

# Equivalence kernels of sequential functions and sequential observation synthesis

Paulin Fournier

Université de Nantes, France

Nathan Lhote

University of Warsaw, Poland

---

## Abstract

We show that one can decide if a rational equivalence relation can be given as the equivalence kernel of a sequential letter-to-letter transduction. This problem comes from the setting of games with imperfect information. In [1, p. 6] the authors propose to model imperfect information by a rational equivalence relation and leave open the problem of deciding if one can synthesize a sequential letter-to-letter transducer (Mealy machine) which maps equivalent histories to the same sequence of observations. We also show that knowing if an equivalence relation can be given as the equivalence kernel of a sequential transducer is *undecidable*, even if the relation is given as a letter-to-letter transducer.

**2012 ACM Subject Classification** General and reference → General literature; General and reference

**Keywords and phrases** games, imperfect information, observation function, transducers

**Digital Object Identifier** 10.4230/LIPIcs.CVIT.2016.23

**Funding** *Nathan Lhote*: The second author was supported by the french National Reasearch Agency (ANR) DeLTA project (ANR-16-CE40-0007) and by the European Research Council (ERC) grant under the European Union's Horizon 2020 research and innovation programme (ERC Consolidator Grant LIPA, grant agreement No. 683080).



© Paulin Fournier and Nathan Lhote;  
licensed under Creative Commons License CC-BY

42nd Conference on Very Important Topics (CVIT 2016).

Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:1–23:16

Leibniz International Proceedings in Informatics



**LIPICs** Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## Introduction

**Motivation: games with imperfect information** The motivation for the present article comes from the paper: *Observation and Distinction. Representing Information in Infinite Games* by Dietmar Berwanger and Laurent Doyen, submitted to the arXiv in 2018 [1]. The authors propose an alternative way of representing imperfect information in games. The standard way to model imperfect information for a player is through a Mealy machine which transforms a sequence of game locations (a history) into a sequence of observations, which we call in the following an *observation function*. The proposed model of [1, p. 6] is to give instead a transducer recognizing an *indistinguishability relation*, *i.e.* an equivalence relation over game histories which recognizes those pairs of histories that are indistinguishable from the player’s perspective.

This new model is actually more expressive than the standard one (composing a Mealy machine with its inverse yields a transducer recognizing the indistinguishability relation), and one of the problems left open in [1, p. 22] is to decide when an indistinguishability relation can be transformed into an observation function, given as a Mealy machine.

Given a class **R** of equivalence relations and a class **F** of functions we define the **R,F-observation synthesis problem** as the problem of deciding if an equivalence relation in **R** can be expressed as the equivalence kernel<sup>1</sup> of a function in **F**, and if possible computing such a function.

The main goal of this article is to solve this problem for rational relations and functions given by Mealy machines. Moreover, we also consider the problem of constructing an observation function given, not as a Mealy machine but, as a sequential transducer, *i.e.* the outputs are not restricted to single letters but can be arbitrary words. In terms of observations, Mealy machines characterize the fact that each game move produces exactly one piece of observation (in some finite alphabet), while for sequential transducers, a move might produce several observations, or even none, in which case this step is *invisible* to the player.

**Contributions** We don’t use the vocabulary of games, but that of transducers, which is actually more suited to this problem: most of the proof techniques that we use stem from the theory of transducers. We consider several subclasses of **RatEq**, the set of rational equivalence relations, that is relations realized by transducers. The *equivalence kernel* of a total function  $f$ , is the equivalence relation defined by having the same image under  $f$ . The class **KerSeq** contains the equivalence relations that are the equivalence kernels of *sequential* transductions (a transducer is *sequential* if it is deterministic with respect to the input). The subclass **KerSeq<sup>ll</sup>** is the set of equivalence relations that are the equivalence kernel of a transduction given as a *sequential letter-to-letter* transducer (also known as a Mealy machine).

We start by studying the simpler class of **KerSeq<sup>ll</sup>** in Sec. 2 and then consider the class **KerSeq** in Sec. 3. Our main contribution is to give explicit characterizations for both classes **KerSeq** and **KerSeq<sup>ll</sup>**. For relations satisfying these properties, we exhibit a construction of a sequential, resp. letter-to-letter sequential, transducer whose kernel is the original relation. Finally we show that for rational equivalence relations, membership in **KerSeq<sup>ll</sup>** is *decidable*. In contrast, membership in **KerSeq** is *undecidable* even for letter-to-letter rational relations (also known as automatic, synchronous or regular relations).

---

<sup>1</sup> The equivalence kernel of a total function  $f$  is defined by  $x \sim y \Leftrightarrow f(x) = f(y)$



Note that while the characterization of  $\mathbf{KerSeq}^l$ , as well as the construction were already given in [1, Thm. 29, p. 19], the decidability status was left open. We reprove these results in our framework. Moreover, while extending the construction from  $\mathbf{KerSeq}^l$  to  $\mathbf{KerSeq}^{lp}$  is rather straightforward, obtaining the characterization for this class is difficult and actually the *most* challenging part of this article.

## 1 Words, relations, automata and transducers

**Words, languages and relations** An *alphabet*  $A$  is a set of symbols called *letter*. A word is a finite sequence of letters and we denote by  $A^*$  the set of finite words with  $\epsilon$  denoting the *empty word*. The length of a word  $w$  is denoted by  $|w|$  with  $|\epsilon| = 0$ . Given a non-empty word  $w$  and an integer  $1 \leq i \leq |w|$  we denote by  $w(i)$  the  $i$ th letter of  $w$ , by  $w(:i)$  the prefix of  $w$  up to position  $i$  included, and by  $w(i:)$  the suffix of  $w$  from position  $i$  included. Given two words  $u, v$  we write  $u \preceq v$  (resp.  $u \prec v$ ) to denote that  $u$  is a (resp. strict) prefix of  $v$ , and we write  $u^{-1}v$  the unique word  $w$  such that  $uw = v$ . A *language* over an alphabet  $A$  is a subset of  $A^*$ . A *word relation*  $R$  (or *transduction*) over alphabets  $A, B$  is a subset of  $A^* \times B^*$  and we often write  $uRv$  to denote  $(u, v) \in R$ . Let  $R(u) = \{v \mid uRv\}$ , and if  $R$  is a partial function from  $A^*$  to  $B^*$ , we rather write  $R(u) = v$  instead of  $R(u) = \{v\}$ . The *composition* of two relations  $R$  and  $S$  is  $R \circ S = \{(u, w) \mid \exists v, uSv \text{ and } vRw\}$ . The *inverse* of a relation  $R$  is  $R^{-1} = \{(v, u) \mid uRv\}$ . The *identity relation* over an alphabet  $A$  is  $Id = \{(u, u) \mid u \in A^*\}$ . The *domain* and *range* of a relation  $R$  are respectively:  $\text{dom}(R) = \{u \mid \exists v, uRv\}$  and  $\text{ran}(R) = \{v \mid \exists u, uRv\}$ .

We say that a relation  $S$  is *finer* than  $R$  (or that  $R$  is *coarser* than  $S$ ) if for any words  $u, v$ ,  $uSv \Rightarrow uRv$ , which we denote by  $S \subseteq R$ .

An equivalence relation  $R$  over alphabet  $A$  is a relation over alphabets  $A, A$  such that it is reflexive ( $Id \subseteq R$ ), symmetric ( $R^{-1} \subseteq R$ ) and transitive ( $R \circ R \subseteq R$ ). Taking the terminology of [4, Sec. 2], the (*equivalence*) *kernel* of a total function  $f : A^* \rightarrow B^*$  is the equivalence relation  $\ker(f) = \{(u, v) \mid f(u) = f(v)\} = f^{-1} \circ f$ . A *canonical function* for an equivalence relation  $R$  is a function  $f$  such that  $\ker(f) = R$ . The *transitive closure* of a relation  $R$ , denoted by  $R^+$ , is the finest transitive relation coarser than  $R$ . Given two equivalence relations  $S \subseteq R$  then any equivalence class of  $R$  is a union of equivalence classes of  $S$  and the *index* of  $S$  with respect to  $R$  is the supremum of the number of equivalence classes of  $S$  included in a unique equivalence class of  $R$ . We extend the notion of index to arbitrary relations  $S \subseteq R$ : the index of  $S$  with respect to  $R$  is the value  $\sup_{\substack{u, T \subseteq R(u) \\ \forall v \neq w \in T, v \not\sim_S w}} |T|$ .

