2 such that $T_1 \equiv T_2$ but there is no k such that $T_1 \equiv_k T_2$.

Proof. Consider the complete relation $\Sigma^* \times \Sigma^*$, and let T_1 be the deterministic transducer that first reads all input symbols (up to the endmarker $\dashv$), and then reads all output symbols. Let T_2 be the deterministic transducer that first reads all output symbols and then the input symbols. Obviously, $\mathcal{R}_{T_1} = \mathcal{R}_{T_2} = \Sigma^* \times \Sigma^*$. However, the lag for the two runs of T_1, T_2 on a pair (u, v) is $|v|$ and thus not bounded. ◀

Our main result for DRat is the following, which extends the corresponding result for automatic relations from [6].

▶ **Theorem 18.** *The sequential uniformisation problem for deterministic transducers is decidable.*

The proof uses the game-theoretic approach, building a game between players Input and Output. A winning strategy for player Output then corresponds to a sequential uniformiser. The moves of the game simulate the deterministic transducer T on the pairs of input and output word played by the two players in order to check whether the output indeed matches the input. However, Output might need to delay the moves to gain some lookahead on the input for making the next decisions. The main challenge in the proof is to find a way to keep the lookahead information bounded without losing too much information. It is not sufficient to simply store words of bounded length as lookahead. The information in the lookahead

rather provides information on the behaviour that the lookahead word induces in T. Player Output can delete parts of this information to reduce the size of the lookahead.

The sequential uniformiser that is constructed from the game in the decidability proof can be shown to have bounded delay. So we obtain the following result, showing that sequential k-uniformisation is complete for deterministic transducers.

▶ **Theorem 19.** *Any sequentially uniformisable deterministic transducer is sequentially k-uniformisable for some $k \in \mathbb{N}$ that can be computed from the given transducer.*

6 Conclusion

We have introduced the notion of resynchronisers, which are transformations for synchronisations of transductions. The decision problems of inclusion, equivalence, and sequential uniformisation, which are undecidable for general rational transductions, become decidable modulo rational resynchronisers. Furthermore, we have shown that it is sufficient to consider k-delay resynchronisers in the context of these decision problems. We have analysed two subclasses of transducers, finite-valued transducers and deterministic transducers. For both classes, sequential uniformisation is decidable, and the existence of a sequential uniformiser implies the existence of a sequential k-uniformiser. Additionally, for finite-valued transducers k-inclusion is shown to be complete. One interesting open question is the problem of deciding for a transducer whether it admits a sequential k-uniformiser for some k.

References

1 Marie-Pierre Béal, Olivier Carton, Christophe Prieur, and Jacques Sakarovitch. Squaring transducers: an efficient procedure for deciding functionality and sequentiality. *Theoretical Computer Science*, 292(1):45–63, 2003.

2 Jean Berstel. *Transductions and Context-free Languages*. Teubner-Verlag, December 2009. URL: `http://www-igm.univ-mlv.fr/~berstel/`.

3 Malcolm Bird. The equivalence problem for deterministic two-tape automata. *J. Comput. Syst. Sci.*, 7(2):218–236, 1973. `doi:10.1016/S0022-0000(73)80045-5`.

4 Roderick Bloem, Barbara Jobstmann, Nir Piterman, Amir Pnueli, and Yaniv Sa'ar. Synthesis of reactive(1) designs. *J. Comput. Syst. Sci.*, 78(3):911–938, 2012. `doi:10.1016/j.jcss.2011.08.007`.

5 Mikolaj Bojanczyk. Transducers with origin information. *ICALP*, abs/1309.6124, 2013.

6 Arnaud Carayol and Christof Löding. Uniformization in Automata Theory. In *Proceedings of the 14th Congress of Logic, Methodology and Philosophy of Science Nancy, July 19-26, 2011*, pages 153–178. London: College Publications, 2014.

7 Olivier Carton, Christian Choffrut, and Serge Grigorieff. Decision problems among the main subfamilies of rational relations. *ITA*, 40(2):255–275, 2006. `doi:10.1051/ita:2006005`.

8 Rodrigo de Souza. On the decidability of the equivalence for k-valued transducers. In *Developments in Language Theory, 12th International Conference, DLT 2008, Kyoto, Japan, September 16-19, 2008. Proceedings*, pages 252–263, 2008.

9 Rodrigo de Souza and Nami Kobayashi. A combinatorial study of k-valued rational relations. *Journal of Automata, Languages and Combinatorics*, 13(3/4):207–231, 2008.

