The code problem for traces—improving the boundaries ⋆

Hendrik Jan Hoogeboom

Leiden University, Dep. of Computer Science, P.O.Box 9512, 2300 RA Leiden, The Netherlands

Anca Muscholl

Universität Stuttgart, Institut für Informatik, Breitwiesenstr. 20-22, 70565 Stuttgart, Germany

1 Introduction

First studied in combinatorics [5], the theory of traces was introduced in computer science through the initiating work of Mazurkiewicz [14]. The initial idea was to provide a convenient model for the behaviour of labelled 1-safe Petri nets. Trace theory now represents a well-investigated mathematical framework for studying concurrent systems (for a detailed overview see [10]). The basic underlying concept is to consider a system as a finite set of actions Σ, together with a fixed symmetric independence relation $I \subseteq \Sigma \times \Sigma$, denoting pairs of actions which can be scheduled in parallel. In the concurrent setting defined by (Σ, I) we identify sequential observations (i.e., strings over Σ) modulo commutations $ab = ba$ for $(a, b) \in I$. This yields a free partially commutative monoid, called trace monoid by Mazurkiewicz and denoted $\mathbb{M}(\Sigma, I)$.

In practice, the behaviour of a concurrent system can be described in a concise form by e.g. finite automata or rational expressions. For partial commutations these two notions are different – automata recognizability is less expressive than rationality. This also leads to interesting decision problems for rational trace languages [3,1,17]. It is known e.g. that the question whether the intersection of two rational trace languages is empty or not is undecidable, in general. More precisely, the intersection problem is decidable if and only if the graph associated to the independence relation contains neither a cycle nor a path on four vertices as induced subgraph [1] (this property is known to characterize precisely comparability graphs of forests [19]). Further decision problems of interest arise in the context of coding theory. Consider for example

⋆ Research supported by the ESPRIT Basic Research Action No. 6317 (ASMICS 2).

the following two basic questions [15]: first, given two trace monoids $\mathbb{M}(\Sigma, I)$, $\mathbb{M}(\Sigma', I')$, can $\mathbb{M}(\Sigma', I')$ be encoded into $\mathbb{M}(\Sigma, I)$, i.e. does an injective homomorphism $h : \mathbb{M}(\Sigma', I') \to \mathbb{M}(\Sigma, I)$ exist? Second, given a homomorphism $h : \mathbb{M}(\Sigma', I') \to \mathbb{M}(\Sigma, I)$, is h a coding, i.e. is h injective? Partial results exist for both questions. The coding existence problem has been shown to be decidable (NP-complete) [9] for a natural subclass of trace homomorphisms, so-called strong homomorphisms, introduced in [4]. But in general the decidability of this problem remains open.

The question whether a homomorphism is a coding or not is known to be undecidable, in general. The undecidability holds even if the first monoid $\mathbb{M}(\Sigma', I')$ is free, i.e. if $I' = \emptyset$ (see [12, Satz 14,p. 102] and [6]). A further special case for undecidability is given by so-called clique-preserving morphisms, which are trace homomorphisms given by means of relations $H \subseteq \Sigma \times \Sigma'$ preserving independence cliques (for details see [9]). We are interested here in the first special case above and we denote as *code problem for* (Σ, I) the following question: given a finite set $X \subset \mathbb{M}(\Sigma, I)$, does X have the code property, i.e. does every $x \in X^+$ have a unique factorization over X?

The code problem for (Σ, I) is known to be undecidable if the graph associated to the independence alphabet (Σ, I) contains an induced C_4 (chordless cycle on four vertices); it is decidable whenever this graph contains neither C_4 nor P_4 (the chordless path on four vertices) as induced subgraph [6,1]. Due to the close connection between the code problem and the intersection problem for rational trace languages one may expect to have the same boundaries w.r.t. decidability for both problems. We show in this paper that this is not the case. We have two surprising results, which are in some sense negative. First, not all remaining cases are undecidable, in particular the code problem is decidable when (Σ, I) is exactly P_4. Independently, Yu. Matiyasevich observed the decidability of the code problem in this particular case, [13]. Second, we exhibit independence alphabets (Σ, I) containing P_4, but no C_4 as induced subgraph, for which the code problem is undecidable. This shows for the first time that not all cases without induced C_4 are decidable.

The paper is organized as follows: we first introduce our basic definitions and notations. Section 3 is devoted to decidability results, whereas Section 4 exhibits a new family of independence alphabets (Σ, I) with an undecidable code problem.

2 Preliminaries

Let (Σ, I) denote a finite *independence alphabet*, i.e. a finite alphabet Σ with an associated symmetric and irreflexive independence relation $I \subseteq \Sigma \times \Sigma$.

(Thus, the complement relation $D = (\Sigma \times \Sigma) \setminus I$ (dependence relation) is symmetric and reflexive.) Usually, we specify I by giving half of the relation, out of which I can be found by taking the symmetric closure. Let $\equiv_I$ denote the congruence on Σ^* induced by $\{ab = ba \mid (a,b) \in I\}$. The associated quotient monoid $\Sigma^*/\equiv_I$ is called *trace monoid over* (Σ, I) and is denoted by $\mathbb{M}(\Sigma, I)$. Hence, elements of $\mathbb{M}(\Sigma, I)$ –called traces– are congruence classes of words. Equivalently, the trace $[w]_{\equiv_I} \in \mathbb{M}(\Sigma, I)$ associated to $w = a_1 \dots a_n$, $a_i \in \Sigma$, has a unique representation (up to isomorphism) as labelled acyclic directed graph $[V, E, \lambda]$ –called dependence graph– as follows. The vertex set $V = \{1, \dots, n\}$ is labelled by $\lambda : V \to \Sigma$, $\lambda(i) = a_i$ for $1 \leq i \leq n$; the edge set E is given by $(i,j) \in E$ if and only if $i < j$ and $(a_i, a_j) \in D$. The product of two dependence graphs is given by adding to their disjoint union edges from vertices of the first factor to vertices of the second one, whenever their labellings are dependent.

