Generating All Minimal Petri Net Unsolvable Binary Words*

Evgeny Erofeev[1], Kamila Barylska[2], Łukasz Mikulski[2], and Marcin Piątkowski[2]

[1] Parallel Systems, Department of Computing Science
Carl von Ossietzky Universität, D-26111 Oldenburg, Germany
evgeny.erofeev@informatik.uni-oldenburg.de
[2] Faculty of Mathematics and Computer Science
Nicolaus Copernicus University, 87-100 Toruń, Poland
{kamila.barylska,lukasz.mikulski,marcin.piatkowski}@mat.umk.pl

Abstract. Sets of finite words, as well as some infinite ones, can be described using finite systems, e.g. automata. On the other hand, some automata may be constructed with the use of even more compact models, like Petri nets. We call such automata Petri net solvable. In this paper we consider the solvability of singleton languages over a binary alphabet (i.e. binary words). An unsolvable (i.e. not solvable) word w is called minimal if each proper factor of w is solvable. We present a complete language-theory characterisation of the set of all minimal unsolvable binary words. The characterisation utilises morphic-based transformations which expose the combinatorial structure of those words, and allows to introduce a pattern matching condition for unsolvability.

Keywords: binary words, labelled transition systems, generations, Petri nets, synthesis

1 Introduction

To deal with infinite sets of words we need to specify them in a finite way. Finite automata which are known as a classical model for describing regular languages, are equivalent to finite labelled transition systems [9]. Some sets may be expressed with use of even more compact system models.

In this paper we investigate the synthesis problem with a specifications given in the form of labelled transition systems. The sought system model is a free-labelled place/transition Petri net [12], with its reachability graph as a natural bridge between specification and implementation. Namely, we are concerned with finding a net, whose reachability graph is isomorphic to a given labelled transition system. Labelled Petri nets are known to be more powerful than finite automata, and hence labelled transition systems [10]. On the other hand, the class of free-labelled Petri net languages is a subset of the class of all Petri net languages. In the present paper we draw attention to the following question: what classes of automata can or cannot be generated by free-labelled Petri nets.

To address this issue one may use the theory of regions [1]. For a given labelled transition system, the solution of a number of linear inequations systems provided by the theory of regions exists if and only if there exists an implementation in a net form.

* This research has been partially supported by the Polish grant No.2013/09/D/ST6/03928, and by DFG (German Research Foundation) through grant Be 1267/14-1 CAVER (Design and Analysis Methods for Real-Time Systems) and Graduiertenkolleg GRK-1765 SCARE (System Correctness under Adverse Conditions).

Moreover, solutions of such linear inequations systems are usually utilised during the synthesis of the resulting system (see Synet [5] and APT [13]).

Our aim is to suggest a combinatorial approach and to provide a complete characterisation of a generative nature for a special kind of labelled transition systems – non-branching and acyclic transition systems having at most two labels (i.e. binary words) [2]. More precisely, we characterise all minimal unsolvable binary words.

The paper is organized as follows. First we give some basic notions and notations concerning labelled transition systems, Petri nets and theory of regions. After that we present a necessary condition for minimal unsolvability in the form of extended regular expressions [6]. It allows to formulate possible shapes of minimal unsolvable words. In section 4 we introduce the notion of (base) extendable and non-extendable binary unsolvable words. In the following sections we provide the main results of this paper: a generic characterisation of all minimal unsolvable binary words (section 5) and its utilization for an efficient verifying procedure (section 6). We conclude the paper with a short section containing some directions for further research.

Due to the page limitation, most of technical proofs were omitted. The extended version of this paper containing all the proofs and a detailed argumentation is available for more inquisitive readers (see: [3]).

2 Basic notions

In this section we introduce notions used throughout the paper.

Words
A *word* (or a *string*) *over alphabet* T is a finite sequence $w \in T^*$, and it is *binary* if $|T| = 2$. For a word w and a letter t, $\#_t(w)$ denotes the number of times t occurs in w. A word $w' \in T^*$ is called a *subword* (or *factor*) of $w \in T^*$ if $\exists u_1, u_2 \in T^* \colon w = u_1 w' u_2$. In particular, w' is called a *prefix* of w if $u_1 = \varepsilon$, a *suffix* of w if $u_2 = \varepsilon$, and an *infix* of w if $u_1 \neq \varepsilon$ and $u_2 \neq \varepsilon$. For a word $w = x_1 x_2 \cdots x_n$ we use a notation for a factor $w[i..j] = x_i \cdots x_j$ and for a single letter $w[i] = x_i$.

A mapping $\phi : \Sigma_1^* \to \Sigma_2^*$ is called a *morphism* if we have $\phi(u{\cdot}v) = \phi(u){\cdot}\phi(v)$ for every $u, v \in \Sigma_1^*$ whenever all operations are defined. A morphism ϕ is uniquely determined by its values on the alphabet. Moreover, ϕ maps the neutral element of Σ_1^* into the neutral element of Σ_2^*.

Transition systems
A *finite labelled transition system* (or simply *lts*) with an initial state is a tuple $TS = (S, T, \to, s_0)$ with nodes S (a finite set of states), edge labels T (a finite set of letters), edges $\to \subseteq (S \times T \times S)$, and an initial state $s_0 \in S$.[1] A label t is enabled at $s \in S$, denoted by $s[t\rangle$, if $\exists s' \in S \colon (s, t, s') \in \to$. A state s' is reachable from s through the execution of $\sigma \in T^*$, denoted by $s[\sigma\rangle s'$, if there is a directed path from s to s' which edges are labelled consecutively by σ. The set of states reachable from s is denoted by $[s\rangle$. A sequence $\sigma \in T^*$ is enabled, or firable, at a state s, denoted by $s[\sigma\rangle$, if there is some state s' such that $s[\sigma\rangle s'$.[2] Two labelled transition systems $TS_1 = (S_1, \to_1, T, s_{0_1})$ and $TS_2 = (S_2, \to_2, T, s_{0_2})$ are isomorphic if there is a bijection $\zeta \colon S_1 \to S_2$ with $\zeta(s_{0_1}) = s_{0_2}$ and $(s, t, s') \in \to_1 \Leftrightarrow (\zeta(s), t, \zeta(s')) \in \to_2$, for all $s, s' \in S_1$.

