

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

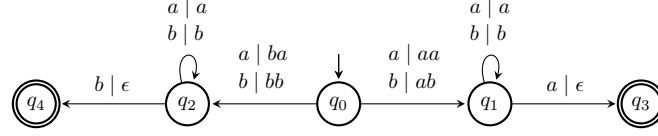
Funding *Emmanuel Filiot*: He is a senior research associate at the National Fund for Scientific Research (F.R.S.-FNRS) in Belgium. His work is supported by the FNRS project T011724F.

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_{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_{last} is not sequential, in other words, the latter transducer is not determinisable. It turns out that f_{last} is *almost* sequential, in the sense that it is *close to* some sequential function, for instance the identity function 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_{last} and 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 ϵ . 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 ρ_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 ρ 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, Δ, 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 ρ 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 ρ is $u = a_1 \dots a_n$ and the *output word* of ρ 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 denote¹ 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 ∞ 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 $bbaa$ 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. Since $d(xu^ky, x'v^{ck}y') \leq C$, we get $|u| = |v^c|$. Otherwise, as k increases the length difference of xu^ky and $x'v^{ck}y'$ increases, and hence their distance will not be bounded.

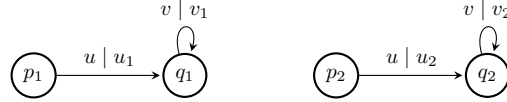
¹ This notation should not be confused with the split-sum operator of [3], which is semantically different.

Since $|u| = |v^c|$, either both u and v are nonempty, or $u = v = \epsilon$. In the latter case, u and v are conjugate. Assume that u and v are nonempty words. Take $k = 2^{|u|+|v|}$. Since $d(xu^ky, x'v^{ck}y') \leq C$ there exist large portions of u 's and v 's that match. In fact, u 's and v 's overlap at least of length $|u| + |v|$. By Fine and Wilf's² theorem, the primitive roots of u and v are conjugate. \blacktriangleleft

► **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 $i \in [\max(|u_1|, |u_2|), \min(|u_1v_1|, |u_2v_2|)]$, $(u_1v_1)[i] = (u_2v_2)[i]$.*

² The Fine and Wilf's theorem states that if some powers of two words u and v share a common factor of length $|u| + |v| - \text{gcd}(u, v)$ then their primitive roots are conjugate.

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

Proof. Let \mathcal{T} be a trim transducer which satisfies **TP**.

We first show it satisfies **ATP**. Consider a 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. Then, $\text{delay}(u_1, u_2) = \text{delay}(u_1 v_1, u_2 v_2) = \text{delay}(u_1 v_1^i, u_2 v_2^i)$ for all $i \geq 0$, which implies that $|v_1| = |v_2|$. Therefore, for all $n \geq 0$, there exists i such that v_1^i and v_2^i have a common factor of length at least n . By Fine and Wilf's theorem, it implies that v_1 and v_2 have conjugate primitive roots, and since $|v_1| = |v_2|$, we get that v_1 and v_2 are conjugate.

Now, consider **STP**. Let $(p, q_1, p, q_2, u, v, u_1, v_1, u_2, v_2)$ be some twinning pattern where p, q_1, q_2 are in the same SCC. Since \mathcal{T} is trim, p is accessible from the initial state q_0 , by a run $q_0 \xrightarrow{\alpha|\beta} p$. Note that $(q_0, q_1, q_0, q_2, \alpha u, v, \beta u_1, v_1, \beta u_2, v_2)$ is also a twinning pattern. Hence by the twinning property, $\text{delay}(\beta u_1, \beta u_2) = \text{delay}(\beta u_1 v_1, \beta u_2 v_2)$, from which we can conclude as $\text{delay}(\beta u_1, \beta u_2) = \text{delay}(u_1, u_2)$ and $\text{delay}(\beta u_1 v_1, \beta u_2 v_2) = \text{delay}(u_1 v_1, u_2 v_2)$.

Finally, if the **HTP** is not satisfied, then considering a twinning pattern as in the definition of the **HTP**, there is a position i such that $(u_1 v_1)[i] \neq (u_2 v_2)[i]$, and therefore $\text{delay}(u_1 v_1, u_2 v_2) \neq \text{delay}(u_1 v_1 v_1, u_2 v_2 v_2)$, which fails the **TP**, contradiction. ◀

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. Let us suppose that \mathcal{S} satisfies $P \in \{\mathbf{ATP}, \mathbf{STP}, \mathbf{HTP}\}$, and show that so does \mathcal{T} . The converse is symmetric.

For ATP Let $(p_1, p_2, q_1, q_2, u, v, u_1, u_2, v_1, v_2)$ be an instance of the twinning pattern for \mathcal{T} with p_1 and q_1 initial. Since \mathcal{T} is trim, there exist words w, w', w_1, w_2 and two accepting states p_f, q_f such that:

$$\begin{array}{ccccccc} p_1 & \xrightarrow{u|u_1} \mathcal{T} & p_2 & \xrightarrow{v|v_1} \mathcal{T} & p_2 & \xrightarrow{w|w_1} \mathcal{T} & p_f \\ q_1 & \xrightarrow{u|u_2} \mathcal{T} & q_2 & \xrightarrow{v|v_2} \mathcal{T} & q_2 & \xrightarrow{w'|w_2} \mathcal{T} & q_f \end{array}$$

Since $\text{dom}(\mathcal{T}) = \text{dom}(\mathcal{S})$, for all $i \geq 1$, $uv^i w, uv^i w' \in \text{dom}(\mathcal{T}) = \text{dom}(\mathcal{S})$. Iterating the loop of \mathcal{T} on v sufficiently many times causes \mathcal{S} to also loop on some power of v . Formally, there exist $a, b, c \in \mathbb{N}$, states $p'_1, p'_2, q'_1, q'_2, p'_f, q'_f$ and words $u'_1, u'_2, v'_1, v'_2, w'_1, w'_2$ such that:

$$\begin{array}{ccccccc} p_1 & \xrightarrow{uv^a|u_1 v_1^a} \mathcal{T} & p_2 & \xrightarrow{v^b|v_1^b} \mathcal{T} & p_2 & \xrightarrow{v^c w|v_1^c w_1} \mathcal{T} & p_f \\ q_1 & \xrightarrow{uv^a|u_2 v_2^a} \mathcal{T} & q_2 & \xrightarrow{v^b|v_2^b} \mathcal{T} & q_2 & \xrightarrow{v^c w'|v_2^c w_2} \mathcal{T} & q_f \\ p'_1 & \xrightarrow{uv^a|u'_1} \mathcal{S} & p'_2 & \xrightarrow{v^b|v'_1} \mathcal{S} & p'_2 & \xrightarrow{v^c w|w'_1} \mathcal{S} & p'_f \\ q'_1 & \xrightarrow{uv^a|u'_2} \mathcal{S} & q'_2 & \xrightarrow{v^b|v'_2} \mathcal{S} & q'_2 & \xrightarrow{v^c w'|w'_2} \mathcal{S} & q'_f \end{array}$$

Let $C \in \mathbb{N}$ such that $d(\mathcal{T}(x), \mathcal{S}(x)) \leq C$ for all $x \in \text{dom}(\mathcal{T})$. Then in particular for all $k \geq 0$:

$$d(u_1 v_1^{a+kb+c} w_1, u'_1 v_1^k w'_1) \leq C \text{ and } d(u_2 v_2^{a+kb+c} w_2, u'_2 v_2^k w'_2) \leq C.$$

From the above inequalities, using Proposition 2, we get that ρ_{v_i} and $\rho_{v'_i}$, the primitive roots of v_i and v'_i respectively, are conjugate and $|v_i^b| = |v'_i|$ for $i \in [2]$. Since \mathcal{S} satisfies the **ATP**, we also have that v'_1 and v'_2 are conjugate, and hence their primitive roots are conjugate and $|v'_1| = |v'_2|$. By transitivity of the conjugacy relation, we get that ρ_{v_1} and ρ_{v_2} are conjugate. Further, $|v_1| = |v_2|$ since $|\rho_{v_1}| = |\rho_{v_2}|$ and $|v_1^b| = |v'_1| = |v'_2| = |v_2^b|$. Therefore, v_1 and v_2 are conjugate, and thus the **ATP** holds for \mathcal{T} as well.

For STP We suppose that \mathcal{S} satisfies the **STP**, and show that \mathcal{T} satisfies it too. Let us pick a twinning pattern $(p, q_1, p, q_2, u, v, u_1, v_1, u_2, v_2)$ in \mathcal{T} such that p, q_1, q_2 are in the same strongly connected component, as in the definition of the **STP** (Definition 6):

$$\begin{aligned} \rho_1 : & \quad q_I \xrightarrow{x|x_0} \mathcal{T} p \xrightarrow{u|u_1} \mathcal{T} q_1 \xrightarrow{v|v_1} \mathcal{T} q_1 \xrightarrow{w^+|w_1} \mathcal{T} p \xrightarrow{z|z_0} \mathcal{T} q_F, \\ \rho_2 : & \quad q_I \xrightarrow{x|x_0} \mathcal{T} p \xrightarrow{u|u_2} \mathcal{T} q_2 \xrightarrow{v|v_2} \mathcal{T} q_2 \xrightarrow{w^-|w_2} \mathcal{T} p \xrightarrow{z|z_0} \mathcal{T} q_F. \end{aligned}$$

Here the runs with inputs x and z are witnesses of the fact that \mathcal{T} is trim and the runs with inputs w^+ and w^- are witnesses of the fact that p, q_1, q_2 are in the same strongly connected component. Similar to the proof of **ATP**, we use these runs to build an instance of **STP** in \mathcal{S} . To that end, let N denote the number of states of \mathcal{S} . We study the run of \mathcal{S} on the word :

$$y = xy_1 y_2 \dots y_{2N} z, \text{ where } y_i = (uv^{N^2} w^+)^i uv^{N^2} w^- \text{ for every } 1 \leq i \leq 2N.$$

Remark that $y \in \text{dom}(\mathcal{T})$, as a run over y can be obtained by pumping and combining the subruns of ρ_1 and ρ_2 . Therefore, as $\text{dom}(\mathcal{T}) = \text{dom}(\mathcal{S})$, there is also a run of \mathcal{S} on y , and since \mathcal{S} has N states this run will visit some state r three times between the y_i . Formally, there exist $1 \leq i < j < k \leq 2N$ such that, if we let $\bar{x} = xy_1 y_2 \dots y_{i-1}$ and $\bar{z} = y_k y_{k+1} \dots y_{2N} z$, we get:

$$r_I \xrightarrow{\bar{x}|\bar{x}_0} \mathcal{S} r \xrightarrow{y_i \dots y_{j-1}|y'} \mathcal{S} r \xrightarrow{y_j \dots y_{k-1}|y''} \mathcal{S} r \xrightarrow{\bar{z}|\bar{z}_0} \mathcal{S} r_F.$$