A trace language over (Σ, I) is called *rational* if it belongs to the least family of subsets of $\mathbb{M}(\Sigma, I)$ containing $\emptyset$ and $\{a\}$, $a \in \Sigma$, which is closed under union, concatenation and Kleene star.

A subset $X \subseteq \mathbb{M}(\Sigma, I)$ has the *code property* if for all $u_i, v_j \in X$, $1 \leq i \leq n$, $1 \leq j \leq m$: $u_1 \cdots u_n = v_1 \cdots v_m$ implies $n = m$ and $u_i = v_i$ for every $1 \leq i \leq n$ [2]. We consider in the following finite sets X, only.

For $t \in \mathbb{M}(\Sigma, I)$, $a \in \Sigma$, we denote by $|t|_a$ the number of occurrences of the letter a in t. By $\mathrm{alph}(t)$ we denote the alphabet of t, i.e. the set $\{a \in \Sigma \mid |t|_a > 0\}$. For $\Gamma \subseteq \Sigma$ let π_Γ denote the canonical projection on $\mathbb{M}(\Gamma, I \cap (\Gamma \times \Gamma))$.

We will also use the usual notion of factor of a trace. Let $u, v \in \mathbb{M}(\Sigma, I)$, then u is a *factor* of v (denoted by $u \subseteq v$) if $v = tut'$ holds for some $t, t' \in \mathbb{M}(\Sigma, I)$. Equivalently, the dependence graph G_v of v contains an induced subgraph G_u isomorphic to the dependence graph of u, such that every directed path between any two vertices in G_u lies entirely in G_u (see e.g. [8]). We will consider in particular the situation where u is a suffix of v, i.e. with $t' = 1$ above. For $u, v \in \mathbb{M}(\Sigma, I)$ we denote by $u \leq v$ that u is a prefix of v, i.e. $t = 1$ in the above equation. Let u, v be factors of some w. Then $u \subseteq v$ means in addition to u being a factor of v, that G_u is a subgraph of G_v within G_w.

Whenever factorizations of traces are considered, the basic situation is settled by the so-called Levi's Lemma, which we state in the following (for details and further basic notions for traces see [10]):

Lemma 1 ([7]) *Let $x, x', w, w' \in \mathbb{M}$ be such that $xw = x'w'$. Then there exist unique traces $r, \alpha, \beta, s \in \mathbb{M}$ such that*

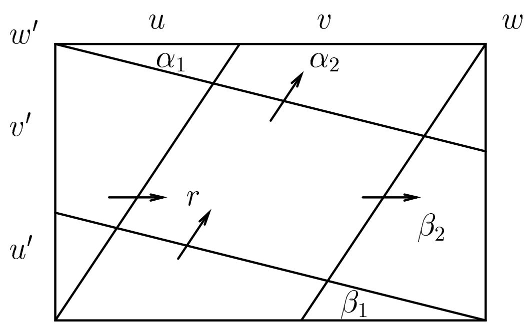


Fig. 1. generalized Levi Lemma

$x = r\alpha$, $w = \beta s$, $x' = r\beta$, $w' = \alpha s$, and
$alph(\alpha) \times alph(\beta) \subseteq I$.

Applying Levi's Lemma twice to the equation $uvw = u'v'w'$ yields directly (see also Fig. 1):

Lemma 2 *Let $u, v, w, u', v', w' \in \mathbb{M}(\Sigma, I)$ be such that $uvw = u'v'w'$. Then there exist traces r, α, β with*

$$uv = r\alpha,\ u'v' = r\beta \text{ and } alph(\alpha) \times alph(\beta) \subseteq I.$$

Moreover, α and β can be uniquely factorized as $\alpha = \alpha_1\alpha_2$ and $\beta = \beta_1\beta_2$ such that α_1 (α_2, β_1, β_2 resp.) is a suffix of u (v, u', v' resp.).

To conclude this section we note that we consider throughout the paper properties of independence alphabets (Σ, I) by viewing them as undirected graphs.

3 Decidability Results

The aim of this section is to show the decidability of the code problem over the independence alphabet P_4, i.e. over $\mathbb{M}(\Sigma, I)$ with $\Sigma = \{a, b, c, d\}$, $I = \{(a, b), (b, c), (c, d)\}$. Traces in this particular monoid $\mathbb{M}(\Sigma, I)$ have dependence graphs of the form represented in Fig. 2 (as Hasse diagram of the partial order, i.e. without redundant edges), where

- $A_i \in a\{a, c\}^*a \cup \{a\}$ and $D_j \in d\{b, d\}^*d \cup \{d\}$, for $2 \le i \le n$ and $1 \le j \le n$,
- $A_1, A_{n+1} \in a\{a, c\}^*a \cup \{1, a\}$,
- $k_i, l_j \ge 0$ for $1 \le i \le n + 1$, $0 \le j \le n + 1$, where $l_0 = 0$ if $A_1 = 1$, resp. $l_{n+1} = 0$ if $A_{n+1} = 1$.

Let $X \subset \mathbb{M}(\Sigma, I)$ be a finite set, for which we ask whether or not it satisfies the code property. In order to be a code, X may contain at most one element

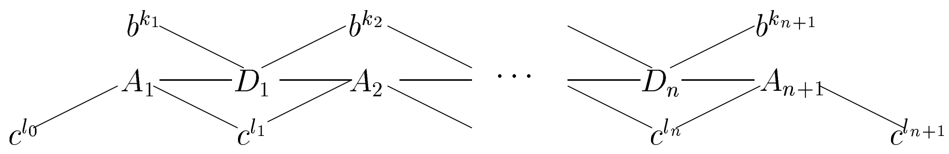


Fig. 2. dependence graph in P_4

x_0 such that $|x_0|_a = |x_0|_d = 0$, since any two such traces commute. Hence, we assume from now on that X contains at most one trace with this property, which we then denote by x_0.

For traces $w_1, w_2, \dots$ let $w[i]$ denote $w_1 \cdots w_i$.

