COMPLEXITY AND DECIDABILITY FOR CHAIN CODE PICTURE LANGUAGES*

I.H. SUDBOROUGH

Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60201, U.S.A.

E. WELZL

Institute for Information Processing, IIG, Technical University of Graz and Austrian Computer Society, A-8010 Graz, Austria

Communicated by A. Salomaa
Received March 1984
Revised August 1984

Abstract. A picture description is a word over the alphabet $\{u, d, r, l\}$, where u means "go one unit line up from the current point", and d, r, and l are interpreted analogously with down, right, and left instead of up. By this, a picture description describes a walk in the plane—its trace is the picture it describes. A set of picture descriptions describes a (chain code) picture language.

This paper investigates complexity and decidability questions for these picture languages. Thus it is shown that the membership problem is NP-complete for regular picture languages (i.e., picture languages described by regular languages of picture descriptions), and that it is undecidable whether two regular picture description languages describe a picture in common. After this we investigate so-called stripe picture languages (all pictures are within a stripe defined by two parallel lines), providing 'better' complexity and decidability results: Membership is decidable in linear time for regular stripe picture languages. Emptiness of intersection and equivalence is decidable for regular stripe picture languages.

Introduction

There is a big variety of formal models for picture recognition and description. In such a formal approach, pictures are often treated as sets of (labeled or unlabeled) cells or unit lines from the Cartesian integer grid—'labeled' if colored pictures are considered—'unlabeled' if only the shapes of pictures are investigated. Rosenfeld [12] surveys some of these models, being mainly acceptor models (automata), while generative models (grammars) constitute only a minor part of this survey.

Maurer et al. [9] initiated the investigation of pictures (respectively languages of pictures) consisting of unit lines from the integer grid, using the following description concept: A picture is described by a word over the alphabet $\{u, d, r, l\}$, whose letters

* This research has been carried out during the first author's visit at the IIG. The second author is supported by the Austrian "Fonds zur Förderung der wissenschaftlichen Forschung".

mean "move (and draw) a unit line up (down, right, left, respectively) from the current point".

Informally, such a word—we call it picture description—constitutes a traversal in the plane, and its trace describes a picture consisting of unit lines. Picture languages are now simply described by languages of picture descriptions. This concept resembles that of a chain code (see [3]), that is why we refer to these picture languages as *chain code picture languages* (in order to distinguish this approach from other models for picture languages).

We continue here the systematic mathematical study of chain code picture languages as started in [9] and put effort on complexity and decidability questions for the classes of picture languages described by classes of picture description languages from the classical Chomsky hierarchy.

The paper is organized as follows. We briefly recall necessary notions from formal language theory and from chain code picture languages in Section 2. For more detailed presentations we refer to the standard literature (as, e.g., Maurer [8], Hopcroft and Ullman [7], Salomaa [13] and Harrison [6]) for formal language theory and to Maurer et al. [9] for chain code picture languages. In Section 3 we investigate the shortest description of a picture in context-free, linear, and regular picture description languages, provided the picture is described at all. In Section 4 we show that the membership problem is NP-complete for regular (linear) picture languages (i.e., picture languages described by regular (linear) picture description languages). The undecidability of the emptiness of the intersection problem for regular picture languages is provided in Section 5. In Section 6 we consider so-called stripe picture languages (i.e., picture languages within the stripe between two parallel lines). The results for this restricted class are: It is decidable whether a context-free picture language is a stripe picture language. Membership in regular stripe picture languages is decidable in linear time. It is decidable whether two regular stripe picture languages have a nonempty intersection or whether they coincide.

1. Preliminaries

We assume the reader to be familiar with basic formal language theory. Perhaps only the following notational matters require an additional comment: $\mathbb{Z}$ denotes the set of integers. For an integer n, $|n|$ denotes the absolute value of n. For a set A, 2^A denotes the set of its subsets and if A is finite, then $|A|$ denotes the cardinality of A. For sets A and B, $A-B$ denotes the set-theoretical difference. λ denotes the empty word. For a word w, $|w|$ denotes its length and if b is a letter, then $\#_b(w)$ denotes the number of occurrences of b in w. The set of prefixes of w is denoted by $\text{pref}(w)$ and for a language L, $\text{pref}(L)=\bigcup_{w\in L}\text{pref}(w)$.

