Noname manuscript No.
(will be inserted by the editor)

Synchronizing Relations on Words

Diego Figueira · **Leonid Libkin**

Received: date / Accepted: date

Abstract While the theory of languages of words is very mature, our understanding of *relations* on words is still lagging behind. And yet such relations appear in many new applications such as verification of parameterized systems, querying graph-structured data, and information extraction, for instance. Classes of well-behaved relations typically used in such applications are obtained by adapting some of the equivalent definitions of regularity of words for relations, leading to non-equivalent notions of recognizable, regular, and rational relations.

The goal of this paper is to propose a systematic way of defining classes of relations on words, of which these three classes are just natural examples, and to demonstrate its advantages compared to some of the standard techniques for studying word relations. The key idea is that of a *synchronization* of a pair of words, which is a word over an extended alphabet. Using it, we define classes of relations via classes of regular languages over a fixed alphabet, just $\{1,2\}$ for binary relations. We characterize some of the standard classes of relations on words via finiteness of parameters of synchronization languages, called shift, lag, and shiftlag. We describe these conditions in terms of the structure of cycles of graphs underlying automata, thereby showing their decidability. We show that for these classes there exist canonical synchronization languages, and every class of relations can be effectively re-synchronized using those canonical representatives. We also give sufficient conditions on synchronization languages, defined in terms of injectivity and surjectivity of their Parikh images, that guarantee closure under intersection and complement of the classes of relations they define.

CR Subject Classification F.4.3 Formal Languages

Keywords Word Relations · Regular · Rational · Recognizable

The current article is the full version of [19].

University of Edinburgh
10 Crichton street, Edinburgh EH8 9AB, UK
Tel.: +44 131 650 5132 Fax: +44 131 651 1426
E-mail: {dfigueir, libkin}@inf.ed.ac.uk

1 Introduction

Foundations of formal language theory have been largely developed in the 1960s and 1970s, and used heavily in practically all areas of computer science. The field itself stayed somewhat dormant for a while, but that changed over the past 10–15 years due to new application areas requiring techniques that could not have been foreseen 30 or 40 years earlier. Among consumers of results in formal language theory are verification (for instance, automata-based approaches to model-checking are now part of standard industrial verification tools [7,25]) and data management (standards for describing and querying XML documents, for instance, are rooted in both word and tree automata [27,31], and emerging graph data models are borrowing many formal language concepts [3]).

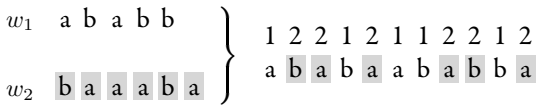
Of interest to us in this paper are *relations on words*. That is, for a given finite alphabet $\mathbb{A}$, we deal with binary relations $R \subseteq \mathbb{A}^* \times \mathbb{A}^*$. Their study goes back to Elgot, Mezei, Nivat in the 1960s [17,28] with much subsequent work done later (see, e.g., surveys [9,15]). The standard notions of regularity that generate the same class of languages —recognizability by finite monoids, definability by automata, or by regular expressions— give rise to different classes of relations, called *recognizable, regular, and rational* relations. Their properties may differ significantly from properties of regular languages: for instance, rational relations are not closed under intersection and it is even undecidable whether the intersection of two such languages is non-empty. Recognizable relations are just unions of products of regular languages; examples of regular relations are prefix, equality, or equal length of words; and examples of rational relations are suffix, subword (for instance, *bb* is a subword of *aabbaa*), and subsequence (*bb* is a subsequence of *abaaba*: letters need not be consecutive).

There has been renewed interest in relations on words as of late. One motivation comes from verification of safety and liveness properties of parameterized systems, where such relations describe transitions [1,12,23,32]. Another comes from graph databases, which are actively studied as a suitable model for RDF data, social networks data, and others [3]. Paths in graph databases are described by their labels, and need to be compared, for instance, for their degree of similarity, e.g., their edit distance [4,6,26]. Yet another example is the study of formal models underlying IBM's tools for information extraction [18].

Many of the basic questions that arise in these new applications, however, are not the kind of questions that had been addressed previously. Just to give an example, it is well known that checking nonemptiness of the intersection of a rational relation and a regular relation is an undecidable problem. But what about *really used* rational relations such as subword, suffix, subsequence (as opposed to artificial codings of the halting problem) – can we test if their intersection with regular relations is nonempty? However natural these questions are, they were answered only recently [5].

An even more basic question relates to the very choice and structure of the main classes of relations: recognizable, regular, and rational. They appeared in a somewhat ad hoc way, just as analogs of different ways of defining regularity of languages, but is there another way to explain these, and perhaps other classes as well? This is the main point of our paper: we argue that there is a natural way to study relations on words, and we do it by explaining how positions in words are *synchronized*.

As an example of synchronization, consider words $w_1 = ababb$ and $w_2 = baaaba$. We can represent this pair as a single word over $\{a,b\}$, by shuffling w_1 and w_2, i.e., interspersing letters of w_1 among letters of w_2. For each position in the shuffle, we remember which word it came from – this is indicated by the symbols 1 or 2 above the letters in the figure.



When we read the letters marked i, for $i = 1,2$ we get the word w_i. The word over $\{1,2\}$ provides a *synchronization* of the pair (w_1,w_2) – in our example, 12212112212. We show that the commonly occurring classes of relations over words follow the same principle:

1. to decide whether (w_1,w_2) is in the relation, one runs an automaton over the shuffle;
2. classes of relations are then determined by the classes of allowed synchronizations.

For instance, recognizable relations are given by synchronizations from 1^*2^*, length-preserving regular relations by synchronizations from $(12)^*$, arbitrary regular relations by synchronizations from $(12)^*(1^*|2^*)$, and rational relations by synchronizations from $(1|2)^*$.

For relations, we have proper inclusions *recognizable* $\subsetneq$ *regular* $\subsetneq$ *rational* [9], making them very different from languages. This raises the question: since every recognizable language is regular, and yet 1^*2^* is *not* contained in $(12)^*(1^*|2^*)$, there must be multiple ways of synchronizing relations to obtain even known classes. What are these ways, and how can they be characterized? And will those characterizations lead to new naturally appearing classes?

These are the questions we answer. We define three parameters of regular languages in $(1|2)^*$: the *shift* says how often we switch between 1s and 2s, the *lag* says how big the difference between the numbers of 1 and 2 is allowed to get, and *shiftlag* combines the two in a certain way. Then finite shift characterizes recognizability, while finite shiftlag characterizes regularity of relations. Finite lag, which appears to be a natural measure then, captures another known class of relations.

We provide automata characterizations of classes of synchronization languages in terms of the structure of cycles in the graph representations of automata. All these turn out to be decidable. This shows one advantage of dealing with relations in terms of their synchronizations. For instance, it is known that checking whether a given rational relation is regular, is an undecidable problem (assuming the input is a transducer, i.e., an automaton with output [9]). However, if the input to the problem is a synchronization language, then it *is* decidable whether the relations it describes are all regular.

Another advantage of describing relations by their synchronizations is the ability to find classes closed under intersection or complementation (rational relations, for

instance, are not). We do it by imposing decidable conditions on Parikh images of synchronization languages to guarantee closure properties of classes of relations they give rise to.

We also look at re-synchronization of relations. For each class of relations, there may be many different regular synchronizing languages over $\{1,2\}$. We show that in the standard cases, there exist canonical synchronizing languages, and relations can be effectively resynchronized using those canonical languages.

2 Recognizable, regular, and rational relations

We start with some basic notations. Throughout the paper, $\mathbb{A}$ stands for a finite alphabet, $\mathbb{N} = \{1,2,\dots\}$ for the set of positive natural numbers, and $\mathbb{N}_0$ for $\mathbb{N} \cup \{0\}$. The set of all words over $\mathbb{A}$ is denoted by $\mathbb{A}^*$, and the length of w in $\mathbb{A}^*$ is denoted by $|w|$. If $w = a_1 \dots a_n$, then $w[i,j]$ stands for the subword $a_i \dots a_j$; in particular, $w[i]$ is the letter a_i.

Recall that there are three standard ways of defining regular languages:

- Recognizability by finite monoids: the set $\mathbb{A}^*$, equipped with the concatenation operation (denoted by '$\cdot$', whose unit is the empty word 'ε') is a monoid. A set $L \subseteq \mathbb{A}^*$ is recognizable if there is a finite monoid M and a homomorphism $\langle \mathbb{A}^*, \cdot, \varepsilon \rangle \to M$ so that $L = f^{-1}(M_0)$ for some $M_0 \subseteq M$.
- Definability by finite automata, say NFAs.
- Definability by regular (sometimes called rational) expressions, i.e., those built from the empty word and alphabet letters using union, concatenation, and the Kleene star.

Classical formal language theory tells us that these definitions generate the same class of languages, known as regular languages. We now adapt them to binary relations on words.

Recognizable relations Since $\langle \mathbb{A}^*, \cdot, \varepsilon \rangle$ is a monoid, $\mathbb{A}^* \times \mathbb{A}^*$ has the structure of a monoid too. We can thus define *recognizable relations* as sets $R \subseteq \mathbb{A}^* \times \mathbb{A}^*$ for which there is a finite monoid M and a morphism $f : \mathbb{A}^* \times \mathbb{A}^* \to M$ such that $R = f^{-1}(M_0)$ for some $M_0 \subseteq M$. This class will be denoted by REC.

Regular relations Let $\perp \notin \mathbb{A}$ be a new alphabet letter. A pair (w_1, w_2) of words from $\mathbb{A}^*$ can be encoded by a single word of length $\max(|w_1|, |w_2|)$ over the alphabet $(\mathbb{A} \cup \{\perp\}) \times (\mathbb{A} \cup \{\perp\})$: its ith letter is the pair containing the ith letter of w_1 and the ith letter of w_2, with $\perp$ used when i is greater than the length of w_1 or w_2. For example, the encoding for the words of the figure of page 3 is $(a,b)(b,a)(a,a)(b,a)(b,b)(\perp,a)$. A regular relation R is given by an automaton over this alphabet: it contains pairs (w_1, w_2) whose encodings are accepted by the automaton. The class of regular relations is denoted by REG.

Rational relations There are two equivalent ways of defining them. One uses regular expressions, which are now built from pairs in $(\mathbb{A} \cup \{\varepsilon\}) \times (\mathbb{A} \cup \{\varepsilon\})$ using the same operations of union, concatenation, and Kleene star. Alternatively, rational relations can be defined by means of 2-tape automata, that have 2 heads for the

tapes and one additional control; at every step, based on the state and the letters it is reading, the automaton can enter a new state and move some (not necessarily all) tape heads. The class of rational relations is denoted by RAT.

Relations in REC are exactly the finite unions of products of regular languages over $\mathbb{A}$ [9,17]. Examples of relations in REG $\setminus$ REC are prefix, equality, or equal length. Examples of relations in RAT $\setminus$ REG are suffix, given by $\left(\bigcup_{a\in\mathbb{A}}(\varepsilon,a)\right)^*\cdot\left(\bigcup_{a\in\mathbb{A}}(a,a)\right)^*$; subword: $\left(\bigcup_{a\in\mathbb{A}}(\varepsilon,a)\right)^*\cdot\left(\bigcup_{a\in\mathbb{A}}(a,a)\right)^*\cdot\left(\bigcup_{a\in\mathbb{A}}(\varepsilon,a)\right)^*$, and subsequence: $\left(\bigcup_{a\in\mathbb{A}}(\varepsilon,a)\cup(a,a)\right)^*$.

Note that unlike in the case of languages, where the three notions coincide, we have REC $\subsetneq$ REG $\subsetneq$ RAT. The classes REC and REG are closed under intersection; however the class of rational relations is not. In fact, one can find $R\in$ REG and $S\in$ RAT so that $R\cap S\notin$ RAT. However, if $R\in$ REC and $S\in$ RAT, then $R\cap S\in$ RAT.

Relations in REC and REG inherit all the closure/decidability properties of regular languages. If $R\in$ RAT, then each of its projections is a regular language, and can be effectively constructed. Hence, the nonemptiness problem is decidable for RAT. However, testing nonemptiness of the intersection of two rational relations is undecidable. We refer to [9,14,30] for basic information on these relations and their decision problems.

3 Synchronizations of relations

We now formalize the idea of synchronizations informally described in the introduction. We write $\mathbf{k}$ for the set $\{1,\dots,k\}$. A *synchronization* of a pair (w_1,w_2) of words in $\mathbb{A}^*$ is a word over $\mathbf{2}\times\mathbb{A}$ so that the projection on $\mathbb{A}$ of positions labeled i is exactly w_i, for $i=1,2$ (see the figure on page 3). Every word w in $(\mathbf{2}\times\mathbb{A})^*$ is a synchronization of a uniquely determined pair (w_1,w_2), where w_i is the sequence of $\mathbb{A}$-letters corresponding to the symbol i in the first position of $\mathbf{2}\times\mathbb{A}$. We denote such (w_1,w_2) by $[\![w]\!]$ and extend it to languages $S\subseteq(\mathbf{2}\times\mathbb{A})^*$ by $[\![S]\!]=\{[\![w]\!]\mid w\in S\}$.

For two words $u=a_1\cdots a_n\in\mathbb{A}^*$ and $v=b_1\cdots b_n\in\mathbb{B}^*$, we write $u\otimes v$ for the word $(a_1,b_1)\cdots(a_n,b_n)\in(\mathbb{A}\times\mathbb{B})^*$. The main idea of our approach to relations on words comes from two different ways of viewing words in $(\mathbf{2}\times\mathbb{A})^*$.

- Every word $w\in(\mathbf{2}\times\mathbb{A})^*$ is a synchronization of a pair $[\![w]\!]=(w_1,w_2)$.
- Every word $w\in(\mathbf{2}\times\mathbb{A})^*$ is of the form $u\otimes v$ with $u\in\mathbf{2}^*$ and $v\in\mathbb{A}^*$.

This makes it possible to define relations consisting of pairs $[\![w]\!]$ with restricted synchronizations, i.e., $w=u\otimes v$ and u belongs to a given language $L\subseteq\mathbf{2}^*$.

