

# Approximate Problems for Finite Transducers

Emmanuel Filiot ✉ 

Université libre de Bruxelles

Ismaël Jecker ✉ 

Université de Franche-Comté

Khushraj Madnani ✉ 

Max Planck Institute for Software Systems

Saina Sunny ✉ 

Indian Institute of Technology Goa

---

## Abstract

Finite (word) state transducers extend finite state automata by defining a binary relation over finite words, called *rational relation*. If the rational relation is the graph of a function, this function is said to be rational. The class of sequential functions is a strict subclass of rational functions, defined as the functions recognised by input-deterministic finite state transducers. The class membership problems between those classes are known to be decidable. We consider approximate versions of these problems and show they are decidable as well. This includes the *approximate functionality problem*, which asks whether given a rational relation (by a transducer), is it *close* to a rational function, and the *approximate determinisation problem*, which asks whether a given rational function is close to a sequential function. We prove decidability results for several classical distances, including Hamming and Levenshtein edit distance. Finally, we investigate the *approximate uniformisation problem*, which asks, given a rational relation  $R$ , whether there exists a sequential function that is close to some function uniformising  $R$ . As its exact version, we prove that this problem is undecidable.

**2012 ACM Subject Classification** Theory of computation → Transducers; Theory of computation → Quantitative automata

**Keywords and phrases** Finite state transducers, Edit distance, Determinisation, Functionality

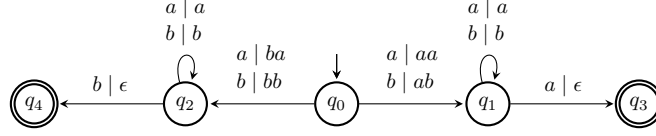
**Funding** *Emmanuel Filiot*: He is a senior research associate at the National Fund for Scientific Research (F.R.S.-FNRS) in Belgium. His work is supported by the FNRS project T011724F.

## 1 Introduction

Finite (state) transducers are a fundamental automata model to compute functions from words to words. The literature on finite state transducers is rich, and dates back to the early days of computer science, where they were called generalised sequential machines [29, 20]. See also [28, 19] and the references therein. Finite state transducers extend finite automata with outputs on their transitions, allowing them to produce none or several symbols. While finite automata recognise languages of (finite) words, finite transducers recognise binary relations from words to words, called *rational relations*. When the rational relation is the graph of a function, it is called a *rational function*. This subclass is decidable within the class of rational relations. In particular, given a finite transducer  $T$ , it is decidable in PTIME whether  $T$  recognises a function [21, 10]. In that case,  $T$  is said to be *functional*. Beyond the fact that it is a natural restriction, the class of functional transducers is of high importance, as many problems known to be undecidable for transducers (such as inclusion and equivalence), become decidable under the functional restriction.

**Determinisation** It turns out that non-determinism is needed for finite transducers to capture rational functions. A canonical example is the function  $f_{\text{last}} : \{a, b\}^* \rightarrow \{a, b\}^*$  that

moves the last symbol upfront. For example,  $f_{\text{last}}(abb) = bab$  and  $f_{\text{last}}(aba) = aab$ . Since the number of symbols that have to be read before reading the last symbol is arbitrarily large, a finite transducer recognising  $f_{\text{last}}$  needs non-determinism to guess the last symbol, as illustrated by the following transducer:



So non-determinism, unlike finite automata, brings some extra expressive power when it comes to finite transducers. On the other hand, non-determinism also yields some inefficiency issues when the input is received sequentially as a stream, because the whole input may have to be stored in memory until the first output symbol can be produced. This motivates the class of *sequential functions*, as the rational functions recognised by *input-deterministic* finite transducers, and the *determinisation problem*: given an arbitrary finite transducer, does it recognise a sequential function? In other words, can it be (input-) determinised? This well-studied problem is known to be decidable in PTIME [10]. The determinisation problem is a central problem in automata theory. It has been for instance extensively considered for weighted automata [27], and a long-standing open problem is whether this problem is decidable for  $(\mathbb{N}, \max, +)$ -automata [25].

**Approximate determinisation** The function  $f_{\text{last}}$  is not sequential, in other words, the latter transducer is not determinisable. It turns out that  $f_{\text{last}}$  is *almost* sequential, in the sense that it is *close to* some sequential function, for instance the identity function  $\text{id}$ . "Close to" can be defined in different ways, by lifting standard distances between words to functions and relations. Two classical examples are the Hamming distance and the Levenshtein distance, which respectively measure the minimal number of letter substitutions (respectively letter substitutions, insertion and deletion) to rewrite a word into another. A distance  $d$  between words is lifted to functions  $f_1, f_2$  with the same input domain, by taking the supremum of  $d(f_1(u), f_2(u))$  for all words  $u$  in their domain. Coming back to our example,  $f_{\text{last}}$  and  $\text{id}$  are close for the edit distance, in the sense that  $d(f_{\text{last}}, \text{id})$  is finite for  $d$  the edit distance, but they are not close for the Hamming distance. This raises a natural and fundamental problem, called *approximate determinisation problem* (for a distance  $d$ ): given a finite transducer recognising a function  $f$ , does there exists a sequential function  $s$  such that  $d(f, s)$  is finite? The approximate determinisation problem has been extensively studied for weighted automata [4, 8, 5] and quantitative automata [6, 7], but, to the best of our knowledge, nothing was known for transducers. However, if both  $f$  and  $s$  are given (by finite transducers), checking whether they are close (for various and classical edit distances) is known to be decidable, even if  $s$  is rational but not sequential [2]. This can be seen as the verification variant of approximate determinisation, while approximate determinisation is rather a synthesis problem, for which only  $f$  is given, and which asks to generate  $s$  if it exists.

**Contributions** In this paper, our main result is the decidability of approximate determinisation of finite transducers, for a family  $\mathcal{E}$  of edit distances, which include Hamming and Levenshtein distances. For exact determinisation, determinisable finite transducers are characterised by the so called *twinning property (TP)* [10, 9, 13], a pattern that requires that the delay between any two outputs on the same input must not increase when taking synchronised cycles of the transducer. As noticed in [2], bounded (Levenshtein) edit distance

is closely related to the notion of word conjugacy. In this paper, we consider an approximate version of the twinning property (**ATP**), with no constraints on the delay, but instead requires that the output words produced on the synchronised loops are conjugate. It turns out that **ATP** is not sufficient to characterise approximately determinisable transducers, and an extra property is needed, the strongly connected twinning property (**STP**), which requires that the **TP** holds within SCCs of the finite transducer. We show that a transducer  $\mathcal{T}$  is approximately determinisable (for Levenshtein distance) iff both **ATP** and **STP** hold and, if they do, we show how to approximately determinise  $\mathcal{T}$ . We also prove that **ATP** and **STP** are both decidable.