We denote by  $S \subseteq_k R$  that the index of  $S$  with respect to  $R$  is at most  $k$ , by  $S \subseteq_{\text{fin}} R$  that the index of  $S$  with respect to  $R$  is finite, and by  $S \subseteq_{\infty} R$  that the index of  $S$  with respect to  $R$  is infinite.

The *valuedness* of a relation  $R$  is the supremum of the cardinal of the image set of a word, i.e.  $\sup_u |R(u)|$ .

**Automata and transducers** A *finite automaton* (or just automaton) over an alphabet  $A$  is a tuple  $\mathcal{A} = (Q, \Delta, I, F)$  where  $Q$  is a finite set of *states*,  $\Delta \subseteq Q \times A \times Q$  is a finite *transition relation* and  $I, F \subseteq Q$  are the sets of *initial states* and *final states*, respectively. A *run* of  $\mathcal{A}$  over a word  $w \in A^*$  is a word  $r \in Q^*$  of length  $|w| + 1$  such that for  $1 \leq i \leq |w|$ ,  $(r(i), w(i), r(i+1)) \in \Delta$ . We use the notation  $p \xrightarrow{w}_{\mathcal{A}} q$  (or just  $p \xrightarrow{w} q$  when  $\mathcal{A}$  is clear from context) to denote that there exists a run  $r$  of  $\mathcal{A}$  over  $w$  such that  $r(1) = p$  and  $r(|r|) = q$ . Let  $r$  be a run of  $\mathcal{A}$ , if  $r(1) \in I$  then  $r$  is called *initial*, if  $r(|r|) \in F$  then  $r$  is called *final* and a run which is both initial and final is called *accepting*. A word  $w$  is *accepted* by  $\mathcal{A}$  if



© Paulin Fournier and Nathan Lhote;  
licensed under Creative Commons License CC-BY

42nd Conference on Very Important Topics (CVIT 2016).

Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:3–23:16

Leibniz International Proceedings in Informatics



LIPIC Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

General case	Length-preserving
<b>RatEq</b>	$\mathbf{KerRat}^{lp} = \mathbf{KerRat}^{ll}$
$\cup$	$= \mathbf{RatEq}^{lp} = \mathbf{RatEq}^{ll}$
<b>KerRat</b>	$\mathbf{KerSeq}^{lp}$
$\cup$	$\mathbf{KerSeq}^{ll}$
<b>KerSeq</b>	

■ **Figure 1** Classes of rational equivalence classes.

there is an accepting run over it and the set of words accepted by  $\mathcal{A}$  is called the *language recognized* by  $\mathcal{A}$  and denoted by  $\llbracket \mathcal{A} \rrbracket$ . A language is called *rational* if it is recognized by some automaton.

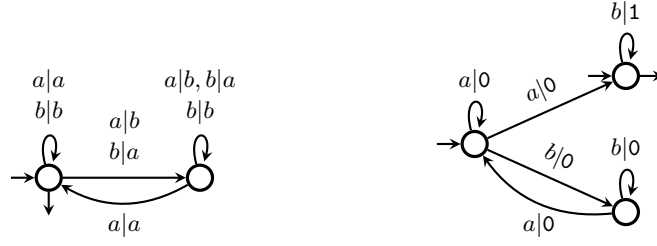
An automaton is called *deterministic* if it has a unique initial state, and for any pair of transitions  $(p, a, q_1), (p, a, q_2) \in \Delta$  we have  $q_1 = q_2$ .

A *finite transducer* over alphabets  $A, B$  is an automaton over  $A^* \times B^*$ . We define the natural projections  $\pi_A : (A^* \times B^*)^* \rightarrow A^*$  and  $\pi_B : (A^* \times B^*)^* \rightarrow B^*$ . We say that a pair of words  $(u, v) \in A^* \times B^*$  is *realized* by a transducer  $\mathcal{T}$  if there exists a word  $w$  such that  $\mathcal{T}$  has an accepting run  $r$  over  $w$ ,  $\pi_A(w) = u$  and  $\pi_B(w) = v$ , and we write  $(u, v) \in \llbracket \mathcal{T} \rrbracket$  with  $\llbracket \mathcal{T} \rrbracket$  denoting the *relation realized* by  $\mathcal{T}$ . A relation realized by a transducer is called *rational*. Given a transducer  $\mathcal{T} = (Q, \Delta, I, F)$  we define  $\pi_A(\mathcal{T})$  the *input automaton* of  $\mathcal{T}$  by  $(Q, \pi_A(\Delta), I, F)$ , where  $\pi_A(\Delta) = \{(p, a, q) \mid \exists b \in B^* (p, a, b, q) \in \Delta\}$ . A transducer is called *real-time* if its transitions are over the alphabet  $A \times B^*$  and *letter-to-letter* if its transitions are over  $A \times B$ . A real-time transducer whose input automaton is deterministic is called *sequential* and the function it realizes is also called sequential. We say that a relation  $R$  is *length-preserving* if for any words  $u, v$ ,  $uRv \Rightarrow |u| = |v|$ . A letter-to-letter transducer realizes a length-preserving relation and it is known that any length-preserving rational relation can be given as a letter-to-letter transducer. However, one can easily see that a sequential length-preserving function cannot in general be given as a letter-to-letter sequential transducer. For instance the function mapping  $aa$  to  $aa$  and  $ab$  to  $bb$  is sequential and length-preserving yet cannot be given as a sequential letter-to-letter transducer.

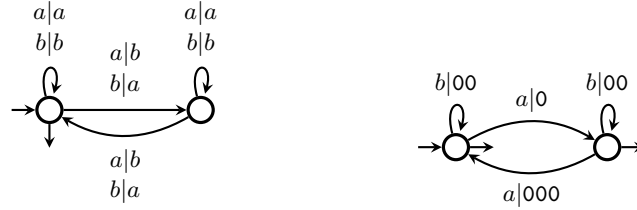
**Classes of rational equivalence relations** We define classes of equivalence relations: **RatEq** the class of all rational equivalence relations, **KerRat** the class of relations which are kernels of rational functions and **KerSeq** the class of relations which are kernels of sequential functions. For each of the previous classes  $\mathbf{C}$ , we define  $\mathbf{C}^{lp}$  as the class of *length-preserving* relations of  $\mathbf{C}$ . Similarly we define  $\mathbf{C}^{ll}$  by restricting to letter-to letter transducers, and we have obviously that  $\mathbf{C}^{ll} \subseteq \mathbf{C}^{lp}$ . For instance  $\mathbf{RatEq}^{ll}$  is the class of equivalence relations which are given by letter-to-letter transducers while  $\mathbf{KerSeq}^{ll}$  is the class of relations which are kernels of letter-to-letter sequential transducers. Fig. 1 gives the relative inclusions of the classes considered in this article, and a similar one can be found in [4, Fig. 1].

It is not known whether the classes **RatEq** and **KerRat** are equal or not. The generic problem we want to study is: given a rational equivalence relation, can we effectively decide if it is in **KerSeq**? Let  $R$  be a length-preserving equivalence relation given by a transducer  $\mathcal{T}$ , we know (*e.g.* from [3, Thm. 5.1]) that there is a canonical function given by a transducer which maps any word to the minimum, for the lexicographic order, of its equivalence class. Hence we have that  $\mathbf{RatEq}^{ll} = \mathbf{RatEq}^{lp} = \mathbf{KerRat}^{ll} = \mathbf{KerRat}^{lp}$  and  $\mathbf{KerSeq}^{ll} \subsetneq \mathbf{KerSeq}^{lp}$ , as we have seen above. We give in Fig. 2 an example of length-





■ **Figure 2** On the left a transducer recognizing an equivalence relation. On the right a transducer realizing a canonical function for it. Two words are equivalent if their last  $a$  is at the same position.