Formally, if $L\subseteq\mathbf{2}^*$, we say that $u\otimes v$ is *L-controlled* if $u\in L$; a language is L-controlled if all its words are. We now look at relations given by L-controlled synchronizations, i.e., for a regular language $L\subseteq\mathbf{2}^*$, let

$$\textsc{Rel}(L)\;=\;\{[\![S]\!]\;\mid\;S\text{ is a regular }L\text{-controlled language}\}\tag{1}$$

If $\mathscr{C}$ is a class of relations over $\mathbb{A}^*$, then $L\subseteq\mathbf{2}^*$ is a *synchronization* for $\mathscr{C}$ if $\textsc{Rel}(L)\subseteq\mathscr{C}$, that is, all relations given by L-controlled synchronizations belong to $\mathscr{C}$. We remark that a similar approach to defining relations was used in [21], although the questions considered were completely different.

Procedurally, each relation in $\textsc{Rel}(L)$ is obtained as follows:

1. Choose an automaton over $\mathbf{2} \times \mathbb{A}$;
2. consider words $u \otimes v$ it accepts so that $u \in L$,
3. view v as a synchronization of (w_1, w_2) and add the pair to the relation.

This view suggests natural candidates for capturing classes $\mathsf{REC}, \mathsf{REG}$, and RAT. For REC, relations are unions of products of regular languages, so synchronizations are of the form 1^*2^*: one starts by going over the first word, and then over the second. For REG, they are from $(12)^*(1^*|2^*)$: we first go over two words letter-by-letter, and then write out the rest of the longer word. For RAT, there are no restrictions. Indeed, we can show the following.

Proposition 1

(I) $\textsc{Rel}(1^*2^*) = \mathsf{REC}$.
(II) $\textsc{Rel}((12)^* \cdot (1^*|2^*)) = \mathsf{REG}$.
(III) $\textsc{Rel}((1|2)^*) = \mathsf{RAT}$.

Proof (I)-[$\supseteq$] The fact that $\textsc{Rel}(L)$ contains any union of products of regular languages (and hence that $\mathsf{REC} \subseteq \textsc{Rel}(L)$) is straightforward. Note that $\textsc{Rel}(L)$ is closed under finite union, and for any two regular languages L_1, L_2 we have that $L_1 \times L_2 \in \textsc{Rel}(L)$ because $(1 \otimes L_1) \cdot (2 \otimes L_2)$ is an L-controlled language.

(I)-[$\subseteq$] On the other hand, let $R \in \textsc{Rel}(L)$, defined by a L-controlled language S. Let S be described by an NFA A_S with statespace Q, initial state q_0 and final states F. Let $L_i^{q,q'}$ be the language consisting of all words $v \in \mathbb{A}^*$ so that there is a partial run of A_S on $i \otimes v$ starting in state q and ending in state q'. Note that $L_i^{q,q'}$ is regular. Hence, $R = [\![S]\!] = \bigcup_{q \in Q, q' \in F} L_1^{q_0,q} \times L_2^{q,q'}$, and thus any relation in $\textsc{Rel}(L)$ is a finite union of products of languages. Therefore, $\textsc{Rel}(L) \subseteq \mathsf{REC}$.

(II)-[$\supseteq$] Let $R \subseteq \mathbb{A}^* \times \mathbb{A}^*$ be a regular relation, represented by an NFA A over the alphabet $(\mathbb{A} \cup \{\bot\})^2$, where (u_1, u_2) is in R if and only if $u_1' \otimes u_2' \in L(A)$, where $u_i' \in (\mathbb{A} \cup \{\bot\})^*$ is the result of padding u_i with a suffix of $\max(|u_1|, |u_2|) - |u_i|$ letters $\bot$. Let Q be the statespace of A. We produce an NFA A' over $\mathbf{2} \times \mathbb{A}$ so that $[\![L(A')]\!] = R$ and $L(A')$ is L-controlled. Let $Q' = \mathbf{2} \times Q$. For any transition $(q, (a,b), q')$ of A, where $a, b \neq \bot$, we have two transitions $((1,q),(1,a),(2,q))$ and $((2,q),(2,b),(1,q'))$ in A'; and for any transition $(q, (\bot,b), q')$ (resp. $(q, (a,\bot), q')$) of A, we have a transition $((1,q),(2,b),(1,q'))$ (resp. $((1,q),(1,a),(1,q'))$) in A'. It follows that a pair (u,v) is accepted by the relation represented by $L(A')$ if, and only if, (u,v) is in $[\![L(A')]\!]$. Further, it is plain that by the behavior of A (*i.e.*, once it reads a letter $(a, \bot)$ for $a \in \mathbb{A}$, it reads only $\bot$ in the second component, and likewise for the first component) A' must be L-controlled.

(II)-[$\subseteq$] Let A be an NFA over $\mathbf{2} \times \mathbb{A}$ so that $L(A)$ is L-controlled, with statespace Q. Note that Q can be partitioned into four sets Q_1, Q_2, Q_1', Q_2', so that the transition relation δ of A is such that

$$\delta \subseteq \bigcup_{i \in \mathbf{2}} Q_i \times \{i\} \times \mathbb{A} \times (Q \setminus Q_i) \cup \bigcup_{i \in \mathbf{2}} Q_i' \times \{i\} \times \mathbb{A} \times Q_i' \qquad (\dagger)$$

that is, all outgoing transitions from Q_i, Q_i' read letters from $\{i\} \times \mathbb{A}$, and there is an alternation between Q_1 and Q_2 until a state from Q_i' is reached, and after that it stays only in Q_i'. We can build an automaton A' over $(\mathbb{A} \cup \{\bot\})^2$ representing the same relation as follows. For every two transitions $(q_1,(1,a),q_2)$ and $(q_2,(2,b),q')$ of A where $q_i \in Q_i$, we have a transition $(q_1,(a,b),q')$ in A'; for every transition $(q_1,(1,a),q_1')$ where $q_1, q_1' \in Q_1 \cup Q_1'$, we have a transition $(q_1,(a,\bot),q_1')$ in A'; and for every transition $(q_2,(2,b),q_2')$ where $q_2, q_2' \in Q_2 \cup Q_2'$, we have a transition $(q_2,(\bot,b),q_2')$ in A'. By (†), it follows that A' represents the relation $[\![L(A)]\!]$.

(III) Note that $L = (1|2)^* = \mathbf{2}^*$ imposes no constraint on $\textsc{Rel}(L)$. That is, $\textsc{Rel}(L)$ is the set of all relations $[\![S]\!]$ so that $S \subseteq \mathbf{2} \times \mathbb{A}$ is regular. Any automaton A over $\mathbf{2} \times \mathbb{A}$ can be alternatively seen as a two-tape automaton A', having one head on each tape, where a transition $(q,(i,a)q,')$ in A corresponds to a transition in A' from q to q' reading letter a from tape i. Conversely, any two-tape automaton A' can be converted into an NFA A over $\mathbf{2} \times \mathbb{A}$. For both directions, the set of relations accepted by A' is $[\![L(A)]\!]$. These are precisely the relations in RAT, and hence the statement follows. □

It is easy to see that $\textsc{Rel}(L)$ is closed under union, alphabetic morphisms, and inverse alphabetic morphisms, and that $L_1 \subseteq L_2$ implies $\textsc{Rel}(L_1) \subseteq \textsc{Rel}(L_2)$.

Remark One may ask why we need to take both S and L regular in the definition (1) of $\textsc{Rel}(L)$. The reason why S needs to be regular is that even with regular L (e.g., 1^*), $\textsc{Rel}(L)$ would otherwise contain non-rational relations (e.g., $\{(a^n b^n, \varepsilon) \mid n \in \mathbb{N}\}$). If, on the other hand, L is not regular, strange things may happen. For instance, it could be that all relations in $\textsc{Rel}(L)$ are finite, although L is infinite. Indeed, take L as the set of all words 1^p for prime p. Note that there is no infinite regular L-controlled language, since it would imply that an infinite number of distinct primes is semi-linear. Thus, all regular L-controlled languages are finite, and $\textsc{Rel}(L)$ is the set of all finite relations on $\mathbb{A}^* \times \{\varepsilon\}$ so that the first component is of prime length.

4 Synchronizations for recognizable, regular, and rational relations

We have seen examples of languages characterizing the classes of recognizable, regular, and rational relations, but those are not unique. There are trivial examples such as $\textsc{Rel}(1^*2^*) = \textsc{Rel}(2^*1^*) = \mathsf{REC}$, and $\textsc{Rel}((12)^*(1^*|2^*)) = \textsc{Rel}((21)^*(1^*|2^*)) = \mathsf{REG}$, but others as well, e.g., the fact that $\textsc{Rel}(1^*2^*1^*2^*)$ is the same class as REC, or $\textsc{Rel}(((12)^*1(12)^*2)^*(1^*|2^*)) = \mathsf{REG}$.

What kind of parameters guarantee that $L \subseteq \mathbf{2}^*$ synchronizes relations in a class $\mathscr{C}$, for the classes we study here? That is, what parameters guarantee that with the synchronization language L, we are guaranteed that the resulting relations are in $\mathscr{C}$?

We now answer this question, but first we need some definitions. Given a word w over some finite alphabet, and a letter a in the alphabet, we define $\#_a(w)$ as the number of occurrences of a in w. Given a word $w \in \mathbf{2}^*$, a position $i \leq |w|$, and $\delta \in \mathbb{N}$, we say i is

- δ-*lagged* if $|\#_1(w[1,i]) - \#_2(w[1,i])| = \delta$;
- $\geq\delta$-*lagged* if $|\#_1(w[1,i]) - \#_2(w[1,i])| \geq \delta$;

– $\leq\delta$-*lagged* if $|\#_1(w[1,i]) - \#_2(w[1,i])| \leq \delta$.

That is, these parameters show by how much the numbers of 1s and 2s in $w \in \mathbf{2}^*$ differ.

A *shift* of w is a position $i \in \{1,\ldots,|w|-1\}$ so that $w[i] \neq w[i+1]$. Two shifts $i < j$ are *consecutive* if there is no shift l so that $i < l < j$.

Let $\mathit{shift}(w)$ be the number of shifts of w, let $\mathit{lag}(w)$ be the maximum lag of a position in w, and let $\mathit{shiftlag}(w)$ be the maximum $n \in \mathbb{N}$ so that w contains n consecutive shifts which are $>n$-lagged. We lift these notions to languages by taking maxima, e.g., $\mathit{shift}(L) = \max_{w\in L} \mathit{shift}(w)$, and likewise for $\mathit{lag}(L)$ and $\mathit{shiftlag}(L)$. If words of arbitrarily large lag (shift, or shiftlag) occur in L, we write $\mathit{shift}(L) = \infty$ (and likewise for the other parameters).

Observe that finite shift and finite lag imply that shiftlag is finite, but the converse is not true: for $L = (12)^*1^*$ we have $\mathit{shiftlag}(L) < \infty$ and yet $\mathit{lag}(L) = \mathit{shift}(L) = \infty$.

It turns out that finiteness of the shiftlag parameter corresponds to synchronizing regular languages, and finiteness of shift corresponds to synchronizing recognizable languages. An arbitrary regular $L \subseteq \mathbf{2}^*$ is guaranteed to synchronize rational languages.

As for the finite lag, it corresponds to a class of languages that is known as well. The class REG^{bld} of *bounded length discrepancy* relations [20,30] is defined as follows. Recall the definition of rational relations using two-tape automata. For a rational relation to be in REG^{bld} it is required that there be $\delta \geq 0$ so that in accepting runs of such automata, the heads for the two tapes are never more than δ positions apart. It also follows from [20,30] that REG^{bld} is the class $\bigcup_{k\in\mathbb{N}_0} \textsc{Rel}(L_k)$, for $L_k = (12)^*(1^k|2^k)$. Note that $\textsc{Rel}(L_0)$ is the class of length preserving relations. A closely related class $R_{\leq} = \{(w_1,w_2) \in \mathbb{A}^* \times \mathbb{A}^* \mid |w_1| \leq |w_2|\}$ [24] can be equally defined by $\textsc{Rel}((12|2)^*)$.

Now we can state the characterization result.

Theorem 1 *Let $L \subseteq \mathbf{2}^*$ be a regular language. Then:*

(I) *L synchronizes recognizable relations iff $\mathit{shift}(L) < \infty$,*
(II) *L synchronizes regular relations iff $\mathit{shiftlag}(L) < \infty$,*
(III) *L synchronizes relations in REG^{bld} iff $\mathit{lag}(L) < \infty$,*
(IV) *L synchronizes rational relations.*

In order to prove Theorem 1, we first need two lemmas.

Lemma 1 *For every $s \geq 1$, we have $\textsc{Rel}((1^*2^*)^s) = \mathsf{REC}$.*

Proof This is a consequence of a synchronization theorem, Theorem 3-(II), which implies that for every $(1^*2^*)^s$-controlled language S there is a (1^*2^*)-controlled language S' so that $[\![S]\!] = [\![S']\!]$. This fact, in conjunction with Proposition 1-(I), shows the statement. □

In the lemma below, we extend the notion of concatenation to classes of relations in the natural way, i.e., element-wise.

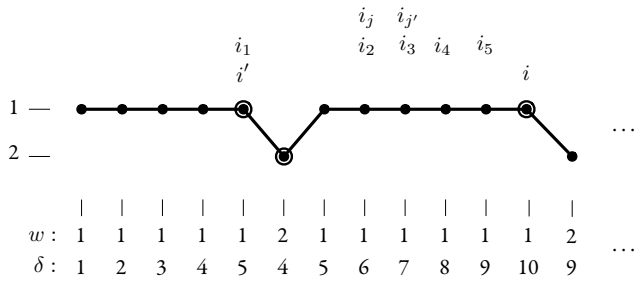


Fig. 1 Example, where w has a prefix 1111121111112 after which it has $n-1$ shifts, $n=4$, $\delta'=9$, and $|Q|=1$. Shift positions are circled.

Lemma 2 *For every $\delta \in \mathbb{N}$, we have* $\textsc{Rel}(L_{\leq \delta\text{-lag}}) \subsetneq \mathsf{REG}$ *and* $\textsc{Rel}(L_{\leq \delta\text{-lag}}) \cdot \mathsf{REC} = \mathsf{REG}$.

Proof Note that any relation $R \in \textsc{Rel}(L_{\leq \delta\text{-lag}})$ only contains pairs (u,v) so that $-\delta \leq |u|-|v| \leq \delta$. Hence the regular relation $\{(u,\varepsilon) \mid u \in \mathbb{A}^*\}$ is not in $\textsc{Rel}(L_{\leq \delta\text{-lag}})$, and thus $\textsc{Rel}(L_{\leq \delta\text{-lag}}) \neq \mathsf{REG}$. On the other hand, we have that any $R \in \textsc{Rel}(L_{\leq \delta\text{-lag}})$ is regular, since it can be recognized by a nondeterministic automaton on two tapes with a look-ahead of δ, which can be simulated in the states of the automaton. Hence, $\textsc{Rel}(L_{\leq \delta\text{-lag}}) \subsetneq \mathsf{REG}$.

Since the concatenation of a regular relation and a recognizable relation is regular [9], we are only left to show $\mathsf{REG} \subseteq \textsc{Rel}(L_{\leq \delta\text{-lag}}) \cdot \mathsf{REC}$. It is easy to see from their automata description that every regular relation $R \in \mathsf{REG}$ can be factored into a finite union of relations $R_1 \cdot R_2$ so that R_1 is $(12)^*$-controlled and R_2 is $(1^*|2^*)$-controlled. Since $(12)^* \in \textsc{Rel}(L_{\leq \delta\text{-lag}})$ for $\delta = 1$, it follows that $\mathsf{REG} \subseteq \textsc{Rel}(L_{\leq 1\text{-lag}}) \cdot \mathsf{REC}$. Note that for every $\delta \leq \delta'$ we have $L_{\leq \delta\text{-lag}} \subseteq L_{\leq \delta'\text{-lag}}$. Then, by the above and monotonicity, $\mathsf{REG} \subseteq \textsc{Rel}(L_{\leq \delta\text{-lag}}) \cdot \mathsf{REC}$ for every $\delta \geq 1$. □

We can now prove the theorem.

Proof (of Theorem 1) (II)**-(if)** Let $n \in \mathbb{N}$ so that $\mathit{shiftlag}(L) < n$. Since L is regular, this implies that there is some δ' where *all* shifts of every $w \in L$ are $\leq\delta'$-lagged for some δ', except perhaps the last $n-1$ shifts.

Claim 1 *There is some δ' so that for all $w \in L$ and for all shifts i of w that are not among the last $n-1$ shifts, we have that they are $\leq\delta'$-lagged.*

Proof Remember that L is regular. Let A_L be an NFA accepting the language L with a state space Q. Let $\delta' = n(|Q|+1)+1$. Suppose, by means of contradiction, that there is $w \in L$ with a shift $i \in \{1,\ldots,|w|\}$ that is $>\delta'$-lagged, so that there are at least $n-1$ shifts to the right of i. Let us assume, without any loss of generality, that $\#_1(w[1,i]) - \#_2(w[1,i]) > \delta'$. Figure 1 contains an example. Since $w \in L$, let $\rho : \{0,\ldots,|w|\} \to Q$ be an accepting run of A_L on w. Let $i' \leq i$ be

- the largest shift $i' < i$ that is $\leq n$-lagged, if there is any, or
- $i' = 1$ otherwise.

Note that in $[i',i]$ there cannot be more than n shifts, since otherwise w would have n consecutive $>n$-lagged shifts contradicting $\mathit{shiftlag}(w) < n$. Also, in $[i',i]$ there must be $k = \delta' - n$ positions $i' \leq i_1 < \cdots < i_k \leq i$ so that for every $j \in \{1,\ldots,k-1\}$

$$\#_1(w[i_j+1, i_{j+1}]) - \#_2(w[i_j+1, i_{j+1}]) = 1, \tag{2}$$

where, by definition of δ', $k = n|Q|+1$ (*cf.* Figure 1). Remember that there are no more than n shifts in $[i',i]$ and i is itself a shift; hence, since $k > n|Q|$, there must be $|Q|+1$ such positions $i_{j_1} < \cdots < i_{j_{|Q|+1}}$ so that there is no shift in $[i_{j_1}, i_{j_{|Q|+1}} - 1]$. Then, there must be two distinct positions $i_j, i_{j'} \in \{i_{j_1}, \ldots, i_{j_{|Q|+1}}\}$, $i_j < i_{j'}$, so that $\rho(i_j) = \rho(i_{j'})$ and there is no shift in $[i_j, i_{j'} - 1]$ (*cf.* Figure 1). We show that we can then "pump" the subword of w inside $[i_j, i_{j'}]$ to obtain a larger word $w' \in L$ that has n shifts $>n$-lagged, that is, where $\mathit{shiftlag}(w') \geq n$. Indeed, for any $l \in \mathbb{N}$, we have that

$$w' = w[1, i_j] \cdot (w[i_j+1, i_{j'}])^l \cdot w[i_{j'}+1, |w|] \ \in \ L.$$

Note that w' has as many shifts as w. Moreover, shift i in w corresponds now to shift $\hat{i} = i + (l-1) \cdot |[i_j+1, i_{j'}]|$ in w', and we have

$$\#_1(w'[1,\hat{i}]) - \#_2(w'[1,\hat{i}]) > (l-1) + \delta'$$

since for every iteration of $w[i_j+1, i_{j'}]$ we add more letters 1 than letters 2, as a consequence of (2).

If we take $l = |w|+1$, we then have that

- w' has at least n shifts in $[\hat{i}, |w'|]$, because w has at least n shifts in $[i, |w|]$ and $w'[\hat{i}, |w'|] = w[i, |w|]$, and
- $\#_1(w'[1,\hat{i}]) - \#_2(w'[1,\hat{i}]) > |w| + \delta'$.

Therefore, the last n shifts of w' are all $>n$-lagged, contradicting $\mathit{shiftlag}(L) < n$. The contradiction comes from assuming that for all δ' there is $w \in L$ and a $>\delta'$-lagged shift i of w that is not among the last $n-1$ shifts. □

As a consequence of the above Claim 1, there must be some δ'' where *all* the positions occurring before the last n shifts are $\leq\delta''$-lagged.

Claim 2 *There is some δ'' so that for all $w \in L$ and all i so that w has at least n shifts in $[i, |w|]$, we have that i is $\leq\delta''$-lagged.*

Proof Let δ' be as in Claim 1. Take any position i so that there are at least n shifts in $[i, |w|]$. Take also the two positions $i_1 \leq i \leq i_2$ so that

- i_2 is a shift,
- i_1 is a shift or $i_1 = 1$, and
- there are no shifts in $[i_1+1, i_2-1]$.

By Claim 1, it follows that both i_1 and i_2 are $\leq\delta'$-lagged. Since $w[i_1+1, i_2]$ is a string of only 1's or only 2's, it cannot be that $|w[i_1+1, i_2]| > 2\delta'$, as otherwise either i_1 or i_2 would not be $\leq\delta'$-lagged. It then follows that i must be $\leq 2\delta'$-lagged. Hence, taking $\delta'' = 2\delta'$, the statement follows. □

A direct consequence of Claim 2 is that there is some δ'' so that

$$L \subseteq L_{\leq \delta''\text{-lag}} \cdot (1^*|2^*)^n \tag{3}$$

because $(1^*|2^*)^n$ contains all words with at most n shifts, and $L_{\leq\delta''\text{-lag}}$ is the (regular) language of all words with $\leq\delta''$-lagged positions. Since $\textsc{Rel}(L') = \mathsf{REC}$ for $L' = (1^*|2^*)^n$ by Lemma 1, we obtain that $\textsc{Rel}(L'') = \mathsf{REG}$ for $L'' = L_{\leq\delta''\text{-lag}} \cdot (1^*|2^*)^n$ by Lemma 2. Finally, as stated in (3), we have that $L \subseteq L''$ where $\textsc{Rel}(L'') = \mathsf{REG}$. Applying monotonicity, we then have $\textsc{Rel}(L) \subseteq \mathsf{REG}$.

(II)**-(only if)** Suppose that $\mathit{shiftlag}(L) = \infty$. Note that this means that for every $s, \delta \in \mathbb{N}$ there is some $w \in L$ that has s consecutive shifts $>\delta$-lagged (because in particular there is some $w \in L$ so that $\mathit{shiftlag}(w) > \max(s,\delta)$). We build an L-controlled relation $S \subseteq (\mathbf{2} \times \mathbb{A})^*$ so that $[\![S]\!] \in \mathsf{RAT} \setminus \mathsf{REG}$.

Let $\mathbb{A}$ be any two-letter alphabet $\{a,b\}$. Let $S \subseteq (\mathbf{2} \times \{a,b\})^*$ consisting of all words $u \otimes v \in (\mathbf{2} \times \{a,b\})^*$ so that $u \in L$, and for every $i \in \{1,\ldots,|v|\}$,

- $v[i] = a$ if i is a shift of u, and
- $v[i] = b$ otherwise.

It is plain that S is a regular L-controlled relation since L is regular, and hence that $[\![S]\!] \in \textsc{Rel}(L)$ is a rational relation. Next we show that $[\![S]\!] \notin \mathsf{REG}$.

Note that every pair in the relation has almost the same number of a's:

$$\text{For every } (u',v') \in [\![S]\!], \quad -1 \leq \#_a(u') - \#_a(v') \leq 1. \tag{$\dagger$}$$

Suppose, by means of contradiction, that $[\![S]\!]$ is regular and therefore, by Proposition 1, $[\![S]\!] \in \textsc{Rel}(L')$ for $L' = (12)^*(1^*|2^*)$. Hence, there must be some L'-controlled relation $S' \subseteq (\mathbf{2} \times \{a,b\})^*$ so that $[\![S']\!] = [\![S]\!]$. Let A_S be an NFA accepting S with statespace Q, and let $A_{S'}$ be an NFA accepting S' with statespace Q'.

Let $s = 2|Q'| + 2$, and let us define the constant $K = s^2|Q|$. We hence define $\delta = 2K$. There must then be some $w = u \otimes v \in S$ with s consecutive shifts that are $>\delta$-lagged. Let $1 \leq i_1 < \cdots < i_s \leq |u|$ be the shifts in question. Let us assume, without any loss of generality, that w is minimal in length and that $\#_1(u[1,i_1]) - \#_2(u[1,i_1]) > \delta$.

Due to minimality of w, it can be shown through a pumping argument, that the lengths of $w[i_1,i_s]$ and of $w[i_s+1,|w|]$ are bounded by a function on s and $|Q|$.

Claim 3 $|w| - i_1 \leq s^2|Q| = K$.

Proof Let $\rho : [0,|w|] \to Q$ be an accepting run of A_S on w. For any $l \in \mathbf{s}$ we have that $u[i_l+1, i_{l+1}]$ is a string of 1's or a string of 2's.

Suppose that $u[i_l+1,i_{l+1}]$ is a string of 2's, and suppose that the string has length greater than $|Q|$. Then there are two distinct elements $i,j \in [i_l+1,i_{l+1}]$ so that $i < j$, $u[i] = u[j] = 2$ and $\rho(i) = \rho(j)$. We then have that $w' = w[1,i] \cdot w[j+1,|w|] \in S$ and it has s consecutive $>\delta$-lagged shifts, because we only removed positions labeled with 2. But this is not possible by minimality of w. Hence, $u[i_l+1,i_{l+1}]$ cannot contain more than $|Q|$ elements 2, and thus

$$\#_2(u[i_1+1,i_s]) \leq (s-1)|Q|. \tag{4}$$

Now suppose that $u[i_l+1, i_{l+1}]$ is a string of 1's, and suppose that the string has length greater than $s|Q|$. Then, there are two distinct elements $i, j \in [i_l+1, i_{l+1}]$ so that $u[i] = u[j] = 1$, $\rho(i) = \rho(j)$ and $i - j \leq |Q|$. We then have that $w' = w[1,i] \cdot w[j+1, |w|] \in S$. Further, w' has s consecutive $>\delta$-lagged shifts, because although we removed some positions marked with 1, we left sufficiently many (at least $(s-1)|Q|$) to make sure that, by (4),

$$\#_1(u[i_1+1, i] \cdot u[j+1, i_{l+1}]) - \#_2(u[i_1+1, i] \cdot u[j+1, i_{l+1}]) \geq 0,$$

and hence that there are still s shifts $>\delta$-lagged in w'. However, this is not possible by minimality of w. Hence, $u[i_l+1, i_{l+1}]$ cannot contain more than $s|Q|$ positions labeled 1, and thus

$$\#_1(u[i_1+1, i_s]) \leq (s-1)s|Q|. \tag{5}$$

Then, by (4) and (5), the length of $u[i_1, i_s]$ is bounded by $(s-1)|Q| + (s-1)s|Q| + 1$.

A simpler consequence of the minimality of w is that

$$|[i_s+1, |w|]| \leq |Q|. \tag{6}$$

Then, summing up, $[i_1, |w|]$ is bounded by

$$\underbrace{|Q|}_{\text{by (6)}} + \underbrace{(s-1)|Q|}_{\text{by (4)}} + \underbrace{(s-1)s|Q|}_{\text{by (5)}} + 1 = s^2|Q| + 1.$$

Thus, $|w| - i_1 \leq s^2|Q| = K$. □

Since $\delta = 2K < \#_1(u[1, i_1]) - \#_2(u[1, i_1])$ and $\#_2(u[i_1+1, |w|]) \leq K$ by Claim 3, we have that

$$\#_1(u) - \#_2(u) > K. \tag{7}$$

Let $w' = u' \otimes v' \in S'$ be the corresponding word in S', so that $[\![w]\!] = [\![w']\!]$. Let $\rho' : [0, |w'|] \to Q'$ be an accepting run of $A_{S'}$ on w'. Note that u' can be factored into $u' = u'_1 \cdot u'_2$ with $u'_1 \in (12)^*$ and $u'_2 \in 1^*$. (The other possibility, $u'_2 \in 2^*$, is only easier.)

Notice that $u[|u| - K, |u|]$ contains s shifts, by Claim 3, and in particular $s/2$ shifts labeled with 1. Therefore, $w[|u| - K, |u|]$ contains at least $s/2$ letters $(1, a)$ by definition of S. By (7), we have that $|u'_2| \geq K$. Thus, u'_2 must contain at least $s/2$ positions labeled with a. Since $s/2 = |Q'| + 1$, there must be two distinct positions $|u'_1| < i < j \leq |w'|$ labeled with a so that $\rho'(i) = \rho'(j)$. Consider then $w'' = w'[1, i] \cdot (w'[i+1, j])^4 \cdot w'[j+1, |w'|]$. Note that $w'' \in S'$. By property (†), we had that $[\![w']\!]$ has the same quantity of a's (plus-minus one) in the first and second components. Therefore, $[\![w'']\!]$ has at least two more a's in its first component than in its second component. Hence, due to property (†), it cannot be that $[\![w'']\!] \in [\![S]\!]$, and thus $[\![S]\!] \neq [\![S']\!]$. The contradiction comes from assuming that there exists an L'-controlled language S' so that $[\![S']\!] = [\![S]\!]$. Hence, $[\![S]\!] \notin \mathsf{REG}$.

(I)-**(if)** Let $\mathit{shift}(L) < n$. Note that $L' = (1^*2^*)^n$ contains all words with less than n shifts. Hence, $L \subseteq L'$. By Lemma 1, $\textsc{Rel}(L') = \mathsf{REC}$, and since $L \subseteq L'$, it follows that $\textsc{Rel}(L) \subseteq \mathsf{REC}$ by monotonicity.

(I)**-(only if)** Suppose $shift(L) = \infty$. We exhibit a relation of $\mathrm{REL}(L)$ which is not in REC. We use the same relation as a previous part of this proof, but we repeat it here for the reader's convenience. Let $\mathbb{A}$ be any two-letter alphabet $\{a,b\}$. Let $S \subseteq (\mathbf{2} \times \{a,b\})^*$ consisting of all words $u \otimes v \in (\mathbf{2} \times \{a,b\})^*$ so that $u \in L$, and for every $i \in \{1,\ldots,|v|\}$,

- $v[i] = a$ if i is a shift of u, and
- $v[i] = b$ otherwise.

It is plain that S is a regular L-controlled relation since L is regular, and hence that $[\![S]\!] \in \mathrm{REL}(L)$ is a rational relation. Next we show that $[\![S]\!] \notin \mathsf{REC}$.

Note that every pair in the relation has almost the same number of a's:

$$\text{For every } (u,v) \in [\![S]\!], \quad -1 \leq \#_a(u) - \#_a(v) \leq 1. \tag{‡}$$

By means of contradiction, suppose that $[\![S]\!] \in \mathsf{REC}$. Then, by Proposition 1-(I), there is a 1^*2^*-controlled language $S' \subseteq (\mathbf{2} \times \{a,b\})^*$ so that $[\![S']\!] = [\![S]\!]$. Let $A_{S'}$ be an NFA recognizing S' with statespace Q'. Let $u \otimes v \in S$ be a word so that u has more than $|Q'|$ shifts, and hence $[\![u \otimes v]\!]$ has more than $|Q'|$ letters a (that is, the the sum of occurrences of a's in both components is greater than $|Q'|$). Since $[\![S']\!] = [\![S]\!]$ there is some $w' = u' \otimes v' \in S'$ so that $[\![u' \otimes v']\!] = [\![u \otimes v]\!]$. Let $\rho' : [0,|w'|] \to Q'$ be an accepting run of $A_{S'}$ on w'. Note that u' has at most one shift. Let i be the only shift of u' (if u' has no shifts the reasoning is only easier). Since v' has more than than $|Q'|$ a's, there must be two positions j_1, j_2 of w' so that $\rho'(j_1) = \rho'(j_2)$, $v'[j_1] = v'[j_2] = a$ and either $1 \leq j_1 < j_2 \leq i$ or $i < j_1 < j_2 \leq |w'|$ (as a consequence of S' being 1^*2^*-controlled). Note then that $w'[1,j_1] \cdot (w'[j_1+1,j_2])^n \cdot w'[j_2+1,|w'|] \in S'$ for every $n \in \mathbb{N}$. Take $n = 4$, and let $w'' = w'[1,j_1] \cdot (w'[j_1+1,j_2])^4 \cdot w'[j_2+1,|w'|] \in S'$. Note that $[\![w'']\!]$ has at least two more a's in one component than in the other, because w' has at most a difference of one a between its components, due to (‡). Hence, w'' is in contradiction with (‡), and it cannot be that $[\![S']\!] = [\![S]\!]$. Therefore, $[\![S]\!] \notin \mathsf{REC}$ and thus $\mathrm{REL}(L) \not\subseteq \mathsf{REC}$.

(III) This is direct by definition of REG^{bld}.

(IV) This is direct from definition of $\mathrm{REL}(L)$ and Proposition 1-(III). □

We conclude the section with a couple of examples of applications of the main result. First, we show that $\mathrm{REL}((112)^*) \not\subseteq \mathsf{REG}$. Indeed, note that for every s, δ, the word $w = (112)^{\delta+s}$ is in $(112)^*$ and the last s shifts of w are $\geq\delta$-lagged. Hence, there must be some L-controlled regular language $S \subseteq (\mathbf{2} \times \mathbb{A})^*$ so that $[\![S]\!]$ is *not* a regular relation.

As another example, we get more ways of synchronizing regular relations: given $L_1 = (1^k \cdot 2^k)^*$, $L_2 = (1^* \cdot 2^*)^k$ for some fixed k, we have $\mathrm{REL}(L_i) \subseteq \mathsf{REG}$ (in fact, $\mathrm{REL}(L_2) \subseteq \mathsf{REC}$).

Finally, we consider the (r/s)-synchronized relations [30, p.660] studied in [13]. This class can be defined as $\mathrm{REL}(L_{r/s})$, where

$$L_{r/s} = (1^r 2^s)^* \Big(\bigcup_{r'<r} (1^{r'} 2^*) \mid \bigcup_{s'<s} (1^* 2^{s'}) \Big). \tag{8}$$

It is easy to see that $shiftlag(L_{r/s}) = \infty$ whenever $r \neq s$, and hence that (r/s)-synchronized relations (with $r \neq s$) are not in REG.

4.1 Automata theoretic characterizations

We characterized classes of relations via conditions imposed on their synchronization languages: finite shift, lag, or shiftlag. Now we show that these conditions themselves can be characterized using automata, or more precisely, the underlying labeled graphs of automata. It turns out that the structure of the cycles provides the desired characterizations.

Since in this section we deal with synchronization languages, we consider automata over the alphabet $\{1,2\}$. For a given NFA A, we consider the *transition graph* G_A of A as the usual representation of the transition relation, where G_A is a directed graph where states are vertices and edges are labeled by transitions. A *path* is a finite sequence of edges of G_A so that the arriving vertex of each edge is equal to the departing vertex of the next one. A *cycle* is a path whose first and last vertices are equal. A *simple cycle* is a cycle whose only repetition of vertex is the first and last ones. Given a cycle C of G_A, we define $\#_a(C)$ as the number of edges in C labeled with transitions reading letter a. In a *heterogeneous* cycle C we have $\#_1(C) > 0$ and $\#_2(C) > 0$; otherwise a cycle is *homogeneous*. A cycle C is *balanced* if $\#_1(C) = \#_2(C)$, otherwise it is *unbalanced* (these definitions are closely related to the notions of balanced/unbalanced oriented cycles in digraphs, cf. [22]). Note that all balanced cycles are also heterogeneous.

Recall that the trim automaton is the result of removing all states which are not reachable from the initial state, and all states from which no final state is reachable.

Theorem 2 *For any trim NFA A over the alphabet* **2***, and its transition graph G_A,*

(I) *$shiftlag(L(A)) = \infty$ iff*
- *G_A contains a heterogeneous unbalanced cycle, or*
- *G_A contains a path from a homogeneous to a heterogeneous cycle,*

(II) *$shift(L(A)) = \infty$ iff G_A has a heterogeneous cycle,*

(III) *$lag(L(A)) = \infty$ iff G_A has an unbalanced cycle.*

Proof Let Q be the statespace of A. Given $w \in L(A)$ and an accepting run $\rho : [0,|w|] \to Q$ of A on w, the *path P on G_A induced by w,ρ* is defined as the sequence of edges $e_1 \cdots e_{|w|}$ of G_A, so that e_i is the edge between $\rho(i-1)$ and $\rho(i)$ labeled with $(\rho(i-1), w[i], \rho(i))$.

(I)**-(if)** Let $n \in \mathbb{N}$. We show that assuming one of the two properties is met, there is some $w \in L(A)$ with $shiftlag(w) \geq n$.

If G_A has a heterogeneous cycle C_{het} with $\#_1(C_{het}) \neq \#_2(C_{het})$, one can iterate this cycle to obtain a word w with $shiftlag(w) > n$. In other words, suppose that $w \in L(A)$ with an accepting run $\rho : [0,|w|] \to Q$ so that the path P induced by w,ρ contains a heterogeneous unbalanced cycle C_{het} between the positions $i \leq j$ where we assume, without any loss of generality, $\#_1(C_{het}) > \#_2(C_{het}) > 0$. Since this means that $\rho(i-1) = \rho(j)$, we have that

$$w_m = w[1,i-1] \cdot (w[i,j])^m \cdot w[j+1,|w|] \in L(A)$$

for every $m \in \mathbb{N}$, and $\#_1(w[i,j]) > \#_2(w[i,j]) > 0$ because $\#_1(C_{het}) > \#_2(C_{het}) > 0$. Hence, if we take $m = |w| + 2n$, it is easy to see that w_m has n consecutive shifts that are $>n$-lagged. Thus, $shiftlag(w_m) \geq n$.

If, on the other hand, there is a path from a homogeneous cycle C_{hom} to a heterogeneous cycle C_{het} in G_A, then we show that we can iterate both cycles enough times to obtain a word $w \in L(A)$ so that $shiftlag(w) > n$. Suppose $w \in L(A)$ with an accepting run $\rho : [0,|w|] \to Q$, so that the path P induced by w, ρ contains both cycles, where C_{hom} occurs before C_{het}. That is, there are $0 < i < j \leq i' < j' \leq |w|$ so that

- $\rho(i) = \rho(j)$ and C_{hom} is the cycle induced by $w[i,j]$, $\rho|_{[i-1,j]}$, and
- $\rho(i') = \rho(j')$ and C_{het} is the cycle induced by $w[i',j']$, $\rho|_{[i'-1,j']}$.

Note that for any $m, l \in \mathbb{N}$ we have

$$w_{m,l} = \underbrace{w[1,i] \cdot (w[i+1,j])^m \cdot w[j+1,i']}_{u_m} \cdot \underbrace{(w[i'+1,j])^l \cdot w[j',|w|]}_{v_l} \in L(A).$$

If we take $m = (n+2)|w|$ and $l = n$, we obtain that

- $|\#_1(u_m) - \#_2(u_m)| > (n+1)|w|$,
- $|v_l| \leq n|w|$, and
- $shift(v_l) > n$.

Therefore, $w_{m,l} = u_m \cdot v_l$ is so that $shiftlag(w_{m,l}) \geq n$.

Thus, if any of the conditions in (I) is met, we must have that $shiftlag(L(A)) = \infty$.

(I)**-(only if)** Suppose now that $shiftlag(L(A)) = \infty$. We choose $n = 2|Q| + 1$, and show that any accepting run of A on $w \in L(A)$ so that $shiftlag(w) \geq n$ must induce a path P containing either

(i) a heterogeneous cycle C_{het} with $\#_1(C_{het}) \neq \#_2(C_{het})$, or
(ii) a homogeneous cycle C_{hom} and a heterogeneous cycle C_{het}, so that C_{hom} occurs *before* C_{het} in P.

Note that once this is verified, the statement follows.

Let $\rho : [0,|w|] \to Q$ be an accepting run of A on w so that $shiftlag(w) > n$. Consider the path P on G_A induced by ρ, w. By definition of $shiftlag(w) > n$, there must be n consecutive $>n$-lagged shifts $1 \leq a_1 < a_2 < \cdots < a_n \leq |w|$ in w. Without any loss of generality, assume that

$$\#_1(w[1,a_1]) - \#_2(w[1,a_1]) > n, \tag{$\dagger$}$$

and that for every odd index i, $w[a_i] = 1$ and for every even index i, $w[a_i] = 2$. Since $n > 2|Q|$, it follows that there must be $a_i < a_j < a_l$ with $\rho(a_i) = \rho(a_j) = \rho(a_l)$, and thus there must be a heterogeneous cycle inside P (the one defined between positions $i+1$ and l). Further, by ($\dagger$), there are positions $0 \leq b_1 < \cdots < b_n \leq a_1$ so that $\#_1(w[b_i + 1, b_{i+1}]) - \#_2(w[b_i+1, b_{i+1}]) = 1$ for every $i \in \mathbf{n} - \mathbf{1}$. Since $n > |Q|$, there must be two $b_i < b_j$ so that $\rho(b_i) = \rho(b_j)$. Hence the cycle C of P induced by $w[b_i+1, b_j], \rho|_{[b_i,b_j]}$ necessarily verifies

$$\#_1(C) > \#_2(C). \tag{$\ddagger$}$$

Now there are two possibilities.

- If $\#_2(C) > 0$ then C is heterogeneous and with $\#_1(C) \neq \#_2(C)$ by ($\ddagger$), verifying condition (i).

– The other possibility is that C is homogeneous. Since there is a path from C to a heterogeneous cycle C_{het}, the condition (ii) is met.

(II)**-(if)** Suppose that G_A contains a heterogeneous cycle C_{het}. Then, there must be some word $w \in L(A)$ with an accepting run $\rho : [0,|w|] \to Q$ so that the path P induced by w, ρ contains C_{het} between positions $i \leq j$ of P. Therefore $\rho(i-1) = \rho(j)$, and $w_n = w[1,i-1] \cdot (w[i,j])^n \cdot w[j+1,|w|] \in L(A)$ for any $n \in \mathbb{N}$. Note that as a consequence of C_{het} being heterogeneous, $w[i,j]$ contains at least one letter 1 and one letter 2. Thus, w_n contains at least n shifts, and therefore $\mathit{shift}(L(A)) = \infty$.

(II)**-(only if)** Suppose that $\mathit{shift}(L(A)) = \infty$, that is, for every $n \in \mathbb{N}$ there is a word $w \in L(A)$ so that $\mathit{shift}(w) > n$. Take $n = 2|Q|$, and let $w \in L(A)$ so that $\mathit{shift}(w) > n$. There must be more than $|Q|$ shifts in w with the same letter $i \in \mathbf{2}$. Without any loss of generality, suppose there are shifts $1 \leq i_1 < \cdots < i_{|Q|+1} \leq |w|$ so that $w[i_j] = 1$ for all $j \in \{1,\ldots,|Q|+1\}$. Then there must be two $i_{j_1} < i_{j_2}$ so that $\rho(i_{j_1}) = \rho(i_{j_2})$. Hence, the word $w[i_{j_1}+1, i_{j_2}]$ has length ≥ 2, and contains at least one letter 1 (the last letter) and at least one letter 2 (the first letter, as otherwise i_{j_1} would not be a shift with letter 1). It then follows that the path on G_A induced by $w[i_{j_1}+1, i_{j_2}], \rho|_{[i_{j_1}, i_{j_2}]}$ is indeed a heterogeneous cycle.

(III) This is shown in [30, Lemma 6.7, p. 603]. □

Corollary 1 *Checking whether* $\textsc{Rel}(L(A)) \subseteq \mathsf{REG}$, $\textsc{Rel}(L(A)) \subseteq \mathsf{REC}$ *or whether* $\textsc{Rel}(L(A)) \subseteq \mathsf{REG}_2^{bld}$ *can be done in polynomial time in the size of* A.

Note that Corollary 1 does *not* mean that it is decidable whether a relation $R \in \mathsf{RAT}$ is in REG (in fact, this problem is undecidable [9, Theorem 8.4-(vi)]). What one can check is whether a synchronized relation has a "safe" control, in the sense that it synchronizes regular relations. Hence, for any relation R controlled by $L(A)$, if $\textsc{Rel}(L(A)) \subseteq \mathsf{REG}$ then $R \in \mathsf{REG}$, but the opposite does not necessarily hold. For example, if we take $L' = (1|2)^*$, we have that $\textsc{Rel}(L') \not\subseteq \mathsf{REG}$ but the universal relation $\mathbb{A}^* \times \mathbb{A}^*$ is obviously in REG.

5 Resynchronizing relations

We saw that different languages in $\mathbf{2}^*$ can generate the same class relations, and yet for the commonly used classes, we have synchronization languages that somehow look canonical: for instance, $(12)^*(1^*|2^*)$ for REG. Thus, we now address the question whether we can resynchronize relations using those canonical synchronization languages, and if so, can we do it effectively?

To pose this formally, suppose two different languages $S, S' \subseteq (\mathbf{2} \times \mathbb{A})^*$ controlled by $L, L' \subseteq \mathbf{2}^*$ respectively represent the same relation, i.e., $[\![S]\!] = [\![S']\!]$. Then we say that S is an *L-resynchronization* of S'. Given a class $\mathscr{C}$ of regular languages over $\mathbf{2}$, we say that $L_0 \in \mathscr{C}$ is a *canonical representative* of $\mathscr{C}$ if for every $L \in \mathscr{C}$ and every L-controlled language S there exists an L_0-resynchronization of S. In other words, for every $L \in \mathscr{C}$ and $R \in \textsc{Rel}(L)$, there is an L_0-controlled $S' \in (\mathbf{2} \times \mathbb{A})^*$ so that

$[\![S']\!] = R$. If, in addition, there is a recursive procedure that constructs such an L_0-resynchronization of S, then we say that L_0 is an *effective canonical representative* of $\mathscr{C}$.

Let RL_{all} be the class of all regular languages over $\mathbf{2}$, and let $\mathsf{RL}^{fin}_{param}$ stand for the class of regular languages $L \subseteq \mathbf{2}^*$ with finite parameter *param*, where *param* is lag, or shift, or shiftlag. We also let $\mathsf{RL}_{lag\leq\delta}$ denote the class of all regular languages $L \subseteq \mathbf{2}^*$ with $lag(L) \leq \delta$.

Example 1 Take, for example, $L_1 = (1122)^*1^*2^*$ *and* $L_2 = (12)^*(1^*|2^*)$*, and a* L_1*-controlled relation* S_1*. Since* $shiftlag(L_1) < \infty$*,* $[\![S_1]\!] \in \mathsf{REG}$ *by Theorem 1. Further, since by Proposition 1-*(II) $\textsc{Rel}(L_2) = \mathsf{REG}$*, there must be some* L_2*-controlled relation* S_2 *so that* $[\![S_2]\!] = [\![S_1]\!]$*. In other words* S_2 *is the* L_2*-resynchronization of* S_1*. Since* $\textsc{Rel}(L_2) = \mathsf{REG}$ *in fact* L_2 *is a canonical representative of* $\mathsf{RL}^{fin}_{shiftlag}$*.*

Theorem 3 (Resynchronization theorem)

(I) $(12)^*(1^*|2^*)$ *is an effective canonical representative of* $\mathsf{RL}^{fin}_{shiftlag}$*;*
(II) 1^*2^* *is an effective canonical representative of* $\mathsf{RL}^{fin}_{shift}$*;*
(III) *there is no canonical representative of* RL^{fin}_{lag}*;*
(IV) $(12)^*(1^{\leq\delta}|2^{\leq\delta})$ *is an effective canonical representative of* $\mathsf{RL}_{lag\leq\delta}$*;*
(V) $\mathbf{2}^*$ *is an effective canonical representative of* RL_{all}*.*

If the relations are given as NFA, the synchronization procedures are in exponential time.

For the proof of the Theorem above wee need to introduce some standard notions. The *shuffle* $sh(U,V)$ of two languages $U,V \subseteq \mathbb{A}^*$ is the set of all words $u_1 \cdot v_1 \cdots u_k \cdot v_k$ so that $u_1 \cdots u_k \in U$, $v_1 \cdots v_k \in V$. The *strongly connected components* (henceforth SCC) of G_A are its maximal strongly connected subgraphs. An SCC is *heterogeneous* if it contains a heterogeneous cycle; an SCC is *homogeneous* if it contains a cycle and all cycles it contains are homogeneous; otherwise, an SCC without cycles (that is, a single vertex) is an *edgeless* SCC. The *condensation* of G_A (written $con(G_A)$) is the directed acyclic graph (henceforth DAG) induced by the SCC's of G_A. This is the DAG where nodes are SCC's of G_A and there is an edge labeled $(q,(i,a),q')$ from vertex v to (a different) vertex v' if q belongs to the SCC of v, q' belongs to the SCC of v' and there is an edge labeled $(q,(i,a),q')$ from q to q' in G_A (in other words, $(q,(i,a),q')$ is a transition of A).

For the proof of Theorem 3 we use the following lemma.

Lemma 3 (Bounds for *shiftlag*, *shift*, *lag*) *Given an NFA* A *over the alphabet* $\mathbf{2}$ *with statespace* Q*,*

(I) *if* $shiftlag(L(A)) < \infty$*, then* $shiftlag(L(A)) \leq |Q|$*;*
(II) *if* $shift(L(A)) < \infty$*, then* $shift(L(A)) \leq |Q|$*;*
(III) *if* $lag(L(A)) < \infty$*, then* $lag(L(A)) \leq |Q|$*.*

Proof Assume without any loss of generality that A is trim. Given a set of vertices S, let $A|_S$ be the NFA whose set of initial states is S, and its transition relation corresponds to the subgraph of G_A induced by all the vertices reachable from S.

(I) By Theorem 2-(I) every SCC S of G_A is so that

(a) S is edgeless, or
(b) S is homogeneous and all SCC's S' reachable from S are homogeneous or edgeless, or
(c) S is heterogeneous, and all simple cycles C in S are so that $\#_1(C) = \#_2(C)$.

Let us analyze each case separately. Let $S_1, \ldots, S_n$ be the set of SCC's reachable from S (excluding S).

(a) Then, $\mathit{shiftlag}(L(A|_S)) \leq 1 + \mathit{shiftlag}(L(A|_{S_1 \cup \cdots \cup S_n}))$.
(b) Then, any word w accepted by $A|_S$ is contained in $(1^*2^*)^{\leq l}$, where l is the number of SCC's in $G_{A|_S}$. Therefore, $\mathit{shift}(L(A|_S)) \leq l$ and therefore $\mathit{shiftlag}(L(A|_S)) \leq l$.
(c) We then have that any word w in $L(A|_S)$ is of the form $w = u \cdot v$ where $u \in \bigcup_{i \leq |S|} (sh(1^i, 2^i))^*$ and $v \in L(A|_{S_1 \cup \cdots \cup S_n})$. Recall that $sh(1^i, 2^i)$ represents the set of shuffles of 1^i and 2^i (*i.e.*, all the words over $\mathbf{2}$ having exactly i 1's and i 2's). Note that
   – there are no positions $>|S|$-lagged in u, and
   – position $|u|$ is 0-lagged in w.
   Thus, $\mathit{shiftlag}(L(A|_S)) \leq \max(|S|, \mathit{shiftlag}(L(A|_{S_1 \cup \cdots \cup S_n})))$.

Combining (a), (b) and (c), and by the fact that $\mathit{con}(G_A)$ is a DAG, we obtain that $\mathit{shiftlag}(L(A)) \leq |Q|$.

(II) By Theorem 2-(II) there are no heterogeneous cycles in G_A, and every SCC S of G_A is hence homogeneous or edgeless. Shifts can hence only occur in transitions *between* SCC's in G_A (*i.e.*, transitions that involving states from two different SCC's). Since the condensation of G_A is a DAG, there are not more than $|Q|$ different SCC that an accepting run of A for a word can go through. Hence, $\mathit{shift}(L(A)) < n$, where n is the number of SCC's of A minus one. Since $n \leq |Q|$, the statement follows.

(III) By Theorem 2-(III) all cycles C in G_A are so that $\#_1(C) = \#_2(C)$. By means of contradiction, suppose that there is some $w \in L(A)$ with $\mathit{lag}(w) > |Q|$, and an accepting run $\rho : [0, |w|] \to Q$ of A on w, where Q is the statespace of A. Further, suppose that w is minimal in length; that is, any word w' shorter than w is so that $\mathit{lag}(w') \leq |Q|$. Since $|w| > |Q|$, let $0 \leq i < j \leq |w|$ be any two indices so that $\rho(i) = \rho(j)$. Note that the path induced by $w[i, j-1], \rho|_{[i,j]}$ is a cycle C, and by hypothesis it must be so that $\#_1(C) = \#_2(C)$. Therefore, $\#_1(w[i, j-1]) = \#_2(w[i, j-1])$. Consider then the word $w' = w[1, i-1] \cdot w[j, |w|]$. We have that $w' \in L(A)$ and that $\mathit{lag}(w') = \mathit{lag}(w)$ because we removed a subword with equal number of letters 1 and 2. This is an absurd by minimality of w. Thus, it cannot be that $\mathit{lag}(L(A)) > |Q|$ and the statement follows. □

We are now in conditions to prove Theorem 3.

Proof (of Theorem 3) We start by showing (II) and (IV) because we use these items in the proof of (I).

(II) Let $S \subseteq (\mathbf{2} \times \mathbb{A})^*$ be an L-controlled regular language with $\mathit{shift}(L) < \infty$. We assume, without any loss of generality, that $L = \{u \mid u \otimes v \in S\}$. Let A be an NFA recognizing S with statespace Q, initial state q_0 and set of final states Q_F. Note that, since S is L-controlled, one can build in linear time an automaton A_L recognizing L,

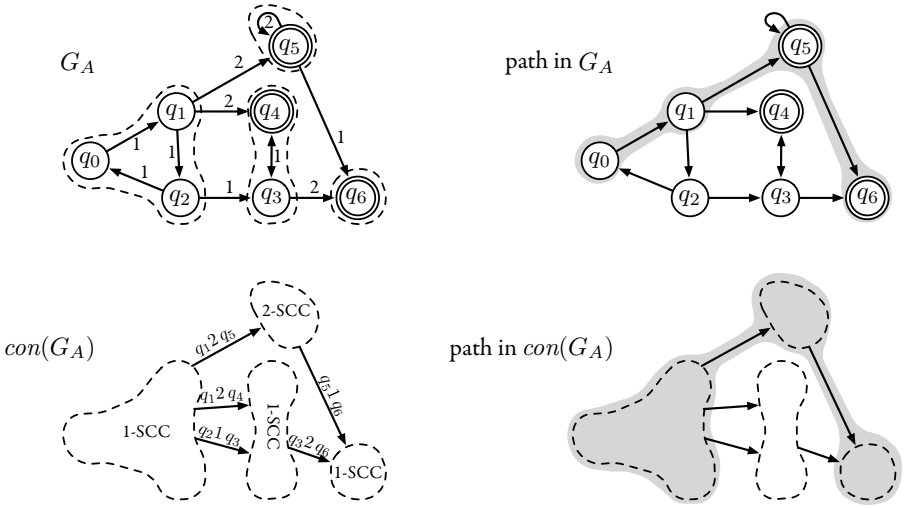


Fig. 2 Example of path in G_A and corresponding path in $con(G_A)$. For simplicity, we assume that the alphabet is singleton $\mathbb{A} = \{a\}$, and we therefore omit 'a' in the transitions.

having the same statespace Q (the transformation consists in replacing every transition $(q,(i,a),q')$ with (q,i,q')). Hence, by Lemma 3-(II), $shift(L) \leq |Q|$.

Let us call 1-edge (resp. 2-edge) an edge of G_A labeled with a transition reading the letter 1 (resp. 2) in its first component. Note that every SCC of G_A is homogeneous or edgeless by Theorem 2-(II). Hence, if a SCC has only 1-edges, we call it a *1-SCC*. Otherwise (if it has only 2-edges), we call it a *2-SCC*. For the purpose of this proof, it is indifferent whether we categorize edgeless SCC's as 1-SCC's or 2-SCC's, but just to fix nomenclature, let us call them 1-SCC's. Hence, every SCC in G_A is a 1-SCC or a 2-SCC.

Note that any path on G_A induces a (possibly empty) path on $con(G_A)$ (*cf.* Figure 2). By acyclicity there are at most exponentially many paths in $con(G_A)$.

For any (possibly empty) path P in $con(G_A)$ and final state $q \in Q_F$, let $S_{P,q}$ be the set of all words $w \in S$ with an accepting run of A ending in q and inducing the path P in $con(G_A)$. Hence,

$$S = \bigcup \{ S_{P,q} \mid P \text{ is a path of } con(G_A) \text{ and } q \in Q_F \}.$$

We conclude the proof by showing that for every path P in $con(G_A)$ and $q \in Q_F$ we can build, in polynomial time, a (1^*2^*)-controlled automaton $A_{P,q}$ so that $[\![L(A_{P,q})]\!] = [\![S_{P,q}]\!]$.

Claim 4 *For every path P in $con(G_A)$ and $q \in Q_F$, an automaton $A_{P,q}$ so that*

- $[\![L(A_{P,q})]\!] = [\![S_{P,q}]\!]$ *and*
- $L(A_{P,q})$ *is* (1^*2^*)*-controlled*

is computable in polynomial time in $|A|$.

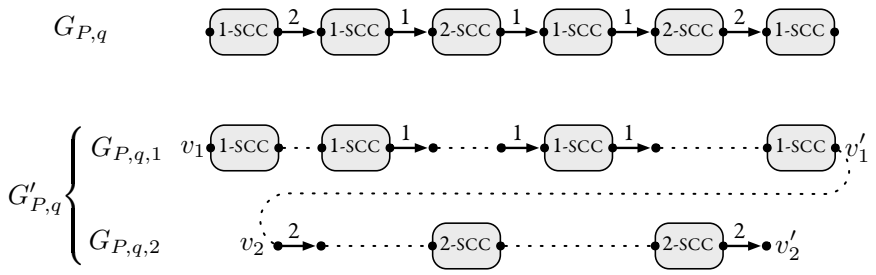


Fig. 3 Example of construction of $G'_{P,q}$ from $G_{P,q}$. The SCC are abstracted as grey boxes, labeled "1-SCC" or "2-SCC" depending on the sort of SCC they are. Edges are also labeled depending on whether they are 1-edges or 2-edges. Dotted lines are used to identify two vertices as being the same.

Proof We can assume, without any loss of generality, that P is not empty, and contains

- a vertex corresponding to a 1-SCC, or a 1-edge, and
- a vertex corresponding to a 2-SCC, or a 2-edge,

since otherwise $S_{P,q}$ would be trivially (1^*2^*)-controlled and an automaton can be easily built in polynomial time in $|A|$.

Let $G_{P,q}$ be the transition graph of the NFA recognizing $S_{P,q}$, which is the result of removing from A

- all the states from SCC's that are not in P and its associated transitions, and
- all transitions $(q,(i,a),q')$ not appearing in P, so that q,q' do not belong to the same SCC.

Note that $con(G_{P,q})$ is a directed chain, where there is at most one edge traveling between two vertices from different SCC's; the shape of $G_{P,q}$ is depicted in the top picture of Figure 3 (the path in this Figure is unrelated to the path of the previous Figures). Let $states_{q_0,q}(P)$ be the sequence of states appearing in P, prefixed with q_0 and suffixed with q; that is, if

$$P = (v_1,(q_1,(i_1,a_1),q'_1),v_2),\ldots,(v_n,(q_n,(i_n,a_n),q'_n),v_{n+1}),$$

then $states_{q_0,q}(P) = q_0,q_1,q'_1,\ldots,q_n,q'_n,q$. The idea is that $states_{q_0,q}(P)$ represents the sequence of states that any accepting run of the automaton recognizing $S_{P,q}$ has to go through (there could, however, be some repetitions of states if the incoming and outgoing state of a SCC are the same in P). For example, in the path P depicted in Figure 2, we have $states_{q_0,q_6} = q_0,q_1,q_5,q_5,q_6,q_6$, note that it includes, for every SCC, the incoming and outgoing states (q_0,q_1 for the first, q_5,q_5 for the second, and q_6,q_6 for the third SCC). In the top picture of Figure 3, the vertices in $states_{q_0,q}(P)$ are depicted as bullets. Consider the graph $G_{P,q,1}$ as the result of

1. removing all 2-edges from $G_{P,q}$,
2. removing all vertices without incoming or outgoing edges that remain, and
3. associating vertices to make it a connected graph, so that the relative appearance of the 1-SCC's and 1-edges given by P is preserved.

This construction is shown in Figure 3. Let v_1, v'_1 be the first and last vertices in the construction of $G_{P,q}$ (*cf.* Figure 3). That is, v_1 corresponds to the first vertex in $states_{q_0,q}(P)$ that has an outgoing 1-edge in $G_{P,q}$, and v'_1 corresponds to the last vertex in $states_{q_0,q}(P)$ that has an incoming 1-edge in $G_{P,q}$.

We define $G_{P,q,2}$ and v_2, v'_2 analogously to $G_{P,q,2}$ and v_1, v'_1, but removing 1-edges instead (*cf.* Figure 3). Now, let $G'_{P,q}$ be the transition graph resulting from composing $G_{P,q,1}$ with $G_{P,q,2}$ by associating v'_1 with v_2 (*cf.* Figure 3). Let us define the automaton $A_{P,q}$ as having the transition relation defined by $G'_{P,q}$, where the initial state is v_1 and the set of final states is $\{v'_2\}$. We then have that $A_{P,q}$ is (1^*2^*)-controlled and $[\![L(A_{P,q})]\!] = [\![S_{P,q}]\!]$. □

The statement follows directly from the previous claim, defining

$$S' = \bigcup_{P,q} L(A_{P,q})$$

for every path P of $con(G_A)$ and $q \in Q_F$, and defining $A_{S'}$ as the union of all automata $A_{P,q}$'s. Then, S' is a (1^*2^*)-resynchronization of S, and $A_{S'}$ can be built in exponential time.

We now show another claim concerning (1^*2^*)-controlled languages, that will be useful in the proof of (I).

Claim 5 *For any (1^*2^*)-controlled automaton A one can build, in polynomial time, $(12)^*$-controlled automata $A_1^{head}, \ldots, A_t^{head}$ as well as $(1^*|2^*)$-controlled automata $A_1^{tail}, \ldots, A_t^{tail}$ so that*

$$[\![L(A)]\!] = \bigcup_{i \in \mathbf{t}} [\![L(A_i^{head}) \cdot L(A_i^{tail})]\!].$$

Proof In the scope of this proof, let Q be the statespace of A, with initial state q_{init} and set of final states Q_F. Let us define the automaton A' over the same alphabet as A with the statespace $Q \times Q \times \mathbf{2}$, with a transition

- $((q_1,q_2,1),(1,a),(q'_1,q_2,2))$ if $(q_1,(1,a),q'_1)$ is a transition of A and $q_2 \in Q$, and
- $((q_1,q_2,2),(2,a),(q_1,q'_2,1))$ if $(q_2,(2,a),q'_2)$ is a transition of A and $q_1 \in Q$.

Note that for every $q_1, q_1, q'_1, q'_2 \in Q$, $A'[(q_1,q_2,1),(q'_1,q'_2,1)]$ is $(12)^*$-controlled. Also, note that for every $q'_1, q_2, q'_2 \in Q$ and $q_f \in Q_F$,

$$\underbrace{L(A'[(q_{init},q_2,1)(q_2,q'_2,1)])}_{L_1^{head}} \cdot \underbrace{(L(A[q'_2,q_f]) \cap (\{2\} \times \mathbb{A})^*)}_{L_1^{tail}}$$

and

$$\underbrace{L(A'[(q_{init},q_2,1)(q'_1,q_f,1)])}_{L_2^{head}} \cdot \underbrace{(L(A[q'_1,q_2]) \cap (\{1\} \times \mathbb{A})^*)}_{L_2^{tail}}$$

are $(12)^*(1^*|2^*)$-controlled, and that automata recognizing L_i^{head}, L_i^{tail} can be obtained in polynomial time.

From the definition of L_i^{head} and L_i^{tail} and the previous observation, we show that for any word $w \in L_i^{head} \cdot L_i^{tail}$ there is some $w' \in L$ so that $[\![w]\!] = [\![w']\!]$, and vice versa.

Observe that for any $q_1, q_1', q_2, q_2' \in Q$, $w \in L(A'[(q_1,q_2,1)(q_1',q_2',1)])$ if, and only if, $w_{odd} \in L(A[q_1,q_1']) \cap (\{1\} \times \mathbb{A})^*$ and $w_{even} \in A[q_2,q_2'] \cap (\{2\} \times \mathbb{A})^*$, where w_{odd} (resp. w_{even}) is the subword of w of odd (resp. even) positions.

From any accepting run of $A'[(q_{init},q_2,1),(q_2,q_2',1)]$ on w_1 and an accepting run of $A[q_2',q_f]$ on $w_2 \in (\{2\} \times \mathbb{A})^*$ one can build an accepting run of A on $(w_1)_{odd} \cdot (w_1)_{even} \cdot w_2$, where $[\![(w_1)_{odd} \cdot (w_1)_{even} \cdot w_2]\!] = [\![w_1 \cdot w_2]\!]$. Similarly, from an accepting run of

$$A'[(q_{init},q_2,1),(q_1',q_f,1)]$$

on w_1 and an accepting run of $w_2 \in (\{1\} \times \mathbb{A})^*$ on $A[q_1',q_2]$ one can build an accepting run of A on $(w_1)_{odd} \cdot w_2 \cdot (w_1)_{even}$, where $[\![(w_1)_{odd} \cdot w_2 \cdot (w_1)_{even}]\!] = [\![w_1 \cdot w_2]\!]$. Indeed, note that in both cases, $(w_1)_{odd} = (w_1)_{\{1\}\times\mathbb{A}}$ and $(w_1)_{even} = (w_1)_{\{2\}\times\mathbb{A}}$.

Conversely, for every accepting run of A on w, let w' be the interleaving of $w_{\{1\}\times\mathbb{A}}[1,m]$ and $w_{\{2\}\times\mathbb{A}}[1,m]$, where $m = \min(|w_{\{1\}\times\mathbb{A}}|, |w_{\{2\}\times\mathbb{A}}|)$ (more formally, it is the word $w' \in sh(w_{\{1\}\times\mathbb{A}}[1,m], w_{\{2\}\times\mathbb{A}}[1,m])$ so that $w' \in (((\{1\} \times \mathbb{A}) \cdot (\{2\} \times \mathbb{A}))^*)$. If $|w_{\{1\}\times\mathbb{A}}| \leq |w_{\{2\}\times\mathbb{A}}|$ then for some $q_2, q_2' \in Q$ and $q_f \in Q_F$ there is an accepting run of $A'[(q_{init},q_2,1),(q_2,q_2',1)]$ on w', and accepting run of $A[q_2',q_f]$ on $w[2m+1,|w|] \in (\{2\} \times \mathbb{A})^*$. Similarly, if $|w_{\{1\}\times\mathbb{A}}| > |w_{\{2\}\times\mathbb{A}}|$ then for some $q_2, q_1' \in Q$ and $q_f \in Q_F$ there is an accepting run of $A'[(q_{init},q_2,1)(q_1',q_f,1)]$ on w', and accepting run of $A[q_1',q_2]$ on $w[2m+1,|w|] \in (\{1\} \times \mathbb{A})^*$. In both cases, observe that $[\![w' \cdot w[2m+1,|w|]]\!] = [\![w]\!]$.

Summing up, for every pair $(u,v) \in \mathbb{A}^* \times \mathbb{A}^*$, there is a word $w \in L_i^{head} \cdot L_i^{tail}$ with $[\![w]\!] = (u,v)$ for some $i \in \mathbf{2}$ if, and only if, there is some $w' \in L(A)$ with $[\![w']\!] = (u,v)$.

Hence, defining L' as the union of all the above $L_1^{head} \cdot L_1^{tail}$ and $L_2^{head} \cdot L_2^{tail}$ languages for all possible $q_2, q_2', q_1' \in Q$ and $q_f \in Q_F$, it follows that $[\![L(A)]\!] = [\![L']\!]$. Since every L_i^{head} is $(12)^*$-controlled and every L_i^{tail} is $(1^*|2^*)$-controlled, and since automata for these languages can be built in polynomial time, the statement follows. □

(IV) This follows from [30, Proposition 6.9, pp. 604–605]. Although in the cited work the complexity is not given, it follows from the proof that it can be built in exponential time. In fact, note that it suffices to build an automaton whose every state has a *buffer* of $lag(L)$ letters.

(I) Let S be an L-controlled regular language $S \subseteq (\mathbf{2} \times \mathbb{A})^*$ with $shiftlag(L) < \infty$. Let A be an NFA recognizing S with statespace Q, initial state q_0 and set of final states Q_F.

Note that since the projection of S onto $\mathbf{2}$ is inside L, we can apply Theorem 2-(I) to A, obtaining that there are no paths from homogeneous SCC's to heterogeneous SCC's in G_A (and there are no heterogeneous cycles C with $\#_1(C) \neq \#_2(C)$). Let Q_{hom} be the set of all vertices of G_A that are reachable from a vertex of a homogeneous SCC. Note that Q_{hom} includes all vertices in homogeneous SCC's, plus some vertices from edgeless SCC's. Also, note that the subgraph of G_A induced by Q_{hom} has no heterogeneous cycles. Let $Q_{het} = Q \setminus Q_{hom}$. Hence, Q_{het} includes all vertices in heterogeneous SCC's and some vertices in edgeless SCC's. Also, by the property

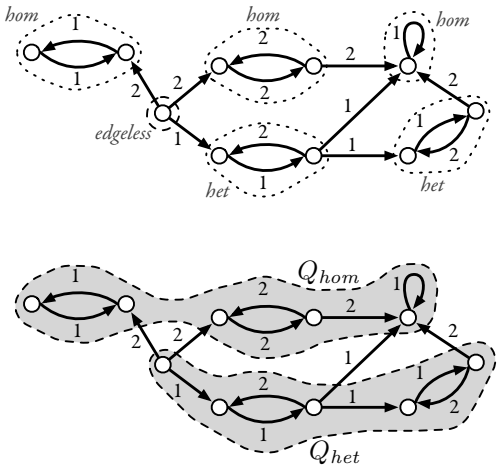


Fig. 4 Example of G_A with the subgraphs induced by Q_{hom} and Q_{het}. For simplicity we assume that $\mathbb{A} = \{a\}$ and we hence omit the letter a when depicting edges labeled by (i,a).

before, the subgraph of G_A induced by Q_{het} is connected. Figure 4 contains an example. Further, any path $\hat{P}$ in G_A is of the form (1) $\hat{P} \cdot (q,\tau,q') \cdot \hat{P}'$, (2) $\hat{P}$, or (3) $\hat{P}'$, where

- $\hat{P}$ is a (possibly empty) path of the subgraph of G_A induced by Q_{het},
- $\hat{P}'$ is a (possibly empty) path of the subgraph of G_A induced by Q_{hom},
- $q \in Q_{het}$, $q' \in Q_{hom}$, and τ is a transition of A.

Let A^{het} be A restricted to Q_{het}, and let A^{hom} be A restricted to Q_{hom}. For every pair of states $q_{het} \in Q_{het}$ and $q_{hom} \in Q_{hom}$, let $L_{q_{het},q_{hom}}$ be the union of all

$$L(A^{het}[q_0,q_{het}]) \cdot \{(i,a)\} \cdot L(A^{hom}[q_{hom},q_f])$$

for every $q_f \in Q_F$ and $(i,a) \in \mathbf{2} \times \mathbb{A}$ so that $(q_{het},(i,a),q_{hom})$ is a transition of A (if there is no such (i,a), let $L_{q_{het},q_{hom}} = \emptyset$). We remind the reader that $A^{het}[q,q']$ (resp. $A^{hom}[q,q']$) where q or q' are not in Q_{het} (resp. Q_{hom}) denotes the automaton accepting the empty language. Let $L_{hom} = \bigcup_{q_f \in Q_F} L(A^{hom}[q_0,q_f])$ and $L_{het} = \bigcup_{q_f \in Q_F} L(A^{het}[q_0,q_f])$. It follows that

$$S = L_{hom} \cup L_{het} \cup \bigcup_{q_{het} \in Q_{het}, q_{hom} \in Q_{hom}} L_{q_{het},q_{hom}} .$$

We show that we can build, in exponential time, a $(12)^*(1^*|2^*)$-controlled automaton for each of these languages. Since the case of $L_{q_{het},q_{hom}}$ is more general than L_{hom} and L_{het}, we will only prove this case.

Note that by definition of A^{het} and A^{hom}, and since G_A has no unbalanced heterogeneous cycles, for every $q_{het} \in Q_{het}, q_{hom} \in Q_{hom}, q_f \in Q_F$ we have that both $lag(L(A^{het}[q_0,q_{het}])) < \infty$ and $shift(L(A^{hom}[q_{hom},q_f])) < \infty$. Hence, by Lemma 3,

- $lag(L(A^{het}[q_0,q_{het}])) \leq n$,

– $\mathit{shift}(L(A^{hom}[q_{hom}, q_f])) \leq n$,

for $n = |A|$.

By the already shown item (II), let $A^{hom}_{q_{hom},q_f}$ be a (1^*2^*)-controlled automaton so that $[\![L(A^{hom}[q_0, q_{hom}])]\!] = [\![L(A^{hom}_{q_0,q_{hom}})]\!]$. By item (IV), let $A^{het}_{q_0,q_{het}}$ be a $(12)^*(1^{\leq n}|2^{\leq n})$-controlled automaton so that $[\![L(A^{het}[q_0, q_{het}])]\!] = [\![L(A^{het}_{q_0,q_{het}})]\!]$. These automata can be built in exponential time.

We finally show that a $(12)^*(1^*|2^*)$-controlled automaton for $L_{q_{het},q_{hom}}$ can be built from $A^{het}_{q_0,q_{het}}$ and all the $A^{hom}_{q_{hom},q_f}$'s for all $q_f \in Q_F$ in polynomial time, and thus the statement follows.

Claim 6 *A $(12)^*(1^*|2^*)$-controlled automaton for $L_{q_{het},q_{hom}}$ can be built from $A^{het}_{q_0,q_{het}}$ and all the $A^{hom}_{q_{hom},q_f}$'s in polynomial time.*

Proof From $A^{hom}_{q_{hom},q_f}$ (which is 1^*2^*-controlled) one can easily build 1^*-controlled automata $B^{hom\text{-}1^*}_1, \ldots, B^{hom\text{-}1^*}_t$ and 2^*-controlled automata $B^{hom\text{-}2^*}_1, \ldots, B^{hom\text{-}2^*}_t$ in polynomial time so that

$$L(A^{hom}_{q_{hom},q_f}) = \bigcup_{i \in \mathbf{t}} \big(L(B^{hom\text{-}1^*}_i) \cdot L(B^{hom\text{-}2^*}_i)\big).$$

Also, it is easy to see that from $A^{het}_{q_0,q_{het}}$ (which is $(12)^*(1^{\leq n}|2^{\leq n})$-controlled) one can build $(12)^*$-controlled automata $A^{het\text{-}(12)^*}_1, \ldots, A^{het\text{-}(12)^*}_s$, $1^{\leq n}$-controlled automata $A^{het\text{-}1^*}_1, \ldots, A^{het\text{-}1^*}_s$ and $2^{\leq n}$-controlled automata $A^{het\text{-}2^*}_1, \ldots, A^{het\text{-}2^*}_s$ in polynomial time, so that

$$L(A^{het}_{q_0,q_{het}}) = \bigcup_{i \in \mathbf{s}} \big(L(A^{het\text{-}(12)^*}_i) \cdot L(A^{het\text{-}1^*}_i) \cdot L(A^{het\text{-}2^*}_i)\big).$$

We then have that

$$[\![L_{q_{het},q_{hom}}]\!] = \bigcup_{\ell \in \mathbf{2}} [\![\bigcup_{i \in \mathbf{t}, j \in \mathbf{s}} L(A^{het\text{-}(12)^*}_j) \cdot L(A^{het\text{-}1^*}_j) \cdot L(A^{het\text{-}2^*}_j) \cdot L_{\ell,q_{het},q_{hom}} \cdot \cdot L(B^{hom\text{-}1^*}_i) \cdot L(B^{hom\text{-}2^*}_i)]\!]$$

where $L_{\ell,q_{het},q_{hom}} = \{(\ell, a) \mid (q_{het}, (\ell, a), q_{hom}) \text{ in } A\}$. Note that, for $\ell = 1$ we have

$$\begin{aligned}
&[\![\bigcup_{i \in \mathbf{t}, j \in \mathbf{s}} L(A^{het\text{-}(12)^*}_j) \cdot L(A^{het\text{-}1^*}_j) \cdot L(A^{het\text{-}2^*}_j) \cdot L_{1,q_{het},q_{hom}} \cdot \\
&\qquad\qquad \cdot L(B^{hom\text{-}1^*}_i) \cdot L(B^{hom\text{-}2^*}_i)]\!] \\
&= [\![\bigcup_{i \in \mathbf{t}, j \in \mathbf{s}} L(A^{het\text{-}(12)^*}_j) \cdot L'_{1,i,j}]\!], \text{ where} \\
L'_{1,i,j} &= L(A^{het\text{-}1^*}_j) \cdot L_{1,q_{het},q_{hom}} \cdot L(B^{hom\text{-}1^*}_i) \cdot L(A^{het\text{-}2^*}_j) \cdot L(B^{hom\text{-}2^*}_i).
\end{aligned}$$

Since $L'_{1,i,j}$ is 1^*2^*-controlled, we can apply the previous Claim 5 on $L'_{1,i,j}$ obtaining $(12)^*$-controlled automata $A^{head}_1, \ldots, A^{head}_m$ and $(1^*|2^*)$-controlled automata $A^{tail}_1, \ldots, A^{tail}_m$ so that

$$[\![L'_{1,i,j}]\!] = \bigcup_{k \in \mathbf{m}} [\![L(A^{head}_k) \cdot L(A^{tail}_k)]\!].$$

in polynomial time. Defining $L''_{1,i,j} = \bigcup_{k\in\mathbf{m}} L(A_k^{head}) \cdot L(A_k^{tail})$, we obtain that $L''_{1,i,j}$ is $(12)^*(1^*|2^*)$-controlled, and an automaton for $L''_{1,i,j}$ can be computed in polynomial time. Thus,

$$\begin{aligned}
&[\![\bigcup_{i\in\mathbf{t},j\in\mathbf{s}} L(A_j^{\textit{het-}(12)^*}) \cdot L(A_j^{\textit{het-}1^*}) \cdot L(A_j^{\textit{het-}2^*}) \cdot L_{1,q_{het},q_{hom}} \cdot \\
&\qquad\qquad \cdot L(B_i^{\textit{hom-}1^*}) \cdot L(B_i^{\textit{hom-}2^*})]\!] \\
&= [\![\bigcup_{i\in\mathbf{t},j\in\mathbf{s}} L(A_j^{\textit{het-}(12)^*}) \cdot L''_{1,i,j}]\!],
\end{aligned}$$

where note that $L'''_1 = \bigcup_{i\in\mathbf{t},j\in\mathbf{s}} L(A_j^{\textit{het-}(12)^*}) \cdot L''_{1,i,j}$ is $(12)^*(1^*|2^*)$-controlled, and an automaton for L'''_1 can be built in polynomial time.

For $\ell = 2$ we apply a similar reasoning,

$$\begin{aligned}
&[\![\bigcup_{i\in\mathbf{t},j\in\mathbf{s}} L(A_j^{\textit{het-}(12)^*}) \cdot L(A_j^{\textit{het-}1^*}) \cdot L(A_j^{\textit{het-}2^*}) \cdot L_{2,q_{het},q_{hom}} \cdot \\
&\qquad\qquad \cdot L(B_i^{\textit{hom-}1^*}) \cdot L(B_i^{\textit{hom-}2^*})]\!] \\
&= [\![\bigcup_{i\in\mathbf{t},j\in\mathbf{s}} L(A_j^{\textit{het-}(12)^*}) \cdot L'_{2,i,j}]\!]
\end{aligned}$$

this time taking

$$L'_{2,i,j} = L(A_j^{\textit{het-}1^*}) \cdot L(B_i^{\textit{hom-}1^*}) \cdot L_{2,q_{het},q_{hom}} \cdot L(A_j^{\textit{het-}2^*}) \cdot L(B_i^{\textit{hom-}2^*}).$$

and obtaining, through Claim 5, a $(12)^*(1^*|2^*)$-controlled language L'''_2 so that

$$\begin{aligned}
&[\![\bigcup_{i\in\mathbf{t},j\in\mathbf{s}} L(A_j^{\textit{het-}(12)^*}) \cdot L(A_j^{\textit{het-}1^*}) \cdot L(A_j^{\textit{het-}2^*}) \cdot L_{2,q_{het},q_{hom}} \cdot \\
&\qquad\qquad \cdot L(B_i^{\textit{hom-}1^*}) \cdot L(B_i^{\textit{hom-}2^*})]\!] = [\![L'''_2]\!].
\end{aligned}$$

Hence,

$$[\![L_{q_{het},q_{hom}}]\!] = \bigcup_{\ell\in\mathbf{2}} [\![L'''_\ell]\!]$$

and an automaton recognizing $\bigcup_{\ell\in\mathbf{2}} L'''_\ell$ can be built in polynomial time. □

(III) For any $L \in \mathsf{RL}^{\textit{fin}}_{\textit{lag}}$ with $\textit{lag}(L) = k$, consider the singleton language $L' = \{1^{k+1}\} \in \mathsf{RL}^{\textit{fin}}_{\textit{lag}}$. Note that any nonempty L'-controlled relation cannot have a L-resynchronization. Thus, there cannot be a canonical representative of $\mathsf{RL}^{\textit{fin}}_{\textit{lag}}$.

Note that, however, the class $\mathsf{RL}^{\textit{fin}}_{\textit{lag}} \cup \{(12)^*(1^*|2^*)\}$ has $(12)^*(1^*|2^*)$ as an effective canonical representative by item (I).

(V) This is straightforward since $\mathbf{2}^*$ contains all languages over $\mathbf{2}$, and therefore all relations are $\mathbf{2}^*$-controlled. □

6 Closure via Parikh images

It is well known that the class REG is effectively closed under Boolean operations. Although RAT is a natural generalization of REG, it is not a Boolean algebra (let alone an effective one), not being closed under intersection or complement [9]. Even testing whether a rational relation is regular, or whether it has an empty intersection with a regular relation is undecidable [9]. Since regular relations are characterized via finite shiftlag, it is natural to ask whether infinite shiftlag somehow describes "dangerous" classes of relations. That is, does this mean for example that for any $L \subseteq \mathbf{2}^*$ with $\mathit{shiftlag}(L) = \infty$ the intersection problem is undecidable for $\textsc{Rel}(L)$? The answer to this question is negative: take for instance $L = (122)^*$ with $\mathit{shiftlag}(L) = \infty$. However, it is not hard to see that $\textsc{Rel}(L)$ is effectively closed under intersection.

This raises the question of whether there are classes $\mathscr{C} \subseteq \mathsf{RAT}$ that are natural, expressive, and well-behaved, that is, so that

- $\mathsf{REC} \subsetneq \mathscr{C}$,
- $\mathscr{C}$ is effectively closed under union, intersection and complementation (i.e., is an *effective Boolean algebra*); and
- $\mathscr{C}$ corresponds to a natural condition on the language.

Note that REG is one such example. Here we address the question from our perspective in terms of control languages. The idea is to show sufficient conditions of synchronization languages L so that $\textsc{Rel}(L)$ is effectively closed under intersection, or an effective boolean algebra. We state those in terms of Parikh images of languages.

Recall that the *Parikh image* of a word $w \in \mathbf{k}^*$, written $\Pi(w)$, is the vector of $\mathbb{N}_0^k$ whose ith component contains $\#_i(w)$, the number of occurrences of i in w. The Parikh image of a language L is $\Pi(L) = \{\Pi(w) \mid w \in L\}$. It is well known that for regular and context-free languages L, sets $\Pi(L)$ are exactly the semi-linear sets in $\mathbb{N}_0^k$, see [29].

A language $L \subseteq \mathbf{k}^*$ is

- *Parikh-injective* if the function $\Pi : L \to \mathbb{N}_0^k$ is injective, and
- *Parikh-surjective* if the function $\Pi : L \to \mathbb{N}_0^k$ is surjective.

Example 2

- $(12)^*(1^*|2^*)$ *and* 1^*2^* *are Parikh-injective, while* $(1|2)^*$ *is not.*
- *It can easily be shown that* $L = w_1^* \cdot w_2^* \cdots w_\ell^* \subseteq \mathbf{k}^*$ *is Parikh-injective if* $\ell \leq k$ *and* $\{\Pi(w_1), \ldots, \Pi(w_\ell)\}$ *generate a linear subspace of* $(\mathbb{N}_0)^k$ *of dimension* ℓ*. For example,* $(122)^*(112)^*$ *is Parikh-injective.*
- $(12)^*(1^*|2^*)$*,* 1^*2^**, and* $(1|2)^*$ *are Parikh-surjective, but* $(122)^*(112)^*$ *is not Parikh-surjective.*
- *It is easy to see that* $L_{r/s}$ *as defined in* (8) *is Parikh-injective and Parikh-surjective for any choice of* r, s*. For example, if* $r = 1$*,* $s = 2$*, then* $L_{r/s} = (122)^*(2^*|1^*2|1^*)$*, which is Parikh-injective and Parikh-surjective, since , as shown in Figure 5, every element of* $(\mathbb{N}_0)^2$ *is covered, and there is only one way to reach any element of* $(\mathbb{N}_0)^2$*.*

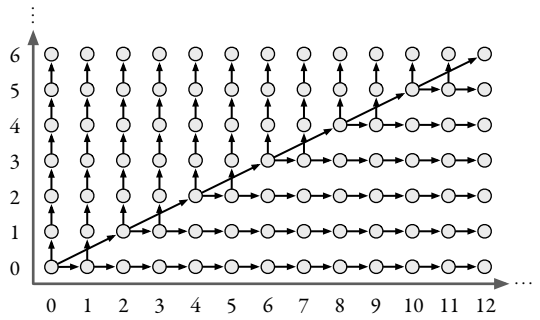


Fig. 5 Example of a Parikh-injective and Parikh-surjective language.