For Hamming distance, which only allows letter substitutions, determinizable transducers are characterized by both **STP** and another property called Hamming twinning property (**HTP**), which roughly requires that outputs on synchronised cycles have same length and do not mismatch. We also show that **HTP** is decidable, which entails decidability of approximate determinisation for Hamming distance.

We also consider another fundamental problem: the *approximate functionality problem*. Informally, the approximate functionality problem asks whether a given rational relation  $R$  is *almost* a rational function, in the sense that  $d(R, f)$  is finite for some rational function  $f$ , where  $d(R, f)$  is now the supremum, for all  $(u, v) \in R$ , of  $d(v, f(u))$ . We prove that the approximate functionality problem is decidable for all the distances in  $\mathcal{E}$ . We prove this problem to be decidable for classical distances, including Hamming and Levenshtein.

Finally, we consider the *approximate (sequential) uniformisation problem*. In its exact version, this problem asks whether for a given rational relation  $R$ , there exists a sequential function  $f$  with the same domain, and whose graph is included in  $R$ . This problem is closely related to a synthesis problem, but is unfortunately undecidable [11, 17]. We consider its approximate variant, where instead of requiring that the graph of  $f$  is included in  $R$ , we require that it is close to some function whose graph is included in  $R$ . However, despite this relaxation, we show that the problem is still undecidable.

**Other related works** Variants of the determinisation problem have been considered in the literature [17]. However, this work considers the *exact* determinisation of a transducer  $\mathcal{T}$ , by some sequential transducer which is *close* to  $\mathcal{T}$ , for some notions of structural similarity between transducers. Robustness and continuity notions for finite transducers have been introduced in [22]. While those notions are also based on word distances, the problems considered are different from ours.

## 2 Preliminaries

For every  $k \in \mathbb{N}$ , we let  $[k]$  denote the set  $\{1, \dots, k\}$ .

**Words** Let  $A$  or  $B$  denote finite alphabets of letters. A word is a sequence of letters. The empty word is denoted by  $\epsilon$ . The length of a word is denoted by  $|w|$ , in particular  $|\epsilon| = 0$ . The  $i$ th letter of a word  $w$ , for  $i \in \{1, \dots, |w|\}$ , is denoted by  $w[i]$ . The primitive root of  $w$  is the shortest word  $\rho_w$  such that  $w \in \rho_w^*$ . The set of all finite words over the alphabet  $A$  is denoted by  $A^*$ . A relation  $R \subseteq A^* \times B^*$  is sometimes called a *transduction*, and is said to be *functional* if it is the graph of a function. We let  $\text{dom}(R) = \{u \in A^* \mid \exists v \in B^*, (u, v) \in R\}$  be the *domain* of  $R$ , and for all  $u \in A^*$ , we let  $R(u) = \{v \in B^* \mid (u, v) \in R\}$ . Note that  $u \in \text{dom}(R)$  iff  $R(u) \neq \emptyset$ .

**Finite State Automata and Transducers** A (non-deterministic) finite automaton over an alphabet  $A$  is denoted as a tuple  $\mathcal{A} = (Q, I, \Delta, F)$  where  $Q$  is the finite set of states,  $I \subseteq Q$  the set of initial states,  $F \subseteq Q$  the set of final states, and  $\Delta \subseteq Q \times A \times Q$  the transition relation. A *run* on a word  $\sigma_1 \dots \sigma_n$  is a sequence of states  $q_1 \dots q_{n+1}$  such that  $(q_i, \sigma_i, q_{i+1}) \in \Delta$  for all  $i \in [n]$ . It is accepting if and only if  $q_1 \in I$  and  $q_{n+1} \in F$ . We denote by  $L(\mathcal{A})$  the language accepted by  $\mathcal{A}$ , i.e. the set of words on which there is an accepting run. An automaton is said to be *trim* if for any state  $q$ , there exists at least one accepting run visiting  $q$ . When  $\mathcal{A}$  is deterministic, we denote the transition function as  $\delta : Q \times A \rightarrow Q$ .

A *rational transducer*  $\mathcal{T}$  over an input alphabet  $A$  and an output alphabet  $B$  is a (non-deterministic) finite automaton over a finite subset of  $A^* \times B^*$ . A transition  $(p, (u, v), q)$  of  $\mathcal{T}$  is often denoted  $p \xrightarrow{u|v} q$ . More generally, we write  $p \xrightarrow{u_1 \dots u_n | v_1 \dots v_n} q$  whenever there exists a run  $\rho$  of  $\mathcal{T}$  from  $p$  to  $q$  on  $(u_1, v_1) \dots (u_n, v_n)$ . The *relation recognised* by  $\mathcal{T}$ , denoted as  $R_{\mathcal{T}} \subseteq A^* \times B^*$ , is defined as  $R_{\mathcal{T}} = \{(u, v) \mid \exists q_0 \in I, q_f \in F, q_0 \xrightarrow{u|v} q_f\}$ . The relations recognised by rational transducers are called *rational relations*.

When rational transducers recognise functions, it is often convenient to restrict their transitions to input words of length one exactly, modulo the possibility of producing a word on accepting states. This defines the so-called class of real-time transducers, which is expressively equivalent to rational transducers when restricted to functions. Formally, a *real-time transducer* (or simply, a transducer in the sequel)  $\mathcal{T}$  over input alphabet  $A$  and output alphabet  $B$  is given by a tuple  $\mathcal{T} = (Q, I, \Delta, F, \lambda)$  where  $(Q, I, \Delta, F)$  is a finite automaton over a finite subset of  $A \times B^*$ , and  $\lambda : F \rightarrow B^*$  is a final output function.

