JOURNAL OF COMPUTER AND SYSTEM SCIENCES 50, 53–63 (1995)

Decision Problems for Patterns*

TAO JIANG

Department of Computer Science, McMaster University, Hamilton, Ontario, Canada L8S 4K1

ARTO SALOMAA

Academy of Finland and Mathematics Department, University of Turku, 20500 Turku, Finland

KAI SALOMAA

Mathematics Department, University of Turku, 20500 Turku, Finland

AND

SHENG YU

Department of Computer Science, University of Western Ontario, London, Ontario, Canada N6A 5B7

Received September 1, 1993; revised March 23, 1994

We settle an open problem, the inclusion problem for pattern languages. This is the first known case where inclusion is undecidable for generative devices having a trivially decidable equivalence problem. The study of patterns goes back to the seminal work of Thue and is important also, for instance, in recent work concerning inductive inference and learning. Our results concern both erasing and nonerasing patterns.

1. INTRODUCTION: THE MAIN RESULT

Instead of an exhaustive definition for a language [7], it is sometimes better to give more leeway in the definition and try to find *patterns* common to all words in a sample set. Such an approach is especially appropriate if the sample set is growing, for instance, through some learning process. It may happen that several "equally good" patterns are found. For instance, consider the following finite sample

$$F = \{010100,\ 00100100,\ 01101100,\ 0001000100,\ 0111011100,\ 010110101100,\ 001010010100\}.$$

Each of the words in F is of the form $0x0x0^2$, where x ranges over all nonempty words of the alphabet $\Sigma = \{0, 1\}$. Similarly, all words in F are of the forms x^20^2, $0x^20$, x^2y^2, $0x0^2$, $0x10^2$ and, in addition, of the form $0x10x10^2$ if x ranges over all words of Σ, including the empty word λ. Our theory will concern only variables ranging over all words or all nonempty words of some alphabet. However, other types of variables are also possible such as x^R and x^P, the reverse of the word x and a permutation of the word x. Then each word in F is also of the form xx^P and $0x10x^R10^2$. The following inclusion diagram holds between the sets of words generated by the eight *patterns* we suggested for F.

It is clear that the patterns generating the smallest sets, in this case $0x10x10^2$ and $0x10x^R10^2$, are the best descriptions for F. If F is not static but rather a subsample of a bigger sample which we want to describe, then further decisions between possible patterns can be made by queries or just by generating new words from the bigger sample.

The study of patterns that are descriptive for a sample, as well as the study of *pattern languages*, in the sense understood in this paper was initiated by Angluin [1, 2]. The inclusion problem was mentioned at the end of [2] as the most important open problem in the area. As we will see, it is closely connected also with some other basic theoretical problems.

Trying to infer a pattern that is common to all words in a given sample such as F is a very typical instance of the process of *inductive inference*, that is, the process of inferring general rules from specific examples. The interrelation with the *theory of learning* is also obvious, especially if the sample is not fixed but is supplemented by new words, or, even better, if one may enquire whether or not some specified words belong to the set. For instance, either one of the queries 0^21110^2 or 01100110^3 would decide between the patterns $0x10^2$ and $0x0x0^2$. We refer to [3, 9, 15] for such interrela-

* The work reported here has been supported by the Natural Sciences and Engineering Research Council of Canada Grants OGP0041630 and OGP0046613 and Project 11281 of the Academy of Finland. Send correspondence to Sheng Yu.

0022-0000/95 $6.00

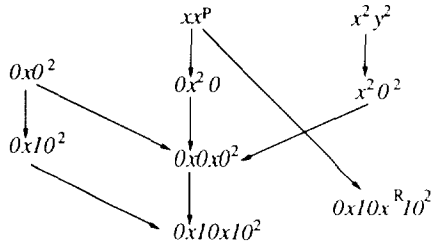

FIG. 1. Inclusion diagram.

tions and to [10] for some background information and interconnections with random numbers and Kolmogorov complexity.

Our main result can be stated very simply, without assuming any previous knowledge on the part of the reader. Our alphabet, if not explicitly specified, is the union $V \cup \Sigma$, where the letters of V are referred to as *variables* and those of Σ as *terminals*. A *pattern* is a word α over $V \cup \Sigma$. The language $L(\alpha)$ defined by the pattern α consists of all words obtained from α by leaving the terminals unchanged and substituting a terminal word for each variable x. The substitution has to be *uniform*: different occurrences of x have to be replaced by the same terminal word. In Angluin's original approach [1, 2] the variables have to be replaced always by *nonempty* words. Such patterns will be referred to as *nonerasing*, or *NE-patterns*, in the sequel. The situation is essentially different if the empty word is allowed in the substitutions (that still have to be uniform). The study of such *erasing patterns*, or *E-patterns*, was initiated in [10].

A little reflection will show that two NE-patterns define the same language if and only if they are identical, up to an eventual renaming of the variables [2]. Hence, the decidability of the *equivalence problem* for nonerasing pattern languages is trivial.

In view of the simplicity of the equivalence problem, our main result is very surprising: the *inclusion problem* is undecidable for nonerasing pattern languages. It seems also that people who have worked with this problem (for instance, see [2]) would have expected the opposite result.

We show that the inclusion problem is undecidable for E-patterns. The decidability status of the equivalence problem for E-patterns remains open. We conjecture that it is decidable. In Section 7 we formulate strong necessary conditions for the equivalence of two E-patterns.

2. BASIC NOTATIONS

Consider two disjoint alphabets Σ (the alphabet of *terminals*) and V (the alphabet of *variables*). Words over $\Sigma \cup V$ are referred to as *patterns*. The length of a word α is denoted $|\alpha|$. Naturally, the length of the empty word λ, $|\lambda|$, is zero. The number of occurrences of $a \in \Sigma \cup V$ in α is denoted $|\alpha|_a$. The set of variables of V appearing in α is denoted $\mathrm{var}(\alpha)$. For an arbitrary set S, the cardinality of S is denoted $\mathrm{card}(S)$.

Let Σ and V be given, and let $H_{\Sigma,V}$ be the set of morphisms h: $(\Sigma \cup V)^* \to (\Sigma \cup V)^*$. The language generated by an E-pattern $\alpha \in (\Sigma \cup V)^*$ is defined as

$$L_{\mathrm{E},\Sigma}(\alpha) = \{w \in \Sigma^* \mid w = h(\alpha) \text{ for some } h \in H_{\Sigma,V} \text{ such that } h(a) = a \text{ for each } a \in \Sigma\}.$$

The language generated by an NE-pattern $\alpha \in (\Sigma \cup V)^*$ is

$$L_{\mathrm{NE},\Sigma}(\alpha) = \{w \in \Sigma^* \mid w = h(\alpha) \text{ for some } \lambda\text{-free } h \in H_{\Sigma,V} \text{ such that } h(a) = a \text{ for each } a \in \Sigma\}.$$

If Σ is understood, we use also the notations $L_{\mathrm{E}}(\alpha)$ and $L_{\mathrm{NE}}(\alpha)$. A morphism $h \in H_{\Sigma,V}$ such that $h(a) = a$ for each $a \in \Sigma$ is usually defined just as a mapping $V \to (\Sigma \cup V)^*$ in the following.

3. WHY IS THE INCLUSION PROBLEM HARD

The inclusion $L_{\mathrm{NE}}(\alpha) \subseteq L_{\mathrm{NE}}(\beta)$ can hold for two patterns α and β without α and β having seemingly any connection. The inclusion holds if $\alpha = h(\beta)$, for some nonerasing morphism h that keeps the terminals fixed, but the existence of such a morphism is by no means necessary for the inclusion to hold. For instance, if $\Sigma = \{0, 1\}$ and α is an arbitrary pattern with length at least 6, then $L_{\mathrm{NE}}(\alpha)$ is included in $L_{\mathrm{NE}}(xyyz)$. The pattern α is not necessarily a morphic image of $xyyz$; the position of the square yy varies in different words of $L_{\mathrm{NE}}(\alpha)$.

Thue's classical results, [16], can be expressed in terms of noninclusion among NE-pattern languages as follows:

If Σ consists of two letters then, for all n, $L_{\mathrm{NE}}(x_1x_2\cdots x_n)$ contains words not belonging to $L_{\mathrm{NE}}(xyyyz)$. If Σ contains at least three letters then, for all n, $L_{\mathrm{NE}}(x_1x_2\cdots x_n)$ contains words not belonging to $L_{\mathrm{NE}}(xyyz)$.

The above results concerning squares and cubes have been extended to arbitrary terminal-free patterns in [5]. We present the definitions in our terminology.

A terminal-free pattern α is termed *unavoidable* (on an alphabet Σ) iff, for some n,

$$L_{\mathrm{NE}}(x_1x_2\cdots x_n) \subseteq L_{\mathrm{NE}}(x\alpha z),$$

where the x_i's are distinct variables and x and z are variables not occurring in α. Otherwise, α is *avoidable*.

Thus, yy is unavoidable on two letters but avoidable on three letters. The pattern yyy is avoidable also on two letters.

The paper [5] gives a recursive characterization of unavoidable terminal-free patterns. A rather tricky example is the pattern

$$\alpha = xyxzx'yxy'xyx'zx'yx',$$

which is unavoidable for a suitable Σ. Given a pattern α that is unavoidable on Σ, it is of interest to find the *smallest* n such that every word of length at least n possesses a subword of pattern α. The following result, which is obvious from the definitions, shows the interconnection with the inclusion problem:

Any algorihm for solving the inclusion problem for NE-pattern languages can be converted into an algorithm for computing the smallest n such that a given unavoidable pattern cannot be avoided on words of length at least n.

We mention, finally, that if we allow the use of variables of other types, such as the variables x^P considered in Section 1, then also problems of different kinds can be reduced to the inclusion problem. For instance, the celebrated new result of [11] can be expressed as follows:

If Σ contains at least four letters then, for all n, $L_{NE}(x_1x_2\cdots x_n)$ contains words not belonging to $L_{NE}(xyy^Pz)$.

The above result is not valid for smaller alphabets. Indeed, if Σ consists of three letters and α is a pattern of length at least 10, then

$$L_{NE}(\alpha) \subseteq L_{NE}(xyy^Pz).$$

4. EXCURSION: DECIDABILITY OF EQUIVALENCE VERSUS DECIDABILITY OF INCLUSION

It has been a challenging task in language theory to find natural examples of language families with a decidable equivalence problem and an undecidable inclusion problem. Of course, reverse examples are not possible. This is also one way to approach the borderline between decidable and undecidable. We mention here four such examples. In the first case, the decidability of one of the problems is still open.

(i) The inclusion problem for deterministic context-free languages is undecidable. The decidability status of the equivalence problem is open [7].

(ii) The inclusion problem for simple languages (s-languages) is undecidable but the equivalence problem is decidable [7].

(iii) The inclusion problem for languages accepted by finite deterministic multi-tape automata is undecidable but the equivalence problem is decidable [6].

(iv) The inclusion problem for nonerasing pattern languages is undecidable but the equivalence problem is decidable.

Assume that for some language family (a) the inclusion problem is undecidable, whereas (b) the equivalence problem is decidable. Since we are crossing here the borderline of decidability, it is intuitively clear that if one of the proofs for (a) and (b) is easy, the other one is difficult. In (iii) the inclusion part is easy and, hence, the equivalence part is difficult. The same holds true as regards (i). (In fact, the equivalence part will be difficult also if undecidability holds, because this part will bring us much closer to the borderline.) In (ii) the situation is rather balanced; neither part is very difficult. Among these examples, (iv) is the only one where the equivalence part is easy.

5. THE INCLUSION PROBLEM FOR E-PATTERN LANGUAGES

We show that the inclusion problem for E-pattern languages is undecidable by reducing to this problem the question whether a nondeterministic two-counter automaton without input has an accepting computation. A configuration of a two-counter automaton can be represented by two unary strings and the internal state. The fact that the stacks can be restricted to be over a unary alphabet is essential for our technical constructions. A nondeterministic 2-counter automaton without input, cf. [8], is denoted as a quintuple.

$$M = (Q, q_0, Q_F, 0, \delta), \tag{1}$$

where Q is the finite set of states, $q_0 \in Q$ is the initial state, $Q_F \subseteq Q$ is the set of final states, $0 \notin Q$ is the single symbol of the stack alphabet, and

$$\delta \subseteq Q \times \{0, 1\}^2 \times Q \times \{-1, 0, 1\}^2$$

is the transition relation of the automaton. The set of configurations of M is $\text{conf}(M) = 0^*Q0^*$, where in a configuration u_1qu_2; $u_1, u_2 \in 0^*$ and $q \in Q$, the words u_i, $i = 1, 2$, denote the contents of the counters and q denotes the current state of the finite control. The relation $\Rightarrow_M$ on the set of configurations is determined by the transition relation δ as follows: Let $q_i \in Q$ and $u_i, v_i \in 0^*$, $i = 1, 2$. Then

$$u_1q_1u_2 \Rightarrow_M v_1q_2v_2$$

iff there exists $(q_1, x_1, x_2, q_2, y_1, y_2) \in \delta$, $x_1, x_2 \in \{0, 1\}$, y_1, $y_2 \in \{-1, 0, 1\}$, such that for $i = 1, 2$,

$$x_i = \begin{cases} 0, & \text{if } u_i = \lambda, \\ 1, & \text{if } u_i \neq \lambda, \end{cases}$$

and

$$|v_i| = |u_i| + y_i.$$

We assume that if $x_i = 0$ then $y_i \geqslant 0$, $i = 1, 2$, i.e., the contents of a counter cannot be made negative.

Let # be a new symbol not appearing in $Q \cup \{0\}$. An accepting computation of M is a word $W_c \in (Q \cup \{0, \#\})^*$ that can be written as

$$W_c = \# C_1 \# C_2 \# \cdots \# C_m \#, \tag{2}$$

where $m \geq 1$, $C_1 = q_0$, $C_m = u_1 q u_2$, $q \in Q_F$, $u_1, u_2 \in 0^*$, $C_i \in \text{conf}(M)$, $i = 2, ..., m-1$, and $C_i \Rightarrow_M C_{i+1}$, $i = 1, ..., m-1$. It is well known that the emptiness problem for deterministic 2-counter automata is undecidable; cf., e.g., [4, 8, 13]. Thus it is also clearly undecidable whether a nondeterministic 2-counter automaton without input has an accepting computation.

If $u, v \in \Sigma^*$ and u is a subword of v, this is denoted as $u \leqslant_{\text{sub}} v$. Now we can state the main result of this section.

THEOREM 5.1. *Given a terminal alphabet Σ, a set of variables V, and two arbitrary patterns $\beta_1, \beta_2 \in (\Sigma \cup V)^*$, it is in general undecidable whether*

$$L_{E,\Sigma}(\beta_1) \subseteq L_{E,\Sigma}(\beta_2).$$

Proof. Let $M = (Q, q_0, Q_F, 0, \delta)$ be an arbitrary 2-counter automaton as in (1). We construct patterns β_1, $\beta_2 \in (\Sigma \cup V)^*$ such that $L_{E,\Sigma}(\beta_1) \not\subseteq L_{E,\Sigma}(\beta_2)$ iff M has an accepting computation. Choose

$$\Sigma = Q \cup \{0, \$, \bigstar, @, \#, \&\},$$

where $\$$, $\bigstar$, $@$, $\#$, $\&$ are new symbols not belonging to $Q \cup \{0\}$. The set of variables V will consist of all variables appearing in the patterns β_1 and β_2 constructed below. Define

$$\beta_1 = \bigstar\$\bigstar x \bigstar y \bigstar \$\bigstar\bigstar p_0 \bigstar\bigstar \$@,$$

where x and y are variables and p_0 is a constant pattern (over Σ) of the form

$$p_0 = @@w_1@@w_2@@\cdots@@w_k@@,$$
$$w_i \in \Sigma^*,\ i = 1, ..., k.$$

The choice of the integer k and of the words $w_1, ..., w_k \in \Sigma^*$ will be explained later.

The pattern β_2 is defined as

$$\beta_2 = x_1 \cdots x_k \$ x_1 r_1 x_1 s_1 x_1 \cdots x_k r_k x_k s_k x_k \$ x_1 x_1 z_1^{\text{left}} t_1 z_1^{\text{right}}$$
$$\cdot x_1 x_1 \cdots x_k x_k z_k^{\text{left}} t_k z_k^{\text{right}} x_k x_k \$ z_1 \cdots z_k.$$

Here $x_1, ..., x_k$, $z_1, ..., z_k$ are distinct variables and for $i \in \{1, ..., k\}$,

$$z_i^{\text{left}} = z_i z_i y_{i,1} z_i z_i y_{i,2} z_i z_i \cdots z_i z_i y_{i,i-1} z_i z_i, \tag{3}$$

and

$$z_i^{\text{right}} = z_i z_i y_{i,i+1} z_i z_i \cdots z_i z_i y_{i,k} z_i z_i. \tag{4}$$

Above $y_{i,j}$ is a distinct variable for each pair (i, j). Thus, z_i^{left} (resp. z_i^{right}) is a catenation of i words (resp. $k - i + 1$ words) $z_i z_i$, where one inserts a new variable $y_{i,j}$ between each consecutive occurrence of the word $z_i z_i$. The symbols r_i, s_i, t_i denote terminal-free patterns (in V^*) and their construction will be explained later. We always assume that for every $i \in \{1, ..., k\}$, the variables of $\text{var}(r_i) \cup \text{var}(s_i) \cup \text{var}(t_i)$ do not appear anywhere else in β_2 except in the subpatterns r_i, s_i, and t_i. (For a fixed i, r_i, s_i, and t_i can have common variables.)

The set of all mappings $\{x, y\} \to \Sigma^*$ is denoted by H. For each $i = 1, ..., k$ we define a unary predicate P_i on the set H as follows: For $h \in H$, $h \in P_i$ iff there exists a mapping

$$g\colon \text{var}(r_i) \cup \text{var}(s_i) \cup \text{var}(t_i) \to \Sigma^*$$

such that

$$g(r_i) = h(x), \qquad g(s_i) = h(y), \qquad g(t_i) = w_i.$$

If g is as above we say also that h *satisfies the predicate* P_i *via the mapping* g. The predicate P_i is completely determined by the four-tuple (w_i, r_i, s_i, t_i). We denote

$$\text{var}(P_i) = \text{var}(r_i) \cup \text{var}(s_i) \cup \text{var}(t_i),$$

$i = 1, ..., k$. Our aim is to choose k and the tuples (w_i, r_i, s_i, t_i), $i = 1, ..., k$, so that for all $h \in H$,

(i) $h(\beta_1) \in L_{E,\Sigma}(\beta_2)$ iff $h \in \bigcup_{i=1}^k P_i$;

(ii) $h \notin \bigcup_{i=1}^k P_i$ iff $h(x)$ is an accepting computation of M (as in (2)) and $h(y) \in 0^*$ is longer than the catenation of any two 0-subwords of $h(x)$. (The latter condition on $h(y)$ enables us to define the condition "$h(x)$ is an accepting computation" using the predicates P_i.)

Note that $L_{E,\Sigma}(\beta_1) \subseteq L_{E,\Sigma}(\beta_2)$ iff $(\forall h \in H)\, h(\beta_1) \in L_{E,\Sigma}(\beta_2)$. Thus from the above conditions (i) and (ii) it follows that $L_{E,\Sigma}(\beta_1) \subseteq L_{E,\Sigma}(\beta_2)$ iff the automaton M deos not have an accepting computation.

We say that a mapping $h \in H$ is of *good form* if

$$h(x) \in (\Sigma - \{\$, \bigstar, @, \&\})^* \quad \text{and} \quad h(y) \in 0^*. \tag{5}$$

First, for a suitable $k_1 < k$ we define the predicates $P_1, ..., P_{k_1}$ so that

$$(\forall h \in H) \quad h \text{ is not of good form iff } h \in \bigcup_{i=1}^{k_1} P_i.$$

Let $k_1 = \text{card}(\Sigma) + 3$. The predicates P_i, $i = 1, 2, 3$, are given by the condition:

$$r_i = a_1 b a_2, \quad s_i = c, \quad t_i = b, \quad w_i = A, \tag{6}$$

where a_1, a_2, b, c are variables and A assumes the value $\$$, $\bigstar$, $\&$. Then clearly $h \in H$ satisfies P_i defined by (6) iff the word $h(x)$ contains the symbol A. Note again that the

properties P_i (i.e., subpatterns w_i, r_i, s_i, t_i) are defined using variables that do not appear anywhere else in β_2.

Because of technical reasons for the symbol @ we need a slightly different predicate P_4:

$$r_4 = a_1 b a_2, \quad s_4 = c, \quad t_4 = dbd, \quad w_4 = \&@\&,$$

where a_1, a_2, b, c, and d are variables. Again it is easy to verify that $h \in P_4$ iff $|h(x)|_@ \neq 0$. (Note that if h satisfies P_4 via a mapping g then either $g(b) = @$, $g(d) = \&$ or $g(b) = \&@\&$, $g(d) = \lambda$.)

Similarly, the predicates P_i, $i = 5, ..., k_1 - 1$, are defined by the condition

$$r_i = c, \quad s_i = a_1 b a_2, \quad t_i = b, \quad w_i = A,$$

where a_1, a_2, b, c are variables and the constant A assumes all values from $\Sigma - \{0, @\}$. Analogously to P_4 above we defined the predicate P_{k_1} corresponding to the symbol @:

$$r_{k_1} = c, \quad s_{k_1} = a_1 b a_2, \quad t_{k_1} = dbd, \quad w_{k_1} = \&@\&.$$

Thus $h \notin \bigcup_{i=5}^{k_1} P_i$ iff $h(y) \in 0^*$. The following claim follows immediately from the above observaions.

CLAIM 1. *$h \in H$ is of good from iff $h \notin \bigcup_{i=1}^{k_1} P_i$.*

In the following, the predicates $P_1, ..., P_{k_1}$ are called BAD_FORM predicates and the as yet undefined predicates P_i, $k_1 < i \leqslant k$, are called INV_COMP (invalid computation) predicates. Exactly two of the constant words w_i, $1 \leqslant i \leqslant k_1$, are equal to \$ (resp. ★) and in all other cases w_i does not contain the symbol \$ (resp. ★). Similarly, exactly two of the words w_i, $1 \leqslant i \leqslant k_1$, are equal to &@& and in all other cases w_i does not contain the symbol @. We still postpone the definition of the INV_COMP predicates but here we make the restriction that none of the constant words w_i, $k_1 < i \leqslant k$, contains any of the symbols \$, ★, or @. Thus the following conditions hold:

$$|p_0|_\$ = 2; \tag{7}$$

$$|p_0|_\star = 2, \quad p_0 \notin \{\star\}\Sigma^* \cup \Sigma^*\{\star\} \cup \Sigma^*\{\star\star\}\Sigma^*; \tag{8}$$

$$p_0 \text{ contains } k+1 \text{ occurrences of the subword @@ that are all pairwise nonoverlapping.} \tag{9}$$

The following claim holds independently of how we define the INV_COMP predicates, assuming only that the abovementioned restrictions on the words w_i guaranteeing (7), (8), and (9).

CLAIM 2. *Let $h \in H$. Then $h(\beta_1) \in L_{E,\Sigma}(\beta_2)$ iff $h \in \bigcup_{i=1}^{k} P_i$.*

Proof of Claim 2. Assume first that $h \in P_i$ for some i, $1 \leqslant i \leqslant k$, i.e., there exists a mapping g: $\text{var}(P_i) \to \Sigma^*$ such that $g(r_i) = h(x)$, $g(s_i) = h(y)$, and $g(t_i) = w_i$. Let g': $\text{var}(\beta_2) \to \Sigma^*$ be the extension of g determined by the conditions:

(i) $g'(a) = g(a)$ for $a \in \text{var}(P_i)$,

(ii) $g'(y_{i,j}) = w_j$ for $j = 1, ..., i-1, i+1, ..., k$,

(iii) $g'(x_i) = \star$, $g'(z_i) = @$, and

(iv) $g'(a) = \lambda$ in all cases not covered by (i)–(iii).

Above the variables $y_{i,j}$ are as in (3) and (4). According to our definition of the predicates P_i the variables of $\text{var}(P_i)$ do not appear anywhere else in β_2 except in the subpatterns r_i, s_i, and t_i. Similarly, the variables $y_{m,j}$, $j = 1, ..., m-1, m+1, ..., k$, appear only in the subpatterns z_m^{left} and z_m^{right}. Thus one can always define g' as in (i)–(iv) above. Now for all $j \neq i$,

$$g'(r_j) = g'(s_j) = g'(t_j) = g'(z_j^{\text{left}}) = g'(z_j^{\text{right}}) = \lambda.$$

Hence,

$$\begin{aligned} g'(\beta_2) &= \star\$\star g'(r_i)\star g'(s_i)\star\$\star\star@@w_1@@\cdots \\ &\quad \cdots @@w_{i-1}@@g'(t_i)@@w_{i+1}@@\cdots \\ &\quad \cdots @@w_k@@\star\star\$@ \\ &= \star\$\star h(x)\star h(y)\star\$\star\star@@w_1@@\cdots \\ &\quad \cdots @@w_i@@\cdots @@w_k@@\star\star\$@ \\ &= h(\beta_1). \end{aligned}$$

Thus $h(\beta_1) \in L_{E,\Sigma}(\beta_2)$.

Conversely, assume that $h(\beta_1) \in L_{E,\Sigma}(\beta_2)$. By Claim 1, if h is not of good form then h satisfies one of the BAD_FORM predicates. Thus without loss of generality we may assume that $h(x)$ and $h(y)$ are as in (5). Let g: $\text{var}(\beta_2) \to \Sigma^*$ be a mapping such that $g(\beta_2) = h(\beta_1)$. Since h is of good form it follows from (7) that $h(\beta_1)$ contains exactly five occurrences of the symbol \$. Thus clearly none of the words $g(x_i)$, $i = 1, ..., k$, contains the symbol \$ and this implies that necessarily

$$g(x_1 \cdots x_k) = \star;$$

i.e., there exists $m \in \{1, ..., k\}$ such that

$$g(x_j) = \begin{cases} \star, & \text{if } j = m; \\ \lambda, & \text{if } j \neq m,\ 1 \leqslant j \leqslant k. \end{cases}$$

Again by (7), none of the words $g(z_i)$, $i = 1, ..., k$, contains the symbol \$ and thus there exists $n \in \{1, ..., k\}$ such that

$$g(z_j) = \begin{cases} @, & \text{if } j = n; \\ \lambda, & \text{if } j \neq n,\ 1 \leqslant j \leqslant k \end{cases}$$

Since h is of good form it follows from (8) that $h(\beta_1)$ contains exactly two occurrences of the subword $\bigstar\bigstar$. These must coincide with the two subwords $g(x_m x_m) = **$ of the word $g(\beta_2)$ and it follows that necessarily $m = n$. This implies that

$$g(r_m) = h(x), \quad g(s_m) = h(y), \quad g(z_m^{\text{left}} t_m z_m^{\text{right}}) = p_0, \tag{10}$$

and for all $j \neq m$,

$$g(r_j) = g(s_j) = g(t_j) = g(z_j^{\text{left}}) = g(z_j^{\text{right}}) = \lambda.$$

By (9), p_0 contains exactly $k+1$ nonoverlapping occurrences of the subword @@ and by (10) these must coincide with the $k+1$ subword occurrences $g(z_m z_m) = @@$ in the word $g(z_m^{\text{left}} t_m z_m^{\text{right}})$. Hence, necessarily $g(t_m) = w_m$ and $g(y_{m,j}) = w_j$, $j = 1, ..., m-1, m+1, ..., k$. Thus $h \in P_m$ and this concludes the proof of the claim. ∎

It remains to define the INV_COMP predicates $P_{k_1+1}, ..., P_k$ so that if h does not satisfy any of the BAD_FORM or INV_COMP predicates then $h(x)$ is necessarily a coding of an accepting computation of the automaton M. First we define a predicate P_i such that the negation of P_i forces $h(y)$ to be longer than the catenation of any two 0-subwords of the word $h(x)$. This property of $h(y)$ will be useful in defining the other INV_COMP predicates that are used to check that $h(x)$ is not a coding of any accepting computation. The INV_COMP predicates are defined in (i)–(xi) below. Here the symbols a, b, c, d, e with possible subscripts denote always new variables that do not appear in any other predicate:

(i) "$h(y)$ is a catenation of two subwords of $h(x)$": The predicate P_i is defined by

$$r_i = a_1 b_1 a_2 b_2 a_3, \quad s_i = b_1 b_2, \quad t_i = w_i = \lambda.$$

(ii) "$h(x)$ begins with some symbol other than #": For each $A \in Q \cup \{0\}$ we define a predicate P_i by

$$r_i = ba, \quad s_i = c, \quad t_i = b, \quad w_i = A.$$

(Note that the negation of the BAD_FORM predicates guarantees that the symbols \$, ★, @, & do not appear in the word $h(x)$.)

(iii) "$h(x)$ ends with some symbol other than #": This is analogous to the above condition.

If $h \in H$ does not satisfy any of the BAD_FORM predicates or the predicates defined above then we can write

$$h(x) = \# f_1 \# f_2 \# \cdots \# f_m \#, \tag{11}$$

$m \geq 0$, $f_i \in (Q \cup \{0\})^*$, $i = 1, ..., m$, and $h(y) = 0^n$, $n \geq 1$, where 0^n is longer than the catenation of any two 0-subwords of $h(x)$. Below we assume always that $h(x)$ and $h(y)$ are of this form.

(iv) "Some f_j, $1 \leq j \leq m$, contains more than one state": For each pair $(q_1, q_2) \in Q^2$ we define a predicate P_i by

$$r_i = a_1 b_1 c_1 b_2 a_2, \quad s_i = c_1 c_2, \quad t_i = b_1 b_2, \quad w_i = q_1 q_2.$$

Assume that h satisfies P_i via a mapping g. Then $g(b_1 c_1 b_2) \leq_{\text{sub}} h(x)$, and, since $g(c_1 c_2) = h(y) = 0^n$, $g(b_1 b_2) = q_1 q_2$, it follows that $g(b_1 c_1 b_2) \leq_{\text{sub}} f_j$ and f_j contains the states q_1 and q_2, for some $j \in \{1, ..., m\}$.

(v) "Some f_j, $1 \leq j \leq m$, contains no states": The corresponding predicate P_i is defined by

$$r_i = a_1 b c_1 b a_2, \quad s_i = c_1 c_2, \quad t_i = b, \quad w_i = \#.$$

If h does not satisfy any of the predicates defined up to this point, then in (11) $f_i \in \text{conf}(M)$, $i = 1, ..., m$. It remains to define conditions guaranteeing that f_1 is the initial configuration q_0, f_m is an accepting configuration and $f_j \Rightarrow_M f_{j+1}$ for each $j = 1, ..., m-1$. First, the negation of the conditions (vi)–(viii) forces that $f_1 = q_0$.

(vi) "f_1 does not contain the initial state q_0": For each $q \in Q - \{q_0\}$ we define a predicate P_i by

$$r_i = ac_1 bd, \quad s_i = c_1 c_2, \quad t_i = ab, \quad w_i = \# q.$$

(vii) "The first counter of f_1 is not empty":

$$r_i = ab, \quad s_i = c, \quad t_i = a, \quad w_i = \#0.$$

(viii) "The second counter of f_1 is not empty":

$$r_i = ab, \quad s_i = c, \quad t_i = a, \quad w_i = \# q_0 0.$$

(Note that in (viii) we assume that the negation of (vi) and (vii) holds; i.e., f_1 is of the form $q_0 v$, $v \in 0^*$.)

(ix) "f_m is not an accepting configuration": For each $q \in Q - Q_F$ we define a predicate P_i by

$$r_i = ba_1 c_1 a_2, \quad s_i = c_1 c_2, \quad t_i = a_1 a_2, \quad w_i = q\#.$$

(x) "The transition from f_j to f_{j+1}, $1 \leq j \leq m-1$, increments the first counter by at least two symbols": For every $q \in Q$ the predicate P_i is defined as

$$r_i = ac_1 bc_2 addc_1 e, \quad s_i = c_1 c_2 c_3,$$
$$t_i = ae_1 be_1 d, \quad w_i = \# \& q \& 0.$$

Assume that h satisfies P_i via a mapping g. Since h is of good form and $g(r_i) = h(x)$, the words $g(a)$, $g(b)$, and $g(d)$ do not contain the symbol &. Hence the condition $g(ae_1be_1d) = \#\&q\&0$ implies that $g(e_1) = \&$ and thus $g(a) = \#$, $g(b) = q$, $g(d) = 0$. Note that here we need also the assumption that $h(y)$ is longer than the catenation of any two 0-subwords of $h(x)$.

Analogously with (x) above, one defines predicates describing that a transition from f_j to f_{j+1} increments the second counter or decrements the first or second counter by at least two symbols. In the following we can then assume that any transition from f_j to f_{j+1} makes a change of at most one symbol in both counters. Invalid computation steps of this form can be described using a finite number of the predicates P_i.

(xi) "Assuming that in f_j both counters are nonempty, the transition from f_j to f_{j+1} changes the state from q_1 to q_2, increments the first counter (by one), and decrements the second counter (by one), but this is not a valid computation step": For every pair of states $(q_1, q_2) \in Q^2$ such that

$$(q_1, 1, 1, q_2, 1, -1) \notin \delta$$

defines the condition P_i by

$$r_i = ac_1db_1c_2dac_1ddb_2c_2a, \qquad s_i = c_1c_2c_3,$$
$$t_i = aeb_1eb_2ed, \qquad w_i = \#\&q_1\&q_2\&0.$$

By going through all the possibilities, whether the counters are initially empty or not and whether the transition increments, decrements, or does not change the contents of each counter for given states q_1 and q_2, one defines analogously with the above the remaining 24 different conditions describing an invalid computation step with state transition from q_1 to q_2. This concludes the construction of the INV_COMP predicates.

If $h \in H$ does not satisfy any of the BAD_FORM or INV_COMP predicates, then $h(x)$ given by (11) describes an accepting computation of the automaton M. Conversely, assume that M has an accepting computation W_c as in (2). Denote

$$n = \max\{j \mid 0^j \leqslant_{\text{sub}} W_c\}$$

and define $h \in H$ by setting

$$h(x) = W_c, \qquad h(y) = 0^{2n+1}.$$

Then clearly

$$h \notin \bigcup_{i=1}^{k} P_i.$$

By Claim 2 it follows thus that $L_{E,\Sigma}(\beta_1) \subset L_{E,\Sigma}(\beta_2)$ iff M does not have an accepting computation. This concludes the proof. ∎

In the proof of Theorem 5.1 the pattern β_2 does not contain any other terminal symbols than \$. Thus the following result is an immediate consequence of the proof of Theorem 5.1.

COROLLARY 5.1. *Given two arbitrary patterns $\beta_1 \in (\Sigma \cup V)^*$ and $\beta_2 \in (\{c\} \cup V)^*$, where Σ is a terminal alphabet, $c \in \Sigma$, and V is a set of variables, it is in general undecidable whether or not $L_{E,\Sigma}(\beta_1) \subseteq L_{E,\Sigma}(\beta_2)$.*

Observe that Corollary 5.1 remains valid even if we assume that $\beta_1 \in \Sigma^3 V \Sigma V \Sigma^*$ and $\beta_2 \in V^*(cV^*)^3$.

6. THE INCLUSION PROBLEM FOR NE-PATTERN LANGUAGES

The decidability of the inclusion problem for NE-patterns was left open in [2]. Here we show that the inclusion problem is undecidable for NE-pattern languages. This is done by reducing the question of deciding inclusion for E-pattern languages to the inclusion problem for NE-pattern languages. The former question was shown to be undecidable in the previous section. In the reduction we use the slightly strengthened form of the undecidability result in Corollary 5.1. First we need a simple technical lemma.

LEMMA 6.1. *Let V be a set of variables, Σ be a terminal alphabet, and $\Omega \subseteq \Sigma$. Consider a pattern $p \in (\Omega \cup V)^*$. Then there exist effectively $m \geqslant 1$ and patterns $p_1, \ldots, p_m \in (\Omega \cup V)^*$ such that*

$$L_{E,\Sigma}(p) = \bigcup_{i=1}^{m} L_{NE,\Sigma}(p_i).$$

Proof. The patterns p_i, $i = 1, \ldots, m$, are obtained from p simply by uniformly erasing some number of variables. ∎

THEOREM 6.1. *Given a terminal alphabet Σ, a set of variables V, and two patterns $\alpha_1, \alpha_2 \in (\Sigma \cup V)^*$ it is undecidable in general whether*

$$L_{NE,\Sigma}(\alpha_1) \subseteq L_{NE,\Sigma}(\alpha_2). \tag{12}$$

Proof. We proceed by contradiction. Assuming that for given patterns α_1 and α_2 one can decide whether (12) holds, we show that the problem of Corollary 5.1 would also be decidable. For this purpose let Σ be a terminal alphabet, $0 \in \Sigma$ a fixed symbol, V a set of variables, and $\beta_1 \in (\Sigma \cup V)^*$, $\beta_2 \in (\{0\} \cup V)^*$ arbitrary given patterns. By Lemma 6.1 there exist patterns $p_1, \ldots, p_m \in (\Sigma \cup V)^*$ and $q_1, \ldots, q_n \in (\{0\} \cup V)^*$, $m, n \geqslant 1$, such that

$$L_{E,\Sigma}(\beta_1) = \bigcup_{i=1}^{m} L_{NE,\Sigma}(p_i)$$

and

$$L_{E,\Sigma}(\beta_2) = \bigcup_{j=1}^{n} L_{NE,\Sigma}(q_j).$$

Thus to decide whether

$$L_{E,\Sigma}(\beta_1) \subseteq L_{E,\Sigma}(\beta_2), \tag{13}$$

it is sufficient to decide for each $i = 1, ..., m$, whether

$$L_{NE,\Sigma}(p_i) \subseteq \bigcup_{j=1}^{n} L_{NE,\Sigma}(q_j). \tag{14}$$

Let i be an arbitrary number in $\{1, ..., m\}$. We construct a terminal alphabet Ω, a set of variables Y and patterns

$$p_i', q \in (\Omega \cup Y)^*$$

such that (14) holds iff

$$L_{NE,\Omega}(p_i') \subseteq L_{NE,\Omega}(q). \tag{15}$$

We introduce two new terminal symbols \$ and ★ and let $\Omega = \Sigma \cup \{\$, \bigstar\}$. The set Y will consist of all variables appearing in the patterns p_i' and q constructed below.

By renaming the variables in q_j, $1 \leqslant j \leqslant n$, we can assume that

$$(\forall j, l,\ 1 \leqslant j, l \leqslant n,\ j \neq l) \qquad \mathrm{var}(q_j) \cap \mathrm{var}(q_l) = \varnothing. \tag{16}$$

Denote

$$k = \sum_{j=1}^{n} |q_j|.$$

The pattern p_i' is defined by

$$p_i' = 0^{2n+8} \bigstar\bigstar 0^{2n+8} \$\$ 0^{2n+k+18} \bigstar p_i \bigstar 0^{2n+k+18} \$\$ p_0 \$\$ p_0 \$\$ p_0 \$\$ p_0,$$

where

$$p_0 = 0^4 \bigstar \$ \bigstar\bigstar\bigstar\bigstar 0^4.$$

Let $x_1, ..., x_{n+4}$, y_j, y_j', $j = 1, ..., 6$, u_l, z_l, $l = 1, ..., 4$, be distinct variables not belonging to $\bigcup_{j=1}^{n} \mathrm{var}(q_j)$. We denote

$$r_1 = u_1 z_1, \qquad r_2 = z_2 u_2, \qquad r_3 = z_3 u_3 z_4, \qquad r_4 = u_4.$$

Now the pattern q is defined as

$$q = Q_1 \$\$ Q_2 \$\$ Q_3 \$\$ Q_4 \$\$ Q_5 \$\$ Q_6,$$

where

- $Q_1 = y_1 x_1 x_1 x_2 x_2 \cdots x_{n+4} x_{n+4} y_1'$,
- $Q_2 = y_2 x_1 q_1 x_1 \cdots x_n q_n x_n x_{n+1} r_1 x_{n+1} \cdots x_{n+4} r_4 x_{n+4} y_2'$,
- $Q_j = y_j x_{n+j-2} u_{j-2} x_{n+j-2} y_j'$, $j = 3, ..., 6$.

The intuitive idea of the construction is as follows: Consider $h: \mathrm{var}(p_i') \to \Omega^+$ and assume that $h(p_i') \in L_{NE,\Omega}(q)$. Then the subpattern p_i of p_i' has to be "matched" with some subpattern q_j, $j = 1, ..., n$, or r_l, $l = 1, ..., 4$, of q. If p_i is matched with r_l, $1 \leqslant l \leqslant 4$, then the corresponding instance of p_i necessarily contains an occurrence of a symbol \$ or ★. Thus all Σ-instances of p_i belong to some language $L_{NE,\Sigma}(q_j)$, $1 \leqslant j \leqslant n$, and (14) holds. Below we prove that the conditions (14) and (15) are equivalent:

(i) First assume that (14) holds. Let h be an arbitrary mapping $\mathrm{var}(p_i') \to \Omega^+$.

(a) Assume that $|h(p_i)|_\$ = |h(p_i)|_\bigstar = 0$, i.e., $h(p_i) \in \Sigma^+$. (Note that $\mathrm{var}(p_i') = \mathrm{var}(p_i)$.) By (14), there exists $t \in \{1, ..., n\}$ and a mapping $f: \mathrm{var}(q_t) \to \Sigma^+$ such that $f(q_t) = h(p_i)$. Denote

$$M_1 = \sum_{j=1}^{t-1} |q_j|, \qquad M_2 = \sum_{j=t+1}^{n} |q_j|.$$

Define a mapping $g: \mathrm{var}(q) \to \Omega^+$ as follows:

- $g(x_t) = \bigstar$ and $g(x_j) = 0$ when $j \neq t$, $0 \leqslant j \leqslant n+4$.
- $g(y_1) = 0^{c_1}$, $g(y_1') = 0^{2t}$, where $c_1 = 2n + 8 - 2(t-1)$.
- $(\forall x \in \mathrm{var}(q_t))\ g(x) = f(x)$.
- $(\forall j \neq t,\ 1 \leqslant j \leqslant n)(\forall x \in \mathrm{var}(q_j))\ g(x) = 0$.
- $(\forall x \in \bigcup_{j=1}^{4} \mathrm{var}(r_j))\ g(x) = 0$.
- $g(y_2) = 0^{c_1}$, $g(y_2') = 0^{c_2}$, where $c_1 = (2n + k + 18) - M_1 - 2(t-1) = 2(n-t) + k - M_1 + 20$ and $c_2 = (2n + k + 18) - M_2 - 2(n-t) - 16 = k + 2 - M_2 + 2t$. (Note that $|x_{n+1} r_1 x_{n+1} \cdots x_{n+4} r_4 x_{n+4}| = 16$.)
- $g(y_j) = 0$, $g(y_j') = \bigstar \$ \bigstar\bigstar\bigstar\bigstar 0^4$, $j = 3, ..., 6$.

By (16), the mapping g can always be defined as above. Note that since 0 is the only terminal appearing in the patterns $q_1, ..., q_n$, it follows that $g(q_j) \in 0^+$ for all $j \neq t$. Now it is immediately verified that $g(q) = h(p_i')$; i.e., $h(p_i') \in L_{NE,\Omega}(q)$.

(b) Assume that $|h(p_i)|_\$ > 0$. First consider the case when $h(p_i)$ contains an occurrence of \$ that is not a prefix or suffix of $h(p_i)$; i.e., we can write

$$h(p_i) = w_1 \$ w_2, \qquad w_1, w_2 \in \Omega^+. \tag{17}$$

In this case we define a mapping $g: \mathrm{var}(q) \to \Omega^+$ by the following:

- $g(x_{n+3}) = \bigstar$ and $g(x_j) = 0$, $j = 1, ..., n+2, n+4$.
- $g(y_1) = 0^4$, $g(y_1') = 0^{2n+6}$.
- $(\forall j \in \{1, ..., n\})(\forall x \in \mathrm{var}(q_j))\ g(x) = 0$.
- $g(z_3) = w_1$, $g(u_3) = \$$, $g(z_4) = w_2$.
- $g(z_1) = g(z_2) = g(u_1) = g(u_2) = g(u_4) = 0$.
- $g(y_2) = 0^{10}$, $g(y_2') = 0^{2n+k+15}$.
- $g(y_5) = 0^4$, $g(y_5') = \bigstar\bigstar\bigstar 0^4$.
- $g(y_3) = g(y_4) = g(y_6) = 0$, $g(y_3') = g(y_4') = g(y_6') = \bigstar\$\bigstar\bigstar\bigstar\bigstar 0^4$.

Again it can be verified that $g(q) = h(p_i')$. The case when $h(p_i)$ contains \$ only as a proper prefix (resp. as a proper suffix, or $h(p_i) = \$$) is similar. In this case one chooses $g(x_{n+1}) = \bigstar$ (resp. $g(x_{n+2}) = \bigstar$ or $g(x_{n+4}) = \bigstar$) and matches $h(p_i)$ with $g(r_1)$ (resp. $g(r_2)$, or $g(r_4)$).

(c) Finally, the case when $|h(p_i)|_\bigstar > 0$ is completely analogous to (b) above. Depending on where the symbol $\bigstar$ appears in the word $h(p_i)$, one of the subpatterns $x_{n+j}u_jx_{n+j}$, $j = 1, ..., 4$, is matched to the subword $\bigstar\bigstar\bigstar$ of p_0 and for the corresponding j we define $g(y_{j+2}) = 0^4\bigstar\$\bigstar$, $g(y_{j+2}') = 0^4$.

Thus we have proved that (14) implies that the relation (15) holds.

(ii) Conversely, assume that (15) holds. Let h be an arbitrary mapping $\mathrm{var}(p_i) \to \Sigma^+$. We aim to show that $h(p_i) \in L_{\mathrm{NE},\Sigma}(q_j)$ for some $j \in \{1, ..., n\}$. By (15), there exists g: $\mathrm{var}(q) \to \Omega^+$ such that

$$g(q) = h(p_i'). \tag{18}$$

The word $h(p_i')$ contains exactly five occurrences of the subword \$\$. Thus from (18) it follows that necessarily

$$g(Q_1) = 0^{2n+8}\bigstar\bigstar 0^{2n+8}, \tag{19}$$

$$g(Q_2) = 0^{2n+k+18}\bigstar h(p_i)\bigstar 0^{2n+k+18}, \tag{20}$$

$$g(Q_3) = \cdots = g(Q_6) = p_0. \tag{21}$$

From (19) it follows that there necessarily exists $t \in \{1, ..., n+4\}$ such that $g(x_t) = \bigstar$ and $g(x_j) \in 0^+$ when $j \neq t$. (Since the mapping g is nonerasing it is not possible, for instance, that the symbols $\bigstar$ would correspond to the subword $g(y_1)$ or $g(y_1')$.)

(a) Assume that $t \in \{1, ..., n\}$. Then from (20) it follows that $g(x_tq_tx_t) = \bigstar h(p_i)\bigstar$; i.e., $g(q_t) = h(p_i)$. Thus the restriction of g to $\mathrm{var}(q_t)$ is a mapping with range Σ^+ and $h(p_i) \in L_{\mathrm{NE},\Sigma}(q_t)$.

(b) Finally, consider the possibility that $t = n + j$, $1 \leqslant j \leqslant 4$. Again by (20) it follows that $g(x_{n+j}r_jx_{n+j}) = \bigstar h(p_i)\bigstar$, and thus

$$g(r_j) = h(p_i). \tag{22}$$

On the other hand, by (21) $g(Q_{j+2}) = p_0$. Since $g(x_{n+j}) = \bigstar$, it follows that $g(u_j) \in \{\$, \bigstar\}^+$. This contradicts (22) because $h(p_i) \in \Sigma^+$.

We have shown that the relations (14) and (15) are equivalent. Thus the assumption that the inclusion problem for NE-patterns is decidable contradicts Corollary 5.1. This concludes our proof. ▮

7. RELATED RESULTS

The main result of this section is that the inclusion problem for terminal-free E-pattern languages is decidable. The equivalence problem for E-pattern languages in general is still open. Here, we also show a strong necessary condition for the equivalence of two E-pattern languages.

Let $V = \{x_1, ..., x_n\}$ be a set of variables and Σ be an alphabet such that $\mathrm{card}(\Sigma) \geqslant 2$. For each pair of letters a, b in Σ, $a \neq b$, and an integer $k > 0$, we define a morphism $\tau_{k,a,b}$: $V^* \to \Sigma^*$ by

$$\tau_{k,a,b}(x_i) = ab^{ki+1}aab^{ki+2}aab^{ki+3}a \cdots ab^{k(i+1)}a, \quad 1 \leqslant i \leqslant n.$$

In the above definition, we call each substring of the form ab^+a a segment. Note that for each $x \in V$, $\tau_{k,a,b}(x)$ consists of k distinct segments. Note also that for $x_i, x_j \in V$ and $i \neq j$, $\tau_{k,a,b}(x_i)$ and $\tau_{k,a,b}(x_j)$ do not contain any segment in common.

LEMMA 7.1. *Let $\alpha, \beta \in V^+$ be two arbitrary terminal-free patterns, and a, b be two distinct letters in Σ. Then $\tau_{|\beta|,a,b}(\alpha) \in L_{\mathrm{E},\Sigma}(\beta)$ if and only if there exists a morphism h: $V^* \to V^*$ such that $h(\beta) = \alpha$.*

Proof. The *if* part of the lemma is trivially true. We prove the *only if* part as follows: Let $\alpha = y_1 \cdots y_l$ and $\beta = z_1 \cdots z_k$, where $y_1, ..., y_l, z_1, ..., z_k \in V$, $l, k > 0$. Since $\tau_{k,a,b}(\alpha) \in L_{\mathrm{E},\Sigma}(\beta)$, there exists a morphism ν: $V^* \to \Sigma^*$ such that $\nu(\beta) = \tau_{k,a,b}(\alpha)$; i.e.,

$$\nu(\beta) = \nu(z_1) \cdots \nu(z_k) = \tau_{k,a,b}(\alpha) = \tau_{k,a,b}(y_1) \cdots \tau_{k,a,b}(y_l).$$

Consider the decomposition of $\tau_{k,a,b}(\alpha)$ into k substrings $\nu(z_1)$, ..., and $\nu(z_k)$. It is clear that, for each variable y_i in α, $1 \leqslant i \leqslant l$, there are the following two cases:

(1) $\tau_{k,a,b}(y_i)$ is a substring of $\nu(z_j)$ for some variable z_j in β, $1 \leqslant j \leqslant k$; or

(2) $\tau_{k,a,b}(y_i)$ is split into m parts, for some integer $m > 1$; i.e., $\tau_{k,a,b}(y_j) = u_1 \cdots u_m$ such that u_1 is a suffix of $\nu(z_j)$, $u_2 = \nu(z_{j+1})$, ..., $u_{k-1} = \nu(z_{j+m-2})$, and u_k is a prefix of $\nu(z_{j+m-1})$.

In (2) above, it is clear that $m \leqslant k$. By the definition of τ, $\tau_{k,a,b}(y_i)$ consists of exactly k segments of the form ab^+a. So, the k segments of $\tau_{k,a,b}(y_i)$ contain at most $k-1$ splitting points of the decomposition $\tau_{k,a,b}(\alpha) = \nu(z_1) \cdots \nu(z_k)$.

Therefore, for each variable y_i of α, there must exist at least one segment in $\tau_{k,a,b}(y_i)$ that is not split by the above decomposition. Following the same principle, we can verify a stronger statement: For each variable $y \in \text{var}(\alpha)$, which may appear in α multiple times, there exists a segment s in $\tau_{k,a,b}(y)$ such that none of the occurrences of s in $\tau_{k,a,b}(\alpha)$ is split by the above decomposition. For each $y \in \text{var}(\alpha)$, we choose such a segment to be the *anchor segment* of y in $\tau_{k,a,b}(\alpha)$ with respect to β. We also say that this segment *anchors* y.

We now define a morphism $h: V^* \to V^*$ by the following: For each $z \in \text{var}(\beta)$, $h(z)$ is obtained from $v(z)$ by first deleting from $v(z)$ all but the anchor segments, and then replacing each anchor segment by the variable it anchors; for each $z \notin \text{var}(\beta)$, define $h(y) = \lambda$. It is easy to verify that $h(\beta) = \alpha$, since each appearance of $\tau_{k,a,b}(y)$ in $\tau_{k,a,b}(\alpha)$ has exactly one anchor segment of y. ∎

THEOREM 7.1. *Let $\alpha, \beta \in X^+$ be two arbitrarily given terminal-free patterns. Then $L_{E,\Sigma}(\alpha) \subseteq L_{E,\Sigma}(\beta)$ if and only if there exists a morphism $h: V^* \to V^*$ such that $h(\beta) = \alpha$.*

Proof. The *only if* part is proved by Lemma 7.1. The *if* part is trivial. ∎

From the above results, we can easily obtain the following corollaries.

COROLLARY 7.1. *Let $\alpha, \beta \in V^+$ be two arbitrary terminal-free patterns and Σ be an alphabet, $\text{card}(\Sigma) \geq 2$. Then $L_{E,\Sigma}(\alpha) \subseteq L_{E,\Sigma}(\beta)$ if and only if $\tau_{|\beta|,a,b}(\alpha) \in L_{E,\Sigma}(\beta)$ for $a, b \in \Sigma$ and $a \neq b$.*

COROLLARY 7.2. *Given two terminal-free patterns $\alpha, \beta \in V^+$ and an alphabet Σ, the inclusion problem, i.e., the question of whether or not $L_{E,\Sigma}(\alpha) \subseteq L_{E,\Sigma}(\beta)$, is decidable.*

Proof. First we assume that $\text{card}(\Sigma) \geq 2$. Then, by Corollary 7.1, the inclusion is decided by whether the word $\tau_{|\beta|,a,b}(\alpha)$ is in $L_{E,\Sigma}(\beta)$. We know that the membership problem for E-pattens is decidable.

Now we consider the case when $\text{card}(\Sigma) = 1$. Let a be the letter in Σ. It is easy to show that, for any pattern γ, the sequence of the lengths of all words in $L_{E,\Sigma}(\gamma)$, in increasing order, is ultimately periodic; i.e., there exist integers n_0, $c > 0$ such that the set $\{a^{ci} \mid i > 0 \ \& \ ci \geq n_0\}$ and $L_{E,\Sigma}(\gamma)$ are the same except for words of length less than n_0. Note that n_0 and c are easily computable. Then it is clear that we can solve the inclusion problem easily. ∎

Note that the inclusion problem for terminal-free NE-pattern languages is still open, as well as the equivalence problem for general E-pattern languages. We conjecture that the latter problem is decidable. The following theorem gives a strong necessary condition for the equivalence of two E-pattern languages.

THEOREM 7.2. *Let $\alpha = \alpha_0 u_1 \alpha_1 u_2 \cdots \alpha_{m-1} u_m \alpha_m$ and $\beta = \beta_0 v_1 \beta_1 v_2 \cdots \beta_{n-1} v_n \beta_n$ be two patterns in $(\Sigma \cup V)^+$, where $\alpha_0, \beta_0, \alpha_m, \beta_n \in V^*$, $\alpha_i \in V^+$ for $1 \leq i < m$, $\beta_j \in V^+$ for $1 \leq j < n$, $u_i \in \Sigma^+$ for $1 \leq i \leq m$, $v_j \in \Sigma^+$ for $1 \leq j \leq n$, $m, n \geq 0$. If $L_{E,\Sigma}(\alpha) = L_{E,\Sigma}(\beta)$ and $\text{card}(\Sigma) \geq 4$, then $m = n$, $L_{E,\Sigma}(\alpha_i) = L_{E,\Sigma}(\beta_i)$, and $u_j = v_j$ for $0 \leq i \leq m$ and $1 \leq j \leq m$.*

Proof. In [6], we have proved that, for $\text{card}(\Sigma) \geq 3$, $L_{E,\Sigma}(\alpha) = L_{E,\Sigma}(\beta)$ implies that $m = n$ and $u_i = v_i$ for all i, $1 \leq i \leq m$. Thus, it suffices to show that

$$L_{E,\Sigma}(\alpha_i) = L_{E,\Sigma}(\beta_i), \qquad 0 \leq i \leq m. \tag{23}$$

Suppose that (23) is not true. Let i be some integer such that $L_{E,\Sigma}(\alpha_i) \neq L_{E,\Sigma}(\beta_i)$. First, we consider the case when $1 \leq i < m$. Let c be the last letter of u_i and d the first letter of u_{i+1}. We choose $a, b \in \Sigma - \{c, d\}$ such that $a \neq b$. Without loss of generality, we assume that $L_{E,\Sigma}(\alpha_i) \not\subseteq L_{E,\Sigma}(\beta_i)$. Then, by Corollary 7.1, it is clear that $\tau_{|\beta|,a,b}(\alpha_i) \in L_{E,\Sigma}(\alpha_i) - L_{E,\Sigma}(\beta_i)$. We choose an arbitrary word $w = w_0 u_1 w_1 \cdots w_{m-1} u_m w_m \in L_{E,\Sigma}(\alpha)$ such that $w_0, \ldots, w_m \in \{a, b\}^*$ and $w_i = \tau_{|\beta|,a,b}(\alpha_i)$. Note that if α_i is between the sth appearance of c and the tth appearance of d in α, then w_i is between the sth appearance of c and the tth appearance of d in w. Assume that w is also generated by β. Then, no variable of β generates a string containing any c or d. Therefore, w_i must be generated by β_i. This is a contradiction.

The argument for $i = 0$ or $i = m$ is similar to the above and simpler. Thus, we have proved the theorem. ∎

Note that in Theorem 7.2, the conditions $m = n$, $L_{E,\Sigma}(\alpha_i) = L_{E,\Sigma}(\beta_i)$, and $u_j = v_j$, for $0 \leq i \leq m$ and $1 \leq j \leq m$, are not sufficient for $L_{E,\Sigma}(\alpha) = L_{E,\Sigma}(\beta)$. For instance, the patterns $\alpha = xaybz$ and $\beta = xaxbx$ satisfy all the above conditions, but it is clear that $L_{E,\Sigma}(\alpha) \neq L_{E,\Sigma}(\beta)$.

REFERENCES

1. D. Angluin, Finding patterns common to a set of strings, "17th Sympos. Theory of Comput., 1979," pp. 130–141.
2. D. Angluin, Finding patterns common to a set of strings, *J. Comput. System Sci.* **21** (1980), 46–62.
3. D. Angluin, Inductive inference of formal languages from positive data, *Inform. and Control* **45** (1980), 117–135.
4. B. S. Baker and R. V. Book, Reversal-bounded multipushdown machines, *J. Comput. System Sci.* **8** (1974), 315–332.
5. D. R. Bean, A. Ehrenfeucht, and G. F. McNulty, Avoidable patterns in strings of symbols, *Pacific J. Math.* **85** (1979), 261–294.
6. T. Harju and J. Karhumäki, The equivalence problem of multitape finite automata, *Theoret. Comput. Sci.* **78** (1991), 347–355.
7. M. A. Harrison, "Introduction to Formal Language Theory," Addison-Wesley, Reading, MA, 1978.
8. O. H. Ibarra, Reversal-bounded multicounter machines and their decision problems, *J. Assoc. Comput. Mach.* **25** (1978), 116–133.
9. O. Ibarra and T. Jiang, Learning regular languages from counter-examples, *J. Comput. System Sci.* **43** (1991), 299–316.

10. T. Jiang, E. Kinber, A. Salomaa, K. Salomaa, and S. Yu, Pattern languages with and without erasing, *Internat. J. Comput. Math.*, submitted.
11. V. Keränen, Abelian squares can be avoided on four letters, *in* "ICALP-92 Proceedings," Lecture Notes in Computer Science, Vol. 623, pp. 41–52, Springer-Verlag, New York/Berlin, 1992.
12. G. S. Makanin, The problem of solvability of equations in a free semigroup, *Soviet Math. Dokl.* **18** (1977), 330–334.
13. M. L. Minsky, Recursive unsolvability of Post's problem of "Tag" and other topics in theory of Turing machines, *Ann. of Math.* **74** (1961), 437–455.
14. A. Salomaa, "Formal Languages," Academic Press, New York/London, 1973.
15. N. Tanida and T. Yokomori, Polynomial-time identification of strictly regular languages in the limit, *IEICE Trans. Inform. Syst.* **E75-D** (1992), 125–132.
16. A. Thue, Über unendliche Zeichenreihen, *Norsk. Vid. Selsk. Skr. I Mat. Nat. Kl. Christiania* 7 (1906), 1–22.