■ **Figure 3** An equivalence relation and a sequential canonical function for it. Two words are equivalent if their number of  $a$ 's is the same modulo 2.

preserving rational equivalence relation  $R$ , and we exhibit a rational canonical function for it. This equivalence relation is not in **KerSeq** and this can be shown using the characterization we prove in Sec. 3. Intuitively, one has to *guess* when reading an  $a$  if it is the last one or not, which cannot be done sequentially. In Fig. 3, we exhibit an equivalence relation which is length-preserving and is the kernel of a sequential function. However it is not the kernel of a *letter-to-letter* sequential function, which we will be able to show using the characterization from Sec. 2.

## 2 Kernels of sequential letter-to-letter functions

The goal of this section is to characterize relations which are kernels of sequential letter-to-letter functions. First, in Sections 2.1 and 2.2 we give two necessary conditions for a relation to be in **KerSeq**<sup>ll</sup>. Then in Sec. 2.3 we provide an algorithm to construct a sequential letter-to-letter canonical function when the two aforementioned conditions are satisfied, showing that they are indeed sufficient and thus characterize **KerSeq**<sup>ll</sup>. Finally in Sec. 2.4, we state the characterization established before and show that it is decidable.

### 2.1 Syntactic congruence

We start by introducing a notion of syntactic congruence associated with an equivalence relation, which will prove crucial throughout the paper. Given a relation  $R$ , we define  $S_R$  the *syntactic congruence* of  $R$  by  $uS_Rv$  if for any word  $w$ , we have  $uwRvw$ . In particular  $S_R$  is finer than  $R$  and  $S_R$  is a (right) congruence meaning that if  $uSv$  then for any letter  $a$  we have  $uaS_Rva$ . Furthermore, if  $R$  is an equivalence relation then so is  $S_R$ .

We now exhibit a first necessary condition to be in **KerSeq**, and *a fortiori* in **KerSeq**<sup>ll</sup>.

► **Proposition 1.** *Let  $R$  be an equivalence relation. If  $R \in \mathbf{KerSeq}$  then  $S_R$  has finite index with respect to  $R$ .*



© Paulin Fournier and Nathan Lhote;  
licensed under Creative Commons License CC-BY

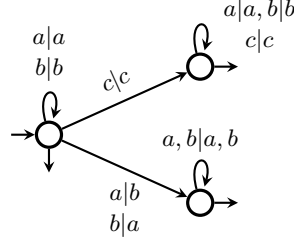
42nd Conference on Very Important Topics (CVIT 2016).

Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:5–23:16

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



■ **Figure 4** An equivalence relation not in **KerSeq**.

**Proof.** Let  $\mathcal{T}$  be a sequential transducer realizing a function  $f$  whose kernel is  $R$ , and let  $n$  be the number of states of  $\mathcal{T}$ . Let  $uRv$ , then we have  $f(u) = f(v)$ . Furthermore, if  $u, v$  reach the same state in  $\mathcal{T}$ , since  $\mathcal{T}$  is sequential,  $f(uw) = f(vw)$  for any word  $w$  which means that  $uS_Rv$ . Let  $u_1Ru_2R\ldots Ru_{n+1}$ . By a pigeon-hole argument, there must be two indices  $1 \leq i < j \leq n+1$ , such that  $u_i$  and  $u_j$  reach the same state in  $\mathcal{T}$ , hence  $u_iS_Ru_j$ . Thus we have shown that the index of  $S_R$  with respect to  $R$  is less than  $n$ , and is thus finite. ◀

We give in Fig. 4 an example of a length-preserving equivalence relation such that its syntactic congruence does not have a finite index with respect to it. Two different words are never syntactically equivalent, however two words of same length without any  $c$ s are equivalent. Thus by Prop. 1, this relation is not in **KerSeq**.

In the next two propositions, we show that 1) the syntactic congruence can be computed for a relation in **RatEq**<sup>lp</sup> and 2) that the finiteness of its index can also be decided.

► **Proposition 2.** *Let  $R$  be an equivalence relation given as a pair-deterministic letter-to-letter transducer. One can compute a transducer recognizing its syntactic congruence in PTIME.*

**Proof.** Let  $R$  be given by a letter-to-letter pair-deterministic transducer  $\mathcal{R}$ , and let  $S_R$  denote its syntactic congruence. Let  $(u, v)$  be a pair of words of equal length, and let us denote by  $p$  the state reached in  $\mathcal{R}$  after reading  $(u, v)$ . Then  $uS_Rv$  if and only if the automaton  $\mathcal{R}_p$  (obtained by taking  $p$  as initial state) recognizes a reflexive relation. This property can be easily checked and thus  $S_R$  is obtained by taking  $\mathcal{R}$  and restricting the final states to states  $p$  such that  $\mathcal{R}_p$  recognizes a reflexive relation. ◀

► **Proposition 3.** *Let  $R$  be a rational relation given as a transducer  $\mathcal{R}$ , and let  $f$  be a rational function given by a transducer  $\mathcal{F}$  such that  $S = \ker(f)$  is finer than  $R$ . Then one can decide if  $S$  has finite index with respect to  $R$  in PTIME.*

**Proof.** Let  $f$  be a rational function such that  $\ker(f) = S$ . We show that the index of  $S$  with respect to  $R$  is equal to the valuedness of  $T = f \circ R$ . We want to show that for any  $u$ ,  $|T(u)| = \max_{\substack{u, X \subseteq R(u) \\ \forall v \neq w \in X, v \not\mathcal{S} w}} |X|$ .

Let  $X \subseteq R(u)$  be such that  $\forall v \neq w \in X, v \not\mathcal{S} w$ . Then  $f$  is injective over  $X$  since  $f$  maps words to the same value if and only if they are  $S$  equivalent, thus  $|X| = |f(X)|$ . Moreover  $f(X) \subseteq T(u)$ , which means that  $|T(u)| \geq \max_{\substack{u, X \subseteq R(u) \\ \forall v \neq w \in X, v \not\mathcal{S} w}} |X|$ .

For each  $v \in T(u)$ , we can find a word  $v' \in f^{-1}(v)$  (for instance the minimum word in the lexicographic order). Let  $X$  be the set of these words, we have by construction  $|X| = |T(u)|$ . Moreover, for each pair of distinct words  $v', w' \in X$ , we have  $f(v') \neq f(w')$  and thus in particular  $v' \not\mathcal{S} w'$ . Thus we have shown  $|T(u)| \leq \max_{\substack{u, X \subseteq R(u) \\ \forall v \neq w \in X, v \not\mathcal{S} w}} |X|$ .



Hence the index of  $S$  with respect to  $R$  is equal to the valuedness of  $T$ . Since finite valuedness can be decided in PTIME [6, Thm. 3.1], then one can decide if  $S$  has finite index with respect to  $R$ , also in PTIME. ◀

► **Corollary 4.** *Let  $R \in \mathbf{RatEq}^{lp}$ , one can decide if its syntactic congruence  $S_R$  has finite index with respect to it.*

**Proof.** From Prop. 2 we can compute a transducer realizing  $S_R$ . According to [3, Thm. 5.1], we can even compute a transducer realizing a function whose kernel is  $S_R$ . Hence from Prop. 3 we can decide the finiteness of the index of  $S_R$  with respect to  $R$ . ◀

## 2.2 Prefix closure

Here we consider a second necessary condition of relations in  $\mathbf{KerSeq}^l$ , namely that they are prefix-closed.

The *prefix closure* of a relation  $R$  is the relation  $P_R$  defined by  $uP_Rv$  if there exists  $u', v'$ , with  $|u'| = |v'|$ , such that  $uu'Rvv'$ . A relation is called *prefix-closed* if it is equal to its prefix closure. We often say that  $u, v$  are *equivalent in the future* when  $uP_Rv$ .