[1] Note that an lts may be considered as a finite automata with no specified set of accepting states.
[2] For compactness, in case of long formulas we write $|_r\, \alpha\, |_s\, \beta\, |_t$ instead of $r\ [\alpha\rangle\ s\ [\beta\rangle\ t$.

A word $w = t_1 t_2 \cdots t_n$ of length $n \in \mathbb{N}$ uniquely corresponds to a finite transition system $TS(w) = (\{0, \ldots, n\}, \{(i-1, t_i, i) \mid 0 < i \leq n \wedge t_i \in T\}, T, 0)$.

Petri nets

An *initially marked (free labelled) Petri net* is denoted as $N = (P, T, F, M_0)$ where P is a finite set of places, T is a finite set of transitions, F is the flow function $F\colon ((P \times T) \cup (T \times P)) \to \mathbb{N}$ specifying the arc weights, and M_0 is the initial marking (where a marking is a mapping $M\colon P \to \mathbb{N}$, indicating the number of tokens in each place). A transition $t \in T$ is enabled at a marking M, denoted by $M[t\rangle$, if $\forall p \in P\colon M(p) \geq F(p,t)$. The firing of t at marking M leads to M', denoted by $M[t\rangle M'$, if $M[t\rangle$ and $M'(p) = M(p) - F(p,t) + F(t,p)$ for every $p \in P$. This can be naturally extended to $M[\sigma\rangle M'$ for sequences $\sigma \in T^*$, and $[M\rangle$ denotes the set of all markings reachable from M. The reachability graph $RG(N)$ of a bounded (such that the number of tokens in each place does not exceed a certain finite number) Petri net N is the labelled transition system with the set of vertices $[M_0\rangle$, labels set T, set of edges $\{(M, t, M') \mid M, M' \in [M_0\rangle \wedge M[t\rangle M'\}$, and initial state M_0,. If a labelled transition system TS is isomorphic to the reachability graph of a Petri net N, we say that N *PN-solves* (or simply *solves*) TS, and that TS is *synthesisable* to N. We say that N solves a word w if it solves $TS(w)$. A word w is then called *solvable*, otherwise it is called *unsolvable*.

Solvability

Theory of regions constitutes the most common tool for proving solvability of labelled transition systems. Let $(S, T, \to, s_0)$ be an lts and $N = (P, T, F, M_0)$ be a Petri net, which we hope to synthesise. The synthesis comprises solving systems of linear inequalities in integer numbers. Those inequalities guaranty satisfiability of the following properties:

State separation property (*ssp* in short)
For every pair $s, s' \in S$ of distinct states ($s \neq s'$) there exists a place $p \in P$ such that $M(p) \neq M'(p)$ for markings $M, M' \in [M_0\rangle$ corresponding to s and s'.

Event/state separation property (*essp* in short)
For every state-transition pair $s \in S$ and $t \in T$ with $\neg(s[t\rangle)$ there exists a place $p \in P$ such that $M(p) < F(p,t)$ for the marking $M \in [M_0\rangle$ corresponding to s.

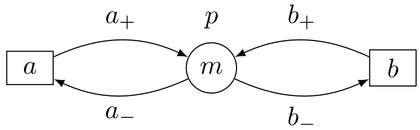


Figure 1. A general form of a place p containing initially m tokens and preventing a transition (a or b) to satisfy essp.

Note that if the lts is defined by a word w then the state separation property is easy to satisfy by introducing a counter place. On the other hand, satisfiability of event/state separation property, for every state-transition pair $s \in S$ and $t \in T$ with $\neg(s[t\rangle)$, requires a place preventing t at s. In the case of binary word $w \in \{a, b\}^*$ such a place $p \in P$ is of the form depicted in figure 2.

The labelled transition systems TS_1 and TS_2 depicted in figure 2 correspond to the words *aabba* and *abbaa*, respectively. The former is PN-solvable, since the reachability

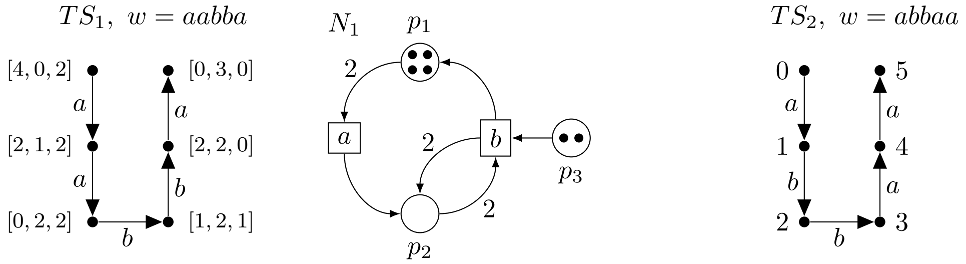


Figure 2. N_1 solves TS_1. No solution of TS_2 exists.

graph of N_1 is isomorphic to TS_1, while the latter contains an unsolvable event/state separation problem represented by event a and state 2 (see [2] for detailed explanation). Note that word *abbaa*, isomorphic to TS_2, is the shortest binary word (modulo swapping a/b) which is not PN-solvable. However, its reverse (*aabba*) is solvable.

Minimal unsolvable words
If a word w is PN-solvable, then all of its subwords w' are. To see this, let the Petri net solving w be executed up to the state before w', take this as the new initial marking, and add a pre-place with $\#_a(w')$ tokens to a and a pre-place with $\#_b(w')$ tokens to b. Thus, the unsolvability of any proper subword of w entails the unsolvability of w. For this reason, the notion of a *minimal unsolvable word* (*muw* in short) is well-defined, namely, as an unsolvable word all of which proper subwords are solvable. A complete list of minimal unsolvable words up to length 110 can be found, amongst some other lists, in [11].

3 Structural classification of minimal unsolvable words

In [2,4] some properties of solvable and of unsolvable words have already been described. In this section we shall indicate some important restrictions which grant all possible shapes of minimal unsolvable words.