Now remark that the input words labeling both loops on r start with the prefix $(uv^{N^2} w^+)^i uv^{N^2}$, which is followed by w^- in the first loop and w^+ in the second. Since \mathcal{S} has N states, the last block of v^{N^2} occurring in this common prefix will visit a synchronised subloop in both loops. Formally, there is a decomposition $\bar{u}\bar{u}'\bar{v}\bar{w}^+$ of $y_i \dots y_{j-1}$ and a decomposition $\bar{u}\bar{u}'\bar{v}\bar{w}^-$ of $y_j \dots y_{k-1}$ satisfying:

$$\begin{aligned} \bar{u} &= (uv^{N^2} w^+)^i, & \bar{u}' &= uv^a, & \bar{v} &= v^b, & \bar{w}^+ &= v^c w^- y_{i+1} \dots y_{j-1}, \\ \bar{w}^- &= v^c w^+ (uv^{N^2} w^+)^{j-i-1} uv^{N^2} w^- y_{j+1} \dots y_{k-1}, \\ \bar{\rho}_1 : & \quad s_I \xrightarrow{\bar{x}|\bar{x}_0} \mathcal{S} r \xrightarrow{\bar{u}|\bar{u}_1} \mathcal{S} r_1 \xrightarrow{\bar{u}'|\bar{u}'_1} \mathcal{S} s_1 \xrightarrow{\bar{v}|\bar{v}_1} \mathcal{S} s_1 \xrightarrow{\bar{w}^+|\bar{w}_1} \mathcal{S} r \xrightarrow{\bar{z}|\bar{z}_0} \mathcal{S} s_F, \\ \bar{\rho}_2 : & \quad s_I \xrightarrow{\bar{x}|\bar{x}_0} \mathcal{S} r \xrightarrow{\bar{u}|\bar{u}_2} \mathcal{S} r_2 \xrightarrow{\bar{u}'|\bar{u}'_2} \mathcal{S} s_2 \xrightarrow{\bar{v}|\bar{v}_2} \mathcal{S} s_2 \xrightarrow{\bar{w}^-|\bar{w}_2} \mathcal{S} r \xrightarrow{\bar{z}|\bar{z}_0} \mathcal{S} s_F. \end{aligned}$$

Since \mathcal{S} satisfies the **STP** by hypothesis, $\text{delay}(\bar{u}_1 \bar{u}'_1, \bar{u}_2 \bar{u}'_2) = \text{delay}(\bar{u}_1 \bar{u}'_1 \bar{v}_1, \bar{u}_2 \bar{u}'_2 \bar{v}_2)$. We now combine this with the fact that the distance between the outputs of \mathcal{T} and \mathcal{S} is bounded by C to show that $\text{delay}(u_1, u_2) = \text{delay}(u_1 v_1, u_2 v_2)$, which concludes the proof of the lemma. First, let us pump the subruns of ρ_1 and ρ_2 to match the inputs of $\bar{\rho}_1$ and $\bar{\rho}_2$:

$$\begin{aligned} \rho'_1 : & \quad q_I \xrightarrow{\bar{x}|x'_0} \mathcal{T} p \xrightarrow{\bar{u}|u_0} \mathcal{T} p \xrightarrow{\bar{u}'|u_1 v_1^a} \mathcal{T} q_1 \xrightarrow{\bar{v}|v_1^b} \mathcal{T} q_1 \xrightarrow{\bar{w}^+|w'_1} \mathcal{T} p \xrightarrow{\bar{z}|z'_0} \mathcal{T} q_F, \\ \rho'_2 : & \quad q_I \xrightarrow{\bar{x}|x'_0} \mathcal{T} p \xrightarrow{\bar{u}|u_0} \mathcal{T} p \xrightarrow{\bar{u}'|u_2 v_2^a} \mathcal{T} q_2 \xrightarrow{\bar{v}|v_2^b} \mathcal{T} q_2 \xrightarrow{\bar{w}^-|w'_2} \mathcal{T} p \xrightarrow{\bar{z}|z'_0} \mathcal{T} q_F. \end{aligned}$$

We then introduce a standard property of the delay, proved for instance in [18] :

▷ **Claim 10.** For every words s_1, s_2, t_1, t_2 we have that $\text{delay}(s_1, s_2) = \text{delay}(s_1 t_1, s_2 t_2)$ if and only if either $t_1 = t_2 = \epsilon$, or $|t_1| = |t_2|$ and there is no mismatch between $s_1 t_1^m$ and $s_2 t_2^m$ for all $m \in \mathbb{N}$.

Therefore, in order to show that $\text{delay}(u_1, u_2) = \text{delay}(u_1 v_1, u_2 v_2)$, we now suppose that $v_1 \neq \epsilon$ or $v_2 \neq \epsilon$, and we show that $|v_1| = |v_2|$ and there is no mismatch between $u_1 v_1^m$ and $u_2 v_2^m$ for all $m \in \mathbb{N}$.

Same output length We first show that $|v_1| = |v_2|$. Observe that for every $\lambda \in \mathbb{N}$ we have :

$$\begin{aligned} \mathcal{S}(\bar{x} \bar{u} u v^{a+\lambda \cdot b} \bar{w}^+ \bar{z}) &= \bar{x}_0 \bar{u}_1 \bar{u}'_1 \bar{v}_1^\lambda \bar{w}_1 \bar{z}_0, & \mathcal{S}(\bar{x} \bar{u} u v^{a+\lambda \cdot b} \bar{w}^- \bar{z}) &= \bar{x}_0 \bar{u}_2 \bar{u}'_2 \bar{v}_2^\lambda \bar{w}_2 \bar{z}_0, \\ \mathcal{T}(\bar{x} \bar{u} u v^{a+\lambda \cdot b} \bar{w}^+ \bar{z}) &= x'_0 u_0 u_1 v_1^{a+\lambda \cdot b} w'_1 z'_0, & \mathcal{T}(\bar{x} \bar{u} u v^{a+\lambda \cdot b} \bar{w}^- \bar{z}) &= x'_0 u_0 u_2 v_2^{a+\lambda \cdot b} w'_2 z'_0. \end{aligned}$$

By the supposition in the statement of the Lemma, $d(\bar{x}_0 \bar{u}_1 \bar{u}'_1 \bar{v}_1^\lambda \bar{w}_1 \bar{z}_0, x'_0 u_0 u_1 v_1^{a+\lambda \cdot b} w'_1 z'_0) \leq C$ and $d(\bar{x}_0 \bar{u}_2 \bar{u}'_2 \bar{v}_2^\lambda \bar{w}_2 \bar{z}_0, x'_0 u_0 u_2 v_2^{a+\lambda \cdot b} w'_2 z'_0) \leq C$. This implies that $|v_1|^b = \bar{v}_1$ and $|v_2|^b = \bar{v}_2$, as otherwise choosing λ large enough yields outputs with a length difference too large to be fixed with at most C number of edits. Hence, it suffices to show $|\bar{v}_1| = |\bar{v}_2|$. This is implied by the fact $\text{delay}(\bar{u}_1 \bar{u}'_1, \bar{u}_2 \bar{u}'_2) = \text{delay}(\bar{u}_1 \bar{u}'_1 \bar{v}_1, \bar{u}_2 \bar{u}'_2 \bar{v}_2)$ along with the Claim 10.

No mismatch As we have shown that $|v_1| = |v_2|$, let us now pick $m \in \mathbb{N}$ and show that there is no mismatch between $u_1 v_1^m$ and $u_2 v_2^m$ to conclude the proof via Claim 10. Remark that the words $u_1 v_1^m$ and $u_2 v_2^m$ appear as subwords of the outputs of the runs ρ'_1 and ρ'_2 . We now derive three key equations on output words (Equation (1), Equation (2) and Equation (3)) by pumping and assembling parts of ρ'_1 , ρ'_2 , $\bar{\rho}_1$ and $\bar{\rho}_2$. We then combine these equations to get the desired result. We begin by introducing some notation. Let $M \in \mathbb{N}$ satisfying

$$M > \frac{m \cdot |v_1| + C + \max(|\bar{x}_0|, |x'_0|)}{b \cdot |v_1|}.$$

Moreover, we consider the following output words :

$$\begin{aligned} \alpha_1 &= u_0 u_1 v_1^{a+Mb}, & \alpha_2 &= u_0 u_2 v_2^{a+Mb}, & \beta_1 &= w'_1, & \beta_2 &= w'_2, \\ \bar{\alpha}_1 &= \bar{u}_1 \bar{u}'_1 \bar{v}_1^M, & \bar{\alpha}_2 &= \bar{u}_2 \bar{u}'_2 \bar{v}_2^M, & \bar{\beta}_1 &= \bar{w}_1, & \bar{\beta}_2 &= \bar{w}_2. \end{aligned}$$

First, remark that for every $\lambda \in \mathbb{N}$ we have :

$$\begin{aligned} \mathcal{S}(\bar{x}(\bar{u} \bar{u}' \bar{v}^M \bar{w}^+)^{\lambda} \bar{z}) &= \bar{x}_0 (\bar{\alpha}_1 \bar{\beta}_1)^{\lambda} \bar{z}_0, & \mathcal{S}(\bar{x}(\bar{u} \bar{u}' \bar{v}^M \bar{w}^-)^{\lambda} \bar{z}) &= \bar{x}_0 (\bar{\alpha}_2 \bar{\beta}_2)^{\lambda} \bar{z}_0, \\ \mathcal{T}(\bar{x}(\bar{u} \bar{u}' \bar{v}^M \bar{w}^+)^{\lambda} \bar{z}) &= x'_0 (\alpha_1 \beta_1)^{\lambda} z'_0, & \mathcal{T}(\bar{x}(\bar{u} \bar{u}' \bar{v}^M \bar{w}^-)^{\lambda} \bar{z}) &= x'_0 (\alpha_2 \beta_2)^{\lambda} z'_0. \end{aligned}$$

Since the edit distance between the outputs of \mathcal{T} and \mathcal{S} is bounded by C , this yields :

$$|\alpha_1 \beta_1| = |\bar{\alpha}_1 \bar{\beta}_1| \text{ and } |\alpha_2 \beta_2| = |\bar{\alpha}_2 \bar{\beta}_2|. \quad (1)$$

Second, for every $\lambda \in \mathbb{N}$ we have :

$$\begin{aligned} \mathcal{S}(\bar{x}(\bar{u} \bar{u}' \bar{v}^M \bar{w}^+ \bar{u} \bar{u}' \bar{v}^M \bar{w}^+ \bar{u} \bar{u}' \bar{v}^M \bar{w}^-)^{\lambda} \bar{z}) &= \bar{x}_0 (\bar{\alpha}_1 \bar{\beta}_1 \bar{\alpha}_1 \bar{\beta}_1 \bar{\alpha}_2 \bar{\beta}_2)^{\lambda} \bar{z}_0, \\ \mathcal{T}(\bar{x}(\bar{u} \bar{u}' \bar{v}^M \bar{w}^+ \bar{u} \bar{u}' \bar{v}^M \bar{w}^+ \bar{u} \bar{u}' \bar{v}^M \bar{w}^-)^{\lambda} \bar{z}) &= x'_0 (\alpha_1 \beta_1 \alpha_1 \beta_1 \alpha_2 \beta_2)^{\lambda} z'_0. \end{aligned}$$

If we set $\lambda = C$, the hypothesis in the statement of the Lemma yields :

$$d(x'_0 (\alpha_1 \beta_1 \alpha_1 \beta_1 \alpha_2 \beta_2)^C z'_0, \bar{x}_0 (\bar{\alpha}_1 \bar{\beta}_1 \bar{\alpha}_1 \bar{\beta}_1 \bar{\alpha}_2 \bar{\beta}_2)^C \bar{z}_0) < C.$$