Given a word  $u = a_1 \dots a_n \in A^*$  where  $a_i \in A$  for all  $i$ , a run  $\rho$  of  $\mathcal{T}$  over  $u$  is a sequence  $q_0 \dots q_n$  such that  $q_0 \in I$  and  $(q_{i-1}, a_i, v_i, q_i) \in \Delta$  for all  $i \in [n]$ . The *input word* of the run  $\rho$  is  $u = a_1 \dots a_n$  and the *output word* of  $\rho$  is  $v_1 \dots v_n \cdot \lambda(q_n)$  if  $q_n$  is a final state; otherwise,  $v_1 \dots v_n$ . As for rational transducers, the relation  $R_{\mathcal{T}}$  recognised by  $\mathcal{T}$  is defined as the set of pairs  $(u, v)$  such that  $u$  (resp.  $v$ ) is the input (resp. output) word of some accepting run. We often confuse  $\mathcal{T}$  with  $R_{\mathcal{T}}$ , and may write  $\text{dom}(\mathcal{T})$  for  $\text{dom}(R_{\mathcal{T}})$ , or  $\mathcal{T}(u)$  for  $R_{\mathcal{T}}(u)$ .

The *underlying automaton* of  $\mathcal{T}$  is the automaton obtained by projection on inputs, i.e. the automaton  $\mathcal{A} = (Q, I, \Delta', F)$  such that  $\Delta' = \{(q, a, q') \mid \exists (q, a, v, q') \in \Delta\}$ . Note that  $\text{dom}(\mathcal{T}) = L(\mathcal{A})$ . The transducer  $\mathcal{T}$  is said to be *trim* if its underlying automaton is trim.

The cartesian product, denoted  $\mathcal{T}_1 \times \mathcal{T}_2$ , of two transducers  $\mathcal{T}_i = (Q_i, I_i, \Delta_i, F_i, \lambda_i)$ ,  $i \in [2]$ , is the transducer  $(Q_1 \times Q_2, I_1 \times I_2, \Delta, F_1 \times F_2, \lambda)$  where  $((p_1, p_2), a, (v_1, v_2), (q_1, q_2)) \in \Delta$  if  $(p_i, a, v_i, q_i) \in \Delta_i$  for  $i \in [2]$ , and,  $\lambda(p_1, p_2) = (\lambda_1(p_1), \lambda_2(p_2))$  for  $(p_1, p_2) \in F_1 \times F_2$ .

**(Sub)classes of rational functions** Let  $\mathcal{T}$  be a real-time transducer. When  $R_{\mathcal{T}}$  is functional,  $\mathcal{T}$  is said to be functional as well, and the functions recognised by functional transducers are called *rational functions*. If the underlying automaton of  $\mathcal{T}$  is unambiguous (i.e., has at most one accepting run on any input),  $\mathcal{T}$  is referred to as an *unambiguous transducer*. It is well-known that a function is recognised by real-time transducer iff, it is recognised by a rational transducer [23] iff, it is recognised by an unambiguous transducer [15].

*Sequential transducers* are those whose underlying automaton is deterministic and they define functions known as *sequential functions*. In that case, we denote the transition function as  $\delta : Q \times A \rightarrow Q \times B^*$ . The functions recognised by transducers that are finite disjoint unions of sequential transducers are called *multi-sequential functions* [14, 24]. The symmetric counterpart of multi-sequential is the class of *series-sequential functions*.

A transducer  $\mathcal{T}$  is *series-sequential* if it is a finite disjoint union of sequential transducers  $\mathcal{D}_1, \dots, \mathcal{D}_k$  where additionally, for every  $1 \leq i < k$ , there is a single transition from a (not necessarily final) state  $q_i$  of  $\mathcal{D}_i$  to the initial state of the next transducer  $\mathcal{D}_{i+1}$ . Moreover, the

initial state of  $\mathcal{T}$  is the initial state of  $\mathcal{D}_1$  (the initial states of  $\mathcal{D}_i$  is not considered as initial in  $\mathcal{T}$ , for all  $2 \leq i \leq k$ ). In particular, non-determinism is allowed, for all  $1 \leq i < k$ , only in state  $q_i$ , from which it is possible to move to  $\mathcal{D}_{i+1}$  or to stay in  $\mathcal{D}_i$ . We denote<sup>1</sup> such a transducer as  $\mathcal{D}_1 \cdots \mathcal{D}_k$ . A function is *series-sequential* if it is recognised by a series-sequential transducer.

**Distances between words and word functions** We recall that a metric on a set  $E$  is a mapping  $d : E^2 \rightarrow \mathbb{R}^+ \cup \{\infty\}$  satisfying the separation, symmetry and triangle inequality axioms. Classical metrics between finite words are the *edit distances*. An edit distance between two words is the minimum number of edit operations required to rewrite a word to another if possible, and  $\infty$  otherwise. Depending on the set of allowed edit operations, we get different edit distances. Table 1 gives widely used edit distances with their edit operations.

Edit Distances	Notation	Edit Operations
Hamming	$d_h$	letter-to-letter substitutions
Longest Common Subsequence	$d_{lcs}$	insertions and deletions
Levenshtein	$d_l$	insertions, deletions and substitutions
Damerau-Levenshtein	$d_{dl}$	insertions, deletions, substitutions and swapping adjacent letters

■ **Table 1** Edit Distances [2]

Distances between words can be lifted to that between functions from words to words.

► **Definition 1** (Metric over Functions [2]). *Let  $d$  be a metric on words over some alphabet  $B$ . Given two partial functions  $f_1, f_2 : A^* \rightarrow B^*$ , the distance between  $f_1$  and  $f_2$  is defined as*

$$d(f_1, f_2) = \begin{cases} \sup \{ d(f_1(w), f_2(w)) \mid w \in \text{dom}(f_1) \} & \text{if } \text{dom}(f_1) = \text{dom}(f_2) \\ \infty & \text{otherwise} \end{cases}$$

It is shown that  $d$  is a metric over functions (Proposition 3.2 of [2]). The distance between two functional transducers is defined as the distance between the functions they recognise. A notion closely related to the distance between functions is *diameter* of a relation. The diameter of a relation  $R$  w.r.t. metric  $d$ , denoted by  $\text{dia}_d(R)$  is defined to be the supremum of the distance of every pair in the relation, i.e.,  $\text{dia}_d(R) = \sup \{ d(u, v) \mid (u, v) \in R \}$ .

The distance between rational functions and diameter of rational relation w.r.t. the metrics given in Table 1 are computable [2]. The computability of distance and diameter relies on the notion of conjugacy. Two words  $u$  and  $v$  are conjugate if there exist words  $x, y$  such that  $u = xy$  and  $v = yx$ . In other words, they are cyclic shifts of each other. For example, words  $aabb$  and  $baaa$  are conjugate with  $x = aa$  and  $y = bb$ . But  $aabb$  and  $abab$  are not conjugate. Conjugacy is an equivalence relation over words.