Basing on [2] we can state the following proposition which provides a sufficient condition for unsolvability:

Proposition 1. SUFFICIENT CONDITION FOR UNSOLVABILITY *If a word over* $\{a, b\}$ *has a subword of the form* (1)*, then it is not PN-solvable.*

$$\boxed{(\ a\ b\ \alpha\ )\ b^*\ (\ b\ a\ \alpha\ )^+\ a\ , \qquad \textit{with}\ \alpha \in T^*} \tag{1}$$

Further in this paper we show that it is also a necessary condition.

Remark: Let us notice that for an arbitrary α the language described by the expression $(ab\alpha)b^*(ba\alpha)^+a$ is not regular, not even context free.

It can be shown ([3]) that, up to swapping a/b, all minimal unsolvable words match one of the following three general patterns:

$$\boxed{\begin{array}{l} ab^{x+k}ab^xa, \text{ with } x > 0,\ k > 2 \quad \text{or} \\ ab^{x+2}(ab^{x+1})^*ab^xa, \text{ with } x > 0 \quad \text{or} \\ ab^{x_1}ab^{x_2}a \cdots ab^{x_n}a, \text{ with } x_1 = x + 1,\ x_n = x,\ x_i \in \{x, x+1\} \text{ for } x > 0,\ n \geq 3 \end{array}} \tag{2}$$

$$\boxed{\begin{array}{l} bab^x(ab^{x+1})^*ab^{x+2}, \text{ with } x > 0 \quad \text{ or} \\ bab^{x_2}ab^{x_3}a\cdots ab^{x_n}, \text{ with } x_2 = x,\ x_n = x+1,\ x_i \in \{x, x+1\} \text{ for } x > 0,\ n \geq 3 \end{array}} \tag{3}$$

$$\boxed{ab^xaa, \text{ with } x > 2 \quad \text{ or } \quad abb(ab)^kaa, \text{ with } k \geq 0} \tag{4}$$

Remark: Let us notice that words of the form (3) start and end with b, while the other start and end with a. For some technical purpose, let us concentrate on words containing not less b's than a's. In the case of equal numbers of a's and b's we concentrate on words starting with a.

Note that both last forms of patterns (2) and (3) do not satisfy (1). In order to prove the necessity of the condition from proposition 1 we restrict them even more, obtaining as a side effect complete characterisation and the compatibility with (1). Moreover, the sets of words generated by all the patterns listed above are mutually disjoint. In the following section we divide them into classes of extendable and non-extendable words.

4 Generative nature of minimal unsolvable binary words

In this section we provide a complete characterisation of minimal unsolvable binary words. The general idea is to split the whole set into two classes: extendable (which are origins for more complex minimal unsolvable words) and non-extendable (which might be also seen as origins of more complex unsolvable, but not minimal, binary words). In the former class we distinguish the simplest extendable muw's, i.e. the words in which the factor α from (1) is of the form a^i or b^i. Such words are called base extendable. After introducing the class of base extendable words, we provide an extension operation based on simple morhisms, which are prefix codes. The code nature is used in subsequent section, where we define the converse operation, called compression.

4.1 Base extendable and non-extendable words

The following definitions must be understood modulo swapping a/b.

Definition 2. Base extendable words

A word $u \in \{a, b\}^*$ is called *base extendable* if it is of the form

$abw(baw)^ka$ with $w = b^j,\ j > 0,\ k \geq 1,$ or

$baw(abw)^kb$ with $w = b^j,\ j \geq 0,\ k \geq 1.$

The class of base extendable words is denoted by $\mathcal{BE}$. □ 2

Definition 3. Non-extendable words

A word $u \in \{a, b\}^*$ is called *non-extendable* if it is of the form

$abb^jb^kbab^ja$ with $j \geq 0,\ k \geq 1.$

The class of all non-extendable words is denoted by $\mathcal{NE}$. □ 3

We now establish that all words from classes $\mathcal{BE}$ and $\mathcal{NE}$ are minimal unsolvable.

Lemma 4. Minimal unsolvability of base extendable and non-extendable words *If w belongs to class $\mathcal{BE}$ or $\mathcal{NE}$, then it is unsolvable and minimal with that property.*

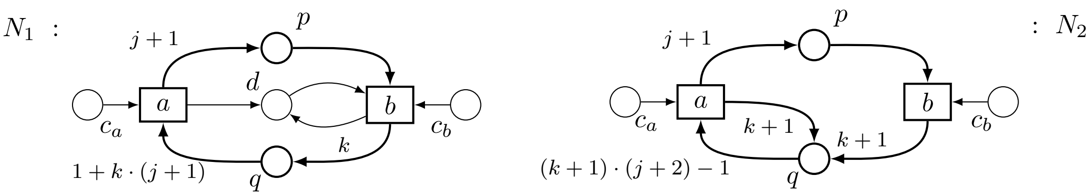
Proof: Let us notice that a word w is a muw if and only if w is unsolvable and every proper prefix and every proper suffix of w is solvable. Every word w from $\mathcal{BE} \cup \mathcal{NE}$ is of the form (1), hence unsolvable. We shall prove the minimality of w by indicating Petri nets solving its proper prefix and suffix.

CASE 1 (base extendable words):

(a) $w = abb^j(bab^j)^k a$

Consider first an arbitrary (modulo swapping a/b) base extendable word of the form $w = abb^j(bab^j)^k a$ with $j \geq 0$ and $k \geq 1$. This form satisfies (1) with $\alpha = b^j$, the star * being repeated zero times, and the plus $^+$ being repeated k times. Due to proposition 1, all binary words of this form are unsolvable.

The maximal proper prefix $abb^j(bab^j)^k$ of this word can be solved by Petri net N_1 in figure 4.1. Place q in this net enables the initial a, and then disables it unless b has been fired $j+2$ times. After the execution of block bb^jb there are $k-1$ tokens more than a needs to fire on place q. These surplus tokens allow a to be fired after each sequence b^jb, but not earlier. Place p has initially 1 token on it, which is necessary to execute block bb^jb after the first a, and this place has only $j+1$ tokens after each next a, preventing b at states where a must occur. Places d and c_b prevent undesirable occurrences of b at the very beginning and at the very end of the prefix, respectively.