Observe that at least one copy of the subword $\alpha_1\beta_1\alpha_1\beta_1\alpha_2\beta_2$ is preserved (i.e., no internal deletion or insertion) while editing $x'_0(\alpha_1\beta_1\alpha_1\beta_1\alpha_2\beta_2)^\lambda z'_0$ into $\bar{x}_0(\bar{\alpha}_1\bar{\beta}_1\bar{\alpha}_1\bar{\beta}_1\bar{\alpha}_2\bar{\beta}_2)^\lambda \bar{z}_0$ via $C-1$ edits. As this copy of $\alpha_1\beta_1\alpha_1\beta_1\alpha_2\beta_2$ can be shifted by at most $C-1$ letters, and as it is originally shifted by $\max(|x'_0|, |\bar{x}_0|)$ letters with respect to its matching copy of $\bar{\alpha}_1\bar{\beta}_1\bar{\alpha}_1\bar{\beta}_1\bar{\alpha}_2\bar{\beta}_2$ in $\bar{x}_0(\bar{\alpha}_1\bar{\beta}_1\bar{\alpha}_1\bar{\beta}_1\bar{\alpha}_2\bar{\beta}_2)^C \bar{z}_0$, there exists $\delta \in \mathbb{N}$ satisfying $|\delta| < C + \max(|\bar{x}_0|, |x'_0|)$ such that :

$$\begin{aligned} \alpha_1\beta_1\alpha_1\beta_1\alpha_2\beta_2[i] &= \bar{\alpha}_1\bar{\beta}_1\bar{\alpha}_1\bar{\beta}_1\bar{\alpha}_2\bar{\beta}_2[i + \delta] \\ \text{for all } \max(0, \delta) < i &\leq \min(|\alpha_1\beta_1\alpha_1\beta_1\alpha_2\beta_2|, |\alpha_1\beta_1\alpha_1\beta_1\alpha_2\beta_2| - \delta). \end{aligned} \quad (2)$$

Finally, as $\text{delay}(\bar{u}_1\bar{u}'_1, \bar{u}_2\bar{u}'_2) = \text{delay}(\bar{u}_1\bar{u}'_1\bar{v}_1, \bar{u}_2\bar{u}'_2\bar{v}_2)$, Claim 10 yields that

$$\bar{\alpha}_1[i] = \bar{\alpha}_2[i] \text{ for every } 1 \leq i \leq \min(|\bar{\alpha}_1|, |\bar{\alpha}_2|). \quad (3)$$

We now combine all equations. For every $\max(0, \delta) < i \leq \min(|\alpha_1|, |\alpha_2| - \delta)$ we get :

$$\begin{aligned} \alpha_1[i] &= \bar{\alpha}_1[i + \delta] && \text{by (2),} \\ &= \bar{\alpha}_2[i + \delta] && \text{by (3),} \\ &= (\bar{\alpha}_1\bar{\beta}_1\bar{\alpha}_1\bar{\beta}_1\bar{\alpha}_2)[|\bar{\alpha}_1\bar{\beta}_1\bar{\alpha}_1\bar{\beta}_1| + i + \delta] \\ &= (\bar{\alpha}_1\bar{\beta}_1\bar{\alpha}_1\bar{\beta}_1\bar{\alpha}_2)[|\alpha_1\beta_1\alpha_1\beta_1| + i + \delta] && \text{by (1),} \\ &= (\alpha_1\beta_1\alpha_1\beta_1\alpha_2)[|\alpha_1\beta_1\alpha_1\beta_1| + i] && \text{by (2),} \\ &= \alpha_2[i]. \end{aligned}$$

If $\delta < 0$ we still need to address some values of i . For every $1 \leq i \leq -\delta$ we get :

$$\begin{aligned} \alpha_1[i] &= (\alpha_1\beta_1\alpha_1)[|\alpha_1\beta_1| + i] \\ &= (\alpha_1\beta_1\alpha_1)[|\bar{\alpha}_1\bar{\beta}_1| + i] && \text{by (1),} \\ &= (\bar{\alpha}_1\bar{\beta}_1)[|\bar{\alpha}_1\bar{\beta}_1| + i + \delta] && \text{by (2) since } i + \delta < 1, \\ &= (\bar{\alpha}_1\bar{\beta}_1\bar{\alpha}_1\bar{\beta}_1)[|\bar{\alpha}_1\bar{\beta}_1\bar{\alpha}_1\bar{\beta}_1| + i + \delta] \\ &= (\bar{\alpha}_1\bar{\beta}_1\bar{\alpha}_1\bar{\beta}_1)[|\alpha_1\beta_1\alpha_1\beta_1| + i + \delta] && \text{by (1),} \\ &= (\alpha_1\beta_1\alpha_1\beta_1\alpha_2)[|\alpha_1\beta_1\alpha_1\beta_1| + i] && \text{by (2) since } i \geq 1, \\ &= \alpha_2[i]. \end{aligned}$$

These last two series of equations ensure that α_1 and α_2 have a large common prefix :

$$\alpha_1[i] = \alpha_2[i] \text{ for every } 1 \leq i \leq \min(|\alpha_1|, |\alpha_2| - \delta). \quad (4)$$

Then we can conclude by noticing that, thanks to the choice of

$$M > \frac{m \cdot |v_1| + C + \max(|\bar{x}_0|, |x'_0|)}{b \cdot |v_1|} > \frac{m}{b} + \frac{|\delta|}{b \cdot |v_1|},$$

the words $\alpha_1 = u_0u_1v_1^{a+Mb}$ and $\alpha_2 = u_0u_2v_2^{a+Mb}$ can be written as $u_0u_1v_1^mv_1^\mu$, respectively $u_0u_2v_2^mv_2^\mu$, for some $\mu \in \mathbb{N}$ such that $|v_1^\mu| = |v_2^\mu| > |\delta|$. Therefore, as desired, Equation (4) implies that there is no mismatch between $u_1v_1^m$ and $u_2v_2^m$.

For HTP Suppose that \mathcal{T} satisfies the **HTP**, and \mathcal{S} does not. Then, there exists a twinning pattern $(p_1, q_1, p_2, q_2, u, v, u_1, v_1, u_2, v_2)$ such that either (1) $|v_1| \neq |v_2|$ or (2) there is a position $i \in]\min(|u_1|, |u_2|), \max(|u_1v_1|, |u_2v_2|]$ such that $(uv_1)[i] \neq (uv_2)[i]$. In both cases, we construct a family of input words u for which the Hamming distance $d_h(\mathcal{S}(uv_i^\lambda), \mathcal{T}(u))$ gets arbitrarily large, which contradicts the assumption.

For case (1), it is immediate, as iterating the loop creates outputs, on the same input, with arbitrarily large length differences.

For case (2), assuming $|v_1| = |v_2|$, it is also immediate, because for all $j \geq 1$, uv_1^j and uv_2^j contain at least j mismatching positions, as for every $1 \leq k \leq j$,

$$(uv_1^j)[i + (k-1)|v_1|] = (uv_1)[i] \neq (uv_2)[i] = (uv_2^j)[i + (k-1)|v_1|]. \quad \blacktriangleleft$$

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

► **Lemma 3.11.** *It is decidable whether a transducer satisfies **ATP**, **STP** and **HTP**.*

Proof. Let \mathcal{T} be a transducer that defines a relation R .

Deciding ATP Let \mathcal{T}^2 be the cartesian product of \mathcal{T} by itself. Verifying whether \mathcal{T} satisfies **ATP** reduces to checking that every loop in \mathcal{T}^2 produces a pair of conjugate output words. For each state $(p, q) \in \mathcal{T}^2$, we define a rational relation $R_{(p,q)}$ consisting of pairs of output words produced by the loops in \mathcal{T}^2 rooted at (p, q) . The transducer defining $R_{(p,q)}$ is obtained from \mathcal{T}^2 by disregarding the inputs and setting both the initial and final state to be (p, q) . It is known that we can decide whether every pair of words in a given rational relation is conjugate [1], which entails decidability of **ATP**.

Deciding STP The twinning property is decidable in polynomial time [10]. To decide **STP**, we first decompose the transducer into SCCs (in polynomial time). Then, for each SCC and each state p , the SCC is seen as a transducer with initial state p on which we check **TP**.

Deciding HTP We prove that it can be decided in PTIME whether a transducer does not satisfy **HTP**. The **HTP** can be decomposed as a conjunction of two properties : **HTP**_{lgth} on lengths and **HTP**_{mism} on mismatches. The (negation of) **HTP**_{lgth} can be directly expressed in the pattern logic of [18], whose model-checking is in PTIME. Then, we reduce the problem of deciding the **HTP**_{mism} to the emptiness problem of a Parikh automaton of polynomial size, in the size of the transducer \mathcal{T} . We recall that a Parikh automaton of dimension d is an NFA extended with vectors in \mathbb{Z}^d on its transitions. It accepts a word if there exists a run of the NFA that reaches an accepting state, and the sum of the vectors seen along the run belongs to some semi-linear set $W \subseteq \mathbb{Z}^d$, given for example as a Presburger formula $\phi(x_1, \dots, x_d)$. The emptiness problem is known to be decidable in PTIME when both d and ϕ are constant [16, 18]. Our reduction follows standard ideas, and falls in this particular case. Let us give a bit more details.

A twinning pattern t can be encoded as a word u_t over $\Delta^2 \cup \{\#\}$, where Δ is the transition relation of \mathcal{T} . In this construction, a run of \mathcal{T} is seen as a sequence of transitions. So, a twinning pattern consists of four runs, r_1, r'_1 and r_2, r'_2 where for all $i = 1, 2$, r_i is the run such that $p_i \xrightarrow{u|u_i} q_i$ and r'_i is the run such that $q_i \xrightarrow{v|v_i} q_i$. The four runs are encoded as a word $u_t = (r_1 \otimes r_2)\#(r'_1 \otimes r'_2)$ where $r_1 \otimes r_2$ is the convolution of r_1 and r_2 , i.e. the overlapping of r_1 and r_2 (and similarly for $r'_1 \otimes r'_2$). It should be clear that the set of words u_t for all twinning pattern t such that p_1 and p_2 are initial states is a regular language, recognizable by an NFA of polysize in the size of \mathcal{T} .

In order to check the condition that there is a mismatch between u_1v_1 and u_2v_2 on a position common to v_1 and v_2 , the NFA is extended with two counters c_1 and c_2 , and the linear acceptance condition $c_1 = c_2 = 0$ (making it a Parikh automaton). Those two counters are used to guess the mismatching position. Initially, using an ϵ -loop, those two counters are incremented in parallel. At the end of this first phase, they therefore hold the same value, say $i \in \mathbb{N}$. Then, the Parikh automaton is built in such a way that it checks that $|u_1| < i \leq |u_1v_1|$ and $|u_2| < i \leq |u_2v_2|$, and $(u_1v_1)[i] \neq (u_2v_2)[i]$. To do so, while reading any transition of r_j producing some word α_j , $j = 1, 2$, c_j is decremented by $|\alpha_j|$. The same is done when reading transitions of r'_j , but the automaton can non-deterministically guess that the counter c_j is equal to 0, and store (in its state) the corresponding letter in α_j . From then on, it never decrements the counter c_j again. When the whole input $(r_1 \otimes r_2)\#(r'_1 \otimes r'_2)$

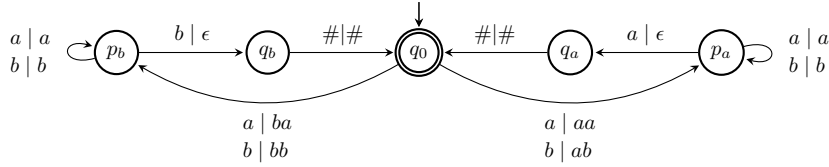
has been read, the automaton has two letters stored in its state. It accepts the input, if those two letters are different, and if $c_1 = c_2 = 0$. ◀

4 Approximate determinisation of rational functions

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

► **Definition 12** (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 13.** The function f_{last} from the introduction is approximately determinisable w.r.t. Levenshtein, while not w.r.t. Hamming distance. The function f_{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.