► **Proposition 5.** *Let  $R$  be an equivalence relation. If  $R \in \mathbf{KerSeq}^l$  then  $R$  is prefix-closed.*

**Proof.** Let  $R$  be an equivalence relation, let  $f$  be realized by a transducer in  $\mathbf{KerSeq}^l$  such that  $\ker f = R$  and let  $P_R$  denote the prefix closure of  $R$ . Let  $uP_Rv$ , then there exist  $u', v'$  with  $|u'| = |v'|$  such that  $f(uu') = f(vv')$ . Since  $f$  is letter-to-letter sequential, we have  $f(u) \preceq f(uu')$ ,  $f(v) \preceq f(vv')$  and  $|f(u)| = |f(v)|$  which means that  $f(u) = f(v)$ . Hence  $uRv$ ,  $R = P_R$  and  $R$  is prefix-closed. ◀

The equivalence relations given in Figures 2 and 3 are *not* prefix-closed, which explains why they are not in  $\mathbf{KerSeq}^l$ , according to Prop. 5.

## 2.3 Construction of a canonical function

The main technical lemma of this section says that the two necessary conditions given above are sufficient:

► **Lemma 6.** *Let  $R \in \mathbf{RatEq}^{lp}$  be prefix-closed with a finite index syntactic congruence with respect to it. Then we can construct a sequential letter-to-letter transducer whose kernel is  $R$ .*

**Proof.** Let  $R \in \mathbf{RatEq}^{lp}$  be given by a transducer  $\mathcal{R} = (Q, \Delta_R, I, F_R)$  which is letter to letter, over the alphabet  $A \times A$ , and such that  $S_R \subseteq_k R = P_R$  for some  $k \in \mathbb{N}$ . Without loss of generality, we assume that  $\mathcal{R}$  is deterministic. A state of  $\mathcal{R}$  will be called *diagonal* if the identity is accepted from that state, and let  $D \subseteq F_R$  be the set of diagonal states. According to Prop. 2, we can obtain a letter-to-letter transducer  $\mathcal{S}$  realizing  $S$  (just by setting  $D$  as the set of final states).

Our goal is to define a sequential letter-to-letter transducer  $\mathcal{T}$  whose kernel is the relation  $R$ . The main idea to obtain this construction is to distinguish three kinds of relationships between two words: 1)  $uSv$  2)  $u\mathcal{S}v$  and  $uRv$  and 3)  $u\mathcal{R}v$ . Then the key idea, as seen in the proof of Prop. 1, is that two words in case number 2) *cannot* end up in the same state in  $\mathcal{T}$ . Two words in situation number 1) might as well reach the same state in  $\mathcal{T}$  since they have the exact same behavior. Then two words in situation 3) may or may not reach the same state, it does not matter since their image by  $\mathcal{T}$  should be different.



© Paulin Fournier and Nathan Lhote;  
licensed under Creative Commons License CC-BY

42nd Conference on Very Important Topics (CVIT 2016).

Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:7–23:16

Leibniz International Proceedings in Informatics



LIPIC Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



For each equivalence class of  $R$  containing  $l \leq k$  different  $S$ -equivalence classes we define  $l$  distinct states. The states will be pairs  $(M, i)$  where  $M \in \mathcal{M}_l(Q)$  is an  $l \times l$  square matrix with values in  $Q$ , the state space of  $\mathcal{R}$ , and  $i \in \{1, \dots, l\}$ . Let  $u_1, \dots, u_l$  be the least lexicographic representatives of the  $l$   $S$ -equivalence classes, in lexicographic order. Then  $M(i, j) = p$  if  $p$  is the state reached in  $\mathcal{R}$  after reading  $(u_i, u_j)$ . Then the state  $(M, i)$  is supposed to be the state reached after reading  $u_i$ , or any other  $S$ -equivalent word. Let us remark that the reachable states will only contain matrices where all states are accepting, *i.e.* with values in  $F_R$ . Moreover, all values on the diagonal are in  $D$ .

Let us define a sequential transducer  $\mathcal{T} = (Q_{\mathcal{M}}, \Delta, \{(M_0, 1)\})$  whose kernel will be the relation  $R$  (we don't specify the final states since all states are final). As we have seen, we define  $Q_{\mathcal{M}} = \bigcup_{l \in \{1, \dots, k\}} \mathcal{M}_l(Q) \times \{1, \dots, l\}$ . Since the word  $\epsilon$  is the only word of length 0, it is alone in its  $R$  and  $S$ -equivalence classes, hence  $M_0$  is the  $1 \times 1$  matrix with value  $q_0$  the initial state of  $\mathcal{R}$ . We have left to define  $\Delta$  and then show that the construction is correct. This will be done by induction on the length of the words. More precisely, let us state the induction hypothesis for words of length  $n$ :

- Hn.1:* Let  $u_1, \dots, u_l$  be the minimal representatives of the  $S$ -equivalence classes of some  $R$ -equivalence class, of words of length  $\leq n$ . Then any word  $uSu_i$  with  $i \in \{1, \dots, l\}$ , reaches the state  $(M, i)$  where  $M(j, j')$  is the state reached in  $\mathcal{R}$  by reading the pair  $(u_j, u_{j'})$ .
- Hn.2:* Two words, of length  $\leq n$ , are  $R$ -equivalent if and only if their outputs in  $\mathcal{T}$  are equal.

This trivially holds for the word of length 0, and let us assume that it holds for words of length  $\leq n$ . Let  $u_1, \dots, u_l$  be the minimal representatives of the  $S$ -equivalence classes of some  $R$ -equivalence class, of words of length  $n$ . Let us consider the corresponding matrix  $M \in \mathcal{M}_l(Q)$ .

Let us define an equivalence relation  $\sim_R$  over  $\{1, \dots, l\} \times A$  which will separate word which are no longer  $R$ -equivalent. Let  $q_{i,j,a,b}$  be the state reached in  $\mathcal{R}$  from  $M(i, j)$  by reading  $(a, b)$ . Two pairs  $(i, a), (j, b)$  are  $\sim_R$ -equivalent if  $q_{i,j,a,b} \in F_R$ . By *Hn.1* we know that this is indeed an equivalence relation. We define a second equivalence relation  $\sim_S$ . Two pairs  $(i, a), (j, b)$  are equivalent if  $q_{i,j,a,b} \in D$ . Finally, we consider a linear order on  $\{1, \dots, l\} \times A$  which is just the lexicographic order (with some fixed order over  $A$ ).

Let  $(i, a) \in \{1, \dots, l\} \times A$ , let us consider the set of minimal  $\sim_S$ -representatives of the  $\sim_R$ -equivalence class of  $(i, a)$ :

$$I_R = \{(j, b) \mid (j, b) \sim_R (i, a) \text{ and } \forall (j', b') < (j, b), (j, b) \not\sim_S (j', b')\}$$

Let  $l'$  denote the cardinal of  $I_R$ , *i.e.* the number of  $\sim_S$  equivalence classes in the  $\sim_R$ -equivalence class of  $i$ . We define the state  $(N, j)$  and the output  $b \in B$  such that  $((M, i), (a, b), (N, j)) \in \Delta$ . The output  $b$  is defined by  $\min I_R$ . The matrix  $N$  has dimension  $l'$  and let  $(i_1, a_1), \dots, (i_{l'}, a_{l'})$  be the elements of  $I_R$  in increasing order. The matrix  $N$  is defined by  $N(j, j') = p$  where  $p$  is the state reached from  $M(i_j, i_{j'})$  by reading  $(a_j, a_{j'})$ . Let  $j$  be the index such that  $(i, a) \sim_S (i_j, a_j)$ , then we have  $((M, i), (a, b), (N, j)) \in \Delta$ .

Let us show *Hn + 1.1*. Let  $uSu_ja$ , we need to show that  $u$  reaches the state  $(N, j)$ . Let  $vc = u$ , with  $c \in A$ . Since  $vcSu_ja$ , we have  $vcSu_ja$ , which means that  $vRu_j$ , since  $R$  is prefix closed. hence there exists  $u_{j'}$  such that  $vSu_{j'}$ . This means that we have  $u_{j'}cSu_ja$  and  $u_{j'}c \geq u_ja$  in the lexicographic order. By induction hypothesis,  $v$  reaches the state  $(M, j')$ , and by construction we have  $((M, j'), (c, b), (N, j)) \in \Delta$ .