10 Samuel Eilenberg. *Automata, Languages, and Machines*. Academic Press, 1974.

11 Diego Figueira and Leonid Libkin. Synchronizing relations on words. *Theory Comput. Syst.*, 57(2):287–318, 2015.

12 Emmanuel Filiot, Ismaël Jecker, Christof Löding, and Sarah Winter. On equivalence and uniformisation problems for finite transducers. *CoRR*, abs/1602.08565, 2016. URL: `http://arxiv.org/abs/1602.08565`.

13 Emmanuel Filiot, Naiyong Jin, and Jean-François Raskin. Antichains and compositional algorithms for LTL synthesis. *Formal Methods in System Design*, 39(3):261–296, 2011.

14 Patrick C. Fischer and Arnold L. Rosenberg. Multitape one-way nonwriting automata. *Journal of Computer and System Sciences*, 2(1):88–101, 1968. doi:10.1016/S0022-0000(68)80006-6.

15 Emily P. Friedman and Sheila A. Greibach. A polynomial time algorithm for deciding the equivalence problem for 2-tape deterministic finite state acceptors. *SIAM J. Comput.*, 11(1):166–183, 1982. doi:10.1137/0211013.

16 Erich Grädel, Wolfgang Thomas, and Thomas Wilke, editors. *Automata, Logics, and Infinite Games*, volume 2500 of *LNCS*. Springer, 2002.

17 Timothy V. Griffiths. The unsolvability of the equivalence problem for lambda-free nondeterministic generalized machines. *Journal of the ACM*, 15(3):409–413, 1968.

18 Eitan M. Gurari and Oscar H. Ibarra. A note on finite-valued and finitely ambiguous transducers. *Theory of Computing Systems*, 16(1):61–66, 1983.

19 Oscar H. Ibarra. The unsolvability of the equivalence problem for ϵ-free NGSM's with unary input (output) alphabet and applications. *SIAM Journal on Computing*, 7(4):524–532, November 1978.

20 Karel Culik II and Juhani Karhumäki. The equivalence of finite valued transducers (on HDT0L languages) is decidable. *TCS: Theoretical Computer Science*, 47, 1986.

21 J. H. Johnson. Do rational equivalence relations have regular cross-sections? In *Lecture Notes in Computer Science*, volume 194 of *LNCS*, pages 300–309. Springer, 1985.

22 Julius Richard Büchi and Lawrence H. Landweber. Solving sequential conditions finite-state strategies. *Trans. Ameri. Math. Soc.*, 138:295–311, 1969.

23 Kojiro Kobayashi. Classification of formal languages by functional binary transductions. *Information and Control*, 15(1):95–109, July 1969.

24 Orna Kupferman, Nir Piterman, and Moshe Y. Vardi. Safraless compositional synthesis. In *Computer Aided Verification, 18th International Conference, CAV 2006*, volume 4144 of *Lecture Notes in Computer Science*, pages 31–44. Springer, 2006.

25 Sylvain Lombardy and Jacques Sakarovitch. Sequential? *Theor. Comput. Sci.*, 356(1-2):224–244, 2006.

26 Maurice Nivat. Transductions des langages de Chomsky. *Ann. de l'Inst. Fourier*, 18:339–456, 1968. in french.

27 Maryse Pelletier and Jacques Sakarovitch. On the representation of finite deterministic 2-tape automata. *TCS: Theoretical Computer Science*, 225, 1999.

28 Amir Pnueli and Roni Rosner. On the synthesis of a reactive module. In *ACM Symposium on Principles of Programming Languages (POPL)*. ACM, 1989.

29 Jacques Sakarovitch. *Elements of Automata Theory*. Cambridge University Press, 2009.

30 Jacques Sakarovitch and Rodrigo de Souza. Lexicographic decomposition of k-valued transducers. *Theory of Computing Systems*, 47(3):758–785, 2010.

31 Wolfgang Thomas. Church's problem and a tour through automata theory. In *Pillars of Computer Science, Essays Dedicated to Boris (Boaz) Trakhtenbrot on the Occasion of His 85th Birthday*, volume 4800 of *Lecture Notes in Computer Science*, pages 635–655. Springer, 2008.

32 Andreas Weber. A decomposition theorem for finite-valued tranducers and an application to the equivalence problem. In *13th International Symposium on Mathematical Foundations of Computer Science, MFCS 1988*, pages 552–562, 1988.

33 Andreas Weber. On the valuedness of finite transducers. *Acta Informatica*, 27(8):749–780, 1989.

34 Andreas Weber and Reinhard Klemm. Economy of description for single-valued transducers. *Information and Computation*, 118(2):327–340, 1995.