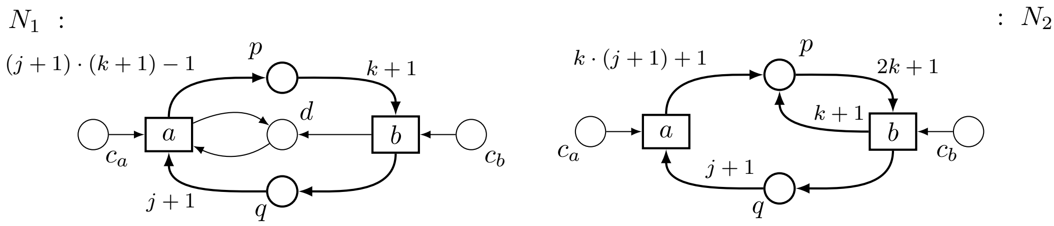
$$M \begin{pmatrix} p \\ q \\ d \\ c_a \\ c_b \end{pmatrix} = \begin{pmatrix} 1 \\ 1 + k \cdot (j+1) \\ 0 \\ k+1 \\ (j+1)(k+1) \end{pmatrix} \qquad M \begin{pmatrix} p \\ q \\ c_a \\ c_b \end{pmatrix} = \begin{pmatrix} j+2 \\ 0 \\ k+1 \\ (k+1)(j+1) \end{pmatrix}$$

Figure 3. N_1 solves the prefix $abb^j(bab^j)^k$. N_2 solves the suffix $bb^j(bab^j)^k a$.

For the general form of maximal proper suffix $bb^j(bab^j)^k a$ of w, one can consider Petri net N_2 on the right-hand side of figure 4.1 as a possible solution. Indeed, place q prevents premature occurrences of a in the first block bb^jb, and enables a only after this and each next block b^jb. Doing so, it collects one additional token after each b^jb, which allows this place to enable the very last a after sequence b^j. The initial marking allows to execute the sequence bb^jb at the beginning, and at most $j+1$ b's in a row after that, thanks to place p. Place c_b restricts the total number of b's allowing only block b^j at the end. Thus we deduce that any word of the form $abb^j(bab^j)^k a$ with $j > 0$ and $k \geq 1$ is a muw.

(b) $w = bab^j(abb^j)^k b$

We can similarly examine arbitrary (modulo swapping a/b) base extendable word of another form $w = bab^j(abb^j)^k b$ with $j \geq 0$ and $k \geq 1$. The word w satisfies (1) with $\alpha = b^j$, the star * being repeated zero times, the plus $^+$ being repeated k times, and a and b swapped. Due to proposition 1, all binary words of this form are unsolvable. Petri nets N_1 and N_2 in figure 4 are possible solutions for maximal proper prefix and for maximal proper suffix of w, respectively.



$$M\begin{pmatrix} p \\ q \\ d \\ c_a \\ c_b \end{pmatrix} = \begin{pmatrix} k+1 \\ j+1 \\ 0 \\ k+1 \\ (k+1)(j+1) \end{pmatrix} \qquad M\begin{pmatrix} p \\ q \\ c_a \\ c_b \end{pmatrix} = \begin{pmatrix} 0 \\ j+2 \\ k+1 \\ (k+1)(j+1) \end{pmatrix}$$

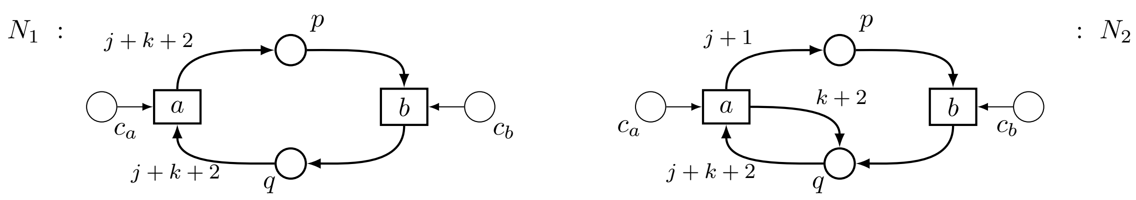
Figure 4. N_1 solves the prefix $bab^j(abb^j)^k$. N_2 solves the suffix $ab^j(abb^j)^k b$.

CASE 2 (non-extendable words):

We now demonstrate that any (modulo swapping a/b) binary word of the form $w = abb^j b^k bab^j a$ with $j \geq 0$ and $k \geq 1$ from class $\mathcal{NE}$ is minimal unsolvable. The word w satisfies (1) with $\alpha = b^j$, the star * being repeated k times, and the plus $^+$ being repeated only once. Due to proposition 1, w is unsolvable. To show minimality of w, we provide Petri nets N_1 and N_2 (see figure 4.1) solving its maximal proper prefix and maximal proper suffix, respectively. $\square$ 4

Remark (*On special structure of Petri nets which solve prefixes and suffixes*):
Petri net N_1 in figure 4.1, which solves maximal proper prefix $abb^j(bab^j)^k$ of word $w = abb^j(bab^j)^k a$ from class $\mathcal{BE}$, has a special structure. Place d serves for preventing undesirable b in the very beginning of w, and places c_a and c_b restrict the total number of a's and b's, correspondingly. So, the internal structure of the word, being executed by N_1, is determined by two places p and q, which prevent b and a, respectively, whenever it is necessary. In what follows, we will call the part of N_1 consisting of these two places (and transitions) the *core part* . So, Petri net N_2 if figure 4.1 has the core part made of places p and q. Similarly, such parts are formed by places p and q for both nets in figure 4 as well as both nets in figure 4.1. In future consideration we shall sometimes concentrate only on such core parts, as the other necessary places may be easily added and does not influence the main behaviour of the nets.

Example 5. Let us consider a word $w = abbbaba$, which is of the form (1), with $\alpha = b$, the star * being repeated zero times, and the plus $^+$ being repeated just once. By definition 2, w is a base extendable word with $j = 1$ and $k = 1$. The word w is unsolvable (by proposition 1) and minimal with that property. We show the minimality by introducing Petri nets solving a proper prefix $abbbab$ and a proper



$$M\begin{pmatrix} p \\ q \\ c_a \\ c_b \end{pmatrix} = \begin{pmatrix} 0 \\ j+k+2 \\ 2 \\ 2\cdot(j+1)+k \end{pmatrix} \qquad M\begin{pmatrix} p \\ q \\ c_a \\ c_b \end{pmatrix} = \begin{pmatrix} j+k+2 \\ 0 \\ 2 \\ 2\cdot(j+1)+k \end{pmatrix}$$

Figure 5. N_1 solves the prefix $abb^jb^kbab^j$. N_2 solves the suffix $bb^jb^kbab^ja$.

suffix *bbbaba* of w. Those Petri nets, constructed on the basis of the proof of lemma 4, are depicted in figure 5.

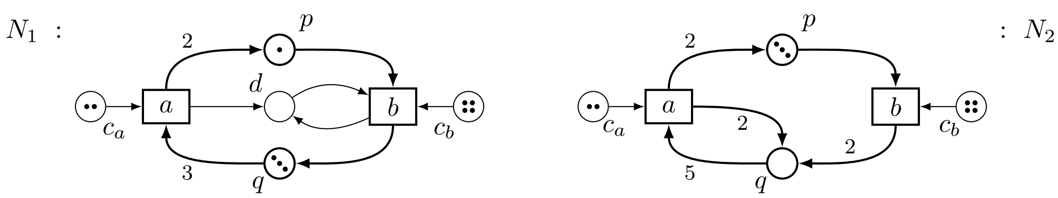


Figure 6. N_1 solves the prefix *abbbab*. N_2 solves the suffix *bbbaba*.

Notice that both Petri nets contain core parts consisting of places p and q, which are responsible for the required behaviour of the nets, as well as auxiliary places – a delay place d and counter places c_a and c_b.

4.2 Extension operation and extendable words

Let us now explain how some minimal unsolvable words can be obtained from other minimal unsolvable words. For this purpose we use the following notion of *extension operation*:

Definition 6. Extension operation

For a word $u = xwx$ ($w \in \{a,b\}^*$, $x \in \{a,b\}$) an extension operation E is defined as follows:

$$E(awa) = \bigcup_{i=1}^{\infty} \left\{abM_{a,i}(w)a^{i+1},\ aM_{b,i}(wa)\right\},$$
$$E(bwb) = \bigcup_{i=1}^{\infty} \left\{baM_{b,i}(w)b^{i+1},\ bM_{a,i}(wb)\right\},$$

where $M_{a,i}$ and $M_{b,i}$ are morphisms defined as follows

$$M_{a,i} = \begin{cases} a \mapsto a^{i+1}b \\ b \mapsto a^ib \end{cases} \qquad \text{and} \qquad M_{b,i} = \begin{cases} a \mapsto b^ia \\ b \mapsto b^{i+1}a \end{cases}.$$

$\square$ 6

In what follows, for a given $w \in \{a, b\}^*$, we shall call $u \in E(w)$ an *extension* of w.

We are now ready to define the class of *extendable words.*

Definition 7. (Derivative) extendable words

For a word $w \in \{a, b\}^*$

1. if $w \in E(v)$ for some base extendable v, then w is (derivative) extendable,
2. if $w \in E(v)$ for some extendable v, then w is (derivative) extendable,
3. there are no other (derivative) extendable words.

The class of all (derivative) extendable words is denoted by $\mathcal{E}$. In what follows we call them simply *extendable words.* □ 7

The following lemmata constitute unsolvability and minimality of all extendable words.

Lemma 8. Unsolvability of extendable words *If $u \in \{a, b\}^*$ is of the form $abv(bav)^k a$ ($k > 0$), then every $w \in E(u)$ is unsolvable.*

Proof: It follows directly by definitions 2 and 7, and proposition 1. □ 8

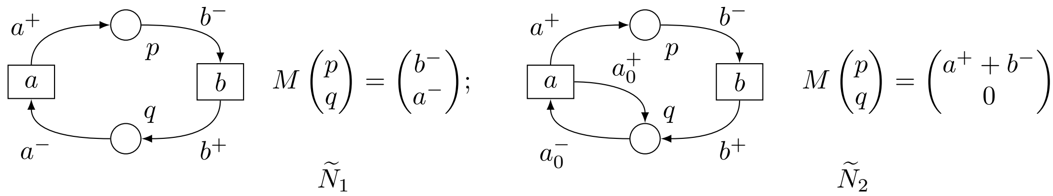


Figure 7. Core parts of Petri nets: $\widetilde{N}_1$ for a net solving prefix, $\widetilde{N}_2$ for a net solving suffix.

Transformations of core part w.r.t. morphisms

As it has been demonstrated above, for every base extendable word w there are Petri nets N_1 and N_2, which solve maximal proper prefix w_1 and maximal proper suffix w_2 of w, respectively. Recall that the nets N_1 and N_2 have a special structure: so called "core" parts $\widetilde{N}_1$ and $\widetilde{N}_2$ (general patterns of $\widetilde{N}_1$ and $\widetilde{N}_2$ are depicted in figure 4.2) determining internal order of firings of a's and b's during execution of w_1 and w_2, while the remaining parts of N_1 and N_2 take responsibility for correct implementation of the beginnings and the ends of w_1 and w_2. Applying operation E to w, one can easily obtain new minimal unsolvable word w'. Moreover, applying appropriate transformation (which is determined by the particular morphism that has been used to gain w' from w) to $\widetilde{N}_1$ or to $\widetilde{N}_2$, one derives new core part $\widetilde{N}'_1$ or $\widetilde{N}'_2$, which correctly implements the internal structure of maximal proper prefix w'_1 or maximal proper suffix w'_2 of w', respectively. In table 1 the correspondence between morphisms from definition 6 and such transformations of nets is provided for general forms of $\widetilde{N}_1$ and $\widetilde{N}_2$. This fact is confirmed throughout the proof of the following lemma

Lemma 9. Minimality of extendable words *If $w \in \mathcal{E}$, then w is minimal unsolvable.*

	$M_{a,i}$	$M_{b,i}$
	$a^+ \longmapsto a^+ + b^-$	$a^+ \longmapsto a^+ + i \cdot (a^+ + b^-)$
	$b^- \longmapsto b^- + i \cdot (a^+ + b^-)$	$b^- \longmapsto a^+ + b^-$
	$b^+ \longmapsto b^+ + i \cdot (a^- + b^+)$	$b^+ \longmapsto a^- + b^+$
$\widetilde{N}_1$	$a^- \longmapsto a^- + b^+$	$a^- \longmapsto a^- + i \cdot (a^- + b^+)$
	$M(p) \longmapsto b^- + i \cdot (a^+ + b^-)$	$M(p) \longmapsto a^+ + b^-$
	$M(q) \longmapsto a^- + b^+$	$M(q) \longmapsto a^- + i \cdot (a^- + b^+)$
	$a^+ \longmapsto a^+ + b^-$	$a^+ \longmapsto a^+ + i \cdot (a^+ + b^-)$
	$b^- \longmapsto b^- + i \cdot (a^+ + b^-)$	$b^- \longmapsto a^+ + b^-$
	$b^+ \longmapsto b^+ + i \cdot (a_0^- + b^+ - a_0^+)$	$b^+ \longmapsto b^+ + a_0^- - a_0^+$
$\widetilde{N}_2$	$a_0^- \longmapsto a_0^- + b^+$	$a_0^- \longmapsto a_0^- + i \cdot (b^+ + a_0^- - a_0^+)$
	$a_0^+ \longmapsto a_0^+$	$a_0^+ \longmapsto a_0^+$
	$M(p) \longmapsto b^- + (i+1) \cdot (a^+ + b^-)$	$M(p) \longmapsto a^+ + (i+1) \cdot (a^+ + b^-)$
	$M(q) \longmapsto 0$	$M(q) \longmapsto 0$

Table 1. Correspondence between morphisms and transformations

Proof: *(Sketch)* Unsolvability follows from lemma 8. By definition 7, for every $w \in \mathcal{E}$ there is a sequence $w_0, w_1, \ldots, w_r$ such that $w_0 \in \mathcal{BE}$, $w_j \in \mathcal{E}$ and $w_j \in E(w_{j-1})$ for $1 \leq j \leq r$, and $w_r = w$. With induction on r and using table 1, one can construct the core parts of Petri nets, solving maximal proper prefix and suffix of w. Additional parts of these nets can be implemented in an uncompliceted way. ☐ 9

Let us note that the extension operation being applied to an extendable word, produces another extendable word which is unsolvable and minimal. On the other hand, from a non-extendable word this operation derives unsolvable but not minimal words.

Lemma 10. Unsolvability of extensions of non-extendable words
If $w \in \mathcal{NE}$, then extension $u \in E(w)$ is unsolvable but not minimal.

Proof:
Follows from definition 3 and 6, and decomposition of the result using proposition 1. ☐ 10

Example 11. Observe again the word $w = abbbaba$. From the previous considerations (see example 5) we know that this word is base extendable, and therefore is a muw. By the application of the extension operation, using the morphism $M_{a,1} = \begin{cases} a \mapsto aab \\ b \mapsto ab \end{cases}$ we obtain word $w_{a,1} = ab\,ababa\,ba\,ababa\,a$, which is of the form (1) with $\alpha = ababa$, the star * being repeated zero times, and the plus $^+$ being repeated just once, hence – by proposition 1 – unsolvable. On the basis of the Petri nets of figure 5, and according to table 1 we construct Petri nets (depicted in figure 11) solving the maximal proper prefix *abababababaababa* and the maximal proper suffix *babababaababaa* of $w_{a,1}$. Thus, $w_{a,1}$ is a minimal unsolvable word.

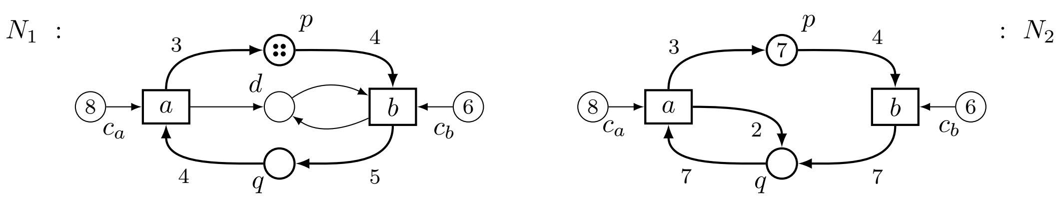


Figure 8. N_1 solves the prefix *ababababaababa* and N_2 solves the suffix *bababaabababaa* of $w_{a,1} = ababababaababaa$.

5 Generation-based classification of minimal unsolvable binary words

Regard minimal unsolvable words w.r.t. the classification obtained earlier. All possible patterns from (2)–(4), can be distinguished into base extendable

$ab(ba)^{k+1}a$, with $k \geq 0$, for the second pattern from (4),
$abb^x(bab^x)^k a$, with $x > 0, k > 0$, for the second pattern from (2),
$bab^x(abb^x)^k b$, with $x > 0, k > 0$, for the first pattern from (3),

non-extendable

$abb^{x-1}baa$, with $x > 2$ for the first pattern from (4),
$abb^x b^{k-1} bab^x a$, with $x > 0$, $k > 2$ for the first pattern from (2),

and the rest, which we call $\mathcal{C}$ (*compressible*)

$ab^{x_1}ab^{x_2}a\cdots ab^{x_n}a$, with $x_1 = x+1$, $x_n = x$, $x_i \in \{x, x+1\}, x > 0$, $n \geq 3$, for the third pattern from (2),
$bab^{x_2}ab^{x_3}a\cdots ab^{x_n}$, with $x_2 = x$, $x_n = x+1$, $x_i \in \{x, x+1\}, x > 0$, $n \geq 3$, for the second pattern from (3).

From this classification we derive that the class of all minimal unsolvable words $\mathcal{MUW} = \mathcal{BE} \cup \mathcal{NE} \cup \mathcal{C}$, where $\mathcal{BE}$, $\mathcal{NE}$ and $\mathcal{C}$ are mutually disjoint classes. Note, that since all words from class $\mathcal{E}$ are unsolvable and minimal with that property, and $\mathcal{E}$ is disjoint with $\mathcal{BE}$ and $\mathcal{NE}$, we have $\mathcal{E} \subseteq \mathcal{C}$.

5.1 Morphic compression and reducibility

In the previous section we showed how to construct new minimal unsolvable words on the basis of extendable words. The purpose of this section is to introduce an inverse transformation,which allows to compress longer minimal unsolvable words into shorter ones.

Definition 12. COMPRESSION FUNCTION

For a word $v = xux$ ($u \in \{a,b\}^*$, $x \in \{a,b\}$) a *compression function* C is defined as follows :

$$\begin{aligned} C(abua^{i+1}) &= aM_{a,i}^{-1}(u)a, & C(baub^{i+1}) &= bM_{b,i}^{-1}(u)b, \\ C(auba) &= aM_{b,i}^{-1}(uba), & C(buab) &= bM_{a,i}^{-1}(uab), \end{aligned} \tag{5}$$

where $i \geq 1$ and $M_{a,i}^{-1}$, $M_{b,i}^{-1}$ are functions defined as follows:

$$M_{a,i}^{-1} : \begin{cases} a^{i+1}b & \mapsto a \\ a^i b & \mapsto b \end{cases} \qquad \text{and} \qquad M_{b,i}^{-1} : \begin{cases} b^i a & \mapsto a \\ b^{i+1} a & \mapsto b. \end{cases}$$

□ 12

It is easy to see that among all possible forms from the classification of minimal unsolvable words, function C can only be applied to patterns from class $\mathcal{C}$. Moreover, the form of a given word from $\mathcal{C}$ explicitly defines the particular function $M_{x,i}^{-1}$ which is used when applying C to the word. Let us also notice that since $\mathcal{E} \subseteq \mathcal{C}$, all words from class $\mathcal{E}$ are compressible with function C.

From definitions 6 and 12 it is clear that the morphisms $M_{x,i}$ are reciprocal to the functions $M_{x,i}^{-1}$ for $x \in \{a, b\}$, $i \geq 1$. The following lemma establishes that the extension operation E and the application of compression function C are complement to each other in the following sense.

Lemma 13. Compression and extension operations

1. *If $v \in \mathcal{BE} \cup \mathcal{E}$ and $u \in E(v)$, then $C(u) = v$;*
2. *If $u \in \mathcal{C}$ and $v = C(u)$, then $u \in E(v)$.*

Proof: Can be ascertained by consecutive application of extension and compression operations, according to definition 6 and 12. □ 13

5.2 Compression of a muw is an unsolvable word

By use of lemma 13, it can be shown that $\mathcal{C} \subseteq \mathcal{E}$, implying that classes of extendable and compressible words coincide. This fact completes the characterisation of all minimal unsolvable words regarding their generative nature, and allows us introduce one of the main results of the paper:

Theorem 14. Generative nature of minimal unsolvable binary words
Let w be a minimal Petri net unsolvable binary word. Then we have the following exclusive alternatives:

- *w is a non-extendable word ($w \in \mathcal{NE}$), or*
- *w is a base extendable word ($w \in \mathcal{BE}$), or*
- *w is an extendable word ($w \in \mathcal{E}$).*

Basing on theorem 14 and proofs of lemmata 4 and 8 we can formulate the following

Corollary 15 (The necessary condition for unsolvability).
If a word over $\{a, b\}$ is not PN-solvable, it has a subword of the form (1).

Generation of maximal partial solutions of minimal unsolvable words

In the last case of the alternative from theorem 14 (case $w \in \mathcal{E}$), applying function C to w consecutively, we can recover a sequence of minimal unsolvable words $w_0, w_1, \ldots, w_r$, such that $w_0 \in \mathcal{BE}$, $w_r = w$, $w_i \in \mathcal{E}$ and $w_{i-1} = C(w_i)$ for $1 \leq i \leq r$.

Moreover, starting from a word w_0, its maximal proper prefix and maximal proper suffix, and Petri nets solving them (in special forms, that have been provided in the paper), using appropriate transformations, we can derive Petri nets solving maximal proper prefix and maximal proper suffix of w_i for all $1 \le i \le r$. We now demonstrate this with the following example:

Example 16. Let us consider word $v = ba\,aabaaabaa\,ab\,aabaaabaa\,b$. It is unsolvable by proposition 1, because it is of the form $ba\alpha\, a^*\, (ab\alpha)^+\, b$ (which is exactly the form (1) – modulo swapping a/b) with $\alpha = aabaaabaa$, the star * being repeated zero times, and the plus $^+$ being repeated just once. Due to theorem 14, if v is minimal, then it belongs to one of the classes $\mathcal{BE}, \mathcal{NE}, \mathcal{E}$. Since it does not fit the patterns of classes $\mathcal{BE}, \mathcal{NE}$, we now aim to check whether $v \in \mathcal{E}$. In order to do this we compress v with function C. It can be easily seen that the word could be written in the form

$$v = b(aaab)(aaab)(aaab)(aab)(aaab)(aab),$$

hence we need to consider the function $M_{a,2}^{-1} : \begin{cases} aaab & \mapsto a \\ aab & \mapsto b \end{cases}$, and by the compression we obtain word $v_{a,2}^{-1} = baaabab$. Let us notice that $v_{a,2}^{-1}$ is dual to the word $w = abbbaba$ (see example 5), up to swapping a/b, hence it is a minimal unsolvable word. Function C cannot be applied to $w = C(v)$, which accord with the fact that $w \in \mathcal{BE}$.

Moreover, starting with the word $w = abbbaba$, together with Petri nets solving its proper prefix and suffix (see figure 5) and applying the morphism $M_{b,2} : \begin{cases} a & \mapsto bba \\ b & \mapsto bbba \end{cases}$ we obtain the word $w_{b,2} = ab\,bbabbbabb\,ba\,bbabbbabb\,a$ which is dual to v up to swapping a/b. By the previous considerations we can easily construct Petri nets solving the maximal proper prefix and the maximal proper suffix of $w_{b,2}$, hence, by swapping letters we can obtain Petri nets for a proper prefix and a proper suffix of v. Such nets are depicted in figure 16. Now we can state that the word v is not only unsolvable, but also minimal with that property.