We now show *Hn + 1.2*. Let  $v_1 = w_1a_1, v_2 = w_2a_2$  be two words of length  $n + 1$ , with  $a_1, a_2 \in A$ . If  $v_1Rv_2$ , then  $w_1Rw_2$  since  $R$  is prefix closed. By induction hypothesis, the outputs over  $w_1$  and  $w_2$  are the same. Moreover, by construction of  $\Delta$ , the final outputs reading  $a_1$  and  $a_2$ , respectively, are the same. If  $w_1 \not R w_2$ , then by induction, their outputs





are different, and so are the outputs over  $v_1, v_2$ . The only remaining case is when  $w_1 R w_2$  and  $v_1 \not R v_2$ . By induction, we have that the outputs over  $w_1, w_2$  are the same, hence we need to show that the outputs from the letters  $a_1, a_2$  are different. By the construction of  $\Delta$ , the outputs are linked with  $\sim_R$  equivalence classes, which means that the outputs corresponding to  $w_1, a_1$  and  $w_2, a_2$  are different. ◀

## 2.4 Characterization of $\text{KerSeq}^l$ and decidability

As a corollary we obtain a characterization of  $\text{KerSeq}^l$ .

► **Theorem 7** (Characterization of  $\text{KerSeq}^l$ ). *Let  $R \in \text{RatEq}$ . The following are equivalent:*

1.  $R \in \text{KerSeq}^l$
2.  $R$  is length-preserving and  $S_R \subseteq_{fin} R = P_R$

**Proof.**  $1. \Rightarrow 2.$  comes from the results of Prop. 1 and Prop. 5. To obtain  $2. \Rightarrow 1.$  we use the construction of Lem. 6. ◀

From the previous result we get an algorithm deciding if an equivalence relation is in  $\text{KerSeq}^l$ .

► **Theorem 8.** *The following problem is decidable.*

1. **Input:**  $\mathcal{R}$  a transducer realizing an equivalence relation  $R$ .
2. **Question:** Does  $R$  belong to  $\text{KerSeq}^l$ ?

**Proof.** Without loss of generality, we can assume that  $\mathcal{R}$  is a letter-to-letter pair-deterministic transducer. From Cor. 4 we can decide if  $S_R$  has finite index with respect to  $R$ . Deciding if  $R$  is prefix-closed, is easy: just check if a reachable state is not final.

According to Thm. 7, we thus have an algorithm to decide the problem. ◀

## 3 Kernels of sequential functions

We turn to the problem of deciding membership in  $\text{KerSeq}^{lp}$ . To tackle this we introduce another kind of transducers called *subsequential*, which are transducers allowed to produce a final output at the end of a computation. A *subsequential transducer* over alphabets  $A, B$  is a pair  $(\mathcal{T}, t)$ , where  $t : F \rightarrow B$  is called the *final output function* ( $F$  being the set of final states of  $\mathcal{T}$ ). We denote by  $\text{KerSub}$  the class of equivalence relations which are kernels of subsequential functions.

Our results are obtained in two steps. First we exhibit sufficient conditions for being in  $\text{KerSub}^l$  very similar to the characterization of  $\text{KerSeq}^l$ . Second we show that  $\text{KerSub}^l = \text{KerSeq}^{lp} = \text{KerSub}^{lp}$ .

### 3.1 Construction for $\text{KerSub}^l$

When studying relations in  $\text{KerSub}^l$ , we lose the property of being prefix-closed. We have to consider instead the transitive closure of the prefix closure.

► **Theorem 9.** *Let  $R \in \text{RatEq}^{lp}$ , let  $P_R$  be the prefix closure of  $R$  such that  $S_R$  has finite index with respect to  $P_R^+$ . Then we can construct a subsequential letter-to-letter transducer whose kernel is  $R$ .*



© Paulin Fournier and Nathan Lhote;  
licensed under Creative Commons License CC-BY

42nd Conference on Very Important Topics (CVIT 2016).

Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:9–23:16

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

**Proof.** From Prop. 2, we can obtain a transducer  $\mathcal{S}$  realizing  $S_R$ . Let  $\mathcal{P}$  be a transducer realizing  $P_R^+$ . Without loss of generality, we assume that  $\mathcal{R}, \mathcal{S}, \mathcal{P}$  are letter-to-letter and deterministic. Let us assume that  $S_R \subseteq_k P_R^+$ . We use the algorithm defined in the proof of Lem. 6 to obtain a transducer which realizes  $P_R^+$ , with state space  $\bigcup_{l \leq k} \mathcal{M}_l(Q)$ , where  $Q = Q_{\mathcal{R}} \times Q_{\mathcal{P}}$ , the product of the state spaces of  $\mathcal{R}$  and  $\mathcal{P}$ . Using the same construction we can obtain a sequential transducer realising  $P_R^+$  with the following properties:

- H.1: Let  $u_1, \dots, u_l$  be the minimal representatives of the  $S_R$ -equivalence classes of some  $P_R^+$ -equivalence class. Then any word  $uS_R u_i$  with  $i \in \{1, \dots, l\}$ , reaches the state  $(M, i)$  where  $M(j, j')$  is the state reached in  $\mathcal{R} \times \mathcal{P}$  by reading the pair  $(u_j, u_{j'})$ .
- H.2: Two words are  $P_R^+$  equivalent if and only if their outputs in  $\mathcal{T}$  are equal.

We only need to define a final output function  $t : Q_{\mathcal{R}} \times Q_{\mathcal{P}} \rightarrow B$  which will differentiate words that are  $P_R^+$  equivalent but not  $R$  equivalent. Let  $u_1, \dots, u_l$  be the minimal representatives of the  $S_R$ -equivalence classes of some  $P_R^+$ -equivalence class, and let  $M \in \mathcal{M}_l(Q)$  be the corresponding matrix such that  $M(i, j)$  is the state reached by reading  $(u_i, u_j)$  in  $\mathcal{R} \times \mathcal{P}$ . Then let us consider the equivalence relation  $\sim_R$  over  $\{1, \dots, l\}$  defined by  $i \sim_R j$  if and only if  $u_i R u_j$ . Then we define  $t(M, i) = \min_{j \sim_R i} j$ .

According to H.1 we only need to show that this construction is correct for minimal lexicographic representatives of  $S$  classes. Let  $u, v$  be two words of same length, and let us assume that  $u \not P_R^+ v$ . Then the images of  $u$  and  $v$  are already different, even without taking the final output into account. Let us assume that  $u P_R^+ v$ , then  $u, v$  have the same image by  $\mathcal{T}$ . If  $u R v$  considering that  $u, v$  are representatives of their respective  $S$  class, we have by definition that  $t(M, i_u) \neq t(M, i_v)$ , where  $(M, i_u)$  and  $(M, i_v)$  are the states reached by reading  $u$  and  $v$ , respectively. Similarly, we show that if  $u R v$ , then final outputs are the same which means that the image of  $u, v$  by  $(\mathcal{T}, t)$  is the same. ◀

### 3.2 Equality of classes

Let us start by stating the obvious inclusions which are just obtained by syntactic restrictions:  $\mathbf{KerSub}^{ll} \subseteq \mathbf{KerSub}^{lp}$  and  $\mathbf{KerSeq}^{lp} \subseteq \mathbf{KerSub}^{lp}$ .

We now show that one can remove the final outputs by adding modulo counting.

► **Lemma 10.**  $\mathbf{KerSub}^{ll} \subseteq \mathbf{KerSeq}$

**Proof.** Let  $(\mathcal{T}, t)$  with  $\mathcal{T} = (Q, \Delta, \{q_0\}, F)$  be a subsequential letter-to-letter transducer over  $A, B$  realizing a function  $f$ , and let  $g$  be the function realized by  $\mathcal{T}$ . Let  $\sim_t$  be an equivalence relation defined over  $F$  by  $p \sim_t q$  if  $t(p) = t(q)$ . Let  $u, v$  be two words that reach states  $p, q$  respectively from  $q_0$ . Then,  $f(u) = f(v)$  if and only if  $g(u) = g(v)$  and  $p \sim_t q$ . We know that the number of equivalence classes of  $\sim_t$  is less than  $n = |B|$ , so we number the equivalence classes from 1 to  $n$ . The main idea is to consider  $g^n$  which multiplies in  $g$  every occurrence of each letter by  $n$ , except for the last letter. Then, the number of occurrences of the last letter encodes, modulo  $n$ , the equivalence class of the state. Hence for any words  $u, v$  we have  $g^n(u) = g^n(v)$  if and only if  $g(u) = g(v)$  and  $p \sim_t q$  if and only if  $f(u) = f(v)$ , which means that the equivalence kernel of  $f$  is equal to that of  $g^n$ .