► **Proposition 2.** *Let  $x, y, x', y', u, v$  be words and  $c, C \in \mathbb{N}$ . For any metric  $d$  in Table 1, if  $d(xu^ky, x'v^{ck}y') \leq C$  for all  $k \geq 0$ , then  $|u| = |v^c|$  and the primitive roots of  $u$  and  $v$  are conjugate.*

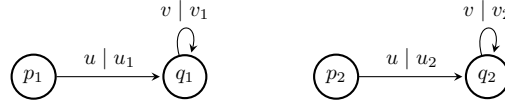
**Proof.** TOPROVE 0 ◀

<sup>1</sup> This notation should not be confused with the split-sum operator of [3], which is semantically different.

► **Proposition 3** ([2]). *Given a rational relation  $R$  defined by a transducer  $\mathcal{T}$ ,  $\text{dia}_d(R) < \infty$  for  $d \in \{d_l, d_{lcs}, d_{dl}\}$  if and only if every pair of input-output words generated by loops in  $\mathcal{T}$  are conjugate.*

### 3 Twinning Properties

The class of sequential functions has been characterised by transducers satisfying the so called *twinning property*. In this section, we recall this property and introduce three variants — approximate twinning property, strongly connected twinning property and Hamming twinning property. We also show that these properties on transducers are decidable as well as independent of the representation of the transducers. All these properties are expressed as particular conditions on twinning patterns. A *twinning pattern* for a transducer  $\mathcal{T} = (Q, I, \Delta, F, \lambda)$  over  $A, B$  is a tuple  $(p_1, q_1, p_2, q_2, u, v, u_1, v_1, u_2, v_2) \in Q^4 \times (A^*)^2 \times (B^*)^4$  such that the following runs (graphically depicted) exist:



The longest common prefix of any two words is denoted by  $u \wedge v$ . The *delay* between  $u$  and  $v$ , denoted by  $\text{delay}(u, v)$ , is the pair  $(u', v')$  such that  $u = (u \wedge v)u'$  and  $v = (u \wedge v)v'$ .

► **Definition 4** (Twinning Property (TP)). *Let  $\mathcal{T}$  be a trim transducer. We say that  $\mathcal{T}$  satisfies twinning property if for each twinning pattern  $(p_1, q_1, p_2, q_2, u, v, u_1, v_1, u_2, v_2)$  such that  $p_1, p_2$  are initial,  $\text{delay}(u_1, u_2) = \text{delay}(u_1v_1, u_2v_2)$  holds.*

It is well-known that a function recognised by a transducer  $\mathcal{T}$  is sequential iff  $\mathcal{T}$  satisfies the twinning property [12, 10]. We now define its approximate variant.

► **Definition 5** (Approximate Twinning Property (ATP)). *A trim transducer  $\mathcal{T}$  satisfies approximate twinning property if for each twinning pattern  $(p_1, q_1, p_2, q_2, u, v, u_1, v_1, u_2, v_2)$  where  $p_1, p_2$  are initial, the words  $v_1$  and  $v_2$  are conjugate.*

► **Definition 6** (Strongly Connected Twinning Property (STP)). *A trim transducer  $\mathcal{T}$  satisfies strongly connected twinning property if for each twinning pattern  $(p_1, q_1, p_2, q_2, u, v, u_1, v_1, u_2, v_2)$  such that  $p_1 = p_2$  (not necessarily initial) and  $p_1, q_1, q_2$  are in the same strongly connected component,  $\text{delay}(u_1, u_2) = \text{delay}(u_1v_1, u_2v_2)$  holds.*

► **Definition 7** (Hamming Twinning Property (HTP)). *A trim transducer  $\mathcal{T}$  satisfies Hamming twinning property if for each twinning pattern  $(p_1, q_1, p_2, q_2, u, v, u_1, v_1, u_2, v_2)$  such that  $p_1, p_2$  are initial, it holds that  $|v_1| = |v_2|$  and there is no mismatch between  $v_1$  and  $v_2$ , i.e. for all position  $i \in [\max(|u_1|, |u_2|), \min(|u_1v_1|, |u_2v_2|)]$ ,  $(u_1v_1)[i] = (u_2v_2)[i]$ .*

► **Proposition 8.** *Each trim transducer satisfying TP also satisfies ATP, STP and HTP.*

**Proof.** TOPROVE 1

The following lemma states that ATP, STP and HTP is preserved between transducers up to finite edit distance, and so in particular between equivalent transducers. This shows that these properties do not depend on the representation of the transductions, not even on the representation of close transductions.

► **Lemma 3.9.** *Let  $\mathcal{T}$  and  $\mathcal{S}$  be two trim transducers satisfying  $\text{dom}(\mathcal{T}) = \text{dom}(\mathcal{S})$ , and such that there exists an edit distance  $d$  in Table 1 and a constant  $C \in \mathbb{N}$  for which*

$$d(\mathcal{T}(u), \mathcal{S}(u)) < C \text{ for all } u \in \text{dom}(\mathcal{T}).$$

*Then for every  $P \in \{\mathbf{ATP}, \mathbf{STP}\}$ ,  $\mathcal{S}$  satisfies  $P$  if and only if  $\mathcal{T}$  satisfies  $P$ . The statement also holds for  $d = d_h$  and  $P = \mathbf{HTP}$ .*

**Proof.** TOPROVE 2 ◀

We now show that checking each of the variants of the twinning property is decidable.

► **Lemma 3.10.** *It is decidable whether a transducer satisfies  $\mathbf{ATP}$ ,  $\mathbf{STP}$  and  $\mathbf{HTP}$ .*

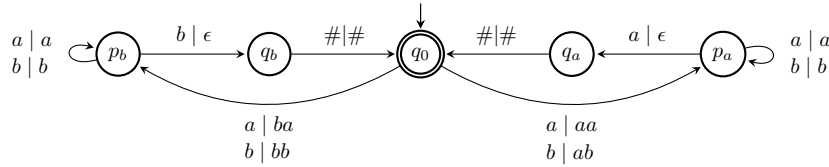
**Proof.** TOPROVE 3 ◀

## 4 Approximate determinisation of rational functions

In this section, we define *approximately determinisable functions* and give decidable properties on transducers to characterise them.