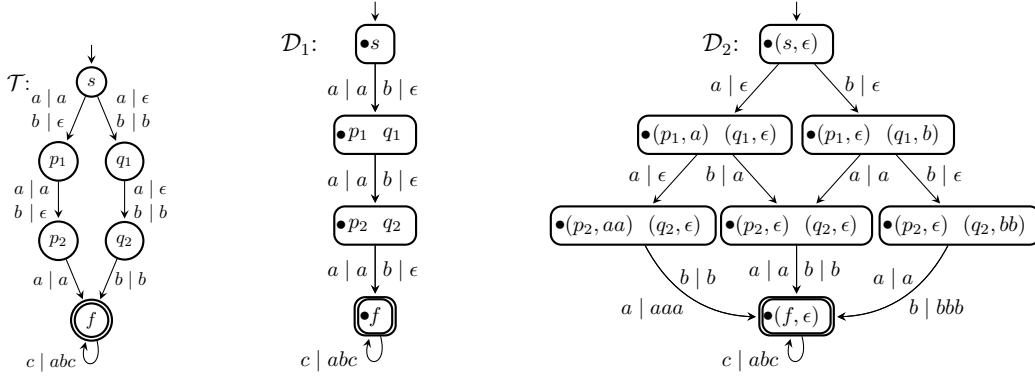
The approximate determinisation problem asks whether a rational function given as a functional transducer is approximately determinisable. We prove the following.

► **Theorem 14.** *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 4.19). Similarly, we establish that **HTP** and **STP** characterise rational functions that can be approximately determinised w.r.t. Hamming distance (Lemma 4.31). Theorem 14 then follows directly, as these three properties are decidable (Lemma 3.11). 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 15 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 4.17) and unambiguous series-sequential (Lemma 4.18) functions. However, for rational functions in general, **ATP** does not suffice. For example, the transducer above for f_{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 4.19).

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.



■ **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.

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 21 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 4.22) and series-sequential (Lemma 4.26) functions, and then we conclude by using **STP** to transform functional transducers into a finite unions of concatenations of sequential transducers (Lemma 4.31).

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 $babcccc$:

$$\begin{aligned} \mathcal{T}(babcccc) &= bbabcbabcabcb, \\ \mathcal{D}_1(babcccc) &= aabcbabcabcb, \\ \mathcal{D}_2(babcccc) &= ababcbabcabcb. \end{aligned}$$

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.³ 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} .

³ The Levenshtein distance remains bounded, as inserting a letter at the start resynchronizes both outputs.

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 15.** *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. Given that the rational function given by \mathcal{T} is approximately determinisable, i.e., there exists a sequential function given by a deterministic transducer \mathcal{D} such that $d(\mathcal{T}, \mathcal{D}) < \infty$. Since \mathcal{D} is a sequential transducer, it satisfies the twinning property, and consequently, \mathcal{D} also satisfies **ATP** and **STP** by Proposition 8. Since $d(\mathcal{T}, \mathcal{D}) < \infty$, we can conclude that \mathcal{T} also satisfies **ATP** as well as **STP** by applying Lemma 3.9. ◀

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 4.16.** *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. Let \mathcal{T}^2 be the cartesian product of \mathcal{T} by itself. By designating all states of \mathcal{T}^2 as final, and disregarding the input word, we obtain a new transducer that defines the relation R_o , consisting of all pairs of output words produced by distinct runs of \mathcal{T} on the same prefix of an input word. Since \mathcal{T} satisfies **ATP**, every pair of output words produced by loops in \mathcal{T}^2 on any input are conjugate. Consequently, since R_o is obtained by ignoring the input word from \mathcal{T}^2 , it satisfies the condition in Proposition 3. Hence, the diameter of R_o w.r.t. Levenshtein family of distances is bounded.

Let $\text{dia}_d(R_o) \leq k$. By the definition of diameter, it follows that $d(u, v) \leq k$ for all $(u, v) \in R_o$. As a result, the distance between any two output words produced by distinct runs of \mathcal{T} on the same word is at most k . Setting $N_{\mathcal{T}} = k$ completes the proof. ◀

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

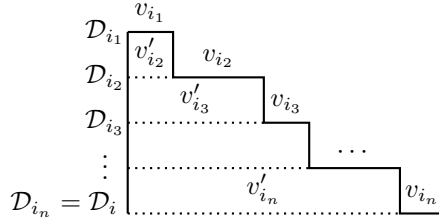
► **Lemma 4.17.** *A multi-sequential function given by a trim transducer \mathcal{T} is approximately determinisable w.r.t. a metric $d \in \{d_l, d_{lcs}, d_{dl}\}$ iff \mathcal{T} satisfies the **ATP**.*

Proof. (\rightarrow) is direct by Proposition 15, and we show (\leftarrow). Since the transducer \mathcal{T} is multi-sequential, it is equivalent to some finite union of sequential transducers $\mathcal{U} = \mathcal{D}_1 \cup \dots \cup \mathcal{D}_k$ for some $k \in \mathbb{N}$ where $\mathcal{D}_i = (Q_i, s_i, \delta_i, F_i, \lambda_i)$ is a sequential trim transducer for $i \in [k]$. By Lemma 3.9 and since \mathcal{T} satisfies **ATP**, the transducer \mathcal{U} also satisfies **ATP**. We construct a sequential transducer \mathcal{D} that approximately determinises \mathcal{T} , which intuitively is simply the cartesian product of $\mathcal{D}_1, \dots, \mathcal{D}_k$ which on each transition produces the output of the smallest index transducer \mathcal{D}_i for which that transition is defined. Let $\mathcal{D} = (Q, s, \delta, F, \lambda)$ where

1. The set of states $Q = Q'_1 \times Q'_2 \times \dots \times Q'_k$ is the cartesian product of the state set Q'_i for $i \in [k]$ such that $Q'_i = Q_i \cup \{d\}$ where d represents a dead state. The initial state is $s = (s_1, s_2, \dots, s_k)$, and the set of final states F is $\{(p_1, p_2, \dots, p_k) \mid \exists i \, p_i \in F_i\}$.
2. The function $\delta : Q \times A \rightarrow Q \times B^*$ is defined as: $\delta((p_1, p_2, \dots, p_k), a) = ((q_1, q_2, \dots, q_k), x)$ where for each $i \in [k]$, either $\delta_i(p_i, a) = (q_i, x_i)$ or, if $p_i = d$ or $\delta_i(p_i, a)$ is undefined, then $q_i = d$. The output x is set to x_j , where $j \in [k]$ is the smallest index for which the transition $\delta_j(p_j, a)$ is defined.
3. The output function $\lambda : F \rightarrow B^*$ is defined as $\lambda((p_1, p_2, \dots, p_k)) = \lambda_i(p_i)$ where $i \in [k]$ is the smallest index such that $p_i \in F_i$.

We show that $d(\mathcal{D}, \mathcal{T}) < \infty$ w.r.t. Levenshtein family. From the construction, it is clear that \mathcal{D} and \mathcal{T} have the same domain. Consider an input word $u \in \text{dom}(\mathcal{T})$. Let $i \in [k]$ be the smallest index such that $u \in \text{dom}(\mathcal{D}_i)$, with output word, say v . The transducer \mathcal{D} on input u produces output, say v' , by concatenating output on each transition over input word produced by the smallest index transducer.

Thus, v' can be decomposed into $v_{i_1} v_{i_2} \dots v_{i_n}$ where $i_1 < i_2 < \dots < i_n = i$ and for each $i_j \in [i]$, v_{i_j} is the output produced by \mathcal{D}_{i_j} along the partial run of u . For each v_{i_j} , let v'_{i_j} denote the prefix of the output produced by \mathcal{D}_{i_j} upto v_{i_j} . The output produced by \mathcal{T} on u via \mathcal{D}_i is $v = v'_i v_i = v'_{i_n} v_{i_n}$.



Observe that $v'_{i_j} v_{i_j}$ and $v'_{i_{j+1}}$ (for $i_1 < i_j < i_n$) is the output produced by \mathcal{D}_{i_j} and $\mathcal{D}_{i_{j+1}}$ on the same prefix of input u . By Lemma 4.16, we obtain $d(v'_{i_j} v_{i_j}, v'_{i_{j+1}}) \leq N_T$. Similarly, since v_{i_1}, v'_{i_2} are the outputs of \mathcal{D}_{i_1} and \mathcal{D}_{i_2} on the same input prefix, we get $d(v_{i_1}, v'_{i_2}) \leq N_T$.

$$\begin{aligned}
d(v_{i_1} v_{i_2} v_{i_3} \dots v_{i_n}, v) &= d(v_{i_1} v_{i_2} v_{i_3} \dots v_{i_n}, v'_{i_n} v_{i_n}) \text{ (Since } v = v'_{i_n} v_{i_n} \text{)} \\
&\leq d(v_{i_1} v_{i_2} v_{i_3} \dots v_{i_n}, v'_{i_2} v_{i_2} v_{i_3} \dots v_{i_n}) + d(v'_{i_2} v_{i_2} v_{i_3} \dots v_{i_n}, v'_{i_3} v_{i_3} \dots v_{i_n}) \\
&\quad + \dots + d(v'_{i_{n-1}} v_{i_{n-1}} v_{i_n}, v'_{i_n} v_{i_n}) \text{ (Applying triangle inequality of } d \text{)} \\
&\leq n \cdot N_T \text{ (using Lemma 4.16)}
\end{aligned}$$

In fact, on any input word, the distances between the outputs produced by \mathcal{D} and \mathcal{T} is less than or equal to $k \cdot N_T$, as there can be at most k switches between runs. Hence, we get that $d(\mathcal{D}, \mathcal{T})$ is bounded. \blacktriangleleft

The characterisation of Lemma 4.17 also holds for series-sequential functions.

► **Lemma 4.18.** *Let $\mathcal{T} = \mathcal{D}_1 \dots \mathcal{D}_k$ 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. (\rightarrow) follows from Proposition 15, and we prove (\leftarrow). Similarly to the proof in Lemma 4.17, we construct a sequential transducer \mathcal{D} using a subset construction of \mathcal{T} such that \mathcal{D} approximately determinises \mathcal{T} . Although \mathcal{T} is functional, and each \mathcal{D}_i , $i \in [k]$, is sequential, the nondeterminism arises between the transitions between one \mathcal{D}_i to the other. We assume an ordering for the states of \mathcal{T} for each \mathcal{D}_i . Intuitively, \mathcal{D} stores a subset of states of \mathcal{T} to capture all possible *active runs* on any input word. It produces the output of the active run of the smallest indexed sequential transducer, called the *producing run*. Upon termination of the producing run, it switches to the next active run of the smallest indexed sequential transducer. Since \mathcal{T} is unambiguous, there is at most one accepting run on any input. Formally, $\mathcal{D} = (Q, s_0, \delta, F, \lambda)$ where

1. Q is the power set of the states in \mathcal{T} . Each set $S \in Q$ has exactly one state marked with a \bullet to denote the currently producing run.
2. s_0 is the initial state of \mathcal{T} and is marked with a \bullet .
3. The transition function $\delta : Q \times A \rightarrow Q \times B^*$ is defined as follows: $\delta(P, a) = (S, x)$ where S consists of all states q in \mathcal{T} reachable from a state $p \in P$ via transition (p, a, q, x_{pq}) in \mathcal{T} . The output word x is set as follows. Let p be the \bullet -marked state in P that belongs to the sequential transducer \mathcal{D}_i for some $i \in [k]$.
 - a. If a transition (p, a, q, x_{pq}) in \mathcal{T} exists within the same sequential transducer \mathcal{D}_i , then $x = x_{pq}$ and q is \bullet -marked in S .
 - b. If no transition exists from p on a within \mathcal{D}_i , but a transition (p, a, q, x_{pq}) in \mathcal{T} exists to a different sequential transducer, i.e., q belongs to \mathcal{D}_{i+1} , then $x = x_{pq}$ and q is \bullet -marked in S . Such a transition is called a *switch* (to \mathcal{D}_{i+1}).
 - c. Otherwise, choose the smallest numbered state $p' \in P$ that belongs to the smallest indexed sequential transducer \mathcal{D}_j where $j \in [k]$ with a defined transition $(p', a, q', x_{p'q'})$ in \mathcal{T} , and set $x = x_{p'q'}$ and \bullet -mark q' in S . Such a transition is also called a switch (to \mathcal{D}_j).
4. F is the set of all states $P \in Q$ such that P contains a final state of \mathcal{T} .
5. The output function $\lambda : F \rightarrow B^*$ is defined, for $P \in F$, as $\lambda(P) = \lambda_{\mathcal{T}}(p)$ for some arbitrary final state $p \in P$ of \mathcal{T} , where $\lambda_{\mathcal{T}}$ is the output function of \mathcal{T} .