Definition 3 *Let $u_1, \dots, u_i, v_1, \dots, v_j \in X$.*

The pair $(u[i], v[j])$ is called a partial solution, *if $u[i]\, s = v[j]\, s'$ holds for some $s, s' \in \mathbb{M}(\Sigma, I)$.*

Let $x \in \{a, d\}$. A partial solution $(u[i], v[j])$ is called x-synchronized, *if one of the following situations holds:*

- $\pi_{a,d} u_i \in \{a, d\}^* x$ *and* $\pi_{a,d} v[j-1] < \pi_{a,d} u[i] \leq \pi_{a,d} v[j]$, *or*
- $\pi_{a,d} v_j \in \{a, d\}^* x$ *and* $\pi_{a,d} u[i-1] < \pi_{a,d} v[j] \leq \pi_{a,d} u[i]$.

Less formally, a partial solution $(u[i], v[j])$ is x-synchronized if the rightmost common vertex of $u[i]$ and $v[j]$ with labelling in $\{a, d\}$ is labelled by x and belongs to both u_i and v_j (if any such common vertex exists).

The following lemma states a property of synchronized partial solutions, which will be crucial for the decision procedure.

Lemma 4 *Let $(u[i], v[j])$ be a x-synchronized partial solution, $x \in \{a, d\}$.*

Then unique factorizations

$$u[i] = r\alpha_0\alpha, \quad v[j] = r\beta_0\beta$$

exist satisfying

(i) $\alpha \subseteq u_i$, $\beta \subseteq v_j$ *and*
(ii) $alph(\alpha_0\alpha) \times alph(\beta_0\beta) \subseteq I$, *where*

$$\alpha_0\beta_0 \in b^* \textit{ if } x = a, \textit{ resp. } \alpha_0\beta_0 \in c^* \textit{ if } x = d.$$

PROOF. By Lem. 2 applied to $u[i-1]u_i s = v[j-1]v_j s'$ for some s, s', we have

$$u[i] = r\alpha_0\alpha, \quad v[j] = r\beta_0\beta \quad \text{with } \mathrm{alph}(\alpha_0\alpha) \times \mathrm{alph}(\beta_0\beta) \subseteq I, \tag{1}$$
$$\text{and } \alpha_0 \subseteq u[i-1],\ \alpha \subseteq u_i,\ \beta_0 \subseteq v[j-1],\ \beta \subseteq v_j. \tag{2}$$

We claim that $(u[i], v[j])$ being x-synchronized yields $\mathrm{alph}(\alpha_0\beta_0) \times \{x\} \subseteq I$. To see this, assume w.l.o.g. that $y \in \mathrm{alph}(\alpha_0)$ for some y with $(x, y) \in D$. Suppose further that $\pi_{a,d}v[j-1] < \pi_{a,d}u[i] \leq \pi_{a,d}v[j]$. Let z_x denote the last vertex labelled x in the dependence graph of u_i, G_{u_i}, whereas z_y denotes the last vertex labelled y in G_{α_0}. Within $G_{u[i]}$ we have an edge from z_y to z_x due to $\alpha_0 \subseteq u[i-1]$ and $u[i] = u[i-1]u_i$. On the other hand, according to the x-synchronization of $u[i], v[j]$, z_x belongs to G_r, hence there is an edge in $G_{u[i]}$ from z_x to z_y, due to $u[i] = r\alpha_0\alpha$. Contradiction. (Note the analogy of the proof with Levi's Lemma: the assumed existence of y as above would contradict Levi's Lemma for the decomposition $u[i-1](u_i s) = v[j]s'$.)

Before describing how synchronized partial solutions can be extended (whenever X is not a code) let us illustrate the problems which may arise by an example.

Example 5 *Consider the set $X = \{a, ada, dad\} \subseteq \{a, d\}^*$. Clearly, X is not a code, e.g. $(a)(dad)(ada) = (ada)(dad)(a)$. Testing the code property for a set of words is equivalent to an emptiness test for a (finite) automaton A_X. The states of A_X are given by all suffixes of elements of X, whereas the transitions are of the form $u \xrightarrow{1} v$, whenever $v \in X^{-1}u \cup u^{-1}X$. With initial states given by $X^{-1}X \setminus 1$ and 1 as the single final state, the above equivalence is readily seen (see also [2]).*

The problem arising for sets of traces is that information about suffixes is no longer bounded, in general. Consider e.g. the set $Y = \{a, d, ac, bd, cab^2, dbc^3, b^2c^3\}$ as a subset of $\mathbb{M}$, which also fails to be a code. By convention we write below $x_0 = b^2c^3$. We have for instance for arbitrary $m \in \mathbb{N}$

$$a \quad \binom{b^2}{ca} \qquad d \quad bd \quad x_0^2 \qquad a \quad \binom{b^2}{ca} \qquad d \quad bd \quad x_0^{2m} \quad (ac)^{2m} \quad a =$$
$$\cdots$$
$$ac \quad a \quad x_0 \quad \binom{db}{c^3} \quad d \qquad ac \quad a \quad x_0^{2m-1} \quad \binom{db}{c^3} \quad d \qquad a \quad \binom{ca}{b^2}^{2m}$$

The basic idea for guessing and verifying different factorizations over Y consists in splitting the information which has to be stored. A finite amount of information can be associated to blocks over $\{a, c\}^*$ (resp. $\{b, d\}^*$), whereas the (unbounded) number of "dangling" b's (resp. c's) is kept in a counter. For instance, after guessing $(a \ \binom{b^2}{ca} \ d \ bd \ x_0^2 \ \cdots \ d \ bd, ac \ a \ x_0 \ \binom{db}{c^3} \ d \ \cdots \ x_0^{2i-1} \ \binom{db}{c^3} \ d)$ the counter contains $6i$ c's, whereas 1 is kept in the finite control (denoting that the $\{d, b\}^*$ part is matched). The main difficulty arises as soon as x_0^{2i} has *to be guessed in the first component. Clearly, we can not decrease the counter by $6i$ and then increase it by $4i$, the number of dangling b's. The trick we use here amounts to store instead of $6i, 4i$ the number of occurrences of x_0, i.e., $2i$. This means that instead of guessing x_0^{2i} we simply reinterpret the value stored in the counter (from c's to b's in the previous situation). In our example, we will store into the counter every second b, resp. every third c.*

Proposition 6 *Assume $X \subset \mathbb{M}$ is not a code and let $w = u_1 \cdots u_n = v_1 \cdots v_m$, $u_i, v_j \in X$, with $(u_1, \ldots, u_n) \neq (v_1, \ldots, v_m)$ be such that $|w|$ is minimal with this property.*

Let $(u[i], v[j])$ be a x-synchronized partial solution, $x \in \{a, d\}$ and $(i, j) \neq (n, m)$.