► **Definition 11** (Approximate determinisation). *A rational function  $f : A^* \rightarrow B^*$  is approximately determinisable w.r.t. metric  $d$  if there exists a sequential function  $g : A^* \rightarrow B^*$  such that  $d(f, g) < \infty$ . In this case, we also say that  $g$  approximately determinises  $f$  w.r.t.  $d$ .*

► **Example 12.** The function  $f_{\text{last}}$  from the introduction is approximately determinisable w.r.t. Levenshtein, while not w.r.t. Hamming distance. The function  $f_{\text{last}}^*$  (depicted below) that maps  $u_1\# \cdots u_n\#$ , for  $n \geq 1$ , to  $f_{\text{last}}(u_1)\# \cdots f_{\text{last}}(u_n)\#$  for some separator  $\#$  is approximately determinisable w.r.t. neither Levenshtein nor Hamming distance.



The approximate determinisation problem asks whether a rational function given as a functional transducer is approximately determinisable. We prove the following.

► **Theorem 13.** *The approximate determinisation problem for rational functions w.r.t. Levenshtein family  $(d_l, d_{lcs}, d_{dl})$  and Hamming  $(d_h)$  distance are decidable.*

To prove the theorem, we show that **ATP** and **STP** characterise rational functions that can be approximately determinised w.r.t. Levenshtein family (Lemma 4.18). Similarly, we establish that **HTP** and **STP** characterise rational functions that can be approximately determinised w.r.t. Hamming distance (Lemma 4.23). Theorem 13 then follows directly, as these three properties are decidable (Lemma 3.10). We now outline the proof strategy. The full proof for Levenshtein family is presented in Section 4.1, while the proof for Hamming distance, which follows a similar approach, is deferred to Section 4.2.



**Levenshtein family** We show with Proposition 14 that **ATP** and **STP** are necessary conditions for approximate determinisation with respect to Levenshtein family, a consequence of Lemma 3.9. The main challenge lies in proving that these properties are sufficient. To prove it, we first show that **ATP** alone suffices for certain subclasses of functional transducers: it enables the approximate determinisation of multi-sequential (Lemma 4.16) and unambiguous series-sequential (Lemma 4.17) functions. However, for rational functions in general, **ATP** does not suffice. For example, the transducer above for  $f_{\text{last}}^*$  satisfies **ATP** but is not approximately determinisable. To extend this result to all rational functions, we incorporate **STP**. Given a functional transducer  $\mathcal{T}$  satisfying **STP**, we transform each strongly connected component of  $\mathcal{T}$  into a sequential transducer, effectively decomposing  $\mathcal{T}$  into a finite union of concatenations of sequential transducers. We then leverage our results for series-sequential and multi-sequential functions to approximate this structure with a sequential function (Lemma 4.18).

Figure 1 illustrates the main construction technique used in these proofs: Starting with a transducer  $\mathcal{T}$  that we aim to approximate, we construct a sequential transducer  $\mathcal{D}_1$  as follows. We apply the powerset construction to  $\mathcal{T}$ , introducing a distinguished state (marked by a  $\bullet$  in the figure) in each subset. The output is determined by the distinguished state's production. If the distinguished state reaches a point where it has no continuation, we simply transition to another distinguished state. We show that **ATP**, combined with a carefully chosen priority scheme for selecting distinguished states, ensures bounded Levenshtein distance.

**Hamming Distance** The proof strategy is similar to the Levenshtein setting: We first show with Proposition 20 that **HTP** and **STP** are necessary conditions for approximate determinisation with respect to Hamming distance, we show that **HTP** alone suffices for the approximate determinisation of multi-sequential (Lemma 4.21) and series-sequential (Lemma 4.22) functions, and then we conclude by using **STP** to transform functional transducers into a finite unions of concatenations of sequential transducers (Lemma 4.23).

While the core ideas remain similar to those used for the Levenshtein family, the constructions required for the Hamming distance, illustrated in Figure 1, are more intricate. Approximating the transducer  $\mathcal{T}$  with respect to the Levenshtein distance (as shown by  $\mathcal{D}_1$  in the figure) allows us, at each step, to select a run, produce its output, and disregard other possible runs. However, for the Hamming distance, it is crucial to carefully track the length difference between the produced output and the potential outputs of alternative runs. For instance, compare the outputs of  $\mathcal{T}$ ,  $\mathcal{D}_1$ , and  $\mathcal{D}_2$  after reading *babcccc* :

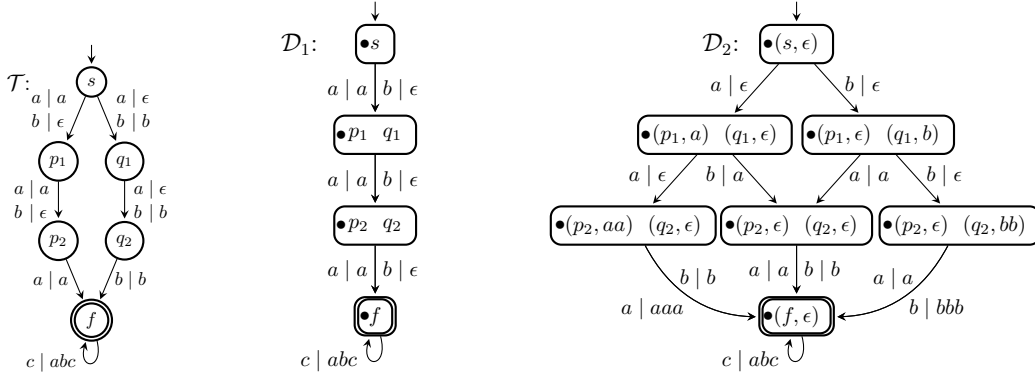
$$\begin{aligned}\mathcal{T}(\text{babcccc}) &= \text{bbabcabcabcab}, \\ \mathcal{D}_1(\text{babcccc}) &= \text{aabcabcabcab}, \\ \mathcal{D}_2(\text{babcccc}) &= \text{ababcabcabcab}.\end{aligned}$$

We observe that after reading the input *bab*,  $\mathcal{D}_1$  realizes that its distinguished state is incorrect and jumps to another state. However, this shift causes a misalignment with  $\mathcal{T}$ , and reading additional *c*'s results in arbitrarily many mismatches.<sup>2</sup> In contrast,  $\mathcal{D}_2$  keeps in memory the delay relative to other runs. Although it may still introduce mismatches along the way, it ensures that when the distinguished run terminates, it adjusts the output while transitioning to another run, preventing long-term misalignment with  $\mathcal{T}$ .