The number of states of \mathcal{D} is exponential in the number of states of \mathcal{T} . We now argue that the number of switches in the run of \mathcal{D} between the active runs of \mathcal{T} on any input word is less than the number of states in \mathcal{T} , and hence finite. Let n_i be the number of states in \mathcal{D}_i for $i \in [k]$. Consider the run of \mathcal{D} on an arbitrary input word. Initially, only the initial state of \mathcal{D}_1 is active in \mathcal{D} . As the run proceeds, if P is the set of states of \mathcal{T} reached so far, then by construction of \mathcal{D} , the only \bullet -marked state in P is the last state of the active run of \mathcal{T} that belongs to the smallest indexed \mathcal{D}_i . If this run eventually dies, then two cases can happen: (1) \mathcal{D} switches to another active run in \mathcal{D}_i , or, (2) if none exists, \mathcal{D} switches to some active run in \mathcal{D}_j , where $j > i$ is minimal. If case (2) happens, then no more states of \mathcal{D}_i are active, so \mathcal{D} will never switch again to \mathcal{D}_i in the future. The number of times case (1) can happen is bounded by n_i . Indeed, at most n_i states of \mathcal{D}_i can be active at any moment, and since \mathcal{D}_i is sequential, the number of active states in \mathcal{D}_i can only decrease.

Now, if \mathcal{D} eventually switches to \mathcal{D}_j , at most n_j states in \mathcal{D}_j are active, so at most n_j switches of type (1) can happen in \mathcal{D}_j , and so on until \mathcal{D} eventually terminates in some sequential transducer \mathcal{D}_l for $l \geq j$.

Therefore, in the worst case, \mathcal{D} switches n_i times in \mathcal{D}_i before switching to \mathcal{D}_{i+1} for all $i \in \{1, \dots, k-1\}$. So, the overall number of switches in the run of \mathcal{D} is at most $N = \sum_{i=1}^k n_i$, which is exactly the number of states in \mathcal{T} .

Now we show that $d(\mathcal{D}, \mathcal{T}) < \infty$ w.r.t. Levenshtein family. From the construction, it is clear that \mathcal{D} and \mathcal{T} have the same domain.

Consider an input word u accepted by \mathcal{T} . Since \mathcal{T} is unambiguous, there is exactly one run in \mathcal{T} that accepts u . Let $v = \mathcal{T}(u)$ and $v' = \mathcal{D}(u)$. From the construction of \mathcal{D} , v' can be decomposed into $v_{s_1} v_{s_2} \dots v_{s_n}$ accommodating $n \in [N]$ switches between the active runs of \mathcal{T} on the prefixes of u , where each v_{s_i} , $i \in [n]$ is the output produced by an active run r_i on the prefix of u . For each v_{s_i} , let v'_{s_i} denote the prefix of the output produced by the run r_i up to v_{s_i} .

Observe that $v'_{s_i} v_{s_i}$ and $v'_{s_{i+1}}$ (for $1 < i < n$) is the output produced by $\mathcal{D}_1 \dots \mathcal{D}_{j_i}$ and $\mathcal{D}_1 \dots \mathcal{D}_{j_{i+1}}$ on the same prefix of u where $j_i, j_{i+1} \in [k]$. By using Lemma 4.16, we obtain $d(v'_{s_i} v_{s_i}, v'_{s_{i+1}}) \leq N_T$. Similarly, since v_{s_1}, v'_{s_2} are the outputs of $\mathcal{D}_1 \dots \mathcal{D}_{j_1}$ and $\mathcal{D}_1 \dots \mathcal{D}_{j_2}$ on the same input prefix, it follows that $d(v_{s_1}, v'_{s_2}) \leq N_T$. As a consequence,

$$\begin{aligned} d(v_{s_1} v_{s_2} \dots v_{s_n}, v) &= d(v_{s_1} v_{s_2} \dots v_{s_n}, v'_{s_n} v_{s_n}) \quad (v = v'_{s_n} v_{s_n} \text{ since } r_{s_n} \text{ is the only accepting run of } u) \\ &\leq d(v_{s_1} v_{s_2} \dots v_{s_n}, v'_{s_2} v_{s_2} \dots v_{s_n}) + d(v'_{s_2} v_{s_2} \dots v_{s_n}, v'_{s_3} v_{s_3} \dots v_{s_n}) \\ &\quad + \dots + d(v'_{s_{n-1}} v_{s_{n-1}} v_{s_n}, v'_{s_n} v_{s_n}) \quad (\text{Applying triangle inequality of } d) \\ &\leq n \cdot N_T \quad (\text{using Lemma 4.16}) \end{aligned}$$

Therefore, $d(\mathcal{D}(u), \mathcal{T}(u)) \leq n \cdot N_T \leq N \cdot N_T$. This holds for any $u \in \text{dom}(\mathcal{T})$, and N being the number of states in \mathcal{T} , we get that $d(\mathcal{D}, \mathcal{T})$ is bounded. \blacktriangleleft

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 4.19.** *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. (\rightarrow) follows from Proposition 15. We prove (\leftarrow). Assume that the rational function is given by an unambiguous transducer \mathcal{T} with set of states Q . Disregarding the labels on transitions, decompose \mathcal{T} into maximal SCCs $S_1, \dots, S_k \subseteq Q$. Consider the set of paths Π of the form $\pi = S_{i_1} t_{i_1} S_{i_2} \dots t_{i_{n-1}} S_{i_n}$ such that S_{i_1} is an SCC which contains an initial state, S_{i_n} is an SCC which contains a final state, and for all $1 \leq k < n$, t_{i_k} is a transition of \mathcal{T} from a state of S_{i_k} to some state of $S_{i_{k+1}}$. Let \mathcal{T}_π denote the trim subtransducer of \mathcal{T} that removes all the transitions in \mathcal{T} except the transitions t_{i_k} ($1 \leq k < n$) and the transitions occurring in the SCCs S_{i_k} ($1 \leq k \leq n$).

Note that since the SCCs are maximal, the set Π is finite. Now, it is straightforward to see that $\mathcal{T} \equiv \mathcal{U} = \bigcup_{\pi \in \Pi} \mathcal{T}_\pi$. From Lemma 3.9, we deduce \mathcal{U} satisfies both **ATP** as well as **STP**. Moreover, given \mathcal{T} 's unambiguity, each input accepted by \mathcal{T} is accepted by exactly one \mathcal{T}_π and, each SCC within a \mathcal{T}_π has a single entry and exit point.

Since \mathcal{U} satisfies **STP**, each SCC in \mathcal{T}_π satisfies **TP**, with initial state being the unique entry point of the SCC. Hence we can determinise each SCC in \mathcal{T}_π and can obtain a series-sequential transducer that is equivalent to \mathcal{T}_π and indeed satisfies **ATP** by Lemma 3.9. From Lemma 4.18, there exists a sequential transducer \mathcal{D}_π for each \mathcal{T}_π , such that $d(\mathcal{D}_\pi, \mathcal{T}_\pi)$ is finite.

Let $d(\mathcal{T}_\pi, \mathcal{D}_\pi) \leq k_\pi$ for some $k_\pi \in \mathbb{N}$. Consequently, the distance between \mathcal{T} and the new transducer $\mathcal{U}' = \bigcup_{\pi \in \Pi} \mathcal{D}_\pi$ is bounded where $d(\mathcal{T}, \mathcal{U}') = d(\mathcal{U}, \mathcal{U}') \leq \max\{k_\pi \mid \pi \in \Pi\}$. Further, since \mathcal{T} satisfies **ATP**, we deduce \mathcal{U}' also satisfies **ATP** by Lemma 3.9. Being

multi-sequential and satisfying **ATP**, \mathcal{U}' can be further approximately determinised using the construction outlined in Lemma 4.17. Let \mathcal{D} be the sequential transducer such that $d(\mathcal{U}', \mathcal{D}) < \infty$. Since d is a metric, $d(\mathcal{T}, \mathcal{D}) \leq d(\mathcal{T}, \mathcal{U}') + d(\mathcal{U}', \mathcal{D})$. Since both $d(\mathcal{T}, \mathcal{U}')$ and $d(\mathcal{U}', \mathcal{D})$ are finite, $d(\mathcal{T}, \mathcal{D})$ is also finite. \blacktriangleleft

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 4.20.** *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 .*

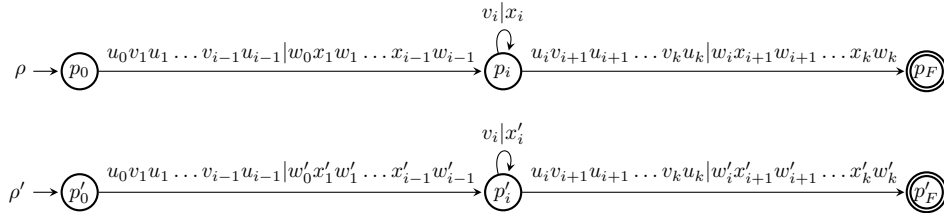
Proof. We establish the lemma by decomposing the runs producing w and w' into short simple paths combined with synchronised looping segments. This decomposition naturally gives rise to multiple occurrences of the Twinning pattern. Since the transducer T satisfies the **HTP**, we can leverage this property to show that all synchronised loops generate output words of the same length. Consequently, any difference in output length can only arise from the short simple paths. Furthermore, the **HTP** also constrains the locations of potential mismatches between the two output words. By quantifying these effects, we derive the desired bounds on both the length difference and the number of mismatched positions.

Formally, let $u \in A^*$ be such that T has two initial runs ρ and ρ' on u , producing the outputs w and w' , respectively. Let $n \in \mathbb{N}$ denote the number of states of T , and let $M \in \mathbb{N}$ be the maximal output length produced by the transition function of T . We set $N_T = Mn^4$.

Since there are at most n^2 distinct pairs of states in T , if the length of u exceeds n^2 , then the runs ρ and ρ' must revisit at least one pair of states while processing u , thus looping synchronously. More precisely, there exists a decomposition of u into nonempty words

$$u = u_0 v_1 u_1 \dots v_k u_k$$

such that $|u_0 u_1 \dots u_k| < n^2$ and both ρ and ρ' loop while processing each v_i for $1 \leq i \leq k$. This decomposition induces k twinning patterns in T , as illustrated below:



Since T satisfies the **HTP**, we have that $|x_i| = |x'_i|$ for every $1 \leq i \leq k$. This allows us to bound the difference in output length between ρ and ρ' after reading each u_i :

$$\begin{aligned}
 ||w_0 x_1 w_1 \dots x_i w_i| - |w'_0 x'_1 w'_1 \dots x'_i w'_i|| &= ||w_0 w_1 \dots w_i| - |w'_0 w'_1 \dots w'_i|| \\
 &\leq \max(|w_0 w_1 \dots w_i|, |w'_0 w'_1 \dots w'_i|) \\
 &\leq M \cdot |u_0 u_1 \dots u_i| \\
 &< Mn^2.
 \end{aligned} \tag{5}$$

Applying Equation (5) to the case $i = k$ immediately yields Item 1:

$$||w| - |w'|| = ||w_0x_1w_1 \dots x_kw_k| - |w'_0x'_1w'_1 \dots x'_kw'_k|| < Mn^2.$$

To establish Item 2, note that Equation (5) implies that for each $1 \leq i \leq k$, the subwords x_i and x'_i in w and w' are shifted by at most Mn^2 . In other words, at most Mn^2 positions of x_i don't overlap x'_i . As T satisfies the **HTP**, every position of x_i that overlaps with x'_i is a match between w and w' . Therefore, the positions where w differs from w' can only occur :

- In one of the w_i , contributing at most $|w_1w_2 \dots w_k| \leq M \cdot |u_1u_2 \dots u_k| < Mn^2$ mismatches.
- In the part of some x_i that does not overlap x'_i , contributing at most $k \cdot Mn^2$ mismatches.