Let us now show that  $g^n$  is sequential. We extend the equivalence relation  $\sim_t$  arbitrarily to non final states, and to simplify things, we assume that the equivalence class of the initial state is  $n$ . Let us define a transducer  $\mathcal{T}^n = (Q \times B, \Delta^n, \{(q_0, b_0)\}, F \times B)$  realizing  $g^n$  (where  $b_0$  is some fixed letter in  $B$ ). Let  $p, q \in Q$  with respective equivalence classes  $i, j \in \{1, \dots, n\}$  such that  $(p, (a, b), q) \in \Delta$ . For any  $c \in B$  we have  $((p, c), (a, c^{n-i}b^j), (q, b)) \in \Delta^n$ . ◀



© Paulin Fournier and Nathan Lhote;  
licensed under Creative Commons License CC-BY

42nd Conference on Very Important Topics (CVIT 2016).

Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:10–23:16



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

We only have left to show that relations in  $\mathbf{KerSeq}^{lp}$  satisfy the sufficient conditions to be in  $\mathbf{KerSub}^{ll}$ .

We need a few technical results before showing the main lemma. The next claim is quite simple and just says that if two words can be equivalent in the future, then they can be equivalent in a near future.

▷ **Claim 11.** Let  $R \in \mathbf{RatEq}^{ll}$  and let  $P_R$  be the prefix closure of  $R$ . There exists  $D \geq 0$  such that for all  $u, v$  with  $uP_Rv$  there exists  $w_1, w_2$  with  $|w_1| = |w_2| \leq D$  and  $uw_1Ruw_2$ .

**Proof.** Let  $\mathcal{R}$  be a letter-to-letter transducer recognizing  $R$ . We assume without loss of generality that  $\mathcal{R}$  is pair-deterministic. Then the relation  $P_R$  is recognized by  $\mathcal{R}'$  which is just  $\mathcal{R}$  where all states that can reach a final state become final. Let  $D$  be the number of states of  $\mathcal{R}$ . If  $uP_Rv$  there exists  $w_1, w_2$  with  $|w_1| = |w_2| \leq D$  and  $uw_1Ruw_2$ . ◀

This next statement is a quite simple consequence of the previous one. If two words are equivalent in the future, then their images by a subsequential kernel function have to be close too.

▷ **Claim 12.**  $R \in \mathbf{KerSub}^{lp}$ , let  $f$  be a subsequential function such that  $\ker(f) = R$  and let  $P_R$  be the prefix closure of  $R$ . There exists  $\delta \geq 0$  such that for all  $u, v$  with  $uP_Rv$ ,  $||f(u)| - |f(v)|| \leq \delta$ .

**Proof.**  $R \in \mathbf{KerSub}^{lp}$ , let  $f$  be a subsequential function such that  $\ker(f) = R$  and let  $P_R$  be the prefix closure of  $R$ . Let  $(\mathcal{T}, t)$  be a subsequential transducer realizing  $f$  and let  $K$  be the maximal size of an output of  $(\mathcal{T}, t)$ . According to Lem. 11, we know that there exists  $D$  such that, if  $uP_Rv$ , then there exists  $w_1, w_2$  with  $|w_1| = |w_2| \leq D$  and  $uw_1Ruw_2$ . This means that  $f(uw_1) = f(vw_2)$ , and thus  $||f(u)| - |f(v)|| \leq 2KD$ . ◀

The next lemma is the most technical part of this section, and its proof is given in App. A due to a lack of space. It says that if two words are *transitively* future equivalent, then their images by a subsequential canonical function have to be close.

► **Lemma 13.**  $R \in \mathbf{KerSub}^{lp}$ , let  $f$  be a subsequential function such that  $\ker(f) = R$  and let  $P_R$  be the prefix closure of  $R$ . There exists  $D \geq 0$  such that for all  $u, v$  with  $uP_R^+v$ ,  $||f(u)| - |f(v)|| \leq D$ .

The previous lemma shows that two words that are *transitively* future equivalent must have close output from a subsequential canonical function. By a pigeon-hole argument we obtain in the next corollary that a relation in  $\mathbf{KerSub}^{lp}$  must have finite index with respect to the transitive closure of the future equivalence.

► **Corollary 14.** Let  $R \in \mathbf{KerSub}^{lp}$ , and let  $P_R$  denote the prefix closure of  $R$ . Then  $S_R$  has finite index with respect to  $P_R^+$ .

**Proof.** Let  $(\mathcal{T}, t)$  be a subsequential transducer realizing  $f$  such that  $\ker f = R$ , and let  $n$  be the number of states of  $\mathcal{T}$ . According to Lem. 13, there exists  $D$  such that for all  $u, v$  with  $uP_R^+v$ ,  $||f(u)| - |f(v)|| \leq D$ . Let  $N = |B|^{D+1}$  and let  $u_1P_R^+u_2P_R^+\dots P_R^+u_{(n+1)N}$ . For all  $i, j \in \{1, \dots, (n+1)N\}$ ,  $||f(u_i)| - |f(u_j)|| \leq D$ . This means that the set  $\{|f(u_i)| \mid 1 \leq i \leq (n+1)N\}$  has cardinality less than  $N$ . Thus there exists  $i_1 < \dots < i_{n+1}$  such that  $f(u_{i_1}) = \dots = f(u_{i_{n+1}})$ . One can see that there must be two indices  $1 \leq j < k \leq n+1$ , such that  $u_{i_j}$  and  $u_{i_k}$  reach the same state in  $\mathcal{T}$ , hence  $u_iRu_j$ , and even  $u_iS_Ru_j$ . Thus we have shown that the index of  $S_R$  with respect to  $P_R^+$  is less than  $(n+1)N$ , and is thus finite. ◀



© Paulin Fournier and Nathan Lhote;  
licensed under Creative Commons License CC-BY

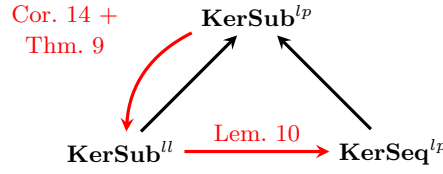
42nd Conference on Very Important Topics (CVIT 2016).

Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:11–23:16

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



■ **Figure 5** Proof of Prop. 15.

► **Proposition 15** (Equality of classes). *The following classes of equivalence relations are identical:*

1.  $\mathbf{KerSeq}^{lp}$
2.  $\mathbf{KerSub}^{ll}$
3.  $\mathbf{KerSub}^{lp}$

**Proof.** The proof is given in Fig. 5. The arrows represent class inclusion. Black arrows are trivial syntactic restrictions.

◀

## 4 Deciding membership in KerSeq

We show here that knowing if a rational equivalence relation is in  $\mathbf{KerSeq}$  is an undecidable problem, and this even if the relation is length-preserving. The trouble lies with computing the equivalence relation  $P_R^+$ . Indeed, transitive closures of even very simple relations are known not to be computable (the next configuration of a Turing machine can be computed by a simple transduction).

Let us first state a characterization of  $\mathbf{KerSeq}^{lp}$ , by combining the results of the previous subsections.

► **Theorem 16** (Characterization of  $\mathbf{KerSeq}^{lp}$ ). *Let  $R \in \mathbf{RatEq}^{ll}$ . The following are equivalent:*

1.  $R \in \mathbf{KerSeq}$
2.  $S_R \subseteq_{fin} R \subseteq_{fin} P_R^+$
3.  $S_R \subseteq_{fin} R \subseteq_{fin} P_R$  and  $\exists k P_R^k = P_R^{k+1}$

**Proof.**  $1 \rightarrow 2$ . Let  $R \in \mathbf{RatEq}^{ll}$ . Let us first assume that  $R \in \mathbf{KerSeq}$ . Then according to Cor. 14, we have  $S_R \subseteq_{fin} R \subseteq_{fin} P_R^+$ .