---

<sup>2</sup> The Levenshtein distance remains bounded, as inserting a letter at the start resynchronizes both outputs.





■ **Figure 1** An unambiguous non-deterministic transducer  $\mathcal{T}$ , along with two sequential approximations  $\mathcal{D}_1$  and  $\mathcal{D}_2$  with respect to the Levenshtein distance, respectively Hamming distance.

#### 4.1 Approximate determinisation for Levenshtein Family

We give a decidable characterisation of approximately determinisable rational functions w.r.t. Levenshtein family of distances — Levenshtein ( $d_l$ ), Longest common subsequence ( $d_{lcs}$ ) and Damerau-Levenshtein ( $d_{dl}$ ). They are all equivalent up to boundedness (Lemma 2.1 and Remark 7 of [2]), i.e., for any two rational functions  $f, g$ ,  $d_{lcs}(f, g) < \infty \iff d_l(f, g) < \infty \iff d_{dl}(f, g) < \infty$ . We show that a rational function is approximately determinisable w.r.t. Levenshtein family if and only if the transducer that defines the function satisfies both **ATP** and **STP**. One direction is a consequence of Lemma 3.9 as follows.

► **Proposition 14.** *If a rational function given by a trim transducer  $\mathcal{T}$  is approximately determinisable w.r.t. a metric  $d \in \{d_l, d_{lcs}, d_{dl}\}$  then  $\mathcal{T}$  satisfies both **ATP** and **STP**.*

**Proof.** TOPROVE 4 ◀

Towards proving the other direction, we prove the following lemma, which provides a bound on the distance between the output words produced by distinct runs of a transducer on the same prefix of an input word using Proposition 3.

► **Lemma 4.15.** *Let  $\mathcal{T}$  be a trim transducer satisfying the **ATP**. Then, there exists a constant  $N_{\mathcal{T}} \in \mathbb{N}$  such that for any two output words  $v, v' \in B^*$  produced via two distinct runs of  $\mathcal{T}$  from an initial state on the same prefix of an input word,  $d(v, v') \leq N_{\mathcal{T}}$  for  $d \in \{d_l, d_{lcs}, d_{dl}\}$ .*

**Proof.** TOPROVE 5 ◀

For subclasses of rational functions, namely multi-sequential and series-sequential functions, we show that **ATP** is a sufficient condition for approximate determinisation.

► **Lemma 4.16.** *A multi-sequential function given by a trim transducer  $\mathcal{T}$  is approximately determinisable w.r.t. a metric  $d \in \{d_l, d_{lcs}, d_{dl}\}$  iff  $\mathcal{T}$  satisfies the **ATP**.*

**Proof.** TOPROVE 6 ◀

The characterisation of Lemma 4.16 also holds for series-sequential functions.

► **Lemma 4.17.** *Let  $\mathcal{T} = \mathcal{D}_1 \cdots \mathcal{D}_k$  be an unambiguous transducer where each  $\mathcal{D}_i$  is a sequential trim transducer for  $i \in [k], k > 1$ . The series-sequential function defined by  $\mathcal{T}$  is approximately determinisable w.r.t. a metric  $d \in \{d_l, d_{lcs}, d_{dl}\}$  if and only if  $\mathcal{T}$  satisfies **ATP**.*

Proof. **TOPROVE 7** ◀

We extend the characterisation to rational functions, where **STP** is also required to decompose the function into a finite union of series-sequential functions, which can then be transformed into a multi-sequential function using the properties of **ATP**.

► **Lemma 4.18.** *A rational function given by a trim transducer  $\mathcal{T}$  is approximately determinisable w.r.t. a metric  $d \in \{d_l, d_{lcs}, d_{dl}\}$  iff  $\mathcal{T}$  satisfies both **ATP** and **STP**.*

Proof. **TOPROVE 8** ◀

## 4.2 Approximate determinisation for Hamming distance

In this subsection, we prove that a rational function is approximately determinisable w.r.t. Hamming distance if and only if the transducer that defines the function satisfies both **HTP** and **STP**. The proof strategy is similar to the setting of Levenshtein family.

► **Lemma 4.19.** *Let  $T$  be a trim transducer satisfying the **HTP**. Then, there exists a constant  $N_T \in \mathbb{N}$  such that for any two output words  $w, w' \in B^*$  produced by  $T$  while processing the same input word via two distinct runs from an initial state, the following properties hold:*

1. *The length difference satisfies  $||w| - |w'|| < N_T$ .*
2. *The number of positions where  $w$  and  $w'$  differ is at most  $N_T$ .*

Proof. **TOPROVE 9** ◀

► **Proposition 20.** *If a rational relation given by a transducer  $\mathcal{T}$  is approximately determinisable w.r.t. Hamming distance then  $\mathcal{T}$  satisfies both **HTP** as well as **STP**.*

Proof. **TOPROVE 10** ◀

► **Lemma 4.21.** *A multi-sequential function given by a transducer  $\mathcal{T}$  is approximately determinisable w.r.t. the Hamming distance if and only if  $\mathcal{T}$  satisfies the **HTP**.*

Proof. **TOPROVE 11** ◀

► **Lemma 4.22.** *Let  $\mathcal{U} = \mathcal{D}_1 \mathcal{D}_2 \cdots \mathcal{D}_k$  be an unambiguous transducer, where  $\mathcal{D}_i$  is a sequential trim transducer for  $i \in [k], k > 1$ . The series-sequential function given by the transducer  $\mathcal{U}$  is approximately determinisable w.r.t. Hamming distance if and only if  $\mathcal{U}$  satisfies the Hamming twinning property.*

Proof. **TOPROVE 12** ◀

► **Lemma 4.23.** *A rational function given by a transducer  $\mathcal{T}$  is approximately determinisable w.r.t. the Hamming distance if and only if  $\mathcal{T}$  satisfies both the Hamming and the strongly connected twinning property.*

Proof. **TOPROVE 13** ◀

## 5 Approximate decision problems for rational relations

In this section, we consider two possible generalisations of the approximate determinisation problem to rational relations. We describe those generalisations informally. The first one asks to decide whether a rational relation is close to some rational function. We call it the approximate functionality problem. The second one, that we still call determinisation problem, amounts to decide, given a rational relation  $R$ , whether it is *almost* a sequential function. The third generalisation we consider is an approximate uniformisation problem, which asks, given  $R$ , whether there exists a sequential function  $s$  which is close to a function  $f$ , whose graph is included in  $R$ . We however show that this problem is undecidable.