As the u_i are nonempty we can bound the number of mismatches between w and w' by

$$Mn^2 + k \cdot Mn^2 \leq Mn^2 + (|u_0u_1 \dots u_k| - 1)Mn^2 < Mn^4. \quad \blacktriangleleft$$

► **Proposition 21.** *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. Let us suppose that the rational relation given by \mathcal{T} is approximately determinisable, i.e., there exists a sequential function given by a deterministic transducer \mathcal{D} such that $d_h(\mathcal{T}, \mathcal{D}) < \infty$. Since \mathcal{D} is a sequential transducer, it satisfies the twinning property, and consequently, \mathcal{D} also satisfies **HTP** and **STP** by Proposition 8. Since $d_h(\mathcal{T}, \mathcal{D}) < \infty$, we can conclude that \mathcal{T} also satisfies **HTP** as well as **STP** by applying Lemma 3.9. \blacktriangleleft

► **Lemma 4.22.** *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. (\rightarrow) By Proposition 21, every transducer defining a relation that is approximately determinisable with respect to the Hamming distance satisfies the **HTP**.

(\leftarrow) Let \mathcal{T} be a multi-sequential transducer satisfying the **HTP**. Then \mathcal{T} is equivalent to a finite union of sequential transducers $\mathcal{U} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup \dots \cup \mathcal{D}_k$, where $\mathcal{D}_i = (Q_i, s_i, \delta_i, F_i, \lambda_i)$ is a sequential trim transducer for all $1 \leq i \leq k$. As \mathcal{T} satisfies the **HTP**, so does \mathcal{U} by Lemma 3.9. We build a sequential transducer \mathcal{D} approximating \mathcal{U} by modifying the *subset construction with delays* used to determinise transducers satisfying the Twinning property [10, 9].

While processing an input word u , the transducer \mathcal{D} simulates the run of each \mathcal{D}_i on u , and tracks the outputs they produce. To that end, the states of \mathcal{D} are tuples of pairs

$$((p_1, v_1), (p_2, v_2), \dots, (p_k, v_k)) \in ((Q_1 \cup \{d\}) \times B^*) \times ((Q_2 \cup \{d\}) \times B^*) \times \dots \times ((Q_k \cup \{d\}) \times B^*).$$

Each p_i is the state reached by \mathcal{D}_i on the current input, and the special “dead” state d is used to denote the fact that \mathcal{D}_i does *not* have a run on the current input. Each v_i is a suffix of the output v'_i produced by \mathcal{D}_i on the current input. Observe that the complete output word v'_i may grow arbitrarily large as u increases in length, and \mathcal{D} cannot store it entirely while maintaining a finite state space. \mathcal{D} decides which suffix to keep in memory through the following procedure : To process an input letter a , \mathcal{D} updates each pair (p_i, v_i) such that $\delta_i(p_i, a) = (q_i, w_i)$ into (q_i, v_iw_i) . Then, \mathcal{D} identifies the *minimal length* $C \in \mathbb{N}$ among all $|v_iw_i|$, truncates the prefix of length C of all these words, and outputs one of the prefixes. While this approach may introduce errors, as the partial outputs produced along the run may not correspond to the run that is eventually accepting, our analysis shows that the satisfaction of the **HTP** ensures bounded Hamming distance between the generated output and the correct one.

Finally, at the end of the input word, \mathcal{D} selects a pair (p, w) from its current state such that p is a final state, and outputs the concatenation of w with the final output of \mathcal{T} at p .

Formal construction We set $\mathcal{D} = (Q, s, \delta, F, \lambda)$, where

1. The set of states is $Q = Q'_1 \times Q'_2 \times \dots \times Q'_k$, with $Q'_i = (Q_i \cup \{d\}) \times B^*$. While this definition allows arbitrarily large words, we will show that only a finite subset of Q is reachable from the initial state.
2. The initial state is $s = ((s_1, \epsilon), (s_2, \epsilon), \dots, (s_k, \epsilon))$.
3. The transition function maps each pair $((p_1, v_1), (p_2, v_2), \dots, (p_k, v_k)), a \in Q \times A$ to

$$\delta(((p_1, v_1), (p_2, v_2), \dots, (p_k, v_k)), a) = (((q_1, x'_1), (q_2, x'_2), \dots, (q_k, x'_k)), w),$$

with the pairs (q_i, x'_i) and the word w defined as follows. First, for all $1 \leq i \leq k$ we set

$$(q_i, x_i) = \begin{cases} (q_i, v_i w_i) & \text{if } p_i \in Q_i \text{ and } \delta_i(p_i, a) = (q_i, w_i); \\ (d, \epsilon) & \text{if } p_i = d, \text{ or if } p_i \in Q_i \text{ and } \delta_i(p_i, a) \text{ is undefined.} \end{cases}$$

Then, we let $C \in \mathbb{N}$ be the minimal element of the set $\{|x_i| \mid 1 \leq i \leq k \text{ and } q_i \neq d\}$, or $C = 0$ if this set is empty. We set $w = x_j$, where j is the smallest index satisfying $|x_j| = C$, and for every $1 \leq i \leq k$ we set x'_i as the word obtained by deleting the C first letters of x_i .

4. The set of final states is $F = \{((p_1, u_1), (p_2, u_2), \dots, (p_k, u_k)) \mid p_i \in F_i \text{ for some } 1 \leq i \leq k\}$.
5. The output function maps each state $p = ((p_1, u_1), (p_2, u_2), \dots, (p_k, u_k)) \in Q$ to

$$\lambda(p) = \begin{cases} u_i \lambda_i(p_i) & \text{if } p \in F, \text{ where } i \text{ is the smallest index such that } p_i \in F_i; \\ \epsilon & \text{otherwise.} \end{cases}$$

Key properties In order to use our construction to prove the Lemma, we explicitly state three key properties. Let us fix an input word $u \in A^*$, and let $p = ((p_1, u_1), (p_2, u_2), \dots, (p_k, u_k)) \in Q$ and $v \in B^*$ denote the state reached and the word produced by \mathcal{D} while reading u . First, observe that if we disregard the output words in the states of \mathcal{D} and remove any occurrences of the “dead” state d , the resulting structure is exactly a subset construction keeping track of which states of T can be reached on the current input. This leads to the following claim :

▷ **Claim 23.** For every $1 \leq i \leq k$ we have $q_i \neq d$ if and only if \mathcal{D}_i has an initial run on u that ends in q_i .

Next, our construction keeps the output suffixes u_i synchronised with the outputs of \mathcal{T} :

▷ **Claim 24.** For every i, i' such that $p_i \neq d$ and $p_{i'} \neq d$, the length difference $|u_i| - |u_{i'}|$ is equal to the length difference between the outputs produced by \mathcal{D}_i and $\mathcal{D}_{i'}$ on u .

Finally, our definition of the transition function ensures that each fragment of output produced by \mathcal{D} while processing u can be traced back to a fragment of output produced by one of the transducers \mathcal{D}_i while reading some prefix of u . This leads to the following claim :

▷ **Claim 25.** For all $u \in \text{dom}(\mathcal{D})$, for all $1 \leq i \leq |\mathcal{D}(u)|$, there exists a run ρ of T on some prefix of u such that the i th letter of u is equal to the i th letter of the word produced by ρ .

Bounding the state space As explained in the definition of the state space of \mathcal{D} , we initially define it as an infinite set of states, and we now show that only a finite subset is reachable from the initial state. To that end, we prove that every state $p = ((p_1, w_1), (p_2, w_2), \dots, (p_k, w_k))$ reachable from the initial state of \mathcal{D} satisfies $|w_i| < N_{\mathcal{T}}$ for every $1 \leq i \leq k$, where $N_{\mathcal{T}}$ is the

constant defined in Lemma 4.20. This implies that the trimmed version of \mathcal{D} contains at most $(|Q_i| + 1) \cdot |A|^{N_{\mathcal{T}}+1}$ states. Let u denote a word so that \mathcal{D} reaches p from the initial state while reading u . First, remark that by construction every pair (p_i, w_i) such that $p_i = d$ satisfies $w_i = \epsilon$, thus $|w_i| = 0 < N_{\mathcal{T}}$. Now, observe that the definition of the transition function implies that if there is at least one index $1 \leq j \leq k$ satisfying $q_j \neq d$, then there exists such an index such that we also have $w_j = \epsilon$. Now for every other index $1 \leq i \leq k$ satisfying $q_i \neq d$, we get that $|w_i| = |w_i| - |\epsilon| = |w_i| - |w_j|$, which is equal to the difference between the length of the outputs produced by D_i and D_j on u by Claim 24. Finally, Lemma 4.20 implies that this difference is smaller than $N_{\mathcal{T}}$.

Bounding the Hamming distance We now proceed with the proof that $d(\mathcal{D}, \mathcal{T}) < \infty$ w.r.t. the Hamming distance. Specifically, we show that every word $u \in \text{dom}(\mathcal{T})$ satisfies $d_h(\mathcal{D}(u), \mathcal{T}(u)) \leq k \cdot N_{\mathcal{T}}$, where $N_{\mathcal{T}}$ is the constant from Lemma 4.20.

First, by Claim 23 and the definition of the final states of D , both \mathcal{D} and \mathcal{T} have the same domain. Fix an input word $u \in \text{dom}(\mathcal{T})$. Remark that Claim 24 and the definition of the output function λ imply that $|\mathcal{T}(u)| = |\mathcal{D}(u)|$. Therefore, in order to bound $d_h(\mathcal{D}(u), \mathcal{T}(u))$ it is sufficient to bound the number of mismatches $\text{mismatch}(\mathcal{D}(u), \mathcal{T}(u)) \in \mathbb{N}$ between $\mathcal{D}(u)$ and $\mathcal{T}(u)$. For each $1 \leq i \leq k$ let u_i denote the longest prefix of u for which \mathcal{D}_i has an initial run, and let v_i denote the output produced by this run. Since $\text{dom}(\mathcal{D}) = \text{dom}(\mathcal{T})$ there exists at least one index $j \in \{1, 2, \dots, k\}$ such that u is in the domain of \mathcal{D}_j , which we now fix for the remainder of the proof. By our notation, we have $u_j = u$ and $v_j = \mathcal{D}(u)$.

By Claim 25, for every $1 \leq m \leq |v|$ there exists $1 \leq i \leq k$ such that $v[m] = v_i[m]$. Consequently, every position where v_j and v differ is also a position where v_j and v_i differ for some i . This yields the desired result :

$$d_h(\mathcal{D}(u), \mathcal{T}(u)) = \text{mismatch}(v_j, v) \leq \sum_{i=1}^k \text{mismatch}(v_j, v_i) \leq \sum_{i=1}^k N_{\mathcal{T}} \leq k \cdot N_{\mathcal{T}}. \quad \blacktriangleleft$$

► **Lemma 4.26.** *Let $\mathcal{U} = \mathcal{D}_1 \mathcal{D}_2 \dots \mathcal{D}_k$ 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. (\rightarrow) By Proposition 21.