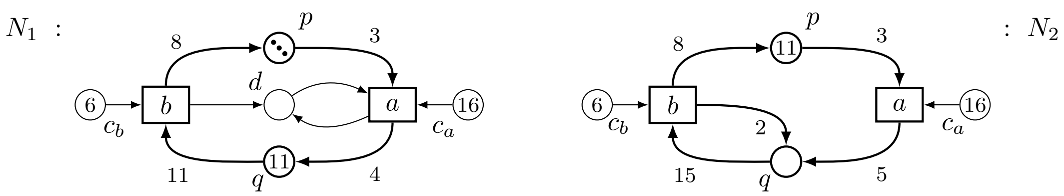


Figure 9. N_1 solves the prefix $baaabaaabaaabaabaaabaa$ and N_2 solves the suffix $aaabaaabaaabaabaaabaab$ of $v = baaabaaabaaabaabaaabaab$.

6 Algorithm for checking unsolvability

The classification of minimal unsolvable words presented in sections 3 and 4 leads to an efficient algorithm for verifying solvability/unsolvability of a binary word. By definition 3 all non-extendable words are of the form (Ia) ab^xab^ya or (Ib) ba^xba^yb, where $x > y+2$, $y \ge 0$, and by definition 2 and 6 all extendable words (including base extendable ones) are of the form (IIa) $abw(baw)^ka$ or (IIb) $baw(abw)^kb$, where $k \ge 1$ and $w \in \{a,b\}^*$.

Recall that a word $v \in \{a,b\}^*$ containing a minimal unsolvable word as a factor is also unsolvable. Moreover, due to theorem 14, v is unsolvable if it contains at least one of the patterns (Ia) (Ib), (IIa) or (IIb). Therefore, checking the solvability of a binary word can be reduced to a pattern-matching problem.

The algorithm described below takes a binary word v as an input and returns true if v is solvable and false otherwise (i.e. any of the above mentioned patterns was found inside v).

As the first step we search for the patterns (Ia) and (Ib). We scan the input word from left to right comparing the sizes of the two blocks of consecutive b's between any three consecutive occurrences of a and the sizes of the two blocks of consecutive a's between any three consecutive occurrences of b. This can be done in $O(n)$ time and $O(1)$ space.

The second step is to search for the patterns (IIa) and (IIb). It utilizes the Knuth-Morris-Pratt failure function called also the border table (see [7]). For any position i in v it contains the length of the longest factor u, which is at the same time a proper prefix and a proper suffix of $v[1..i]$. Such a factor is called a border of $v[1..i]$. For the relation between borders and periods of a word see for instance [8].

The search for the patterns (IIa) and (IIb) is performed as follows. For any possible pair of letters $v[i..i+1] = ab$ ($v[i..i+1] = ba$ respectively) we temporarily swap $v[i]$ with $v[i+1]$ and then build the border table for the suffix of v starting at position i. After discovering a repetition $v[i..j]$ (i.e. difference between j and the lenght of the border divides $j-i+1$) we check whether it is followed by a (b respectively) and report the occurrence of the pattern if needed.

The border table for a single suffix of the input word v can be constructed in $O(n)$ time and $O(n)$ space (see [7]). We have to process at most $O(n)$ suffixes of v, therefore the second step and the whole algorithm runs in $O(n^2)$ time and $O(n)$ space.

7 Conclusions and future work

In this paper we studied the class of binary words which can not be generated by any injectively-labelled Petri net, and which are minimal with that property. We examined in detail all possible shapes of such words. The presented classification of minimal unsolvable words results in the construction of a pattern-matching based algorithm for checking the solvability/unsolvability for binary words. The implementation could be found at [11]. Moreover, we introduced the extension and compression functions, which can be foundations of a fixed-point procedure for the generation of the set of all minimal unsolvable binary words. The non-extendable and base extendable words are defined by simple parametrized formulas (see definitions 3 and 2). Choosing all possible values of the parameters j and k we can generate all non-extendable and base extendable words of a given length. Then by using recursive calls of extension and compression function we can generate all extendable words of a given length.

It would be interesting to examine larger alphabets in the hope of finding analogous regularities. The present work can also be of interest in a wider context – a natural extension of this work would consist in analyzing more complex labelled transition systems in terms of their solvability, utilizing the presented results. For instance, for an unsolvable word w, we might find a net N whose reachability graph consists of only two maximal branches labelled by w and w', for some w'. Then we can deliberate over "approximate solvability" of w.

References

1. E. Badouel, L. Bernardinello, and P. Darondeau: *Petri Net Synthesis*, Texts in Theoretical Computer Science. An EATCS Series, Springer, 2015.
2. K. Barylska, E. Best, E. Erofeev, Ł. Mikulski, and M. Piątkowski: *On binary words being Petri net solvable*, in Proceedings ATAED 2015, vol. 1371, CEUR-WS.org, 2015, pp. 1–15.
3. K. Barylska, E. Erofeev, Ł. Mikulski, and M. Piątkowski: *Generating all minimal Petri net unsolvable binary words - full version*, 2016, `http://folco.mat.umk.pl/papers/generating-binary-muws.pdf`.
4. E. Best, E. Erofeev, U. Schlachter, and H. Wimmel: *Characterising petri net solvable binary words*, vol. 9698 of Lecture Notes in Computer Science, 2016, pp. 39–58.
5. B. Caillaud: 2002, `http://www.irisa.fr/s4/tools/synet`.
6. C. Câmpeanu, K. Salomaa, and S. Yu: *A formal study of practical regular expressions.* International Journal of Foundations of Computer Science, 14(6) 2003, pp. 1007–1018.
7. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, eds., *Introduction to algorithms*, MIT Press, third ed., 2009.
8. M. Crochemore, L. Ilie, and W. Rytter: *Repetitions in strings: Algorithms and combinatorics.* Theoretical Computer Science, 410(50) 2009, pp. 5227–5235.
9. M. Droste and R. M. Shortt: *From petri nets to automata with concurrency.* Applied Categorical Structures, 10(2) 2002, pp. 173–191.
10. J. L. Peterson: *Petri Net Theory and the Modelling of Systems*, Prentice-Hall, 1981.
11. M. Piątkowski et al.: 2015, `http://folco.mat.umk.pl/unsolvable-words`.
12. W. Reisig: *Understanding Petri Nets - Modeling Techniques, Analysis Methods, Case Studies*, Springer, 2013.
13. U. Schlachter et al.: 2013, `http://github.com/CvO-Theory/apt`.