We now proceed with the formal definitions and statements of our results. First, we need to extend the notion of distance from functions of words to binary relations of words. Towards this, we use Hausdorff distance between languages, defined as

$$d_H(L, L') = \max \left\{ \sup_{w \in L} \inf_{w' \in L'} d(w, w'), \sup_{w' \in L'} \inf_{w \in L} d(w, w') \right\}.$$

Given a metric  $d$  on words, and two relations  $R_1, R_2 \subseteq A^* \times B^*$ , the distance between  $R_1$  and  $R_2$  is defined as follows.

$$d(R_1, R_2) = \begin{cases} \sup \{ d_H(R_1(w), R_2(w)) \mid w \in \text{dom}(R_1) \} & \text{if } \text{dom}(R_1) = \text{dom}(R_2) \\ \infty & \text{otherwise} \end{cases}$$

Therefore,  $d(R_1, R_2) < \infty$  iff there exists  $k \in \mathbb{N}$  such that for all word  $u$  in the domain and any output  $v_1$  of  $R_1$  on  $u$ , there exists some output  $v_2$  of  $R_2$  on  $u$ , such that  $d(v_1, v_2) < k$ , and symmetrically. In fact,  $d$  is a metric on relations as shown below.

► **Proposition 24.**  *$d$  is a metric on relations.*

**Proof.** TOPROVE 14 ◀

### Approximate functionality problem

► **Definition 25** (Approximate Functionalisation). *A rational relation  $R$  is approximately functionalisable w.r.t. a metric  $d$  if there exists a rational function  $f$  such that  $d(R, f) < \infty$ .*

The approximate functionality problem asks, given  $R$  represented by a transducer, whether it is approximately functionalisable w.r.t.  $d$ . Towards this, we define the following value for a relation  $R$  and metric  $d$ , which measures how different are output words over the same input. More precisely, it is the maximal distance between any two output words over the same input word, by  $R$ :

$$\text{diff}_d(R) = \sup_{u \in \text{dom}(R)} \sup_{v_1, v_2 \in R(u)} d(v_1, v_2)$$

► **Lemma 5.26.** *For a rational relation  $R$  given as a rational transducer,  $\text{diff}_d(R)$  is computable for all metrics given in Table 1.*

**Proof.** TOPROVE 15 ◀

We now characterise rational relations which are approximately functionalisable.

► **Lemma 5.27.** *A rational relation  $R$  is approximately functionalisable w.r.t. a metric  $d$  if and only if  $\text{diff}_d(R) < \infty$ .*

**Proof.** TOPROVE 16 ◀

Since  $\text{diff}_d(R)$  is computable (see Lemma 5.26), we get the following result.

► **Theorem 28.** *The approximate functionality problem for rational relations given as rational transducers w.r.t. a metric given in Table 1 is decidable.*

**Approximate determinisation problem** A rational relation  $R$  is said to be approximately determinisable for a metric  $d$  if it is almost a sequential function with respect to  $d$ . Formally, it means that there exists a sequential function  $f$  such that  $d(R, f) < \infty$ . We show that the associated decision problem, that we call approximate determinisation problem, is decidable for Levenshtein family of distances. In fact, the characterisation for approximate determinisation of rational functions also holds for rational relations.

► **Lemma 5.29.** *A rational relation defined by a trim transducer  $\mathcal{T}$  is approximate determinisable w.r.t. Levenshtein family  $(d_l, d_{lcs}, d_{dl})$  if and only if  $\mathcal{T}$  satisfies **ATP** and **STP**.*

**Proof.** TOPROVE 17 ◀

Since **ATP** and **STP** are decidable for transducers (Lemma 3.10), we obtain the following.

► **Theorem 30.** *The approximate determinisation problem for rational relations given as rational transducers w.r.t. a metric  $d \in \{d_l, d_{lcs}, d_{dl}\}$  is decidable.*

**Approximate uniformisation problem** Given a relation  $R \subseteq A^* \times B^*$ , a *uniformiser* of  $R$  is a function  $U : A^* \rightarrow B^*$  such that  $\text{dom}(R) = \text{dom}(U)$  and for all  $u \in \text{dom}(R)$ , it holds that  $(u, U(u)) \in R$ . It is known that any rational relation admits a rational uniformiser [26], which is not true if we seek for a sequential uniformiser. Moreover, the problem of deciding whether a given rational relation admits a sequential uniformiser is undecidable [11]. We consider an approximate variant of this problem.

► **Definition 31** (Approximate Uniformisation). *A rational relation  $R$  is approximately uniformisable w.r.t. a metric  $d$  if there exists a uniformiser  $U$  of  $R$  and a sequential function  $f$  such that  $d(U, f) < \infty$ . In that case, we say that  $R$  is  $d$ -approximate uniformisable by a sequential function.*

In other words,  $R$  is  $d$ -approximate uniformisable by a sequential function  $f$  iff there exists an integer  $k \in \mathbb{N}$  such that for all  $u \in \text{dom}(R)$ , there exists  $(u, v) \in R$  with  $d(v, f(u)) \leq k$ .

► **Theorem 32.** *Checking whether a rational relation is  $d$ -approximate uniformisable for  $d \in \{d_l, d_{lcs}, d_{dl}\}$  is undecidable.*

**Proof.** TOPROVE 18 ◀

## 6 Future works

In this paper, we proved that approximate determinisation is decidable for functional transducers, by checking various twinning properties. We have shown that **HTP** and **STP** are decidable in PTIME, which entails that approximate determinization of functional transducers for the Hamming distance is decidable in PTIME. For the other distances, such as Levenshtein distance, the time complexity is doubly exponential, as deciding conjugacy of a rational relation, and hence **ATP**, is doubly exponential [1]. We conjecture that this is suboptimal, and leave as future work finding a better upper-bound.