$2 \rightarrow 1$ . Conversely, let us assume that  $S_R \subseteq_{fin} R \subseteq_{fin} P_R^+$ . According to Theorem 9, we can construct a subsequential letter-to-letter transducer whose kernel is  $R$ . From Lem. 10, we have  $R \in \mathbf{KerSeq}$ .

$2 \rightarrow 3$ . Let us assume  $S_R \subseteq_{fin} R \subseteq_{fin} P_R^+$ . In particular  $S_R \subseteq_{fin} R \subseteq_{fin} P_R$ . Let us assume that  $S_R \subseteq_N P_R^+$ . Let  $uP_R^+v$  and let  $u = u_0P_Ru_1 \dots P_Ru_m = v$  be a chain of minimal length  $m$ . If we assume  $m > N$ , then there must exist  $i, j \leq m$  such that  $u_iS_Ru_j$ . Since  $u_i$  and  $u_j$  are *syntactically* equivalent, this means that  $u_iP_Rw \Leftrightarrow u_jP_Rw$ . Thus we can obtain a strictly smaller chain, which contradicts the assumption, thus  $P_R^N = P_R^{N+1}$ .

$3 \rightarrow 2$ . Finally, let us assume  $S_R \subseteq_{fin} R \subseteq_{fin} P_R$  and  $\exists k P_R^k = P_R^{k+1}$ . Let us assume that  $S \subseteq_N P_R$ . We only have to show  $S_R \subseteq_{N^i} P_R^i$  to conclude the proof. Let us assume that for some  $i$  we have  $S_R \subseteq_{N^i} P_R^i$ . We want to show that  $S_R \subseteq_{N^{i+1}} P_R^{i+1}$ . Let  $u \in A^+$ , let  $T = P_R^i(u)$ . Let  $T' \subseteq T$  be such that  $\forall v \in T, \exists! w \in T', vS_Rw$ . Thus we have  $P_R(T) = P_R(T')$  since  $S_R$  is the syntactic equivalence relation of  $R$ . Moreover, we have  $|T'| \leq N^i$  by assumption, since



© Paulin Fournier and Nathan Lhote;  
licensed under Creative Commons License CC-BY

42nd Conference on Very Important Topics (CVIT 2016).

Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:12–23:16



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Relations \ Kernels	<b>KerSeq<sup>ll</sup></b>	<b>KerSeq</b>	<b>KerRat</b>
<b>RatEq<sup>ll</sup></b>	<b>Dec.</b>	<b>Undec.</b> (Thm. 17)	<b>Yes</b>
<b>RatEq</b>	<b>Dec.</b> (Thm. 8)	<b>Undec.</b>	<b>?</b>

■ **Figure 6** Summary of the results.

for all words  $v, w \in T'$ ,  $v \mathcal{S}_R w$ . For each  $v \in T'$ , for each  $X \subseteq P_R(v)$  verifying  $\forall x, y \ x \mathcal{S}_R y$ , we know by assumption that  $|X| \leq N$ . Thus for any  $Y \subseteq P_R(T) = P_R(T')$  verifying  $\forall x, y \ x \mathcal{S}_R y$ , we know that  $|Y| \leq |T'| \cdot N \leq N^{i+1}$ , which concludes the proof. ◀

From this characterization we obtain two decidability results, one negative and one positive.

► **Theorem 17.** *The following problem is undecidable:*

**Input:**  $\mathcal{R}$  a letter-to-letter transducer realizing an equivalence relation  $R$ .

**Question:** Does  $R$  belong to **KerSeq**?

The proof of this theorem relies on a reduction of the *mortality problem*, see [2, p. 226] and is given in App. B. The next theorem shows that we are able to identify exactly where the undecidability comes from: computing the transitive closure of the relation  $P_R$ .

► **Theorem 18.** *The following problem is decidable:*

**Input:**  $\mathcal{R}, \mathcal{P}$  two transducers realizing equivalence relations  $R, P$ , respectively, such that  $P$  is the transitive closure of the prefix closure of  $R$ .

**Question:** Does  $R$  belong to **KerSeq<sup>lp</sup>**?

**Proof.** To show this we rely on the characterization from Thm. 16. We proceed as in the proof of Thm. 8, except that we want to check whether  $S_R$  has finite index with respect to  $P = P_R^+$  instead of  $R$ . First we can compute a transducer realizing  $S_R$ , according to Prop. 2. Then from [3, Thm. 5.1], we know we can obtain a transducer realizing a function  $f$  whose kernel is  $S_R$ . Then, using Prop. 3, we can decide if  $S_R$  has finite index with respect to  $P$ . ◀

We sum up the decidability of the problem for different classes of equivalence relations in the table of Fig. 6. New results are shown in red.

## Conclusion

We have studied the observation synthesis problem for two classes of observation functions: **KerSeq** and **KerSeq<sup>ll</sup>**. A natural question would be to consider the same problem for different classes of functions. However, the term *observation function* is only justified (and related to games with imperfect information) if the functions considered are *monotone* meaning that if  $h_1 \prec h_2$  denotes that history  $h_1$  is a prefix of history  $h_2$ , then any reasonable class of observation function should ensure that  $f(h_1) \prec f(h_2)$ , for any function  $f$ .

Since bounded memory and monotonicity somehow characterize the sequential functions, this means that such a class of observation functions would have to use unbounded memory, for instance the class of regular function, *i.e.* functions realized by two-way transducers. In terms of observations, this would mean that a single game step could give an arbitrary long (actually linear in the size of the history) sequence of observations.

## Acknowledgements

We would like to thank Bruno Guillon for his help in obtaining the undecidability result.



© Paulin Fournier and Nathan Lhote;  
licensed under Creative Commons License CC-BY

42nd Conference on Very Important Topics (CVIT 2016).

Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:13–23:16

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

## References

- 1 Dietmar Berwanger and Laurent Doyen. Observation and distinction. representing information in infinite games. *CoRR*, abs/1809.05978, 2018. URL: <http://arxiv.org/abs/1809.05978>, arXiv:1809.05978.
- 2 Philip K. Hooper. The undecidability of the turing machine immortality problem. *J. Symb. Log.*, 31(2):219–234, 1966. URL: <https://doi.org/10.2307/2269811>, doi:10.2307/2269811.
- 3 J. Howard Johnson. Do rational equivalence relations have regular cross-sections? In *Automata, Languages and Programming, 12th Colloquium, Nafplion, Greece, July 15-19, 1985, Proceedings*, pages 300–309, 1985. URL: <https://doi.org/10.1007/BFb0015755>, doi:10.1007/BFb0015755.
- 4 J. Howard Johnson. Rational equivalence relations. *Theor. Comput. Sci.*, 47(3):39–60, 1986. URL: [https://doi.org/10.1016/0304-3975\(86\)90132-5](https://doi.org/10.1016/0304-3975(86)90132-5), doi:10.1016/0304-3975(86)90132-5.
- 5 Jarkko Kari and Nicolas Ollinger. Periodicity and immortality in reversible computing. In Edward Ochmanski and Jerzy Tyszkiewicz, editors, *Mathematical Foundations of Computer Science 2008, 33rd International Symposium, MFCS 2008, Torun, Poland, August 25-29, 2008, Proceedings*, volume 5162 of *Lecture Notes in Computer Science*, pages 419–430. Springer, 2008. URL: [https://doi.org/10.1007/978-3-540-85238-4\\_34](https://doi.org/10.1007/978-3-540-85238-4_34), doi:10.1007/978-3-540-85238-4\_34.
- 6 Andreas Weber. On the valuedness of finite transducers. *Acta Inf.*, 27(8):749–780, 1990. URL: <https://doi.org/10.1007/BF00264285>, doi:10.1007/BF00264285.

## A Proof of Lem. 13

**Proof.**  $R \in \mathbf{KerSub}^{lp}$ , let  $f$  be a subsequential function such that  $\ker(f) = R$  and let  $P$  be the prefix closure of  $R$ . Let  $(\mathcal{T}, t)$  be a subsequential transducer realizing  $f$  and let  $K$  be the maximal size of an output of  $(\mathcal{T}, t)$ . Let us remark that there are no loops in  $\mathcal{T}$  that produce nothing. Indeed, if we assume otherwise, then we can find a loop in  $\mathcal{T}$  producing nothing, contradicting the fact that  $R$  is length-preserving. Hence let  $k$  be the smallest ratio of output length over input length for a simple loop in  $\mathcal{T}$ . Thus we have for any words  $u, v$ ,  $k|v| - b \leq ||f(uv)| - |f(u)|| \leq K|v|$ .