(\leftarrow) Let $\mathcal{U} = (Q, s, \delta, F, \lambda)$, and for every $1 \leq i \leq k$ let $\mathcal{D}_i = (Q_i, s_i, \delta_i, F_i, \lambda_i)$. We construct a sequential transducer $\mathcal{D} = (Q', G_S, \delta', F', \lambda')$ that approximately determinises \mathcal{U} by following a similar approach to the one used in the proof of Lemma 4.22. Specifically, we design \mathcal{D} to track the set of states in which \mathcal{U} could currently be, while also keeping track of synchronised output suffixes. At each step, \mathcal{D} produces a part of these suffixes, ensuring that the state space remains finite. The key difference compared to the proof of Lemma 4.22 is that, whereas in the previous case we could pick any suffix of minimal length to be produced, here we must be more careful to ensure that the Hamming distance remains bounded. We show that selecting suffixes corresponding to runs that end in the component \mathcal{D}_i with the smallest index i guarantees the desired outcome.

Formal Construction We begin by selecting a total ordering \prec of the states of \mathcal{U} that respects reachability constraints:

For all $q_i \in Q_i$ and $q_j \in Q_j$, if $i < j$ then $q_i \prec q_j$.

This order is naturally extended to pairs in $Q \times B^*$, where pairs are compared first by the state component according to \prec and then lexicographically by the word component if states are identical.

1. The set of states Q' is the power set $\mathcal{P}(Q \times B^*)$. While this definition allows arbitrarily large words, we will show that only a finite subset of Q' is reachable from the initial state.
2. The initial state is $G_s = \{(s_1, \epsilon)\}$.
3. The transition function operates in two steps. Given $(G, a) \in Q' \times A$, define:

$$\overline{H} = \{(q, uw) \mid \text{there exists } (p, u) \in G \text{ such that } \delta(p, a) = (q, w)\}.$$

Then, let $C \in \mathbb{N}$ be the minimum length of the set $\{|w| \mid (q, w) \in \overline{H}\}$, and let

$$H = \{(q, w_2) \mid (q, w_1 w_2) \in \overline{H} \text{ for some } w_1 \in B^* \text{ satisfying } |w_1| = C\}.$$

We set $\delta'(G, a) = (H, v)$, where v is the C -length prefix of the word w_{\min} occurring in the minimal pair (q_{\min}, w_{\min}) in \overline{H} according to \prec .

4. The set of final states is $F' = \{G \in Q' \mid \exists (p, u) \in G \text{ such that } p \in F\}$.
5. The output function maps each state $G \in Q'$ to

$$\lambda'(G) = \begin{cases} u\lambda(p) & \text{if } p \in F, \text{ where } (p, u) \text{ is the minimal pair in } G \text{ such that } p \in F; \\ \epsilon & \text{otherwise.} \end{cases}$$

Key Properties We establish three fundamental properties linking, for each input word $u \in A^*$, the initial run of \mathcal{D} on u with the runs of \mathcal{U} on u . Let us fix an initial run of \mathcal{D} :

$$\rho : s' \xrightarrow{u|v} G.$$

First, observe that disregarding the output words in the states of \mathcal{D} results in a subset construction keeping track of the reachable states of \mathcal{U} . This leads to the following claim :

▷ **Claim 27.** For all $p \in Q$, \mathcal{U} has an initial run on u ending in p if and only if $(p, w) \in G$ for some $w \in B^*$.

Next, our construction guarantees that the output suffixes stored in G remain synchronized :

▷ **Claim 28.** For all $(p, u), (p', u') \in G$, the length difference $|u| - |u'|$ is equal to the length difference between the outputs produced by the initial runs of \mathcal{U} on u ending in p and p' .

Finally, our definition of the transition function ensures that the output produced by \mathcal{D} while processing u can be traced back to output produced by \mathcal{U} while reading some prefix of u . More precisely, δ' always produces output coming from a run of \mathcal{U} that ends in the smallest state with respect to \prec . To reflect this, we say that an initial run of \mathcal{U} is *optimal* if it reaches the minimal state reachable on its input, and we get :

▷ **Claim 29.** For every $1 \leq i \leq v$, the i th letter of v is equal to the i th letter of an optimal run of \mathcal{U} on some prefix of u .

While \mathcal{U} might have arbitrarily many optimal runs on prefixes of u , we now build a set of $|Q|$ runs such that every optimal run on a prefix of u is a prefix of one of these runs. This will be crucial to bound the Hamming distance between \mathcal{D} and \mathcal{U} .

Let $u = u_1 u_2 \dots u_k$ denote the decomposition of u into (possibly empty) subwords such that for every $1 \leq i \leq k$, $u_1 u_2 \dots u_i$ is the minimal prefix of u for which there exists no run of

\mathcal{U} starting in the initial state and ending in \mathcal{D}_i . For every $1 \leq i \leq k$, let $\bar{Q}_i \subseteq Q_i$ denote the set containing all the states $q \in Q_i$ reachable by reading $u_1 u_2 \dots u_{i-1}$ from the initial state. Moreover, let $\bar{Q} = \bigcup_{i=1}^k \bar{Q}_i$. For every $q \in \bar{Q}$ we define a specific run ρ_q as the concatenation of two runs $\rho'_q \rho''_q$, where :

- ρ'_q is the initial run of \mathcal{U} on $u_1 u_2 \dots u_{i-1}$ ending in q . Note that there is only one such run as \mathcal{U} is an unambiguous trim transducer.
- ρ''_q is the run of \mathcal{D}_i starting from q that processes the longest prefix of u_i (while remaining in \mathcal{D}_i). Note that there is only one such run as \mathcal{D}_i is sequential.

We now argue the following key claim:

▷ **Claim 30.** Every optimal initial run ρ of \mathcal{U} on a prefix of u is a prefix of ρ_q for some $q \in \bar{Q}$.

Proof. Let ρ be an optimal run on some prefix of u , and let $1 \leq i \leq k$ be the index of the component \mathcal{D}_i in which ρ terminates. By definition of the decomposition $u = u_1 u_2 \dots u_k$ the run ρ terminates while processing u_i , thus its input must be of the form $u_1 u_2 \dots u_{i-1} u'_i$ for some prefix u'_i of u_i . Moreover, ρ must have already entered the component \mathcal{D}_i after reading $u_1 u_2 \dots u_{i-1}$. We decompose ρ into two parts $\rho' \rho''$ defined as follows:

$$\rho : p_0 \xrightarrow{u_1 u_2 \dots u_{i-1} | v} q \xrightarrow{u'_i | w} q'.$$

Since T is an unambiguous trim transducer we have $\rho' = \rho'_q$, and as \mathcal{D}_i is sequential ρ'' is a prefix of ρ''_q , which implies that ρ is a prefix of $\rho'_q \rho''_q$. This establishes the claim. ◀

Bounding the state space The state space Q' in the formal definition is infinite. We now show that the set of states reachable from the initial state is finite. More precisely, we let $N_{\mathcal{U}} \in \mathbb{N}$ be the bound given by Lemma 4.20, and show that every initial run

$$\rho : \{(s, \epsilon)\} \xrightarrow{u|v} G = \{(p_1, w_1), (p_2, w_2), \dots, (p_m, w_m)\}$$

satisfies $|w_i| < N_{\mathcal{U}}$ for every $1 \leq i \leq m$. This bounds the number of reachable states by

$$2^{(|Q|) \cdot A^{N_{\mathcal{U}}+1}}.$$

Observe that G is reached by successive applications of the transition function δ' to the initial state, thus, by definition of δ' , there is a pair $(q_j, w_j) \in G$ satisfying $w_j = \epsilon$. Since $|\epsilon| = 0$, we get that for every $(q_i, w_i) \in G$, the length of $|w_i|$ is equal to the difference $|w_i| - |w_j|$. In turn, we can apply Claim 28 to get that the length of $|w_i|$ is equal to the length difference between the outputs of the initial runs of \mathcal{U} on u ending in q_i and q_j . Finally, Lemma 4.20 guarantees that this difference is bounded by $N_{\mathcal{U}}$ as \mathcal{U} satisfies the **HTP**.

Bounding the Hamming distance We now establish that for every word $u \in \text{dom}(\mathcal{U})$, the Hamming distance between $\mathcal{D}(u)$ and $\mathcal{U}(u)$ is bounded by $|Q| \cdot N_{\mathcal{U}}$, where $N_{\mathcal{U}}$ is the constant from Lemma 4.20. This ensures that $d_h(\mathcal{D}, \mathcal{U}) < \infty$.

By Claim 27 and the definition of the final states of \mathcal{D} , we know that \mathcal{D} and \mathcal{U} have the same domain. Fix an input word $u \in \text{dom}(\mathcal{U})$. By Claim 28 and the definition of the output function λ' we get that $|\mathcal{D}(u)| = |\mathcal{U}(u)|$, therefore the distance $d_h(\mathcal{D}(u), \mathcal{U}(u))$ is equal to the number of mismatches between $\mathcal{D}(u)$ and $\mathcal{U}(u)$, denoted $\text{mismatch}(\mathcal{D}(u), \mathcal{U}(u))$.

Combining Claim 29 and Claim 30 yields that for every $1 \leq m \leq |\mathcal{D}(u)|$, the m th letter of $\mathcal{D}(u)$ matches the m th letter of the output v_q of one of the runs ρ_q with $q \in \bar{Q}$. Consequently,

every position where $\mathcal{D}(u)$ and $\mathcal{U}(u)$ differ is also a position where v_q and $\mathcal{U}(u)$ differ for some $q \in Q$. This yields the desired bound through the use of Lemma 4.20 :

$$d_h(\mathcal{D}(u), \mathcal{T}(u)) = \text{mismatch}(\mathcal{D}(u), \mathcal{U}(u)) \leq \sum_{q \in Q} \text{mismatch}(v_q, \mathcal{U}(u)) \leq \sum_{q \in Q} N_{\mathcal{U}} = |Q| \cdot N_{\mathcal{U}}. \blacktriangleleft$$

► **Lemma 4.31.** *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. (\rightarrow) By Proposition 21.

(\leftarrow): First, we assume that the rational function is given by an unambiguous transducer \mathcal{T} and decompose \mathcal{T} into an equivalent finite union of transducers $\mathcal{U} = \bigcup_{\pi \in \Pi} \mathcal{T}_{\pi}$, exactly as in the proof of Proposition 15. Then Lemma 3.9 guarantees that \mathcal{U} satisfies both **HTP** as well as **STP**.

Since \mathcal{U} satisfies **STP**, for every $\pi \in \Pi$ each SCC of \mathcal{T}_{π} satisfies twinning property if we set as initial state the unique entry point of the SCC. Hence we can determinise each SCC in \mathcal{T}_{π} , thus obtaining a series-sequential transducer \mathcal{T}'_{π} equivalent to \mathcal{T}_{π} that still satisfies **HTP** by Lemma 3.9. From Lemma 4.26, we can then turn each \mathcal{T}'_{π} into a sequential transducer \mathcal{D}_{π} for which the distance $d_h(\mathcal{D}_{\pi}, \mathcal{T}'_{\pi}) = d_h(\mathcal{D}_{\pi}, \mathcal{T}_{\pi})$ is finite.