Then there exist $x' \in \{a, d\}$ and $s, t \geq 0$ with $s + t > 0$ satisfying the following properties:

1. *$(u[i + s], v[j + t])$ is x'-synchronized and*
   - *either $s \leq 1$ and $v_{j+q} = x_0$ for all $1 \leq q < t$,*
   - *or $t \leq 1$ and $u_{i+p} = x_0$ for all $1 \leq p < s$,*

   where x_0 denotes the unique element of $X \cap b^*c^*$ (if any).
2. *$s + t$ is minimal such that $(u[i + s], v[j + t])$ is synchronized.*

PROOF. Let $u[i] = r\alpha_0\alpha$, $v[j] = r\beta_0\beta$, with $r, \alpha_0, \alpha, \beta_0, \beta$ as stated in Lem. 4.

Suppose first that $\pi_{a,d}\alpha \neq 1$ (hence, $\pi_{a,d}\beta = 1$ due to $\mathrm{alph}(\alpha) \times \mathrm{alph}(\beta) \subseteq I$). Then let $t > 0$ be minimal such that $\pi_{a,d}v_{j+t} \neq 1$. In this case $(u[i], v[j + t])$ satisfies the requirement of the proposition, with $v_{j+q} = x_0$, for all $1 \leq q < t$ (resp. $t = 1$, if $X \cap b^*c^* = \emptyset$).

Let now $\pi_{a,d}\alpha = \pi_{a,d}\beta = 1$, hence $\mathrm{alph}(\alpha_0\alpha\beta_0\beta) \subseteq \{b, c\}$. Let $s, t > 0$ with $s+t$ minimal be such that $\pi_{a,d}u_{i+s} \neq 1$ and $\pi_{a,d}v_{j+t} \neq 1$. Hence, $u_{i+p} = v_{j+q} = x_0$ for all $1 \leq p < s$ and $1 \leq q < t$, if $X \cap b^*c^* \neq \emptyset$ (otherwise, $s = t = 1$). Note that if $(s-1)(t-1) \neq 0$, then $|w|$ would not be minimal: by $\alpha_0\alpha\beta_0\beta \in b^*c^*$ and $(b, c) \in I$ this would imply $\alpha_0\alpha x_0^{s-2}u_{i+s} \cdots u_n = \beta_0\beta x_0^{t-2}v_{j+t} \cdots v_m$. Therefore either $(u[i + 1], v[j + t])$ is a synchronized partial solution with $v_{j+q} = x_0$ for all $1 \leq q < t$; or $(u[i + s], v[j + 1])$ is a synchronized partial solution with $u_{i+p} = x_0$ for all $1 \leq p < s$.

We show that the code problem is decidable for P_4 by reducing it to the emptiness problem for one-counter automata. A one-counter automaton is a pushdown automaton $\mathcal{A} = (Q, \Sigma, \Gamma, \delta, q_0, Z, F)$ [11] where the pushdown alphabet Γ contains only one symbol $\square$, besides the bottom-of-stack marker Z (i.e. the pushdown content is from $Z\square^*$). With the transition relation $\delta \subseteq$

$Q \times (\Sigma \cup \{1\}) \times \Gamma \times Q \times \Box^*$ the automaton can increment, decrement resp. test the counter for zero. The value of the counter will be assumed to be an integer, by storing the sign in the finite control.

The language $L(\mathcal{A})$ accepted by the one-counter automaton $\mathcal{A}$ defined below will satisfy the condition $L(\mathcal{A}) \neq \emptyset$ if and only if X is not a code. Since the emptiness problem for pushdown automaton is decidable (in polynomial time) [11] this will provide the claimed result.

Whenever X is not a code, the automaton $\mathcal{A}$ will determine two different factorizations over X of a trace $w \in \mathbb{M}(\Sigma, I)$, i.e. $w = u_1 \cdots u_n = v_1 \cdots v_m$ with $u_i, v_j \in X$, $(u_1, \ldots, u_n) \neq (v_1, \ldots, v_m)$. Every configuration of $\mathcal{A}$, i.e. the state $q \in Q$ and the counter content $\gamma \in Z\Box^*$, is associated to a synchronized partial solution. The informal idea is that $\mathcal{A}$ will "read" in blocks A_h (resp. D_h) by guessing a- (resp. d-) synchronized partial solutions (see Fig. 2). Some bounded information concerning A_h (resp. D_h) will be stored in the finite control, whereas the unbounded information concerning the number of b's (resp. c's) will be stored in the counter. With the notations of Lem. 4 the configuration of $\mathcal{A}$ associated to the x-synchronized partial solution $(u[i], v[j])$ is described by (α, β) as part of the state, resp. $|\alpha_0| - |\beta_0| \in \mathbb{Z}$ as content of the counter. Hence, the counter will store (the number of) either b's or c's; it counts b's when $(u[i], v[j])$ is a-synchronized, resp. c's, when $(u[i], v[j])$ is d-synchronized. $\mathcal{A}$ will have initial states (α, β) with $\mathrm{alph}(\alpha) \times \mathrm{alph}(\beta) \subseteq I$, associated to some $u, v \in X$, $u \neq v$, satisfying $u = r\alpha$, $v = r\beta$ for a suitable r (with the counter initially empty, corresponding to $\alpha_0 = \beta_0 = 1$); the final configuration is given by the state $(\alpha = 1, \beta = 1)$ and the empty counter.

Before going into details let us note that we have to distinguish four cases, depending on whether or not X contains a trace $x_0 = b^k c^l$ (with $k, l > 0$ resp. $k > 0 = l$ resp. $l > 0 = k$). We describe in the following the most complex case above $(k, l > 0)$ and briefly discuss at the end the other cases. Hence, assume from now on that $X \cap b^* c^* = \{b^k c^l\}$ with $k, l > 0$.

The difficulty arising in the informal description given above concerns the following situation: assume $(u[i], v[j])$ is a-synchronized and (α, β), resp. $|\alpha_0| - |\beta_0|$ is the associated configuration of $\mathcal{A}$, where $\alpha_0 \beta_0 \in b^*$. Assume further that $(u[i+1], v[j+t])$ is x'-synchronized as in Prop. 6, where $x' = d$. The consistency check for $(u[i+1], v[j+t])$ has to ensure that $\alpha_0 \alpha u_{i+1}$ resp. $\beta_0 \beta x_0^{t-1} v_{j+t}$ have a common upper bound (w.r.t. the prefix order). In particular, after processing $x_0^{t-1} v_{j+t}$ $\mathcal{A}$ has to check the counter for zero (i.e., check that both factorizations have the same number of b's occurring before $x' = d$). On the other side, the content of the counter associated to $(u[i+s], v[j+t])$ has to include the value $-(t-1)l$ for the new occurrences of the letter c from x_0^{t-1}. Obviously, the counter cannot be decreased and then increased in order to store the required information. In order to overcome this problem , the content of the counter for

$(u[i], v[j])$ will actually correspond to $t-1$, i.e. to $(|\alpha_0| - |\beta_0|)$ div k (instead of $|\alpha_0| - |\beta_0|$). The value $|\alpha_0| - |\beta_0|$ mod k will be part of the finite control. For d-synchronized partial solutions we replace k by l.

We describe in the following the transition relation of $\mathcal{A}$. Let us assume that the configuration of $\mathcal{A}$ corresponds to an a-synchronized partial solution $(u[i], v[j])$, with $\alpha_0 \in b^*$, $\beta_0 = 1$. With the notations of Prop. 6 we may assume that $s, t \geq 0$, $x' \in \{a, d\}$ exist such that $0 \leq s \leq 1$, $s + t > 0$, $v_{j+q} = x_0$ $(1 \leq q < t)$ and $(u[i+s], v[j+t])$ is x'-synchronized. If $\mathcal{A}$ guesses $x' \in \{a, d\}$, $s, t \geq 0$ and $u_{i+s}, v_{j+t} \in X$ as above (with the convention $u_{i+s} = 1$ if $s = 0$), then it will test consistency and compute α', β' (resp. update the counter according to $|\alpha'_0| - |\beta'_0|$) such that

- $\alpha_0 \alpha u_{i+s} = r' \alpha'_0 \alpha'$ and $\beta_0 \beta x_0^{t-1} v_{j+t} = r' \beta'_0 \beta'$ for some r', with $\mathrm{alph}(\alpha'_0 \alpha') \times \mathrm{alph}(\beta'_0 \beta) \subseteq I$, and
- $\alpha'_0 \subseteq \alpha_0 \alpha$, $\beta'_0 \subseteq \beta_0 \beta x_0^{t-1}$, $\alpha' \subseteq u_{i+s}$, $\beta' \subseteq v_{j+t}$, if $s = 1$ (the case $s = 0$ is handled accordingly).

The automaton $\mathcal{A}$ distinguishes mainly two cases (with $u_{i+s} = 1$ if $s = 0$):

(i) If $a \in \mathrm{alph}(\beta v_{j+t}) \cap \mathrm{alph}(\alpha u_{i+s})$, $t \geq 1$, then it can be verified that $t < |\alpha u_{i+s}|$, since $(a, c) \in D$ (and noting that $|\beta|_a = 0$, due to the minimality of $s+t$). Hence, $\mathcal{A}$ can verify directly that $\alpha_0 \alpha u_{i+s}$ and $\beta_0 \beta x_0^{t-1} v_{j+t}$ have a common upper bound and compute the new configuration accordingly. Note also that the counter may be tested in this case for equality with a constant bounded by X (hence, for zero).

(ii) Otherwise we have $x' = d$ and $\alpha'_0 \beta'_0 \in c^*$. Since $\alpha'_0 \subseteq \alpha$, $\beta'_0 \subseteq \beta x_0^{t-1}$ we obtain that $||\alpha'_0| - |\beta'_0||$ div l and $t-1$ differ by a value bounded by α, β. Moreover, $x' = d$ implies that $(|\alpha_0| - |\beta_0|)$ div k and $t-1$ differ by a value bounded by α, β, u_{i+s}, v_{j+t}. Hence, it suffices to modify the counter by a value depending on X, only, (and update the finite state accordingly), in order to provide the correct information about $(|\alpha'_0| - |\beta'_0|)$ div l.

As a final remark we note that if $X \cap b^+ c^+ = \emptyset$, then the automaton $\mathcal{A}$ can store directly $|\alpha_0| - |\beta_0|$, since no reinterpretation of the counter is needed.

Theorem 7 *The code problem for* $(\Sigma, I) = (\{a, b, c, d\}, \{(a, b), (b, c), (c, d)\})$ *is decidable in polynomial time.*

Remark 8 *The complexity of the code problem for* P_4 *can be characterized more precisely. This question is NL-complete, i.e. complete for the class of problems which can be solved by nondeterministic Turing machines with logarithmic space. The hardness is provided already in the case of free monoids (over two letters alphabets) [16]. The code problem is shown to belong to NL by noting e.g. that one can test the existence of two different factorizations over a given* $X \subset \mathbb{M}(\Sigma, I)$ *by using a 2-way multihead nondeterministic one-*

counter automaton. With the notations of Lem. 4 this automaton keeps track of α and β using two heads, which point at the corresponding elements of X in the input; the counter is used as in the proof of Thm. 7, whereas the modulo k (resp. l) values are handled by further heads on the input. Since the class of languages accepted by 2-way multihead nondeterministic one-counter automata is known to coincide with NL [18], this provides the result.

The above decidability result can be slightly generalized to independence alphabets (Σ, I) of the following form: there exists a partition of the alphabet $\Sigma = \Sigma_p \dot{\bigcup} \Sigma_b$ and a mapping $\varphi : \Sigma_p \to \Sigma_b$ such that $I = \binom{\Sigma_b}{2} \cup \{(a, \varphi(a)) \mid a \in \Sigma_p\}$. A one-counter automaton synchronizes here on (dependent) letters (pins) $a \in \Sigma_p$, while counting the associated (body) letter $\varphi(a) \in \Sigma_b$. As before, X may contain at most one element x_0 with $\mathrm{alph}(x_0) \subseteq \Sigma_b$.

4 Undecidability Results

Let $[L]_I \subseteq \mathbb{M}(\Sigma, I)$ denote the trace language associated to $L \subseteq \Sigma^*$, i.e. $[L]_I = \{[w]_{\equiv_I} \mid w \in L\}$. It can be seen by induction on rational expressions that a trace language $K \subseteq \mathbb{M}(\Sigma, I)$ is rational if and only if $K = [L]_I$ for some rational word language $L \subseteq \Sigma^*$. The intersection problem for rational trace languages has been shown to be undecidable for the independence alphabet P_4:

Proposition 9 ([1], Thm. 3.5) *Let $(\Sigma, I) = (\{a, b, c, d\}, \{(a, b), (b, c), (c, d)\})$. Given two rational languages $K, L \subseteq \Sigma^*$, it is undecidable whether or not $[K]_I \cap [L]_I = \emptyset$.*

Consider now the family $\mathcal{G}$ of undirected graphs given by $(\Sigma, I) \in \mathcal{G}$ if distinct vertices $a, b, c, d, e, f \in \Sigma$ exist such that $I \cap \{a, b, c, d\}^2 = \{(a, b), (b, c), (c, d)\}$, $(e, f) \notin I$ and $\{b, c\} \times \{e, f\} \subseteq I$ (see Fig. 3, where dashed edges may occur additionally).

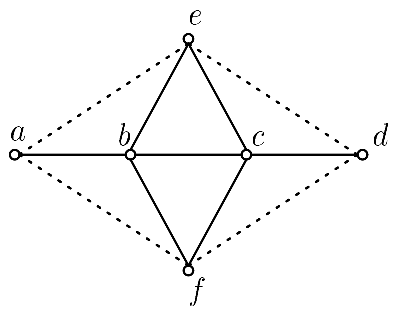


Fig. 3. graph family with undecidable code problem

Theorem 10 *The code problem for $(\Sigma, I) \in \mathcal{G}$ is undecidable.*

PROOF. We give a reduction of the intersection problem for $(\Sigma, I) = (\{a, b, c, d\}, \{(a, b), (b, c), (c, d)\})$ to the code problem for $(\Sigma', I') \in \mathcal{G}$, with $\Sigma' \supseteq \Sigma \cup \{e, f\}$ and $I' \cap \Sigma^2 = \{(a, b), (b, c), (c, d)\}$, $(e, f) \notin I'$, $\{e, f\} \times \{b, c\} \subseteq I'$.

Let $\mathcal{A}_i = (Q_i, \Sigma, \delta_i, J_i, F_i)$, $i = 1, 2$, be two finite automata effectively giving two rational languages $K, L \subseteq \Sigma^*$ (with δ_i denoting the transition relations and J_i resp. F_i the sets of initial resp. final states). We assume that $Q_1 \cup Q_2 \subseteq \{5, 6, \ldots, M\}$, for some integer M – the numbers 0 to 4 will be used for other purposes. Additionally we can assume without loss of generality that neither $\mathcal{A}_1$, nor $\mathcal{A}_2$ have parallel edges, i.e., there is at most one edge leading from one specific state to another specific state.

We describe below a finite set $X \subset \mathbb{M}(\Sigma', I')$ with the property that X is not a code if and only if the trace languages $[K]_I$ and $[L]_I$ have an element in common. Two different factorizations over X will correspond to accepting computations of $\mathcal{A}_1$ and $\mathcal{A}_2$ on two $\equiv_I$-equivalent words over Σ. The letters a, b, c, d retain their original meaning as elements of the input alphabet. The new symbols f and e will be used to encode the states of the automata.

For each $0 \leq i \leq M$ we use $\langle i \rangle$ to denote the string ef^ie. The numbers $1, 2, 3$ denote three types of code words, or actually three types of synchronization between the automata, whereas the numbers 0 and 4 are used to signal begin and end of synchronized computations.

1. For every transition $(i_1, x, i_2) \in \delta_1$ and every $j \in Q_2$, let $\langle i_1 \rangle \langle j \rangle \langle i_2 \rangle \langle 1 \rangle x \langle j \rangle \in X$, when $x \in \{a, d\}$, respectively $\langle i_1 \rangle \langle j \rangle \langle i_2 \rangle \langle 2 \rangle x \langle j \rangle \in X$, when $x \in \{b, c\}$.
2. For every transition $(j_1, x, j_2) \in \delta_2$ and every $i \in Q_1$, let $\langle i \rangle \langle 1 \rangle x \langle j_1 \rangle \langle i \rangle \langle j_2 \rangle \in X$, when $x \in \{a, d\}$, respectively $\langle i \rangle \langle 3 \rangle x \langle j_1 \rangle \langle i \rangle \langle j_2 \rangle \in X$, when $x \in \{b, c\}$.
3. For every pair $i \in Q_1$, $j \in Q_2$, let $\langle i \rangle \langle j \rangle \langle i \rangle \langle 3 \rangle \langle j \rangle \in X$ and $\langle i \rangle \langle 2 \rangle \langle j \rangle \langle i \rangle \langle j \rangle \in X$.
4. X contains $ad\langle 0 \rangle$ and $\langle 4 \rangle ad$. For every pair $i \in J_1$, $j \in J_2$ of initial states let $ad\langle 0 \rangle \langle i \rangle \langle j \rangle \in X$. Similarly, for every pair $i \in F_1$, $j \in F_2$ of final states let X contain $\langle i \rangle \langle j \rangle \langle 4 \rangle ad$.

Consider now $u_i, v_j \in X$, $0 \leq i \leq n$, $0 \leq j \leq m$ such that $u_0 \cdots u_n = v_0 \cdots v_m$, $(u_0, \ldots, u_n) \neq (v_0, \ldots, v_m)$ and $|u_0 \cdots u_n|$ is minimal (in particular, $u_0 \neq v_0$). Since there is no pair $\langle i \rangle \langle 1 \rangle x \langle j \rangle \langle i \rangle \langle j' \rangle, \langle i \rangle \langle 1 \rangle x' \langle j \rangle \langle i \rangle \langle j' \rangle$ in X with $x \neq x'$ (analogously for other elements defined in 1., 2.) we immediately have $u_0 = ad\langle 0 \rangle$ and $v_0 = ad\langle 0 \rangle \langle i_0 \rangle \langle j_0 \rangle$ for some initial states $i_0 \in J_1$, $j_0 \in J_2$.

We show by induction on l that $\pi_{e,f}(v_0 \cdots v_l) = \pi_{e,f}(u_0 \cdots u_l) \langle i \rangle \langle j \rangle$ holds for some $i \in Q_1$, $j \in Q_2$ such that $i \in \delta_1(i_0, u)$ and $j \in \delta_2(j_0, v)$ for some $u, v \in \Sigma^*$, where $[u]_{\equiv_I} = \pi_\Sigma(u_1 \cdots u_l)$, $[v]_{\equiv_I} = \pi_\Sigma(v_1 \cdots v_l)$. Assume the claim holds for $l - 1 \geq 1$. By definition of X there are four possible cases:

1. $u_l = \langle i\rangle\langle j\rangle\langle i'\rangle\langle 1\rangle x\langle j\rangle$ for $x \in \{a,d\}$ and some $(i,x,i') \in \delta_1$, $j \in Q_2$; hence, $v_l = \langle i'\rangle\langle 1\rangle x\langle j\rangle\langle i'\rangle\langle j'\rangle$ for some $(j,x,j') \in \delta_2$.
2. $u_l = \langle i\rangle\langle j\rangle\langle i'\rangle\langle 2\rangle x\langle j\rangle$ for $x \in \{b,c\}$ and some $(i,x,i') \in \delta_1$, $j \in Q_2$; hence, $v_l = \langle i'\rangle\langle 2\rangle\langle j\rangle\langle i'\rangle\langle j\rangle$.
3. $u_l = \langle i\rangle\langle j\rangle\langle i\rangle\langle 3\rangle\langle j\rangle$; hence, $v_l = \langle i\rangle\langle 3\rangle\langle j\rangle x\langle i\rangle\langle j'\rangle$ for some $x \in \{b,c\}$ and $(j,x,j') \in \delta_2$.
4. $u_l = \langle i\rangle\langle j\rangle\langle 4\rangle ad$, hence $v_l = \langle 4\rangle ad$. It follows $u_0 \cdots u_l = v_0 \cdots v_l$, hence $w := \pi_\Sigma(u_0 \cdots u_l) = \pi_\Sigma(v_0 \cdots v_l)$ and thus $w \in [K]_I \cap [L]_I$.

For the converse let $x, y \in \Sigma^*$ be such that $x \equiv_I y$ and $x = x_1 \cdots x_n \in K$, $y = y_0 \cdots y_n \in L$, $x_p, y_q \in \Sigma$. Let $i_0 \in J_1$, $j_0 \in J_2$ denote initial states from which accepting computations for x resp. y exist and let $u_0 = ad\langle 0\rangle$, $v_0 = ad\langle 0\rangle\langle i_0\rangle\langle j_0\rangle$. Consider prefixes $x' = x_1 \cdots x_p$, $y' = y_1 \cdots y_q$ ($1 \le p, q \le n$) with $[x']_{\equiv_I} = ru$, $[y']_{\equiv_I} = rv$ for some $r, u, v \in \mathbb{M}(\Sigma, I)$, where $(u,v) \in b^* \times c^* \cup c^* \times b^*$. We show by induction on $p+q$ that traces $u_1, \ldots, u_l, v_1, \ldots, v_l \in X$ exist such that

$$u_0 u_1 \cdots u_l = su \text{ and } v_0 v_1 \cdots v_l = sv\langle i\rangle\langle j\rangle,$$

where $i \in \delta_1(i_0, x')$, $j \in \delta_2(j_0, y')$.

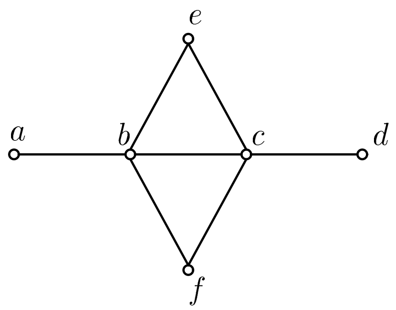
The claim is satisfied for $p + q = 0$. We distinguish in the following cases:

(i) If $p < n$ and $x_{p+1} \in \{b,c\}$ then let $u_{l+1} = \langle i\rangle\langle j\rangle\langle i'\rangle\langle 2\rangle\langle j\rangle x_{p+1}$ and $v_{l+1} = \langle i'\rangle\langle 2\rangle\langle j\rangle\langle i'\rangle\langle j\rangle$, with $(i, x_{p+1}, i') \in \delta_1$ corresponding to the accepting computation on x. The case where $q < n$ and $y_{q+1} \in \{b,c\}$ is handled symmetrically. Note that $u_0 \cdots u_{l+1}$, $v_0 \cdots v_{l+1}$ and $x'x_{p+1}, y'$ (resp. $x', y'y_{q+1}$) satisfy the requirements due to $\{b,c\} \times \{e,f\} \subseteq I$.