Let us assume towards a contradiction that the statement does not hold. This means that for any  $D$ , we can find a sequence  $u_0 P u_1 P \dots P u_N$ , such that  $||f(u_0)| - |f(u_N)|| \geq D$ . Without loss of generality, let us assume that  $|f(u_0)|$  is minimal among  $\{|f(u_i)| \mid 0 \leq i \leq N\}$ . According to Lem. 12 there exists  $\delta$  such that  $||f(u_{i-1})| - |f(u_i)|| \leq \delta$ , for any  $i \in \{1, \dots, N\}$ . Thus, for any integer  $M \in \{|f(u_0)|, |f(u_0)| + 1, \dots, |f(u_N)|\}$ , there exists  $i \in \{0, \dots, N\}$  and  $d \in \{0, \dots, \delta\}$  such that  $|f(u_i)| + d = M$ .

Let  $C > \max(3, K, \frac{2K}{k}, \delta, b)$  be a large enough integer. We extract a subsequence of the  $u_i$ s defined in the following way. Let  $i \geq 0$  be such that  $|f(u_0)| + C^i + \delta \leq |f(u_N)|$ , then there exists  $u \in \{u_0, \dots, u_N\}$  such that  $|f(u)| = C^i + d_i$ , with  $d_i \in \{0, \dots, \delta\}$ , and we set  $v_i = u$ . Let  $v'_i$  be the smallest prefix of  $v_i$  such that  $|f(v'_i)| = |f(u_0)| + d'_i$  with  $d'_i \in \{0, \dots, \delta\}$ . Then we obtain:

$$\begin{aligned} k||v_i, v'_i|| - b &\leq ||f(v_i), f(v'_i)|| \leq K||v_i, v'_i|| \\ k||v_i, v'_i|| - b &\leq C^i + d_i - d'_i \leq K||v_i, v'_i|| \end{aligned}$$

Using these inequalities for  $i + 1$  and  $i$  we have:



© Paulin Fournier and Nathan Lhote;  
licensed under Creative Commons License CC-BY  
42nd Conference on Very Important Topics (CVIT 2016).

Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:14–23:16

Leibniz International Proceedings in Informatics

**LIPICs** Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



$$\begin{aligned}
\|v_{i+1}, v'_{i+1}\| &\geq \frac{C}{K} C^i + \frac{d_{i+1} - d'_{i+1}}{K} \\
&\geq \frac{C}{K} (k\|v_i, v'_i\| - b + d'_i - d_i) + \frac{d_{i+1} - d'_{i+1}}{K} \\
&\geq C \frac{k}{K} \|v_i, v'_i\| + \frac{C(-b + d'_i - d_i) + d_{i+1} - d'_{i+1}}{K} \\
&\geq C \frac{k}{K} \|v_i, v'_i\| - \frac{C(b + \delta) + \delta}{K} \\
&> C \frac{k}{K} \|v_i, v'_i\| - \frac{2C^2 + C}{K}
\end{aligned}$$

It suffices to show  $C \frac{k}{K} \|v_i, v'_i\| - \frac{2C^2 + C}{K} \geq \|v_i, v'_i\|$  in order to obtain  $\|v_{i+1}, v'_{i+1}\| > \|v_i, v'_i\|$ .

$$\begin{aligned}
C \frac{k}{K} \|v_i, v'_i\| - \frac{2C^2 + C}{K} &\geq \|v_i, v'_i\| \\
\iff Ck\|v_i, v'_i\| - (2C^2 + C) &\geq K\|v_i, v'_i\| \\
\iff \|v_i, v'_i\| &\geq \frac{2C^2 + C}{Ck - K}
\end{aligned}$$

Since  $C > \frac{2K}{k}$ , we only have to show  $\|v_i, v'_i\| \geq 2C^2 + C$ . Moreover, we know that  $\|v_i, v'_i\| \geq \frac{C^i - \delta}{K}$ . Thus it suffices to show that  $\frac{C^i - \delta}{C} \geq 2C^2 + C$ , since  $C$  is larger than both  $K$  and  $\delta$ . The inequality holds, as long as  $i \geq 4$ , since  $C$  is larger than 3.

This means that  $\|v_{i+1}, v'_{i+1}\| > \|v_i, v'_i\|$  for any  $i > 3$ . Since all  $v_i$ s have the same length, this means that all  $v'_i$ s have different length, for  $i > 3$ . For  $D$  large enough, we can assume that there are more than  $B^{\delta+1}$   $v'_i$ s of different lengths. Thus there must exist two with the same image, which contradicts the assumption that  $R$  is length-preserving. ◀

## B Proof of Thm. 17

**Proof.** We use a reduction from the following problem, which we call the *bounded configuration problem*:

**Input:**  $M$  a reversible Turing machine

**Question:** Is there a computation  $c_1 \rightarrow c_2 \rightarrow \dots$  which visits an infinite number of configurations.

We first give the reduction and then show that the problem is actually undecidable.

Let  $M$  be a Turing machine with alphabet  $\Sigma$ , a state space  $Q$  and a transition function  $\delta : Q \times \Sigma \rightarrow Q \times \Sigma \times \{\text{left}, \text{right}\}$ . A *configuration* is a word over  $\Sigma \cup Q$ , with exactly one occurrence of a letter in  $Q$ .

We define a letter-to-letter transducer  $\mathcal{R}$  recognizing an equivalence relation  $R$ , with a prefix closure  $P$ . Let  $c_1, c_2$  be a pair of consecutive configurations, then  $\mathcal{R}$  recognizes the pairs  $(c_1\#1, c_2\#2)$ , and  $(c_2\#2, c_1\#1)$  by symmetry. Note that these equivalence classes of  $R$  have size 2. Words of the shape  $c\#$ , with  $c$  a configuration are only equivalent to themselves. Words that are strict prefixes of words of the shape  $c\#$  are all equivalent, if they have the same size. All other words are only equivalent to themselves.

One can easily see that there exists  $k$  such that  $P^k = P^{k+1}$  if and only if computations of  $M$  visit at most  $k + 1$  different configurations. We only have left to check that there are computations of unbounded size if and only if there is an infinite computation. Let  $c_1, c_2, \dots$  be configurations such that from  $c_n$ , the machine  $M$  visits at least  $n$  distinct configurations. Then we can extract a subsequence  $d_1, d_2, \dots$  such that all configurations start in the same state. Extracting a subsequence we can assume that all cells of the tape at distance 1 from the reading head agree. Repeating the operation, we end up with a configuration which visits more than  $n$  configurations for any  $n$ , i.e. is infinite.

▷ **Claim 19.** The bounded configuration problem is undecidable.



© Paulin Fournier and Nathan Lhote;  
licensed under Creative Commons License CC-BY

42nd Conference on Very Important Topics (CVIT 2016).

Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:15–23:16



Leibniz International Proceedings in Informatics

LIPIC Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

**Proof.** This is shown by a reduction from the *mortality problem* which amounts to deciding if a Turing machine has an infinite computation. Note that this is different from the halting problem, because we ask if the machine halts on *all* possible configurations. This problem was shown to be undecidable in [2, p. 226] for Turing machines and in [5, Thm. 7] for *reversible* Turing machines.

Assume that the bounded configuration problem is decidable. Given a reversible machine  $M$ , if it has a computation visiting an infinite number of configurations, then it has an infinite computation. If there is no computation visiting an infinite number of distinct configurations then there is a uniform bound on the number of configurations that a computation can visit. This bound  $k$  can be computed, just by simulating the machine on larger and larger configurations. Then, one can easily see if one of the computations loops, and thus decide if the machine has an infinite computation. ◀

◀



© Paulin Fournier and Nathan Lhote;  
licensed under Creative Commons License CC-BY

42nd Conference on Very Important Topics (CVIT 2016).

Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:16–23:16



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany