Theoretical Computer Science 104 (1992) 185–206
Elsevier

On reversal-bounded picture languages

Changwook Kim*

School of Electrical Engineering and Computer Science, University of Oklahoma, Norman, OK 73019, USA

Ivan Hal Sudborough

Computer Science Program, University of Texas at Dallas, Richardson, TX 75083, USA

Communicated by G. Ausiello
Received October 1988
Revised April 1991

Abstract

Kim, C. and I.H. Sudborough, On reversal-bounded picture languages, Theoretical Computer Science 104 (1992) 185–206.

For an integer $k \geqslant 0$, a *k-reversal-bounded picture language* is a chain-code picture language which is described by a language L over the alphabet $\pi = \{u, d, r, l\}$ such that, for every word x in L, the number of alternating occurrences of r's and l's in x is bounded by k. It is shown that the membership problem can be solved in $O(n^{4k+4})$ time for k-reversal-bounded regular picture languages, for every $k \geqslant 1$, and is NP-complete for 1-reversal-bounded stripe linear picture languages. The membership problem is known to be NP-complete for regular and context-free picture languages without restriction on the number of reversals and solvable in $O(n)$ time ($O(n^{12})$ time) for 0-reversal-bounded regular (context-free) picture languages. Whether the membership problem for stripe context-free picture languages could be solved in polynomial time has been an open problem. Other basic properties of reversal-bounded picture languages are also presented.

1. Introduction

A picture can be described by a word over the alphabet $\pi = \{u, d, r, l\}$, with the following interpretation of the symbols in π: the symbol u (d, r, and l) means "draw one unit line in the two-dimensional Cartesian plane by moving the pen up (down, right, and left) from the current point". A set of pictures described by a language over π is called a (*chain-code*) *picture language*. Maurer et al. [9] initiated the investigation of

* This author's research was supported in part by the 1988 Junior Faculty Summer Research Fellowship, The University of Oklahoma.

the properties of picture languages classified by the Chomsky hierarchy of language families; subsequently, many other research results regarding the properties of picture languages have appeared in the literature [1, 3, 4, 6–8, 10, 12]. Among others, the membership problem is NP-complete for regular, linear, and context-free picture languages [8, 12] and the equivalence, containment, intersection emptiness, and ambiguity problems are all undecidable for regular picture languages [7, 8, 12].

The intractability of major decision problems considered for picture languages motivated the study of restricted classes of picture languages that have better complexity results and still have power for real-world applications. Two such classes are *stripe picture languages* [12], whose pictures fit into a stripe defined by two parallel lines in the plane, and *three-way picture languages* [6], which are defined by a three-letter subset of π. It was shown in [12] that for stripe regular picture languages, the membership problem can be solved in linear time and the equivalence and intersection emptiness problems are decidable. In [6], the membership problem was shown to be solvable in linear time for three-way regular picture languages and in $O(n^{12})$ time for three-way context-free picture languages.

This paper introduces a class of picture languages, called *reversal-bounded picture languages*, which are described by languages over π whose words contain a bounded number of alternating r's and l's, and studies their decision properties. Reversal-bounded picture languages defined in this paper are a straightforward extension of the three-way picture languages introduced in [6] and well classify picture languages which lie between three-way picture languages and general picture languages. For example, for every $k \geqslant 0$, the concept of reversals defines the picture version of the Chomsky hierarchy that separates the families of k-reversal-bounded regular, linear, context-free, context-sensitive, and recursively enumerable picture languages, while it is known that for general picture languages without restriction on the number of reversals, the family of context-sensitive picture languages is identical to the family of recursively enumerable picture languages [9]. There is also an infinite hierarchy of picture languages, defined by bounding the number of reversals, in each family of picture languages classified by the Chomsky hierarchy of grammars. The concept of reversals has been used as a useful restriction or normal form in many other branches of the theory of formal languages and automata. The restriction by the number of reversals in chain-code picture languages as defined in this paper is natural since a reversal corresponds to a reversal of the direction of the head motion of a picture-drawing device.

In Section 2, we shall give necessary notations and notions for chain-code picture languages and define reversal-bounded picture languages. In Section 3, we study some of the basic properties of reversal-bounded picture languages. We discuss decidability and undecidability of descriptability problems arising from the notion of reversal-bounded picture languages and prove several hierarchy theorems that demonstrate relations between families of (reversal-bounded) picture languages. In Section 4, we discuss polynomial-time solvable membership problems. It is shown that, for every $k \geqslant 1$, the membership problem for k-reversal-bounded regular picture languages can be solved in $O(n^{4k+4})$ time. (A 0-reversal-bounded picture language can be represented by the union of two three-way picture languages. So, the membership problem for 0-reversal-bounded picture languages can be solved by using the polynomial-time recognition algorithms for three-way picture languages.) Section 5 presents an NP-completeness result. It is shown that there is a 1-reversal-bounded stripe linear picture language for which the membership problem is NP-complete. This solves a question left unsolved in an earlier publication. Namely, in [12], the membership problem for stripe regular picture languages was shown to be solvable in linear time, but whether the membership problem for stripe context-free picture languages could be solved in polynomial time was left unanswered. In Section 6, we present some undecidability results on the membership problem and other decision problems for reversal-bounded picture languages.

2. Preliminaries

We shall assume the reader to be familiar with the basics of the theory of formal languages and automata [5]. For the picture part, we follow the notions from [9]. Z denotes the set of integers. For a word w, $|w|$ denotes the length of w. The empty string is denoted by λ. For a set A, $|A|$ denotes the cardinality of A and 2^A denotes the power set of A. The empty set is denoted by $\emptyset$. For sets A and B, $A \subseteq B$ denotes the inclusion of A in B, $A \subsetneq B$ denotes the proper inclusion of A in B, and $A - B$ denotes the set-theoretical difference of A and B.

The *universal point set*, denoted by M_0, is the Cartesian product of Z with itself. For a point $v=(m,n)\in M_0$, the *up-neighbor* of v, denoted by $u(v)$, is $(m,n+1)$, the *down-neighbor* of v, denoted by $d(v)$, is $(m,n-1)$, the *right-neighbor* of v, denoted by $r(v)$, is $(m+1,n)$, and the *left-neighbor* of v, denoted by $l(v)$, is $(m-1,n)$. The *neighborhood* of v, denoted by $N(v)$, is the set $\{u(v), d(v), r(v), l(v)\}$. The *universal line set*, denoted by M_1, is the set $\{\{v, v'\} \mid v\in M_0 \text{ and } v'\in N(v)\}$.

An *attached basic picture* p is a finite subset of M_1. Its *point set*, denote by $V(p)$, is the set $\{v\in M_0 \mid \{v, v'\}\in p \text{ for some } v'\in M_0\}$. Thus, an attached basic picture is a graph, whose set of vertices is $V(p)$ and whose set of edges is p. We shall consider only connected pictures, i.e., connected graphs.

An *attached drawn picture* is a triple (p, s, e), where p is an attached basic picture and either p is empty and $s=e$ is an arbitrary point in M_0 or p is nonempty and s, e are points in $V(p)$. Given an attached drawn picture $q=(p,s,e)$, p is called the *base* of q, denoted by base(q), s is called the *start point* of q, denoted by start(q), and e is called the *end point* of q, denoted by end(q).

For attached basic pictures p_1 and p_2, p_1 is a *subpicture* of p_2 if $p_1 \subseteq p_2$. (Each line $\{v, v'\}$ of p_1 or p_2 is considered to be identical to $\{v', v\}$.) For attached drawn pictures q_1 and q_2, q_1 is a *subpicture* of q_2 if base$(q_1) \subseteq$ base(q_2).

For integers m and n, define the translational mapping $t_{m,n}$ from M_0 to M_0 by $t_{m,n}(i,j)=(i+m, j+n)$. The translation $t_{m,n}$ induces a mapping from M_1 to M_1 by

$t_{m,n}(\{v, v'\}) = \{t_{m,n}(v), t_{m,n}(v')\}$. Furthermore, the translation $t_{m,n}$ induces a mapping on subsets of M_1 defined, for any subset A of M_1, by $t_{m,n}(A) = \{t_{m,n}(h) \mid h \in A\}$.

Let p_1 and p_2 be two attached basic pictures. p_1 is *translation-equivalent* to p_2 if there exist integers m and n such that $p_1 = t_{m,n}(p_2)$. Similarly, for two attached drawn pictures $q_1 = (p_1, s_1, e_1)$ and $q_2 = (p_2, s_2, e_2)$, we say that q_1 is *translation-equivalent* to q_2 if there exist integers m and n such that $p_1 = t_{m,n}(p_2)$, $s_1 = t_{m,n}(s_2)$, and $e_1 = t_{m,n}(e_2)$.

The equivalence class containing an attached basic picture p, denoted by $[p]$, is called the *unattached version of* p and is simply referred to as *basic picture*. Similarly, the equivalence class containing an attached drawn picture q, denoted by $\langle q \rangle$, is called the *unattached version of* q and is simply referred to as *drawn picture*.

Consider a four-letter alphabet $\pi = \{u, d, r, l\}$. Every word in π^* is called a *picture description word* (or *π-word*). Every language over π is called a *picture description language* (or *π-language*), every grammar generating a π-language is called a *picture description grammar* (or *π-grammar*), and every automaton accepting a π-language is called a *picture description automaton* (or *π-automaton*).

The *drawn picture described by a π-word* w, denoted by dpic(w), is defined inductively by

$$\text{dpic}(\lambda) = \langle \emptyset, (0,0), (0,0) \rangle,$$

and for all $z \in \pi^*$, where $\text{dpic}(z) = \langle p, s, e \rangle$, and for all $b \in \pi$,

$$\text{dpic}(zb) = \langle p \cup \{\{e, b(e)\}\}, s, b(e) \rangle.$$

If $\text{dpic}(w) = \langle p, s, e \rangle$, then the *basic picture described by* w, denoted by bpic(w), is $[p]$.

Figure 1 shows a drawn picture described by π-word $r^3u^2ld^3r$, where a little circle in the figure indicates the start point of the picture and a little square indicates the end point.

For a given π-language L, the *drawn picture language described by* L, denoted by dpic(L), is $\text{dpic}(L) = \{\text{dpic}(w) \mid w \in L\}$ and the *basic picture language described by* L, denoted by bpic(L), is $\text{bpic}(L) = \{\text{bpic}(w) \mid w \in L\}$. Similarly, for a π-grammar or π-automaton A, the *drawn (basic) picture language described by* $L(A)$ is denoted by dpic(A) (bpic(A)).

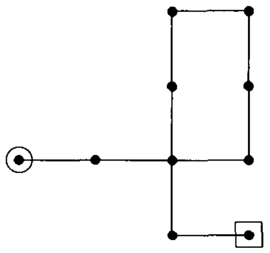


Fig. 1. A drawn picture described by $r^3u^2ld^3r$.

A drawn (basic) picture language P is *regular* (*linear*, *context-free*, *context-sensitive*, or *recursively enumerable*) if there is a regular (linear, context-free, context-sensitive, or recursively enumerable) language L such that $\mathrm{dpic}(L)=P$ ($\mathrm{bpic}(L)=P$).

Now we turn to the notion of reversal-bounded picture languages. Let $\vec{\pi}=\{u,d,r\}$ and $\overleftarrow{\pi}=\{u,d,l\}$. A picture language described by a language over $\vec{\pi}$ or $\overleftarrow{\pi}$ is called a three-way picture language in [6]. The notion of three-way picture languages is extended to reversal-bounded picture languages as follows.

For a π-word x, we say that a *right-to-left reversal* occurs in x if x contains a subword $y \in r\vec{\pi}^*l$ and that a *left-to-right reversal* occurs in x if x contains a subword $y \in l\overleftarrow{\pi}^*r$. A right-to-left or left-to-right reversal in x is called a *reversal* in x. Thus, x has k reversals, $k \geqslant 0$, if the number of alternating occurrences of r's and l's in x is k. For example, the π-word $x=u^2drludrlrul^2$ has five reversals since there are five alternations of r's and l's in x, i.e., each of the subwords rl, $ludr$, lr and rul results in a reversal, where the subword rl occurs twice in x and each counts as a reversal.

For an integer $k \geqslant 0$, a π-language L is a *k-reversal-bounded language* if every π-word in L has at most k reversals. A π-automaton (π-grammar) A is a *k-reversal-bounded automaton* (*grammar*) if $L=L(A)$ for a k-reversal-bounded language L. A picture language described by a k-reversal-bounded π-language is called a *k-reversal-bounded picture language*.

3. Basic properties

In this section we shall observe some basic facts regarding the families of reversal-bounded picture languages. We discuss decidability and undecidability of the descriptability questions for reversal-bounded picture languages and prove several hierarchy theorems that show relations between families of (reversal-bounded) picture languages.

Theorem 3.1. *Let k be a nonnegative integer. It is decidable whether or not an arbitrary context-free π-grammar is a k-reversal-bounded grammar.*

Proof. A π-language is a k-reversal-bounded language if and only if it does not contain a π-word having more than k reversals. The set of all π-words having more than k reversals can be represented in regular expression by

$$E_k=\pi^*\big[r(\vec{\pi}^*l\overleftarrow{\pi}^*r)^{\lfloor (k+1)/2\rfloor}(\vec{\pi}^*l)^{(k+1)\bmod 2}$$
$$+\,l(\overleftarrow{\pi}^*r\vec{\pi}^*l)^{\lfloor (k+1)/2\rfloor}(\overleftarrow{\pi}^*r)^{(k+1)\bmod 2}\big]\pi^*.$$

For a context-free π-grammar G, the π-language $L'=L(G)\cap E_k$ is a context-free π-language, which is constructible from G and E_k. G is k-reversal-bounded if and only if L' is empty. The emptiness problem for context-free languages is decidable [5]. □

Theorem 3.2. *Let k be a nonnegative integer. It is not partially decidable whether an arbitrary context-sensitive π-grammar is a k-reversal-bounded grammar.*

Proof. Reduction from the emptiness problem for context-sensitive languages, which is not partially decidable [5]. Let G be an arbitrary context-sensitive π-grammar. Let E_k be the regular π-language defined in the proof of Theorem 3.1. As the family of context-sensitive languages is effectively closed under concatenation with a regular language, a context-sensitive π-grammar G' such that $L(G')=L(G)\cdot E_k$ can be constructed from G and E_k. It is easy to see that $L(G)=\emptyset$ if and only if G' is k-reversal-bounded. □

Theorem 3.3. *Let k be a nonnegative integer. If G is a regular (linear, context-free, context-sensitive, or recursively enumerable) π-grammar, then the set of all π-words in $L(G)$ having at most k reversals is a regular (linear, context-free, context-sensitive, or recursively enumerable) π-language that can be effectively constructed.*

Proof. The set of all π-words having at most k reversals is $E'_k=\pi^*-E_k$, where E_k is the regular π-language defined in the proof of Theorem 3.1. The family of regular (linear, context-free, context-sensitive, or recursively enumerable) languages is effectively closed under intersection with a regular language. So, $L(G)\cap E'_k$ is a regular (linear, context-free, context-sensitive, or recursively enumerable) π-language, which is constructible if $L(G)$ is. □

Let $\mathscr{D}_{\mathrm{REG}}$ ($\mathscr{D}_{\mathrm{LIN}}$, $\mathscr{D}_{\mathrm{CF}}$, $\mathscr{D}_{\mathrm{CS}}$, and $\mathscr{D}_{\mathrm{RE}}$) be the family of drawn regular (linear, context-free, context-sensitive, and recursively enumerable) picture languages. Similarly, define $\mathscr{B}_{\mathrm{REG}}$, $\mathscr{B}_{\mathrm{LIN}}$, $\mathscr{B}_{\mathrm{CF}}$, $\mathscr{B}_{\mathrm{CS}}$, and $\mathscr{B}_{\mathrm{RE}}$ for the corresponding families of basic picture languages.

Theorem 3.4 (Maurer et al. [9]). *The following relations are true*: (1) $\mathscr{D}_{\mathrm{REG}}\subsetneq\mathscr{D}_{\mathrm{LIN}}\subsetneq\mathscr{D}_{\mathrm{CF}}\subsetneq\mathscr{D}_{\mathrm{CS}}=\mathscr{D}_{\mathrm{RE}}$ *and* (2) $\mathscr{B}_{\mathrm{REG}}\subsetneq\mathscr{B}_{\mathrm{LIN}}\subsetneq\mathscr{B}_{\mathrm{CF}}\subsetneq\mathscr{B}_{\mathrm{CS}}=\mathscr{B}_{\mathrm{RE}}$.

For each integer $k\geqslant 0$, let $\mathscr{D}_{\mathrm{REG}}(k)$ ($\mathscr{D}_{\mathrm{LIN}}(k)$, $\mathscr{D}_{\mathrm{CF}}(k)$, $\mathscr{D}_{\mathrm{CS}}(k)$, and $\mathscr{D}_{\mathrm{RE}}(k)$) be the family of drawn k-reversal-bounded regular (linear, context-free, context-sensitive, and recursively enumerable) picture languages. Similarly, we define, for each $k\geqslant 0$, $\mathscr{B}_{\mathrm{REG}}(k)$, $\mathscr{B}_{\mathrm{LIN}}(k)$, $\mathscr{B}_{\mathrm{CF}}(k)$, $\mathscr{B}_{\mathrm{CS}}(k)$, and $\mathscr{B}_{\mathrm{RE}}(k)$ for the families of basic k-reversal-bounded picture languages.

Theorem 3.5. *For every $k\geqslant 0$, the following relations hold*: (1) $\mathscr{D}_{\mathrm{REG}}(k)\subsetneq\mathscr{D}_{\mathrm{LIN}}(k)\subsetneq\mathscr{D}_{\mathrm{CF}}(k)\subsetneq\mathscr{D}_{\mathrm{CS}}(k)\subsetneq\mathscr{D}_{\mathrm{RE}}(k)$ *and* (2) $\mathscr{B}_{\mathrm{REG}}(k)\subsetneq\mathscr{B}_{\mathrm{LIN}}(k)\subsetneq\mathscr{B}_{\mathrm{CF}}(k)\subsetneq\mathscr{B}_{\mathrm{CS}}(k)\subsetneq\mathscr{B}_{\mathrm{RE}}(k)$.

Proof. Let k be an arbitrary nonnegative integer and let w_k be a π-word defined by

$$w_k=r(dldr)^{\lfloor k/2\rfloor}(dl)^{k \bmod 2}.$$

Let $L_{k,1}=\{r^i d^i w_k \mid i\geqslant 0\}$, $L_{k,2}=\{r^i d^i r^j d^j w_k \mid i\geqslant 0,\ j\geqslant 0\}$, and $L_{k,3}=\{r^i d^i r^i w_k \mid i\geqslant 0\}$. Then, $L_{k,1}, L_{k,2}$, and $L_{k,3}$ are k-reversal-bounded π-languages. Using the iteration theorems for regular, linear, and context-free picture languages [7], it can be easily seen that the picture languages described by $L_{k,1}, L_{k,2}$, and $L_{k,3}$ are not regular, linear, and context-free picture languages, respectively. However, obviously, $L_{k,1}, L_{k,2}$ and $L_{k,3}$ are, respectively, linear, context-free, and context-sensitive π-languages. Hence, we have $\mathscr{D}_{\mathrm{REG}}(k)\subsetneq\mathscr{D}_{\mathrm{LIN}}(k)\subsetneq\mathscr{D}_{\mathrm{CF}}(k)\subsetneq\mathscr{D}_{\mathrm{CS}}(k)$ and $\mathscr{B}_{\mathrm{REG}}(k)\subsetneq\mathscr{B}_{\mathrm{LIN}}(k)\subsetneq\mathscr{B}_{\mathrm{CF}}(k)\subsetneq\mathscr{B}_{\mathrm{CS}}(k)$.

To show the last containment relations, we use a diagonalizational argument. Let $G_1, G_2, \ldots$ be a standard enumeration of all context-sensitive π-grammars. By Theorem 3.3, for each G_i, $i\geqslant 1$, a π-grammar G_i' that generates all π-words in $L(G_i)$ having at most k reversals can be effectively constructed. Let L_{d} be a π-language defined by

$$L_{\mathrm{d}}=\{r^i \mid i\geqslant 1,\ \mathrm{dpic}(r^i)\notin\mathrm{dpic}(G_i')\}.$$

Suppose that $\mathrm{dpic}(L_{\mathrm{d}})$ is a k-reversal-bounded context-sensitive picture language. As every k-reversal-bounded context-sensitive π-grammar is a context-sensitive π-grammar, there is an integer $i\geqslant 1$ such that $\mathrm{dpic}(L_{\mathrm{d}})=\mathrm{dpic}(G_i)=\mathrm{dpic}(G_i')$. Consider the π-word r^i. We have $r^i\in L_{\mathrm{d}}$ if and only if $\mathrm{dpic}(r^i)\notin\mathrm{dpic}(G_i')$ (by definition of L_{d}) if and only if $\mathrm{dpic}(r^i)\notin\mathrm{dpic}(L_{\mathrm{d}})$ (since $\mathrm{dpic}(L_{\mathrm{d}})=\mathrm{dpic}(G_i')$) if and only if $r^i\notin L_{\mathrm{d}}$ (since both the π-word r^i and the π-language L_{d} are retreat-free, where a retreat is a word in $\{ud, du, lr, rl\}$ [9]). This is a contradiction. So, $\mathrm{dpic}(L_{\mathrm{d}})$ is not a k-reversal-bounded context-sensitive picture language.

We shall show, however, that $\mathrm{dpic}(L_{\mathrm{d}})$ is a k-reversal-bounded recursively enumerable picture language. A k-reversal-bounded Turing machine M accepting exactly the π-language L_{d} can be constructed as follows. Given an arbitrary π-word w, checking whether w is of the form r^i for some $i\geqslant 1$ is trivial. If so, the ith context-sensitive π-grammar G_i can be determined. From G_i, a k-reversal-bounded context-sensitive π-grammar G_i' generating all k-reversal-bounded π-words in $L(G_i)$ can be effectively constructed (Theorem 3.3). Note that there are only finitely many π-words that have at most k reversals and describe the picture $\mathrm{dpic}(r^i)$; denote the set of such π-words by R_i. We have $w\in L_{\mathrm{d}}$ if and only if $\mathrm{dpic}(r^i)\notin\mathrm{dpic}(G_i')$ if and only if $L(G_i')\cap R_i=\emptyset$. The intersection emptiness of a context-sensitive language with a finite set is decidable.

It follows that $\mathscr{D}_{\mathrm{CS}}\subsetneq\mathscr{D}_{\mathrm{RE}}$. It is straightforward to observe that a similar argument proves $\mathscr{B}_{\mathrm{CS}}\subsetneq\mathscr{B}_{\mathrm{RE}}$. Thus, the theorem follows. □

Theorem 3.6. *For every* $k\geqslant 0$, *the following relations hold*: (1) $\mathscr{D}_{\mathrm{REG}}(k)\subsetneq\mathscr{D}_{\mathrm{REG}}(k+1)$, (2) $\mathscr{D}_{\mathrm{LIN}}(k)\subsetneq\mathscr{D}_{\mathrm{LIN}}(k+1)$, (3) $\mathscr{D}_{\mathrm{CF}}(k)\subsetneq\mathscr{D}_{\mathrm{CF}}(k+1)$, (4) $\mathscr{D}_{\mathrm{CS}}(k)\subsetneq\mathscr{D}_{\mathrm{CS}}(k+1)$, *and* (5) $\mathscr{D}_{\mathrm{RE}}(k)\subsetneq\mathscr{D}_{\mathrm{RE}}(k+1)$.

Proof. For every $k\geqslant 0$, let w_k be the π-word defined in the proof of Theorem 3.5 and consider the singleton set L_k consisting of w_k. Let $\mathscr{L}$ denote any one of

$\mathscr{D}_{\mathrm{REG}}$, $\mathscr{D}_{\mathrm{LIN}}$, $\mathscr{D}_{\mathrm{CF}}$, $\mathscr{D}_{\mathrm{CS}}$, and $\mathscr{D}_{\mathrm{RE}}$. Then we have $\mathrm{dpic}(L_{k+1}) \in \mathscr{L}(k+1)$, but $\mathrm{dpic}(L_{k+1}) \notin \mathscr{L}(k)$. □

Theorem 3.7. *For every $k \geqslant 0$, the following relations hold*: (1) $\mathscr{B}_{\mathrm{REG}}(k) \subsetneq \mathscr{B}_{\mathrm{REG}}(k+1)$, (2) $\mathscr{B}_{\mathrm{LIN}}(k) \subsetneq \mathscr{B}_{\mathrm{LIN}}(k+1)$, (3) $\mathscr{B}_{\mathrm{CF}}(k) \subsetneq \mathscr{B}_{\mathrm{CF}}(k+1)$, (4) $\mathscr{B}_{\mathrm{CS}}(k) \subsetneq \mathscr{B}_{\mathrm{CS}}(k+1)$, *and* (5) $\mathscr{B}_{\mathrm{RE}}(k) \subsetneq \mathscr{B}_{\mathrm{RE}}(k+1)$.

Proof. Similar to the proof of Theorem 3.6. □

Let $\mathscr{D}_{\mathrm{REG}}(\ )$ ($\mathscr{D}_{\mathrm{LIN}}(\ )$, $\mathscr{D}_{\mathrm{CF}}(\ )$, $\mathscr{D}_{\mathrm{CS}}(\ )$, and $\mathscr{D}_{\mathrm{RE}}(\ )$) be the family of reversal-bounded regular (linear, context-free, context-sensitive, and recursively enumerable) picture languages, i.e., the union over all $k \geqslant 0$ of $\mathscr{D}_{\mathrm{REG}}(k)$ ($\mathscr{D}_{\mathrm{LIN}}(k)$, $\mathscr{D}_{\mathrm{CF}}(k)$, $\mathscr{D}_{\mathrm{CS}}(k)$, and $\mathscr{D}_{\mathrm{RE}}(k)$). The families $\mathscr{B}_{\mathrm{REG}}(\ )$, $\mathscr{B}_{\mathrm{LIN}}(\ )$, $\mathscr{B}_{\mathrm{CF}}(\ )$, $\mathscr{B}_{\mathrm{CS}}(\ )$, and $\mathscr{B}_{\mathrm{RE}}(\ )$ are defined similarly.

Theorem 3.8. *The following relations are true*: (1) $\mathscr{D}_{\mathrm{REG}}(\ ) \subsetneq \mathscr{D}_{\mathrm{LIN}}(\ ) \subsetneq \mathscr{D}_{\mathrm{CF}}(\ ) \subsetneq \mathscr{D}_{\mathrm{CS}}(\ ) \subsetneq \mathscr{D}_{\mathrm{RE}}(\ )$ *and* (2) $\mathscr{B}_{\mathrm{REG}}(\ ) \subsetneq \mathscr{B}_{\mathrm{LIN}}(\ ) \subsetneq \mathscr{B}_{\mathrm{CF}}(\ ) \subsetneq \mathscr{B}_{\mathrm{CS}}(\ ) \subsetneq \mathscr{B}_{\mathrm{RE}}(\ )$.

Proof. The picture languages described by $L_{k,1}$, $L_{k,2}$, and $L_{k,3}$, defined in the proof of Theorem 3.5, are not in $\mathscr{D}_{\mathrm{REG}}(\ )$, $\mathscr{D}_{\mathrm{LIN}}(\ )$, and $\mathscr{D}_{\mathrm{CF}}(\ )$, respectively, for any $k \geqslant 0$, but are in $\mathscr{D}_{\mathrm{LIN}}(\ )$, $\mathscr{D}_{\mathrm{CF}}(\ )$, and $\mathscr{D}_{\mathrm{CS}}(\ )$, respectively. Thus, we have $\mathscr{D}_{\mathrm{REG}}(\ ) \subsetneq \mathscr{D}_{\mathrm{LIN}}(\ ) \subsetneq \mathscr{D}_{\mathrm{CF}}(\ ) \subsetneq \mathscr{D}_{\mathrm{CS}}(\ )$. Similarly, we have $\mathscr{B}_{\mathrm{REG}}(\ ) \subsetneq \mathscr{B}_{\mathrm{LIN}}(\ ) \subsetneq \mathscr{B}_{\mathrm{CF}}(\ ) \subsetneq \mathscr{B}_{\mathrm{CS}}(\ )$.

The last containment relations can be proved by using a diagonalizational argument similar to the one used in the proof of Theorem 3.5. Let $G_1, G_2, \ldots$ be an enumeration of all context-sensitive π-grammars. Then, for each $n \geqslant 1$ and each $k \geqslant 0$, a k-reversal-bounded context-sensitive π-grammar $G_{n,k}$ generating all π-words in $L(G_n)$ having at most k reversals can be effectively constructed (Theorem 3.3). Let f be a bijective mapping from the positive integers onto the pairs (n, k), $n \geqslant 1$, $k \geqslant 0$, and let

$$L_{\mathrm{d}} = \{r^i \mid i \geqslant 1,\ \mathrm{dpic}(r^i) \notin \mathrm{dpic}(G_{f(i)})\}.$$

Suppose that $\mathrm{dpic}(L_{\mathrm{d}})$ is a k-reversal-bounded context-sensitive picture language for some $k \geqslant 0$. Then, $\mathrm{dpic}(L_{\mathrm{d}}) = \mathrm{dpic}(G_n) = \mathrm{dpic}(G_{n,k})$ for some $n \geqslant 1$. Let $f^{-1}((n, k)) = i$. Then, we have $r^i \in L_{\mathrm{d}}$ if and only if $\mathrm{dpic}(r^i) \notin \mathrm{dpic}(G_{n,k})$ if and only if $\mathrm{dpic}(r^i) \notin \mathrm{dpic}(L_{\mathrm{d}})$ if and only if $r^i \notin L_{\mathrm{d}}$; a contradiction.

To show that L_{d} is a reversal-bounded recursively enumerable π-language, suppose that w is an arbitrary π-word. If w is of the form r^i then $f(i) = (n, k)$ can be computed. From this, the grammars G_n and $G_{n,k}$ can be constructed. Let $R_{i,k}$ be the finite set of k-reversal-bounded π-words that describe the picture $\mathrm{dpic}(r^i)$. It needs only to be tested whether or not $L(G_{n,k}) \cap R_{i,k} = \emptyset$, which is decidable, to accept r^i. □

Theorem 3.9. *The following relations are true*: (1) $\mathscr{D}_{\mathrm{REG}}(\ ) \subsetneq \mathscr{D}_{\mathrm{REG}}$, (2) $\mathscr{D}_{\mathrm{LIN}}(\ ) \subsetneq \mathscr{D}_{\mathrm{LIN}}$, (3) $\mathscr{D}_{\mathrm{CF}}(\ ) \subsetneq \mathscr{D}_{\mathrm{CF}}$, (4) $\mathscr{D}_{\mathrm{CS}}(\ ) \subsetneq \mathscr{D}_{\mathrm{CS}}$, *and* (5) $\mathscr{D}_{\mathrm{RE}}(\ ) \subsetneq \mathscr{D}_{\mathrm{RE}}$.

Proof. Let w_k be the π-word defined in the proof of Theorem 3.5. Then $L=\{w_k \mid k\geq 0\}$ is a regular π-language, and so, $\text{dpic}(L)\in\mathscr{D}_{\text{REG}}$. However, $\text{dpic}(L)\notin\mathscr{D}_{\text{RE}}(\ )$ since L is not a k-reversal-bounded π-language for any $k\geq 0$. □

Theorem 3.10. *The following relations are true*: (1) $\mathscr{B}_{\text{REG}}(\ )\subsetneq\mathscr{B}_{\text{REG}}$, (2) $\mathscr{B}_{\text{LIN}}(\ )\subsetneq\mathscr{B}_{\text{LIN}}$, (3) $\mathscr{B}_{\text{CF}}(\ )\subsetneq\mathscr{B}_{\text{CF}}$, (4) $\mathscr{B}_{\text{CS}}(\ )\subsetneq\mathscr{B}_{\text{CS}}$, *and* (5) $\mathscr{B}_{\text{RE}}(\ )\subsetneq\mathscr{B}_{\text{RE}}$.

Proof. Similar to the proof of Theorem 3.9. □

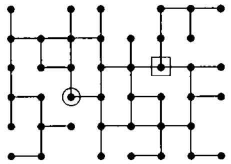
4. Polynomial-time solvable membership problems

The (*picture*) *membership problem* for a π-language L is to decide whether or not a drawn (basic) picture p can be described by a word in L, i.e., whether or not $p\in\text{dpic}(L)$ ($p\in\text{bpic}(L)$). The membership problem is NP-complete for regular, linear, and context-free picture languages [8, 12] and is solvable in $O(n)$ time for stripe regular picture languages [12] and three-way regular picture languages [6] and in $O(n^{12})$ time for three-way context-free picture languages [6]. It is easy to see that every 0-reversal-bounded π-language can be represented by the union of two three-way π-languages. That is, if L is a 0-reversal-bounded π-language, then $L=L_1\cup L_2$, where $L_1=L\cap\vec{\pi}^*$ and $L_2=L\cap\overleftarrow{\pi}^*$. It follows that the membership problem for 0-reversal-bounded picture languages can be solved by the polynomial-time recognition algorithms for three-way picture languages. We shall show that for every $k\geq 1$, the membership problem for k-reversal-bounded regular picture languages can be solved in $O(n^{4k+4})$ time. The discussion on the membership complexity will continue in the next section where an NP-completeness result is presented.

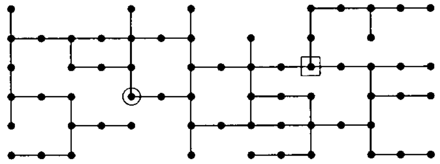
Let k be a positive integer and let L be a k-reversal-bounded regular π-language. Let p be an arbitrary attached drawn picture given as input. If p contains no horizontal line segment then the membership of $\langle p\rangle$ in $\text{dpic}(L)$ can be tested by using the linear-time algorithm for a stripe regular picture language [12]. So, we shall assume that p contains at least one horizontal line segment.

We first consider a transformation of L and p by which our recognition algorithm is simplified. Let h be a homomorphism, mapping π^* into $\{u,d,rr,ll\}^*$, defined by $h(u)=u$, $h(d)=d$, $h(r)=rr$, and $h(l)=ll$. One can easily transform the input picture p into p' so that $\langle p\rangle\in\text{dpic}(L)$ if and only if $\langle p'\rangle\in\text{dpic}(h(L))$. Such a picture p' with its leftmost points on the straight line $x=1$ (and its rightmost points on the straight line $x=2n+1$, for some $n>1$) will be denoted by $\bar{h}(p)$. An example of a drawn picture p and its transformation $\bar{h}(p)$ is given in Fig. 2a, b.

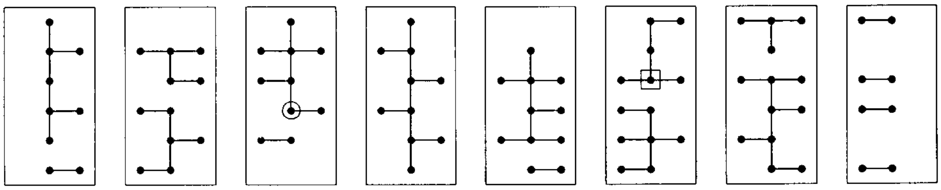
Let $M=(Q,\pi,\delta,q_0,F)$ be a k-reversal-bounded finite π-automaton such that $L(M)=h(L)$, where Q is the set of states, π is the input alphabet, $\delta: Q\times(\pi\cup\{\lambda\})\to 2^Q$ is the transition function, $q_0\in Q$ is the initial state, and $F\subseteq Q$ is the set of accepting states. We shall assume, without loss of generality, that M is a *normalized finite automaton*



(a)



(b)



(c)

Fig. 2. A picture p, the transformation $\bar{h}(p)$, and the partition of $\bar{h}(p)$.

such that there is a unique accepting state q_f (so, $|F|=1$), there is no transition to q_0 from any state, and there is no transition from q_f to any state.

For each integer $i\in Z$, the *ith vertical stripe of* M_0, denoted by M_0^i, is the set of all points $(j,j')\in M_0$ such that $2i-2\leqslant j\leqslant 2i$ and $j'\in Z$. Namely, M_0^i is the vertical stripe of width two centered at $x=2i-1$. The *set of lines in* M_0^i, denoted by M_1^i, is the set $\{\{v,v'\}\in M_1 \mid v,v'\in M_0^i\}$.

Let $\bar{h}(p)=(r,s,e)$. The lines of $\bar{h}(p)$ can be partitioned (or sliced) into n nonempty subpictures as follows. For each $i=1,2,\ldots,n$, the *ith block of* $\bar{h}(p)$, denoted by p_i, is defined by

$$p_i=\begin{cases}(r\cap M_1^i,s,e) & \text{if } s\in M_0^i \text{ and } e\in M_0^i,\\ (r\cap M_1^i,s,\#) & \text{if } s\in M_0^i \text{ and } e\notin M_0^i,\\ (r\cap M_1^i,\#,e) & \text{if } s\notin M_0^i \text{ and } e\in M_0^i,\\ (r\cap M_1^i,\#,\#) & \text{if } s\notin M_0^i \text{ and } e\notin M_0^i,\end{cases}$$

where $\#$ is a special symbol to represent the nonexistence in p_i of the start or end point of $\bar{h}(p)$. Hence, e.g., $(r \cap M_1^i, \#, \#)$ is simply an attached basic picture. Note that the start point and the end point of $\bar{h}(p)$ can each appear in exactly one block according to the above picture-slicing method. Figure 2c shows the partition of the drawn picture described in Fig. 2b.

The recognition algorithm we are going to describe below is based on the "crossing sequence" argument on subpictures $p_1, p_2, \ldots, p_n$ of $\bar{h}(p)$, which is similar to the idea behind the well-known simulation of two-way finite automata by one-way finite automata [5]. Recall that in the simulation of a two-way finite automaton the crossing means the crossing of the boundary of two consecutive cells of the input tape by the simulated two-way finite automaton and a crossing sequence is a sequence of states that can be entered by the simulated automaton when crossing the same boundary in a computation. A one-way finite automaton simulates by moving one way on its input tape and keeping track of a valid crossing sequence at the current boundary; it advances one cell to the right by matching a crossing sequence with the current one at the right end. The simulation works here since the maximum number of reversals of the tape head (or, in other words, the maximum length of a crossing sequence) of the simulated automaton in a shortest accepting computation can be bounded.

We shall proceed similarly. As in the simulation of two-way finite automata, we work from left to right on the blocks $p_1, p_2, \ldots, p_n$. Suppose that $\langle \bar{h}(p) \rangle = \mathrm{dpic}(w)$ for some π-word $w \in L(M)$. Then the crossing here will mean the crossing of the boundary of two consecutive blocks of $\bar{h}(p)$, when the picture $\bar{h}(p)$ is drawn according to w. Note that the crossing can occur only through the unit-size horizontal lines in $p_1, p_2, \ldots, p_n$; call them *thorns* and the points that lie on the left and right ends of the thorns shared with the adjacent blocks the *connection points*. Now, the corresponding concept of a crossing sequence should be certain information that is enough to add (or glue) a new block to the current boundary when we work from left to right on the blocks of $\bar{h}(p)$. Intuitively, it is sufficient to check that there is a way to completely draw the new block, visiting it (and making reversals) certain bounded number of times, and that the so-drawn new block can be glued completely through the connection points dangling from the current block. Since there can be many different crossing sequences that can be validly glued at each boundary, we shall keep track of all possible crossing sequences of the current, rightmost block.

Let us first look more closely at the picture-drawing behavior of M on the blocks of $\bar{h}(p)$. For each $i = 1, 2, \ldots, n$, consider the way the ith block p_i is drawn when $\bar{h}(p)$ is drawn by M (i.e., when $\bar{h}(p)$ is drawn according to a π-word in $L(M)$). As noted before, M draws p_i by visiting it some number of times, each time drawing a nonempty subpicture of p_i. There are several different ways that M visits the block p_i. Namely, there are three ways to enter the block p_i: (1) from the left block, i.e., from p_{i-1}, (2) from the right block, i.e., from p_{i+1}, and (3) from itself, i.e., through start $(\bar{h}(p))$ which is located in p_i. There are also three ways to leave the block p_i: (1) to the left block, i.e., to p_{i-1}; (2) to the right block, i.e., to p_{i+1}; and (3) to itself, i.e., through end$(\bar{h}(p))$ which is located in p_i. The method of visiting p_i using the entering method (3) and the leaving method (3) can be used only when $n=1$. As we assume that $n \geqslant 2$, we can eliminate this case from consideration. Thus, there are basically eight different ways that M visits p_i. It is not difficult to see that, at every visit of p_i, M can make at most one reversal inside p_i. Namely, M always reads r's or l's in a pair and, when M visits p_i, it does so by consuming the second letter of the pair. If M reads an l or r again inside p_i, then it is the first symbol of such a pair, and so, M leaves the block p_i, since it must consume a letter of the same type again (i.e., M must move one step further in the same direction).

Let $\mathscr{C}$ be the set of all 5-tuples in $Q \times Q \times 2^{M_1} \times M_0 \times M_0$. For each $i=1,2,\ldots,n$, a 5-tuple (q,q',r',s',e') in $\mathscr{C}$ is called *p_i-valid* if M can consume a π-word x, going from state q to state q', such that (1) $\mathrm{dpic}(x)=\langle r',s',e'\rangle$; (2) $r' \subseteq \mathrm{base}(p_i)$; (3) if $s' \neq s$ ($e' \neq e$), then s' (e') is a connection point; and (4) $q=q_0$ ($q'=q_f$) if and only if $s'=s$ ($e'=e$).

We classify, for each $i=1,2,\ldots,n$, the p_i-valid 5-tuples into the eight different methods of visiting the block p_i. Let $LL, LR, LS, RL, RR, RS, SL$, and SR be tables of size n, where the letters L, R, and S in the names of the tables mean "left", "right", and "stay", respectively, and the first letter and the second letter indicate, respectively, the method of entering and leaving a block of $\bar{h}(p)$ as interpreted by the letter. For each $i=1,2,\ldots,n$, let $LL[i]$ contain all p_i-valid 5-tuples, say (q,q',r',s',e'), such that (1) M enters p_i from "left" by entering state q; (2) M draws $\langle r',s',e'\rangle$; and (3) M leaves p_i to "left" from state q'. Other tables are defined similarly by replacing the direction "left" in (1) and "left" in (3) of the definition of the LL table with appropriate directions.

Lemma 4.1. *The eight tables $LL, LR, LS, RL, RR, RS, SL$, and SR can be constructed in $O(|p|^3)$ time.*

Proof. Note that, for each $i=1,2,\ldots,n$, the ith block p_i has at most $k+1$ left thorns and at most $k+1$ right thorns since, otherwise, the picture $\bar{h}(p)$ cannot be drawn by a k-reversal-bounded π-word. (That is, between two consecutive visits of the same block, there occurs at least one reversal.) This means that there can be no more than $O(|p_i|^2)$ connected subpictures of p_i of the form (r',s',e') such that (q,q',r',s',e') is possibly p_i-valid for some $q,q' \in Q$. So, there are $O(|Q|^2 \cdot |p_i|^2)=O(|p_i|^2)$ distinct 5-tuples that need to be tested for p_i-validness. These candidate 5-tuples can be easily enumerated in a systematic way. Each candidate 5-tuple can be written in $O(|p_i|)$ time and whether it is p_i-valid or not can be tested in $O(|p_i|)$ time by using the linear-time recognition algorithm for a stripe regular picture language [12]. Therefore, finding all p_i-valid 5-tuples takes $O(|p_i|^3)$ time. The classification of the p_i-valid 5-tuples into the eight visiting methods can be done easily and requires no additional time. Altogether, the construction of the eight tables takes $\sum_{1 \leqslant i \leqslant n} O(|p_i|^3)=O(|\bar{h}(p)|^3)=O(|p|^3)$ time. $\square$

The eight tables constructed above contain and classify all possible 5-tuples that can be chosen by M in a single visit of a block of $\bar{h}(p)$. Let $t=(t_1,t_2,\ldots,t_m)$, $m \geqslant 1$, be a sequence of p_i-valid 5-tuples for some i and let $t_j=(q_j, q'_j, r_j, s_j, e_j)$, $1 \leqslant j \leqslant m$. The sequence t represents a way the automaton M visits the block p_i while drawing $\bar{h}(p)$ if the following conditions hold: (1) if $s \in V(p_i)$ then $t_1 \in SL[i] \cup SR[i]$ else no element of t is in $SL[i] \cup SR[i]$; (2) if $e \in V(p_i)$ then $t_m \in LS[i] \cup RS[i]$ else no element of t is in $LS[i] \cup RS[i]$; (3) for each $j=1,2,\ldots,m-1$, if t_j is in $LL[i]$, $RL[i]$, or $SL[i]$, then t_{j+1} is in $LL[i]$, $LR[i]$, or $LS[i]$; (4) for each $j=1,2,\ldots,m-1$, if t_j is in $LR[i]$, $RR[i]$, or $SR[i]$, then t_{j+1} is in $RL[i]$, $RR[i]$, or $RS[i]$; (5) $\bigcup_{1 \leqslant j \leqslant m} r_j = \text{base}(p_i)$; and (6) $m \leqslant k+1$. A sequence t satisfying these six conditions is called a *p_i-complete m-sequence*. Let X be a table of size n such that, for each $i=1,2,\ldots,n$, $X[i]$ contains all p_i-complete m-sequences, $1 \leqslant m \leqslant k+1$.

Lemma 4.2. *The table X can be constructed in $O(|p|^{2k+3})$ time.*

Proof. By Lemma 4.2, the eight tables can be constructed in $O(|p|^3)$ time. For each $i=1,2,\ldots,n$ and each $j=1,2,\ldots,k+1$, there are $O(|p_i|^{2j})$ distinct j-sequences that need to be tested for p_i-completeness. The test for p_i-completeness can be done in $O(|p_i|)$ time since the picture union operation in step (5) above takes $O(|p_i|)$ time and all other steps take $O(1)$ time. So, the set of all p_i-complete j-sequences can be obtained in $O(|p_i|^{2j+1})$ time. Hence, the time taken to construct the table X from the eight tables is

$$\sum_{i=1}^{n} \sum_{j=1}^{k+1} O(|p_i|^{2j+1}),$$

which is $O(|p|^{2k+3})$. So, the table X can be constructed in $O(|p|^3)+O(|p|^{2k+3}) = O(|p|^{2k+3})$ time. $\square$

Suppose that the picture $\bar{h}(p)$ is drawn by M. The drawing of $\bar{h}(p)$ by M results in n p_i-complete sequences, one for each $i=1,2,\ldots,n$, that represent the picture-drawing behavior of M on the corresponding blocks of $\bar{h}(p)$. We have all possible candidates for such p_i-complete sequences in the table X. Thus, the picture $\bar{h}(p)$ is drawn by M if there exist p_i-complete sequences $x_i \in X[i]$, $1 \leqslant i \leqslant n$, such that, for each $i=1,2,\ldots,n-1$, the elements x_i and x_{i+1} together represent a locally consistent computation by M on the blocks p_i and p_{i+1}. That is, x_i and x_{i+1} can be glued together so that the methods of visiting the blocks p_i and p_{i+1} as indicated in the sequences x_i and x_{i+1} are consistent at the boundary of p_i and p_{i+1}. As M is a k-reversal-bounded automaton, such a local consistency will ensure that the elements $x_1, x_2, \ldots, x_n$ together represent, in fact, a globally consistent computation by M that draws $\bar{h}(p)$, i.e., the total number of reversals involved in $x_1, x_2, \ldots, x_n$ is at most k.

To determine the existence of such sequences, we shall work from left to right on $X[1], X[2], \ldots, X[n]$, checking connections between sequences complete for consecutive blocks of $\bar{h}(p)$ and eliminating all unmatchable sequences. For example, if we have worked on $X[1], X[2], \ldots, X[i]$ and are about to consider $X[i+1]$, then what we have to do is to eliminate all sequences in $X[i+1]$ that cannot be right-matched with any sequence in $X[i]$. When we are done, we will have in $X[n]$ only the elements that can be the last elements of some sequence $x_i \in X[i]$, $1 \leqslant i \leqslant n$, that can be glued completely in sequence. Thus, we need only check finally whether $X[n]$, after we sweep from left to right on $X[1], X[2], \ldots, X[n]$, is not empty to see if $\langle \bar{h}(p) \rangle \in \mathrm{dpic}(M)$.

We now describe how unmatchable sequences can be eliminated when we sweep from left to right on $X[1], X[2], \ldots, X[n]$. Let $t=(t_1, t_2, \ldots, t_m)$ be in $X[i]$ and $\bar{t}=(\bar{t}_1, \bar{t}_2, \ldots, \bar{t}_{m'})$ be in $X[i+1]$, for some $i \geqslant 1$. The 5-tuples in t of the type LL, LS, and SL (i.e., the 5-tuples from $LL[i]$, $LS[i]$, and $SL[i]$) have already been considered when elements of $X[i]$ were matched with those in $X[i-1]$ and are nothing to do with the matching condition for t and $\bar{t}$. (If $i=1$, then there are no such 5-tuples in t since p_1 contains no left thorn.) Similarly, the 5-tuples in $\bar{t}$ of the type RR, RS, and SR will be considered when elements of $X[i+1]$ are matched with those in $X[i+2]$ and are nothing to do with the matching condition for t and $\bar{t}$. (If $i+1=n$, then there are no such 5-tuples in $\bar{t}$ since p_n contains no right thorn.) Let j be the smallest integer such that t_j is of the type LR, RL, RR, RS, or SR and let j' be the smallest integer such that $\bar{t}_{j'}$ is of the type LL, LR, LS, RL, or SL. If there does not exist such a pair of integers, then there is no crossing between p_i and p_{i+1} indicated by t and $\bar{t}$, and so, t and $\bar{t}$ are not matchable. The elements t and $\bar{t}$ start communications (representing the crossings of the boundary between p_i and p_{i+1}) with t_j and $\bar{t}_{j'}$. There are two cases:

(1) The communication is activated by t, i.e., t_j is of the type LR or SR. Then $\bar{t}_{j'}$ must be of the type LL, LR, or LS;

(2) The communication is activated by $\bar{t}$, i.e., $\bar{t}_{j'}$ is of the type RL or SL. Then t_j must be of the type RL, RR, or RS.

It is easy to see that no other case can activate the correspondence between p_i and p_{i+1}. In either case, starting with t_j and $\bar{t}_{j'}$, the 5-tuples in t and $\bar{t}$ having direct relations with each other must represent a sequence of alternating moves of M between p_i and p_{i+1}. Figure 3 illustrates the precedence relations that must be satisfied between the 5-tuples in t and $\bar{t}$ (see the middle two columns). Figure 3 also describes the relative order of the types of consecutive 5-tuples in t (see the left two columns) and the relative order of the types of consecutive 5-tuples in $\bar{t}$ (see the right two columns), which are already satisfied in t and $\bar{t}$ from the conditions for a sequence to be a p_i- or p_{i+1}-complete sequence.

To understand the meaning of the relations between t and $\bar{t}$ described in Fig. 3, suppose that it is the case 1 above. Then M moves from left to right, i.e., from p_i to p_{i+1}. There must be a matching 5-tuple in $\bar{t}$ of the type LL, LR, or LS, i.e., $\bar{t}_{j'}$, as observed before. This fact is indicated in the figure by a solid arrow going from the rectangle containing LR and SR to the rectangle containing LL, LR, and LS. If $\bar{t}_{j'}$ is of the type LL, then M moves next from right to left, i.e., from p_{i+1} to p_i. There must be a matching 5-tuple in t of the type RL, RR, or RS and it must be exactly the next 5-tuple in t. This fact is indicated in the figure by a solid arrow going from LL to the rectangle containing RL, RR, and RS. If $\bar{t}_{j'}$ is of the type LR and there is another

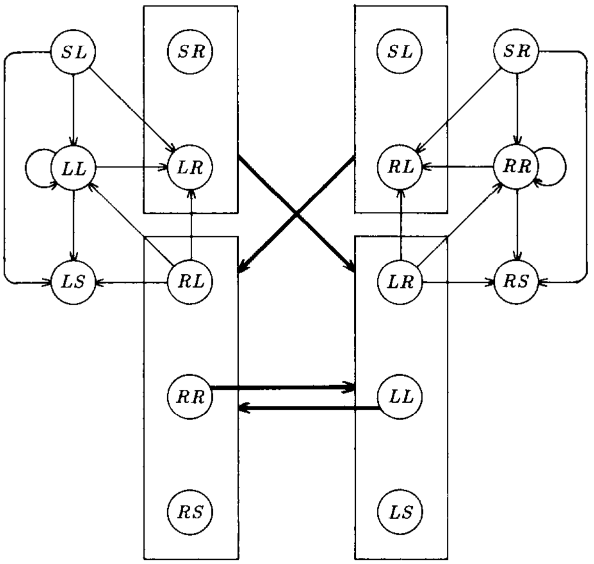


Fig. 3. Illustration of the glueing conditions.

5-tuple in $\bar{t}$ following $\bar{t}_{j'}$ that is not of the type RR or RS, then it must be exactly of the type RL. There must be a matching 5-tuple in t, again the next 5-tuple in t, of the type RL, RR, or RS. This fact is indicated by the portion of the figure containing the types RL, LR, RR, and RS in the third and fourth columns and the solid arrow going from the rectangle containing SL and RL to the rectangle containing RL, RR, and RS. It should be easy to see that the remaining 5-tuples in t and $\bar{t}$ must satisfy the precedence relations illustrated in the figure.

Observe that if we follow the precedence relations described in Fig. 3, alternately moving on t and $\bar{t}$, then either we end up with an unacceptable 5-tuple which is not indicated by a solid arrow in the figure or we eventually arrive at the end of t or $\bar{t}$. The first case means that t and $\bar{t}$ are unmatchable. If it is the end of t, then it must be of the type RL, RS, LL, or LS since a 5-tuple of the type SL can appear only as the first element of t and a 5-tuple of the type SR, LR, or RR pushes us to move to $\bar{t}$ as indicated by the solid arrows. At that point, there must be no 5-tuple left in $\bar{t}$, i.e., all elements in $\bar{t}$ must have already been considered. Similarly, if it is the end of $\bar{t}$, then it must be of the type LR, LS, RR, or RS and there must be no 5-tuple left in t.

Lemma 4.3. *For each $t \in X[i]$ and each $\bar{t} \in X[i+1]$, $1 \leqslant i \leqslant n-1$, whether t and $\bar{t}$ can be matched in sequence or not can be decided in* O(1) *time.*

Proof. The time taken to match two sequences t and $\bar{t}$ is proportional to $|t|+|\bar{t}|$, assuming an appropriate encoding or data structure for t and $\bar{t}$. Both t and $\bar{t}$ are length-bounded by $k+1$, and k is a constant. $\square$

Let MATCH be a function that performs the matching between elements in the table X as described above. That is, for each $i=1,2,\dots,n-1$, MATCH($X[i]$, $X[i+1]$) returns the modified version of $X[i+1]$ after performing matching between the elements of $X[i]$ and the elements of $X[i+1]$. For time consideration, if an element of $X[i+1]$ is going to be removed as the result of matching between $X[i]$ and $X[i+1]$, then MATCH will simply mark the removal of such an element rather than physically remove it.

The membership of $\langle \bar{h}(p)\rangle$ in dpic($h(L)$) (or equivalently, the membership of p in dpic(L)) can now be tested as follows:

Algorithm 4.4. *Membership test for reversal-bounded regular picture languages.*

Input: *An attached drawn picture p and a fixed reversal-bounded regular π-language L.*

Output: *"yes" if $\langle p\rangle \in$dpic(L), and "no" otherwise.*

Method:

(1) *Assume that a normalized finite automaton M such that $L(M)=h(L)$ has been obtained.*

(2) *Compute $\bar{h}(p)$ and let $\bar{h}(p)$ be partitioned into n nonempty subpictures. If $n=1$ then test the membership of $\bar{h}(p)$ in* dpic(M) *by using the recognition algorithm in* [6]. *If $n\geqslant 2$ then proceed to Step* (3).

(3) *Construct the table X from $\bar{h}(p)$ and M.*

(4) *for $i=1$ to $n-1$ step 1 do $X[i+1]:=MATCH(X[i],X[i+1])$.*

(5) *If there remains an element in $X[n]$ then output "yes", else output "no".*

The membership test of a basic picture can be done similarly with a slight modification of the process described so far. Suppose that p is an attached basic picture given as input. We first transform p into $\bar{h}(p)$ as before. As the start point and the end point of p (and $\bar{h}(p)$) are not specified, the tables LS, RS, SL, and SR will contain all p_i-valid 5-tuples such that every point in p_i corresponding to a point of the original picture p is considered to be the start point or the end point. From the eight tables classifying p_i-valid 5-tuples, the table X containing p_i-complete sequences can be constructed. We are now in a similar situation as for a drawn picture. Namely, $\langle \bar{h}(p)\rangle \in$bpic$(M)$ if there exist p_i-complete sequences $x_i \in X[i]$, $i=1,2,\dots,n$, that can be glued together and, in addition, among $x_1, x_2, \dots, x_n$ there is exactly one sequence whose first element is of the type SL or SR and there is exactly one sequence whose last element is of the type LS or RS. This additional condition can be checked by adding two bit vectors S and E to each p_i-complete sequence t. Suppose that the table X contains such augmented sequences (t,S,E) with $S=1$ $(E=1)$ if the first (last) element of t is of the type SL or SR (LS or RS) and $S=0$ $(E=0)$ otherwise. While sweeping from left to right on $X[1], X[2], \dots, X[n]$, we can modify as well the bit vectors S and E of each augmented sequence (t,S,E) in $X[i]$ so that $S=1$ $(E=1)$ if and only if there exist $x_j \in X[j]$, $1\leqslant j\leqslant i$, with $x_i=(t,S,E)$, that can be glued together such that exactly one sequence among $x_1, x_2, \dots, x_i$ is of the type SL or SR (LS or RS).

When we finish sweeping, we can simply check whether there is an element (t, S, E) in $X[n]$ such that $S = E = 1$, to see if $\langle p \rangle \in \mathrm{bpic}(L)$.

Theorem 4.5. *Let k be a positive integer. For every k-reversal-bounded regular π-language L, the drawn and basic versions of the membership problem for L can be solved in* $\mathrm{O}(|p|^{4k+4})$ *time, where p is the input picture.*

Proof. We consider the running time of Algorithm 4.4. As M is a fixed automaton which is not a part of input, the time taken in step 1 is ignored. Step 2 of the algorithm takes $\mathrm{O}(|p|)$ time since the transformation from p into $\bar{h}(p)$ takes $\mathrm{O}(|p|)$ time and the recognition algorithm in [6] takes $\mathrm{O}(|p|)$ time. By Lemma 4.2, step 3 of the algorithm takes $\mathrm{O}(|p|^{2k+3})$ time. Each $X[i]$ contains $\sum_{1 \leqslant j \leqslant k+1} \mathrm{O}(|p_i|^{2j}) = \mathrm{O}(|p_i|^{2k+2})$ elements (see the proof of Lemma 4.2). As the matching between two elements can be done in $\mathrm{O}(1)$ time (Lemma 4.3), the time taken in step 4 is

$$\sum_{i=1}^{n-1} \mathrm{O}(|p_i|^{2k+2}) \cdot \mathrm{O}(|p_{i+1}|^{2k+2}),$$

which is $\mathrm{O}(|p|^{4k+4})$. Step (5) takes $\mathrm{O}(|p|^{2k+2})$ time. Altogether, Algorithm 4.4 takes $\mathrm{O}(|p|^{4k+4})$ time. It is easy to see that the modified algorithm for basic pictures takes the same amount of time. □

5. An NP-completeness result

We show that there is a 1-reversal-bounded linear π-language L_0 describing a stripe picture language for which the membership problem is NP-complete. This resolves a problem left unsolved in [12], where the membership problem was shown to be linear-time solvable for stripe regular picture languages, but whether the membership problem for stripe context-free picture languages could be solved in polynomial time was left unanswered.

Theorem 5.1. *There is a 1-reversal-bounded linear π-language* L_0 *describing a stripe picture language for which the drawn and basic versions of the membership problem are NP-complete.*

Proof. It suffices to show that the given problems are NP-hard since the problems are already in NP [8]. We show a reduction from the *bounded post correspondence problem (BPCP)*, which is NP-complete, to the given problems. BPCP is to determine whether, for two lists $(x_1, x_2, \ldots, x_n)$ and $(y_1, y_2, \ldots, y_n)$, $n \geqslant 1$, of nonempty words over an alphabet Σ and a positive integer $k \leqslant n$, there exists a sequence of integers $i_1, i_2, \ldots, i_m$, $1 \leqslant m \leqslant k$ and $1 \leqslant i_j \leqslant n$ for all $j = 1, 2, \ldots, m$, such that $x_{i_1} x_{i_2} \cdots x_{i_m} = y_{i_1} y_{i_2} \cdots y_{i_m}$. BPCP is NP-complete when $|\Sigma| = 2$ [2].

Let (x, y, k) be an arbitrary instance of BPCP, where $x=(x_1, x_2, \ldots, x_n)$ and $y=(y_1, y_2, \ldots, y_n)$ are two lists of nonempty words over $\Sigma=\{a, b\}$ and $k \leqslant n$ is a positive integer. We construct a drawn picture p from x, y, and k, and a 1-reversal-bounded linear π-language L_0 describing a stripe picture language that does not depend on x, y, and k, and show that (x, y, k) is a positive instance of BPCP if and only if $p \in \mathrm{dpic}(L_0)$.

We shall first describe the construction of p. Let $h_x, h_y : \Sigma^* \to \vec{\pi}^*$ be homomorphisms defined by

$$h_x(a)=urdr, \qquad h_y(a)=drur,$$

$$h_x(b)=u^2rd^2r, \qquad h_y(b)=d^2ru^2r,$$

and let $\phi : \vec{\pi}^* \to \{l\}^*$ be a homomorphism defined by $\phi(u)=\phi(d)=\lambda$ and $\phi(r)=l$. Let $z=h_x(x_1)h_y(y_1)\cdots h_x(x_n)h_y(y_n)$ and

$$\langle ur\text{-}bridge\rangle = ur^2d,$$

$$\langle dl\text{-}bridge\rangle = dl^2u.$$

Let $w=\langle ur\text{-}bridge\rangle(rz\langle ur\text{-}bridge\rangle)^k(\langle dl\text{-}bridge\rangle\phi(z)l)^k\langle dl\text{-}bridge\rangle$ and let $p=\mathrm{dpic}(w)$. An example of such a picture p is shown in Fig. 4, where $n=3$, $k=3$, $x_1=a$, $y_1=b$, $x_2=b$, $y_2=a$, $x_3=bb$, and $y_3=a$.

Let us call each 2×2 square in p a *bridge* and the subpicture of p that lies between each pair of two consecutive bridges a *block*. The long horizontal line segment in each block that joins two bridges will be called a *spine*. The picture p consists of k identical blocks, separated by bridges. In each block, the lists x and y are described above and below, respectively, the spine, intermixing $x_1, x_2, \ldots, x_n$ and $y_1, y_2, \ldots, y_n$ one by one in order, by using the picture components for the symbols in x and y. The subpicture corresponding to each pair (x_i, y_i) in a block is called a *subblock*.

Now we describe the π-language L_0. Let $C=(h_x(a)+h_x(b)+h_y(a)+h_y(b))^*$ and let $\Gamma=\{a, b, \bar{a}, \bar{b}, a', b', \bar{a}', \bar{b}', \#\}$. Let A be a regular language over $\pi \cup \Gamma$ defined by

$$A=\langle ur\text{-}bridge\rangle(rC(a+b)^+(\bar{a}+\bar{b})^+C\langle ur\text{-}bridge\rangle)^+$$

$$\cdot(rC\langle ur\text{-}bridge\rangle)^*\,\#\,(\langle dl\text{-}bridge\rangle l^*)^*$$

$$\cdot(\langle dl\text{-}bridge\rangle l^*(a'+b')^+(\bar{a}'+\bar{b}')^+l^*)^+\langle dl\text{-}bridge\rangle,$$

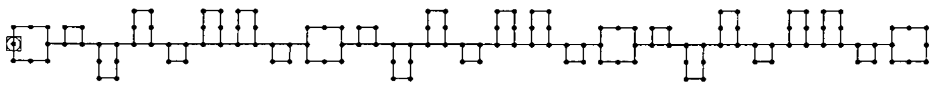


Fig. 4. An example of a picture constructed from an instance of BPCP.

and let $h:(\pi\cup\Gamma)^*\to\{a,b,\#\}^*$ be a homomorphism defined by

$$h(\sigma)=\lambda \quad \text{for each } \sigma\in\pi\cup\{\bar{a},\bar{b},\bar{a}',\bar{b}'\},$$

$$h(a)=h(a')=a,$$

$$h(b)=h(b')=b,$$

$$h(\#)=\#.$$

Let $B=\{z\in A \mid h(z)=w\#w^{\mathrm{R}} \text{ for some } w\in\{a,b\}^+\}$. Then, B is clearly a linear context-free language. For any homomorphism $f:(\pi\cup\Gamma)^*\to\pi^*$, $f(B)$ is also a linear context-free language. Let $L_0=f(B)$, where

$$f(\sigma)=\sigma \quad \text{for each } \sigma\in\pi,$$

$$f(a)=udr^2, \qquad f(b)=u^2d^2r^2,$$

$$f(\bar{a})=dur^2, \qquad f(\bar{b})=d^2u^2r^2,$$

$$f(a')=ldlu, \qquad f(b')=ld^2lu^2,$$

$$f(\bar{a}')=luld, \qquad f(\bar{b}')=lu^2ld^2,$$

$$f(\#)=\lambda.$$

It is easy to see that L_0 describes a stripe picture language. We claim that $(x,y,k)\in$BPCP if and only if $p\in\mathrm{dpic}(L_0)$, where p is the picture drawn for (x,y,k).

Suppose that $(x,y,k)\in$BPCP. Then, by definition, there is a sequence $i_1,i_2,\ldots,i_m$ $(1\leqslant m\leqslant k)$ such that $x_{i_1}x_{i_2}\cdots x_{i_m}=y_{i_1}y_{i_2}\cdots y_{i_m}$. Therefore, the string

$$v=x_{i_1}x_{i_2}\cdots x_{i_m}\#(y_{i_1}y_{i_2}\cdots y_{i_m})^{\mathrm{R}}$$

is in $\{w\#w^{\mathrm{R}} \mid w\in\{a,b\}^+\}$. It follows that there is a string z in B such that $h(z)=v$. Thus, $f(z)\in L_0$. We can in fact choose a string $z=z_1\#z_2$ in B such that

(1) $f(z_1)$ draws a subpicture of p, moving right in the plane, such that, for each $t=1,2,\ldots,m$, the picture components for the symbols a and b in the i_tth subblock of the tth block of p are partially drawn by using $f(a),f(b),f(\bar{a}),f(\bar{b})$, and the picture components for the symbols a and b in all other subblocks (including the subblocks located to the right of the mth block) are completely drawn by the strings in C, and

(2) $f(z_2)$ draws a subpicture of p, moving left in the plane, such that, for each $t=1,2,\ldots,m$, the picture components for the symbols a and b in the i_tth subblock of the tth block of p are partially drawn by using $f(a'),f(b'),f(\bar{a}'),f(\bar{b}')$, and the picture components for the symbols a and b in all other subblocks are drawn by l's.

As $f(a)$ and $f(\bar{a}')$ ($f(b)$ and $f(\bar{b}')$) together can completely draw a picture component for the symbol a (b) used in x and as $f(\bar{a})$ and $f(a')$ ($f(\bar{b})$ and $f(b')$) together can completely draw a picture component for the symbol a (b) used in y, it follows that $p=\mathrm{dpic}(z)$ for our choice of z.

Conversely, let $p=\mathrm{dpic}(f(z))$ for some $z\in B$. Let $z=z_1 \# z_2$. Suppose that, when $f(z_1)$ draws a subpicture of p, the π-words $f(a)$, $f(b), f(\bar{a}), f(\bar{b})$ are used in the i_tth subblock of the tth block of p, for $t=1,2,\ldots,m$ $(1\leqslant m\leqslant k)$. When $f(z_2)$ draws a subpicture of p, each picture component drawn by $f(a)$ $(f(b), f(\bar{a}), f(\bar{b}))$ in $f(z_1)$ must be completed by using $f(\bar{a}')$ $(f(\bar{b}'), f(a'), f(b'))$. Similarly, each picture component drawn by a string from C must be completed by using l's (otherwise, spines cannot be completely drawn) and each picture component drawn by $\langle$*ur-bridge*$\rangle$ must be completed by using $\langle$*dl-bridge*$\rangle$. The fact that $h(z_1)=(h(z_2))^{\mathrm{R}}$ implies that $x_{i_1}x_{i_2}\cdots x_{i_m}=y_{i_1}y_{i_2}\cdots y_{i_m}$. Therefore, $(x, y, k)\in\mathrm{BPCP}$.

The reader can easily observe that the proof given above works for the π-language L_0 and the basic picture p' obtained from p by removing the start and end points. □

6. Undecidability results

In this section, we prove an undecidability result on the membership problem for three-way context-sensitive picture languages and state some undecidability results on other decision problems for three-way picture languages, proved in other papers, in a stronger form when possible. This will give a better view of the reversal-bounded picture languages since every three-way picture language is a 0-reversal-bounded picture language.

In [9], it was shown to be undecidable whether or not $p\in\mathrm{dpic}(G)$ $(p\in\mathrm{bpic}(G))$ for an arbitrary drawn (basic) picture p and an arbitrary context-sensitive π-grammar G. This result is strengthened as follows:

Theorem 6.1. *There is a drawn (basic) picture p such that it is undecidable whether or not $p\in\mathrm{dpic}(G)$ $(p\in\mathrm{bpic}(G))$ for an arbitrary three-way context-sensitive π-grammar G describing a stripe picture language.*

Proof. Reduction from the problem of whether or not a Turing machine accepts the empty string, which is undecidable [5]. Let M be an arbitrary Turing machine accepting a language $L\subseteq\{0,1\}^*$. Then there is a context-sensitive language $L'\subseteq\{0,1,2,3\}^*$, which can be constructed from M, such that (1) L' consists of words of the form 2^i3w, where $i\geqslant 0$ and $w\in L$; and (2) for every $w\in L$, there is an integer $i\geqslant 0$ such that $2^i3w\in L'$ [11]. Thus, M accepts the empty string λ if and only if $2^i3\in L'$ for some $i\geqslant 0$. Define a homomorphism $h:\{0,1,2,3\}^*\to\vec{\pi}^*$ by $h(0)=u$, $h(1)=d$, $h(2)=ud$, and $h(3)=udr$. As the family of context-sensitive languages is closed under λ-free homomorphisms, $h(L')$ is a context-sensitive language. Let G be a context-sensitive grammar such that $L(G)=h(L')$. Then, G is a three-way language describing a stripe picture language. Let $p=\mathrm{dpic}(udr)$ and $q=\mathrm{bpic}(udr)$. We have $2^i3\in L'$ (for some $i\geqslant 0$) if and only if $(ud)^{i+1}r\in h(L')$ if and only if $p\in\mathrm{dpic}(G)$ $(q\in\mathrm{bpic}(G))$ since the π-word udr describes the same picture as the π-word $(ud)^{i+1}r$ does, for every $i\geqslant 0$. □

The *equivalence problem* for drawn-picture languages is to determine whether or not $\mathrm{dpic}(L_1) = \mathrm{dpic}(L_2)$ for two π-languages L_1 and L_2. The *containment problem* and the *intersection emptiness problem* for the same input are to determine whether or not $\mathrm{dpic}(L_1) \subseteq \mathrm{dpic}(L_2)$ and $\mathrm{dpic}(L_1) \cap \mathrm{dpic}(L_2) = \emptyset$, respectively. The *ambiguity problem* for drawn-picture languages is to determine whether or not a π-language L is an ambiguous picture description language, i.e., whether or not there exist two distinct π-words in L that describe the same drawn picture. These problems can be defined for basic-picture languages similarly. The equivalence problem, the containment problem, the intersection emptiness problem, and the ambiguity problem are all known to be undecidable for drawn and basic regular picture languages [7, 8, 12].

Theorem 6.2 (Kim [6]). *It is not partially decidable whether* (1) $\mathrm{dpic}(L_1) = \mathrm{dpic}(L_2)$, (2) $\mathrm{bpic}(L_1) = \mathrm{bpic}(L_2)$, (3) $\mathrm{dpic}(L_1) \subseteq \mathrm{dpic}(L_2)$, *and* (4) $\mathrm{bpic}(L_1) \subseteq \mathrm{bpic}(L_2)$, *for a three-way regular π-language L_1 and a three-way linear π-language L_2 such that both* $\mathrm{bpic}(L_1)$ *and bpic*(L_2) *are stripe picture languages.*

Theorem 6.3 (Kim [6]). *It is not partially decidable whether* (1) $\mathrm{dpic}(L_1) \cap \mathrm{dpic}(L_2) = \emptyset$, *and* (2) $\mathrm{bpic}(L) \cap \mathrm{bpic}(L) = \emptyset$, *for two three-way linear π-languages L_1 and L_2 such that both* $\mathrm{bpic}(L_1)$ *and* $\mathrm{bpic}(L_2)$ *are stripe picture languages.*

Theorem 6.4 (Kim [7]). *It is undecidable whether or not a three-way linear π-language L such that* $\mathrm{bpic}(L)$ *is a stripe picture language is an ambiguous picture description language.*

Theorem 16 in [10] has been successively restated in stronger forms in [12, Theorem 5.11], comprising the concept of stripe picture languages, and then in [6, Theorem 4.4], comprising the concept of three-way picture languages. The same theorem can be stated in a stronger form as follows (the proof can be found in [6] or [10]).

Theorem 6.5. *It is not partially decidable whether* (1) $\mathrm{dpic}(L_1) \cap \mathrm{dpic}(L_2) = \emptyset$, *and* (2) $\mathrm{bpic}(L_1) \cap \mathrm{bpic}(L_2) = \emptyset$, *for a three-way regular π-language L_1 and a 1-reversal-bounded linear π-language L_2 such that both* $\mathrm{bpic}(L_1)$ *and* $\mathrm{bpic}(L_2)$ *are stripe picture languages.*

Acknowledgment

The authors express appreciation to the referees for pointing out mistakes and for suggesting improvements. In particular, the use of the homomorphism h in the polynomial-time algorithm for reversal-bounded regular picture languages (Section 4) was suggested by one of the referees and this simplified the algorithm and improved the running time. The authors also express appreciaton to F. Hinz who made some helpful comments on an earlier version of the paper.

References

[1] K. Culik II and E. Welzl, Two-way finite state generators, in: M. Karpinsky, ed., *Foundations of Computation Theory*, Lecture Notes in Computer Science, Vol. 158 (Springer, Berlin, 1983) 106–114.
[2] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (H. Freeman, San Francisco, 1978).
[3] F. Hinz, Regular chain code picture languages of nonlinear descriptional complexity, in: J. Gruska, B. Rovan and J. Wiedermann, eds., *MFCS 86*, Lecture Notes in Computer Science, Vol. 233 (Springer, Berlin, 1986) 414–421.
[4] F. Hinz, Classes of picture languages that cannot be distinguished in the chain code concept and deletion of redundant retreats, in: B. Monien and R. Cori, eds., *STACS 89*, Lecture Notes in Computer Science, Vol. 349 (Springer, Berlin, 1989) 132–143.
[5] J.E. Hopcroft and J.D. Ullman, *Introduction to Automata Theory, Languages, and Computation* (Addison-Wesley, Reading, MA, 1979).
[6] C. Kim, Complexity and decidability for restricted classes of picture languages, *Theoret. Comput. Sci.* **73** (1990) 295–311.
[7] C. Kim, Picture iteration and picture ambiguity, *J. Comput. System Sci.* **40** (1990) 289–306.
[8] C. Kim and I.H. Sudborough, The membership and equivalence problems for picture languages, *Theoret. Comput. Sci.* **52** (1987) 177–191.
[9] H.A. Maurer, G. Rozenberg, and E. Welzl, Using string languages to describe picture languages, *Inform. and Control* **54** (1982) 155–185.
[10] H.A. Maurer, G. Rozenberg, and E. Welzl, Chain code picture languages, in: H. Ehrig, N. Nagl and G. Rozenberg, eds., *Graph Grammars and Their Application to Computer Science*, Lecture Notes in Computer Science, Vol. 153 (Springer, Berlin, 1983) 232–244.
[11] A. Salomaa, *Formal Languages* (Academic Press, London, 1973).
[12] I.H. Sudborough and E. Welzl, Complexity and decidability for chain code picture languages, *Theoret. Comput. Sci.* **36** (1985) 173–202.