We then set $d_h(\mathcal{D}_{\pi}, \mathcal{T}_{\pi}) = k_{\pi}$ for some $k_{\pi} \in \mathbb{N}$, which yields that the distance between \mathcal{T} and the new transducer $\mathcal{U}' = \bigcup_{\pi \in \Pi} \mathcal{D}_{\pi}$ is bounded :

$$d_h(\mathcal{U}', \mathcal{T}) = d_h(\mathcal{U}', \mathcal{U}) = \max\{k_{\pi} \mid \pi \in \Pi\} < \infty.$$

Then, as \mathcal{T} satisfies **HTP** so does \mathcal{U}' by Lemma 3.9, and as \mathcal{U}' is multi-sequential, Lemma 4.22 allows to construct a sequential transducer \mathcal{D} satisfying $d_h(\mathcal{D}, \mathcal{U}') < \infty$.

Now since both $d_h(\mathcal{D}, \mathcal{U}')$ and $d_h(\mathcal{U}', \mathcal{T})$ are bounded, we get that $d_h(\mathcal{D}, \mathcal{T})$ is also bounded by the triangle inequality. This shows that, as required, \mathcal{T} is approximately determinisable w.r.t Hamming distance. \blacktriangleleft

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 32.** *d is a metric on relations.*

Proof. When d is a metric on words, the Hausdorff distance d_H between languages is also a metric. Therefore, $d(R_1, R_2) = 0 \iff R_1 = R_2$ and $d(R_1, R_2) = d(R_2, R_1)$ for any two relations R_1 and R_2 . It remains to show that $d(R_1, R_2) \leq d(R_1, R_3) + d(R_3, R_2)$ for relations R_1, R_2 and R_3 .

Assume the domains of R_1 and R_2 are different. Then, either $\text{dom}(R_1) \neq \text{dom}(R_3)$ or $\text{dom}(R_3) \neq \text{dom}(R_2)$. In both cases, $d(R_1, R_2)$ and $d(R_1, R_3) + d(R_3, R_2)$ are ∞ . Therefore, assume that the domains of R_1, R_2 and R_3 are the same, call it L . Since for each word w in L , $d_H(R_1(w), R_2(w)) \leq d_H(R_1(w), R_3(w)) + d_H(R_3(w), R_2(w))$ by virtue of d_H being a metric, it follows that

$$\begin{aligned} d(R_1, R_2) &= \sup \{ d_H(R_1(w), R_2(w)) \mid w \in L \} \\ &\leq \sup \{ d_H(R_1(w), R_3(w)) + d_H(R_3(w), R_2(w)) \mid w \in L \} \\ &\leq \sup \{ d_H(R_1(w), R_3(w)) \mid w \in L \} + \sup \{ d_H(R_3(w), R_2(w)) \mid w \in L \} \\ &= d(R_1, R_3) + d(R_3, R_2). \end{aligned}$$

Approximate functionality problem

► **Definition 33** (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 5.34.** *For a rational relation R given as a rational transducer, $\text{diff}_d(R)$ is computable for all metrics given in Table 1.*

Proof. Given a rational relation R , we can construct a rational relation R_o that consists of all pairs of output words of R on any input, i.e., $R_o = \{(v_1, v_2) \mid \exists u \in \text{dom}(R), (u, v_1), (u, v_2) \in R\}$. If R is a relation defined by a transducer \mathcal{T} , then the transducer obtained by $\mathcal{T} \times \mathcal{T}$ (cartesian product of \mathcal{T} by itself) by ignoring the input word is a transducer that defines the relation R_o . Observe that $\text{diff}_d(R)$ is equivalent to the diameter of R_o w.r.t. d . It is shown in [2] that diameter of a rational relation is computable for all metrics given in Table 1. Hence, for those metrics, $\text{diff}_d(R)$ is computable. ◀

We now characterise rational relations which are approximately functionalisable.

► **Lemma 5.35.** *A rational relation R is approximately functionalisable w.r.t. a metric d if and only if $\text{diff}_d(R) < \infty$.*

Proof. (\rightarrow) Assume that R is approximately functionalisable, i.e., there exists a rational function f such that $d(R, f) < \infty$. Therefore, $\text{dom}(R) = \text{dom}(f)$ and there exists an integer k such that for each input word $u \in \text{dom}(R)$, for each output word $v \in R(u)$,

$d(v, f(u)) \leq k$ (since f is a function). By triangle inequality of metric d , the distance between any two arbitrary output words $v_1, v_2 \in R(u)$ on any input $u \in \text{dom}(R)$ is $d(v_1, v_2) \leq d(v_1, f(u)) + d(f(u), v_2) \leq 2k$. Since this holds for any input in the domain of R , we get $\text{diff}_d(R) = \sup_{u \in \text{dom}(R)} \sup_{v_1, v_2 \in R(u)} d(v_1, v_2) \leq 2k$.

(\leftarrow) Assume that $\text{diff}_d(R) < \infty$. Let $\text{diff}_d(R) \leq k$ for some $k \in \mathbb{N}$. We show that any uniformiser of the relation (a function of same domain as the relation whose graph is included in the relation) is a function that approximately functionalises the relation. Let f be a uniformiser of R . We prove that $d(R, f) < \infty$. Since f is a uniformiser of R , $\text{dom}(R) = \text{dom}(f)$, and for all $u \in \text{dom}(R)$, it holds that $(u, f(u)) \in R$. Since $\text{diff}_d(R) \leq k$, the distance between the outputs of R on any input is less than or equal to k . Thus, for any input word $u \in \text{dom}(R)$, for each output word $v \in R(u)$, $d(v, f(u)) \leq k$ (since $f(u)$ is also an output of $R(u)$). Hence, $d(R, f) < \infty$, i.e., R is approximately functionalisable w.r.t. d . \blacktriangleleft

Since $\text{diff}_d(R)$ is computable (see Lemma 5.34), we get the following result.

► **Theorem 36.** *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 5.37.** *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. Let R be a rational relation given by a transducer \mathcal{T} , and let $d \in \{d_l, d_{lcs}, d_{dl}\}$. The proof of direction (\rightarrow) is the same as Proposition 15 when the function is replaced with a relation. For the other direction, assume that \mathcal{T} satisfies both **ATP** and **STP**. We first show that if R is approximately functionalisable then it is approximately determinisable when \mathcal{T} (that defines R) satisfies both **ATP** and **STP**. Let f be a rational function that approximately functionalises R w.r.t. Levenshtein distance, i.e., $d(R, f) < \infty$. Thus, an unambiguous transducer \mathcal{F} that defines f satisfies $d(\mathcal{T}, \mathcal{F}) < \infty$. From Lemma 3.9, because \mathcal{T} satisfies both **ATP** and **STP** and $d(\mathcal{T}, \mathcal{F}) < \infty$, it follows that \mathcal{F} also satisfies **ATP** and **STP**. Using Lemma 4.19, the rational function f is approximately determinisable. Let g be a sequential function that approximately determinises f w.r.t. Levenshtein, i.e., $d(f, g) < \infty$. Since d is a metric on relations (see Proposition 32), $d(R, g) \leq d(R, f) + d(f, g)$. Since both $d(R, f)$ and $d(f, g)$ are finite, we get $d(R, g) < \infty$. Hence, the rational relation R is approximately determinisable.

Now it suffices to show that R is approximately functionalisable w.r.t. Levenshtein family of distance. Towards this, we prove that $\text{diff}_d(R) < \infty$ when \mathcal{T} satisfies **ATP**. Observe that $\text{diff}_d(R) = \text{dia}_d(R_o)$ where $R_o = \{(v_1, v_2) \mid \exists u \in \text{dom}(R), (u, v_1), (u, v_2) \in R\}$ is a rational relation that consists of all pairs of output words of R on any input. Using Lemma 4.16, $\text{dia}_d(R_o) \leq N_{\mathcal{T}}$ when \mathcal{T} satisfies **ATP**. Hence, $\text{diff}_d(R) = \text{dia}_d(R_o) < \infty$, and R is approximately functionalisable w.r.t. d by virtue of Lemma 5.35. \blacktriangleleft

Since **ATP** and **STP** are decidable for transducers (Lemma 3.11), we obtain the following.

► **Theorem 38.** *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 39** (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 40.** *Checking whether a rational relation is d -approximate uniformisable for $d \in \{d_l, d_{lcs}, d_{dl}\}$ is undecidable.*

Proof. Let $\phi_1 : \{1, \dots, k\} \rightarrow B^*$ and $\phi_2 : \{1, \dots, k\} \rightarrow B^*$ be two morphisms defining an instance of the post correspondence problem (PCP), which asks whether there exists $w \in \{1, \dots, k\}^+$ such that $\phi_1(w) = \phi_2(w)$. Let $A = B \uplus \{1, \dots, i, a, b, \#\}$.

Let R be the relation which as input takes any word of the form $w_1 \# w_2 \# \dots w_m \# X$ where $w_i \in \{1, \dots, k\}^*$ for $1 \leq i \leq m$ and $X \in \{a, b\}$. Let us define the outputs:

- If $X = a$, then the only output is $\phi_1(w_1) \# \phi_1(w_2) \# \dots \phi_1(w_m) \#$.
- If $X = b$, then any word of the form $v_1 \# \dots v_m \#$ where $v_i \neq \phi_2(w_i)$ for all $1 \leq i \leq m$ is a valid output.

It can be shown that R is rational, recognizable by some transducer \mathcal{T} . We now show that R is approx-uniformisable iff PCP has no solution iff R is exact-uniformisable. First, suppose that PCP has a solution w and R is approx-uniformisable by some sequential transducer \mathcal{D} , i.e. $d(\mathcal{T}, \mathcal{D}) \leq K$ for some K . Consider inputs of the form $u_\ell = (w\#)^\ell$ for $\ell \geq 0$, and let $\alpha_\ell, \alpha_a, \alpha_b$ be such that $q_0 \xrightarrow{u_\ell \alpha_\ell} \mathcal{D} q, q \xrightarrow{a \alpha_a} \mathcal{D} q_f$ and $q \xrightarrow{b \alpha_b} \mathcal{D} p_f$, where q_0 is the initial state, q is a state and q_f, p_f are final states of \mathcal{D} . Since $d(\mathcal{T}, \mathcal{D}) \leq K$, for all ℓ , $d((\Phi_1(w)\#)^\ell, \alpha_\ell \alpha_a) \leq K$ holds. Similarly, for all ℓ , there exist v_1, v_2, \dots, v_ℓ all different from $\Phi_2(w)$ such that

$$d(v_1 \# v_2 \# \dots v_\ell \#, \alpha_\ell \alpha_b) \leq K.$$

Since $d(\alpha_\ell \alpha_a, \alpha_\ell \alpha_b)$ is uniformly bounded for all ℓ by some M , by applying triangular inequalities, we get that for all ℓ , there exist v_1, \dots, v_ℓ all different from $\Phi_2(w)$ such that

$$d((\Phi_1(w)\#)^\ell, v_1 \# v_2 \# \dots v_\ell \#) \leq 2K + M$$

Take $\ell = 4K + 2M + 2$, and fix a sequence of at most $2K + M$ edits from $(\Phi_1(w)\#)^\ell = (\Phi_1(w)\# \Phi_1(w)\#)^{2K+M+1}$ to $v_1 \# v_2 \# \dots v_\ell \#$. In this sequence, at least one copy of $(\Phi_1(w)\# \Phi_1(w)\#)$ is not edited by the sequence. Therefore, there exists some i such that $\#v_i\# = \# \Phi_1(w)\#$ and hence $v_i = \Phi_1(w)$. It is a contradiction since $v_i \neq \Phi_2(w) = \Phi_1(w)$. Therefore R is not approximately uniformisable.

Conversely, if there is no solution to PCP, then the following sequential function is an exact uniformiser of R : on any input of the form $w_1 \# \dots \# w_m \# X$ it outputs $\Phi_1(w_1) \# \dots \# \Phi_1(w_m)$. ◀

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 Löding. 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.