We now analyze the (effective) closure of classes $\mathrm{REL}(L)$ under Boolean operations. It turns out that closure under union is free, but for closure under intersection and complement, the newly introduced criteria serve as sufficient conditions.

Theorem 4 *Let $L \subseteq \mathbf{2}^*$ be a regular language. Then*

(I) $\mathrm{REL}(L)$ *is effectively closed under union, alphabetic morphisms, and inverse alphabetic morphisms;*
(II) *if L is Parikh-injective, then* $\mathrm{REL}(L)$ *is effectively closed under intersection;*
(III) *if L is both Parikh-injective and Parikh-surjective, then* $\mathrm{REL}(L)$ *is effectively closed under complement.*

Proof (I) Let $S_1, S_2 \subseteq (\mathbf{2} \times \mathbb{A})^*$ be two L-controlled relations. It is immediate that the language $S_\cup = S_1 \cup S_2$ is L-controlled. We then have that $[\![S_1]\!] \cup [\![S_2]\!] = [\![S_\cup]\!]$. The fact that it is closed under (inverse) alphabetic morphisms is immediate from the definition of $\mathrm{REL}(L)$.

(II) Let $S_1, S_2 \subseteq (\mathbf{2} \times \mathbb{A})^*$ be two L-controlled relations. It is immediate that the language $S_\cap = S_1 \cap S_2$ is L-controlled and $[\![S_\cap]\!] \subseteq [\![S_1]\!] \cap [\![S_2]\!]$. We show that $[\![S_1]\!] \cap [\![S_2]\!] \subseteq [\![S_\cap]\!]$. Suppose that $(u,v) \in [\![S_1]\!] \cap [\![S_2]\!]$. Then, there must be $w_1 \in S_1$ and $w_2 \in S_2$ so that $[\![w_1]\!] = [\![w_2]\!] = (u,v)$. Note that the projection onto the first component of w_1 must be equal to the projection onto the first component of w_2 since L is Parikh-injective. Then, we must have that $w_1 = w_2$ and thus $(u,v) \in [\![S_\cap]\!]$.

(III) Let $S \subseteq (\mathbf{2} \times \mathbb{A})^*$ be an L-controlled relation. Let S^c be the complement of S and let $[\![S]\!]^c$ be the complement of $[\![S]\!]$. We show the following,

$$[\![S]\!]^c \;=\; [\![S^c \cap (L \otimes \mathbb{A}^*)]\!],$$

where $L \otimes \mathbb{A}^*$ denotes the set of all words $u \otimes v$ where $|u| = |v|$, $u \in L$ and $v \in \mathbb{A}^*$.

[$\subseteq$] Suppose $(u,v) \notin [\![S]\!]$. We show that there must be some $w \in S^c \cap (L \otimes \mathbb{A}^*)$ so that $(u,v) = [\![w]\!]$. Since L is Parikh-injective, there must be at most one word $w' \in L$ so that $\Pi(w') = (|u|,|v|)$. Since the Parikh image of L is the whole universe $\mathbb{N}_0^2$, there must be at least one word $w' \in L$ so that $\Pi(w') = (|u|,|v|)$. Hence, there is *exactly one* word $w' \in L$ so that $\Pi(w') = (|u|,|v|)$. Let $w = u' \otimes v' \in (\mathbf{2} \times \mathbb{A})^*$ be the only word so that $u' = w'$ and $[\![w]\!] = (u,v)$. Note that $w \notin S$ (otherwise (u,v) would be

in $[\![S]\!]$) and that its projection onto the first component (*i.e.*, w') is in L. Therefore, $w \in S^c \cap (L \otimes \mathbb{A}^*)$.

$[\supseteq]$ Suppose now that $w \in S^c \cap (L \otimes \mathbb{A}^*)$. We show that $[\![w]\!] \notin [\![S]\!]$. Assume, by means of contradiction, that $[\![w]\!] \in [\![S]\!]$. Then, there is some $w' \in S$ so that $[\![w']\!] = [\![w]\!]$. It cannot be that $w' = w$, as it would be in contradiction with $w \in S^c \cap (L \otimes \mathbb{A}^*)$. Since L is Parikh-injective, and both w and w' are L-controlled, it must be that $w = w'$, as otherwise $[\![w']\!] \neq [\![w]\!]$, which is not possible as already observed. The contradiction comes from assuming that $[\![w]\!] \in [\![S]\!]$. Thus, $[\![w]\!] \notin [\![S]\!]$ and $[\![S]\!]^c \supseteq [\![S^c \cap (L \otimes \mathbb{A}^*)]\!]$. □

Corollary 2 *If $L \subseteq \mathbf{2}^*$ is Parikh-injective and Parikh-surjective, then* $\textsc{Rel}(L)$ *is an effective boolean algebra, closed under alphabetic morphisms and inverse alphabetic morphisms.*

Observe that in this context, REG and REC are simply two examples of the (infinitely) many such well-behaved classes.

Example 3

- REC *and* REG *are effective boolean algebras since they correspond to* $\textsc{Rel}(1^*2^*)$ *and* $\textsc{Rel}((12)^*(1^*|2^*))$*, where* 1^*2^**,* $(12)^*(1^*|2^*)$ *are Parikh-injective and Parikh-surjective.*
- $\textsc{Rel}((122)^*(112)^*)$ *is effectively closed under intersection.*
- *It was shown in [13] that the class of (r/s)-synchronized relations is an effective Boolean algebra. Our results provide an alternative proof, since $L_{r/s}$ is Parikh-injective and Parikh-surjective.*

Observation Theorem 4 cannot be generalized to finite unions of Parikh-injective languages, since for instance $\textsc{Rel}(L)$ for $L = ((12)^*1^*)|(1^*(12)^*)$ is not closed under intersection. In fact, its intersection problem is undecidable. This follows from the fact that $\textsc{Rel}(L)$ contains the suffix relation and all regular relations (where the first component is longer than the second). By [5, Theorem V.1], this problem is undecidable.

7 Future work

We presented a new way of looking at relations on words, and this new perspective opens up several directions. An obvious one is to extend results to k-ary relations, for $k > 2$. We know that exact analogs of Proposition 1, Theorem 1, and Theorem 2 continue to hold. Other directions are as follows.

Containment One of the classical language-theoretic problems is language containment, which in this case is formulated as follows: given $L_1, L_2 \subseteq \mathbf{2}^*$, is $\textsc{Rel}(L_1) \subseteq \textsc{Rel}(L_2)$? We would like to understand decidability/complexity issues for containment.

Two-wayness Another way to extend the framework is by using two-way automata. Then, instead of having a synchronization language over the alphabet $\{1,2\}$, we have

it over the alphabet $\{1,2,\bar{1},\bar{2}\}$, where $i,\bar{i}$ are interpreted as moving the ith head to the right or to the left respectively. For example, in this context, $\mathrm{REL}(1^*\bar{2}^*)$ contains the relation $\{(w,w^r) \in \mathbb{A}^* \times \mathbb{A}^* \mid w \text{ is the reverse of } w^r\}$.

Model theory approach One way of capturing regularity is by standard model-theoretic techniques: one can find (so-called universal automatic) first-order structures over Σ^* so that definable relations are regular (or nice subclasses of regular) relations. For instance, using the binary predicates for prefix and equal length, and unary predicates for each letter $a \in \Sigma$ checking if the last letter of a word is a, we get one such structure [10]. By virtue of translation into automata, such structures are decidable, and their model-theoretic properties have been investigated [8]. We would like to extend this investigation and connect definability in infinite structures over Σ^* with different classes of relations of the form $\mathrm{REL}(L)$.

Context-free relations Another natural extension is to look for other classes of relations, say analogs of context-free languages. In particular, one can look at a generalization of rational relations, the *pushdown relations* of [16], which are those recognized by multi-tape automata with a stack or, equivalently, by a context-free grammar. In view of our approach here, this is not the only way of generalizing REC, REG, and RAT with pushdown automata. Indeed, in our framework, the simplest way to generalize these relations with the power of context-free languages, is to consider —instead of $\mathrm{REL}(L)$— the class $\mathrm{REL}^{\mathrm{CF}}(L)$ as the set of all relations $[\![S]\!]$, for any L-controlled context-free language $S \subseteq (\mathbf{2} \times \mathbb{A})^*$.

In this framework we can show that $\mathrm{REL}^{\mathrm{CF}}(\mathbf{2}^*)$, the context-free analog of rational relations, is the class of pushdown relations of [16]. Analogs of recognizable and regular relations are $\mathrm{REL}^{\mathrm{CF}}((12)^*(1^*|2^*))$ and $\mathrm{REL}^{\mathrm{CF}}(1^*2^*)$. Those properly contain REG and REC, respectively, are contained in $\mathrm{REL}^{\mathrm{CF}}(\mathbf{2}^*)$, but are incomparable with each other as well as with RAT. We want to conduct a further study of those, perhaps extending to visibly pushdown languages [2] due to their appeal in both verification and modeling XML.

We also would like to use the structural approach to look for better behaved classes of relational word transducers for verification purposes, and for classes of relations that can be effectively used in querying graph data. Finally, we would like to use it to identify classes of well behaved relations over *data words* [11] and study logics over them, extending the approach of [5,6] with data.

Acknowledgements Work partially supported by EPSRC grants G049165 and J015377.

References

1. Abdulla, J., Jonnson, B., Nilsson, M., Saksena, M.: A survey of regular model checking. In: International Conference on Concurrency Theory (CONCUR'03), pp. 35–48 (2003)
2. Alur, R., Madhusudan, P.: Visibly pushdown languages. In: Symposium on Theory of Computing (STOC'04), pp. 202–211 (2004)
3. Angles, R., Gutiérrez, C.: Survey of graph database models. ACM Comput. Surv. **40**(1) (2008)
4. Anyanwu, K., Sheth, A.: ρ-queries: enabling querying for semantic associations on the semantic web. In: 12th International World Wide Web Conference (WWW'03), pp. 690–699 (2003)

5. Barceló, P., Figueira, D., Libkin, L.: Graph logics with rational relations and the generalized intersection problem. In: Annual IEEE Symposium on Logic in Computer Science (LICS'12), pp. 115–124. IEEE Computer Society Press (2012). DOI 10.1109/LICS.2012.23
6. Barceló, P., Libkin, L., Lin, A.W., Wood, P.T.: Expressive languages for path queries over graph-structured data. ACM Trans. Database Syst. **37**(4), 31 (2012)
7. Ben-Ari, M.: Principles of the Spin model checker. Springer (2008)
8. Benedikt, M., Libkin, L., Schwentick, T., Segoufin, L.: Definable relations and first-order query languages over strings. J. ACM **50**(5), 694–751 (2003)
9. Berstel, J.: Transductions and Context-Free Languages. B. G. Teubner (1979)
10. Blumensath, A., Grädel, E.: Automatic structures. In: LICS, pp. 51–62 (2000)
11. Bojańczyk, M.: Automata for data words and data trees. In: 21st International Conference on Rewriting Techniques and Applications (RTA), *Leibniz International Proceedings in Informatics (LIPIcs)*, vol. 6, pp. 1–4 (2010)
12. Bouajjani, A., Jonsson, B., Nilsson, M., Touili, T.: Regular model checking. In: International Conference on Computer Aided Verification (CAV'00), pp. 403–418. Springer-Verlag, London, UK (2000)
13. Carton, O.: The growth ratio of synchronous rational relations is unique. Theor. Comput. Sci. **376**(1-2), 52–59 (2007)
14. Carton, O., Choffrut, C., Grigorieff, S.: Decision problems among the main subfamilies of rational relations. RAIRO Theoretical Informatics and Applications **40**(2), 255–275 (2006)
15. Choffrut, C.: Relations over words and logic: A chronology. Bulletin of the EATCS **89**, 159–163 (2006)
16. Choffrut, C., Culik II, K.: Properties of finite and pushdown transducers. SIAM J. Comput. **12**(2), 300–315 (1983)
17. Elgot, C.C., Mezei, J.E.: On relations defined by generalized finite automata. IBM J. Res. Dev. **9**(1), 47–68 (1965). DOI 10.1147/rd.91.0047. URL http://dx.doi.org/10.1147/rd.91.0047
18. Fagin, R., Kimelfeld, B., Reiss, F., Vansummeren, S.: A formal framework for information extraction. In: ACM Symposium on Principles of Database Systems (PODS'13) (2013, to appear.)
19. Figueira, D., Libkin, L.: Synchronizing relations on words. In: International Symposium on Theoretical Aspects of Computer Science (STACS'14), *Leibniz International Proceedings in Informatics (LIPIcs)*, vol. 25, pp. 93–104. Leibniz-Zentrum für Informatik, Lyon, France (2014). DOI 10.4230/LIPIcs.STACS.2014.518
20. Frougny, C., Sakarovitch, J.: Synchronized rational relations of finite and infinite words. Theor. Comput. Sci. **108**(1), 45–82 (1993)
21. Harju, T., Mateescu, A., Salomaa, A.: Shuffle on trajectories: The schützenberger product and related operations. In: MFCS, pp. 503–511 (1998)
22. Hell, P., Nešetřil, J.: Graphs and Homomorphisms. Oxford University Press (2004)
23. Jonsson, B., Nilsson, M.: Transitive closures of regular relations for verifying infinite-state systems. In: International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'00), pp. 220–234. Springer-Verlag (2000)
24. Leguy, J.: Transductions rationnelles décroissantes. ITA **15**(2), 141–148 (1981)
25. McMillan, K.L.: Symbolic Model Checking. Kluwer Academic Publishers (1993)
26. Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., Alon, U.: Network motifs: simple building blocks of complex networks. Science **298**(5594), 824–827 (2002)
27. Neven, F.: Automata, Logic, and XML. In: EACSL Annual Conference on Computer Science Logic (CSL'02), pp. 2–26 (2002)
28. Nivat, M.: Transduction des langages de Chomsky. Ann. Inst. Fourier **18**, 339–455 (1968)
29. Parikh, R.: On context-free languages. Journal of the ACM **13**(4), 570–581 (1966)
30. Sakarovitch, J.: Elements of Automata Theory. Cambridge University Press (2009)
31. Schwentick, T.: Automata for XML - a survey. J. Comput. Syst. Sci. **73**(3), 289–315 (2007)
32. To, A.W., Libkin, L.: Algorithmic metatheorems for decidable LTL model checking over infinite systems. In: International Conference on Foundations of Software Science and Computational Structures (FOSSACS'10), pp. 221–236 (2010)