(ii) If $p < n$, $q < n$ and $\{x_{p+1}, y_{q+1}\} \cap \{b,c\} = \emptyset$ then $x_{p+1} = y_{q+1} \in \{a,d\}$ due to $x \equiv_I y$ and $(u,v) \in b^* \times c^* \cup c^* \times b^*$. In this case let $u_{l+1} = \langle i\rangle\langle j\rangle\langle i'\rangle\langle 1\rangle x_{p+1}\langle j\rangle$, $v_{l+1} = \langle i'\rangle\langle 1\rangle y_{q+1}\langle j\rangle\langle i'\rangle\langle j'\rangle$ for $(i, x_{p+1}, i') \in \delta_1$ and $(j, y_{q+1}, j') \in \delta_2$ corresponding to the accepting computations on x resp. y. Note that in this case either $u = 1$ or $v = 1$, and $\mathrm{alph}(uv) \times \mathrm{alph}(x_{p+1}) \subseteq I$. It can be easily verified that $u_0 \cdots u_{l+1}$, $v_0 \cdots v_{l+1}$ and $x'x_{p+1}, y'y_{q+1}$ satisfy the requirements.

Finally, if $p = q = n$ then $i \in F_1$, $j \in F_2$ and we conclude by taking $u_{l+1} = \langle i\rangle\langle j\rangle\langle 4\rangle ad$, $v_{l+1} = \langle 4\rangle ad$.

As a final remark we note that the edge-minimal new independence alphabet (Σ, I) with an undecidable code problem is minimal w.r.t induced subgraphs and undecidability.



I.e., the code problem is decidable for all induced subgraphs $(\Sigma', I|_{\Sigma'\times\Sigma'})$, $\Sigma' \subset \Sigma$, as it can be easily checked:

(i) $\Sigma' = \Sigma \setminus \{x\}$, $x \in \{a, b, c, d\}$ contains neither P_4 nor C_4 as induced subgraph.
(ii) $\Sigma' = \Sigma \setminus \{x\}$, $x \in \{e, f\}$, can be represented as $\Sigma' = \Sigma_b \cup \Sigma_p$ with $\Sigma_p = \{a, d\}$, $\Sigma_b = \{b, c, e, f\} \setminus \{x\}$, as mentioned at the end of Sect. 3.

Acknowledgement

We thank Volker Diekert for bringing to our attention that Yu. Matiyasevich independently observed the decidability of P_4. We are also indebted to the anonymous referees for suggesting improvements in our presentation.

References

[1] IJ. J. Aalbersberg and H. J. Hoogeboom. Characterizations of the decidability of some problems for regular trace languages. *Mathematical Systems Theory*, 22:1–19, 1989.

[2] J. Berstel and D. Perrin. *Theory of Codes*. Pure and Applied Mathematics; 117. Academic Press, Orlando, Florida, 1985.

[3] A. Bertoni, G. Mauri, and N. Sabadini. Equivalence and membership problems for regular trace languages. In *Proceedings of the 9th International Colloquium on Automata, Languages and Programming (ICALP'82)*, number 140 in Lecture Notes in Computer Science, pages 61–71, Berlin-Heidelberg-New York, 1982. Springer.

[4] V. Bruyère, C. De Felice, and G. Guaiana. Coding with traces. In P. Enjalbert, E. Mayr, and K. W. Wagner, editors, *Proceedings of the 11th Annual Symposium on Theoretical Aspects of Computer Science (STACS'94), 1994*, number 775 in Lecture Notes in Computer Science, pages 353–364, Berlin-Heidelberg-New York, 1994. Springer.

[5] P. Cartier and D. Foata. *Problèmes combinatoires de commutation et réarrangements*. Number 85 in Lecture Notes in Mathematics. Springer, Berlin-Heidelberg-New York, 1969.

[6] M. Chrobak and W. Rytter. Unique decipherability for partially commutative alphabets. *Fundamenta Informaticae*, X:323–336, 1987.

[7] R. Cori and D. Perrin. Automates et commutations partielles. *R.A.I.R.O. — Informatique Théorique et Applications*, 19:21–32, 1985.

[8] V. Diekert. *Combinatorics on Traces.* Number 454 in Lecture Notes in Computer Science. Springer, Berlin-Heidelberg-New York, 1990.

[9] V. Diekert, A. Muscholl, and K. Reinhardt. On codings of traces. In E. Mayr and C. Puech, editors, *Proceedings of the 12th Annual Symposium on Theoretical Aspects of Computer Science (STACS'95), 1995*, number 900 in Lecture Notes in Computer Science, pages 385–396, Berlin-Heidelberg-New York, 1995. Springer.

[10] V. Diekert and G. Rozenberg, editors. *The Book of Traces.* World Scientific, Singapore, 1995.

[11] M. A. Harrison. *Introduction to Formal Language Theory.* Addison-Wesley Publishing Company, 1978.

[12] G. Hotz and V. Claus. *Automatentheorie und Formale Sprachen, Band III.* Bibliographisches Institut, Mannheim, 1972.

[13] Yu. Matyiasevich. Cas décidables et indécidables du problème du codage pour les monoïdes partialement commutatifs, 1996. To appear in Quadrature.

[14] A. Mazurkiewicz. Concurrent program schemes and their interpretations. DAIMI Rep. PB 78, Aarhus University, Aarhus, 1977.

[15] E. Ochmański. On morphisms of trace monoids. In R. Cori and M. Wirsing, editors, *Proceedings of the 5th Annual Symposium on Theoretical Aspects of Computer Science (STACS'88)*, number 294 in Lecture Notes in Computer Science, pages 346–355, Berlin-Heidelberg-New York, 1988. Springer.

[16] W. Rytter. The space complexity of the unique decipherability problem. *Information Processing Letters*, 23:1–3, 1986.

[17] J. Sakarovitch. The "last" decision problem for rational trace languages. In I. Simon, editor, *Proceedings of the 1st Latin American Symposium on Theoretical Informatics (LATIN'92)*, number 583 in Lecture Notes in Computer Science, pages 460–473, Berlin-Heidelberg-New York, 1992. Springer.

[18] K. Wagner and G. Wechsung. *Computational complexity.* VEB Deutscher Verlag der Wissenschaften, Berlin, 1986.

[19] E. S. Wolk. A note on the comparability graph of a tree. *Proc. of the Amer. Math. Soc.*, (16):17–20, 1965.