We show that approximate uniformisation is undecidable for Levenshtein family. We leave the case of Hamming distance as future work. The current undecidability proof for

Levenshtein family, based on PCP, heavily requires that the lengths of the output words produced on transitions can differ, which may not guarantee that the total output lengths are the same, which is necessary to have a finite Hamming distance. Our proof also does not extend to the letter-to-letter setting, where both the given transducer and the required uniformiser process *and produce* a single letter on every transition. This problem is closely related to the standard Church synthesis for regular specifications, with the modification that the strategy to be synthesized is allowed to make a bounded number of errors. To our knowledge, this variant has not been studied in the literature.

Finally, we studied approximate decision problems up to finite distance. Another interesting question is to consider their “up to distance  $k$ ” variant, where  $k$  is given as input.

---

## References

---

- 1 C. Aiswarya, Amaldev Manuel, and Saina Sunny. Deciding conjugacy of a rational relation - (extended abstract). In *28th International Conference on Developments in Language Theory, DLT 2024*, volume 14791 of *Lecture Notes in Computer Science*, pages 37–50. Springer, 2024.
- 2 C. Aiswarya, Amaldev Manuel, and Saina Sunny. Edit Distance of Finite State Transducers. In *51st International Colloquium on Automata, Languages, and Programming ICALP 2024*, volume 297 of *LIPIcs*, pages 125:1–125:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024.
- 3 Rajeev Alur, Adam Freilich, and Mukund Raghothaman. Regular combinators for string transformations. In *Joint Meeting of the 23rd EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), CSL-LICS ’14*, pages 9:1–9:10. ACM, 2014.
- 4 Benjamin Aminof, Orna Kupferman, and Robby Lampert. Rigorous approximated determinization of weighted automata. In *Proceedings of the 26th Annual IEEE Symposium on Logic in Computer Science, LICS 2011*, pages 345–354. IEEE Computer Society, 2011.
- 5 Benjamin Aminof, Orna Kupferman, and Robby Lampert. Rigorous approximated determinization of weighted automata. *Theoretical Computer Science*, 480:104–117, 2013.
- 6 Udi Boker and Thomas A. Henzinger. Approximate determinization of quantitative automata. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2012*, volume 18 of *LIPIcs*, pages 362–373. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2012.
- 7 Udi Boker and Thomas A. Henzinger. Exact and approximate determinization of discounted-sum automata. *Logical Methods in Computer Science*, 10(1), 2014.
- 8 Adam L. Buchsbaum, Raffaele Giancarlo, and Jeffery R. Westbrook. An approximate determinization algorithm for weighted finite-state automata. *Algorithmica*, 30(4):503–526, 2001.
- 9 Marie-Pierre Béal and Olivier Carton. Determinization of transducers over finite and infinite words. *Theoretical Computer Science*, 289(1):225–251, 2002.
- 10 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.
- 11 Arnaud Carayol and Christof Loeding. Uniformization in automata theory. In *14th International Congress of Logic, Methodology and Philosophy of Science*, pages 153–178. London: College Publications, 2015.
- 12 Christian Choffrut. Une caractérisation des fonctions séquentielles et des fonctions sous-séquentielles en tant que relations rationnelles. *Theoretical Computer Science*, 5(3):325–337, 1977.
- 13 Christian Choffrut and Serge Grigorieff. Uniformization of rational relations. In *Jewels are Forever, Contributions on Theoretical Computer Science in Honor of Arto Salomaa*, pages 59–71. Springer, 1999.

- 14 Christian Choffrut and Marcel Paul Schützenberger. Décomposition de fonctions rationnelles. In *3rd Annual Symposium on Theoretical Aspects of Computer Science STACS 1986*, volume 210 of *Lecture Notes in Computer Science*, pages 213–226. Springer, 1986.
- 15 Samuel Eilenberg. *Automata, languages, and machines. vol. A*. Pure and applied mathematics. Academic Press, 1974.
- 16 Diego Figueira and Leonid Libkin. Path logics for querying graphs: Combining expressiveness and efficiency. In *30th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2015*, pages 329–340. IEEE Computer Society, 2015.
- 17 Emmanuel Filiot, Ismaël Jecker, Christof Löding, and Sarah Winter. On equivalence and uniformisation problems for finite transducers. In *43rd International Colloquium on Automata, Languages, and Programming ICALP 2016*, volume 55 of *LIPIcs*, pages 125:1–125:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016.
- 18 Emmanuel Filiot, Nicolas Mazzocchi, and Jean-François Raskin. A pattern logic for automata with outputs. *International Journal of Foundations of Computer Science*, 31(6):711–748, 2020.
- 19 Emmanuel Filiot and Pierre-Alain Reynier. Transducers, logic and algebra for functions of finite words. *ACM SIGLOG News*, 3(3):4–19, 2016.
- 20 Seymour Ginsburg and Gene F. Rose. A characterization of machine mappings. *Journal of Symbolic Logic*, 33(3):468–468, 1968.
- 21 Eitan M. Gurari and Oscar H. Ibarra. A note on finitely-valued and finitely ambiguous transducers. *Mathematical Systems Theory*, 16(1):61–66, 1983.
- 22 Thomas A. Henzinger, Jan Otop, and Roopsha Samanta. Lipschitz robustness of finite-state transducers. In *34th International Conference on Foundation of Software Technology and Theoretical Computer Science, FSTTCS 2014*, volume 29 of *LIPIcs*, pages 431–443. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2014.
- 23 J. Berstel. *Transductions and Context-Free Languages*. Teubner, Stuttgart, 1979.
- 24 Ismaël Jecker and Emmanuel Filiot. Multi-sequential word relations. *International Journal of Foundations of Computer Science*, 29(2):271–296, 2018.
- 25 Daniel Kirsten and Sylvain Lombardy. Deciding unambiguity and sequentiality of polynomially ambiguous min-plus automata. In *26th International Symposium on Theoretical Aspects of Computer Science STACS 2009*, volume 3 of *LIPIcs*, pages 589–600. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Germany, 2009.
- 26 K. Kobayashi. Classification of formal languages by functional binary transductions. *Information and Control*, 15(1):95–109, 1969.
- 27 Mehryar Mohri. Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311, 1997.
- 28 Anca Muscholl and Gabriele Puppis. The many facets of string transducers (invited talk). In *36th International Symposium on Theoretical Aspects of Computer Science STACS 2019*, volume 126 of *LIPIcs*, pages 2:1–2:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.
- 29 George N. Raney. Sequential functions. *Journal of the ACM*, 5(2):177–180, 1958.