We will use context-free grammars as tuples $G=(N, \Sigma, P, S)$, where N is the set of nonterminals, Σ the set of terminals (i.e., $N\cap\Sigma=\emptyset$), P is the set of productions, and $S\in N$ is the axiom. If we define a normal form, upper case letters represent nonterminals, lower case letters represent terminals. E.g., the Chomsky normal form would be defined as $(A \to BC, A \to a)$NF. A deterministic finite automaton is a tuple $A = (Q, \Sigma, \delta, q_0, F)$, with Q being the set of states, Σ the input alphabet, δ the transition function, q_0 the initial state, and $F \subseteq Q$ the set of accepting states.

We turn to notions from the 'picture-part'. The two-fold Cartesian product $\mathbb{Z}^2$ of $\mathbb{Z}$ with itself is referred to as the *universal point set* M_0. The *up-neighbor* of $v = (m, n) \in M_0$ is $u(v) = (m, n+1)$, the *down neighbor* of v is $d(v) = (m, n-1)$, the *right neighbor* of v is $r(v) = (m+1, n)$, and, finally, the *left neighbor* of v is $l(v) = (m-1, n)$. The *neighborhood* of v is defined as $N(v) = \{u(v), d(v), r(v), l(v)\}$. The *universal line set* M_1 is defined by $M_1 = \{\{v, v'\} \mid v \in M_0, v' \in N(v)\}$. An *attached basic picture p* is a finite subset of M_1. For an attached basic picture p its *point set* $V(p)$ is defined by $V(p) = \{v \in M_0 \mid \{v, v'\} \in p$ for some $v' \in M_0\}$. If for every pair $v, v' \in V(p)$ with $v \neq v'$, there are points $v_0, v_1, \ldots, v_n$ in $V(p)$ such that $v_0 = v$, $v_n = v'$ and $\{v_i, v_{i+1}\} \in p$, for $0 \leq i \leq n-1$, then p is *connected*. In what follows we assume every picture to be connected!

The notion of a drawn picture is technically useful. An *attached drawn picture q* is a triple $q = (p, s, e)$, such that p is an attached basic picture and either p is empty and $s = e \in M_0$ or p is nonempty and $s, e \in V(p)$; p is the *base* of q, denoted base(q), s is the *start point* of q, and e is the *end point* of q. The *shift* of q is defined by $sh(q) = (x(e) - x(s), y(e) - y(s))$. (For an element $v = (m, n) \in M_0$, $x(v) = m$ and $y(v) = n$.)

In fact we are mainly interested in the relative position of the edges rather than in their absolute position in $\mathbb{Z}^2$. For integers m and n we define $t_{m,n}$ to be a mapping from M_0 to M_0 such that, for $v = (i, j) \in M_0$, $t_{m,n}(v) = (i+m, j+n)$. Simultaneously, we use $t_{m,n}$ as a mapping from M_1 to M_1 (for $\{v, v'\} \in M_1$, $t_{m,n}(\{v, v'\}) = \{t_{m,n}(v), t_{m,n}(v')\}$) and in the obvious way for subsets of M_0 and M_1. Let p_1 and p_2 be two attached basic pictures. We say p_1 *and* p_2 *are equivalent*, denoted $p_1 \sim p_2$, if there exist integers m and n such that $p_1 = t_{m,n}(p_2)$. Let $q_1 = (p_1, s_1, e_1)$ and $q_2 = (p_2, s_2, e_2)$ be two attached drawn pictures. We say q_1 *and* q_2 *are equivalent*, denoted $q_1 \simeq q_2$, if there exist integers m and n, such that $p_1 = t_{m,n}(p_2)$, $s_1 = t_{m,n}(s_2)$, and $e_1 = t_{m,n}(e_2)$.

It is easily seen that both $\sim$ and $\simeq$ are equivalence relations. The equivalence class of $\sim$ containing an attached basic picture p, denoted $[p]$, is called the *unattached version* of p, simply referred to as *basic picture*. Analogously, the equivalence class of $\simeq$ containing an attached drawn picture q, denoted $\langle q \rangle$, is called the *unattached version* of q, simply referred to as *drawn picture*.

We are using the four letter alphabet $\pi = \{u, d, r, l\}$ to describe these pictures. 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*) and every grammar generating a π-language is called a *picture description grammar* (or *π-grammar*).

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

If $w = \lambda$, then dpic$(w) = \langle \emptyset, (0,0), (0,0) \rangle$.

If $w = zb$ for some z in π^*, b in π, where $\text{dpic}(z) = \langle p, s, e\rangle$,
then $\text{dpic}(w) = \langle p \cup \{\{e, b(e)\}\}, s, b(e)\rangle$.

If $\text{dpic}(w) = \langle p, s, e\rangle$, then the *basic picture described by* w, denoted $\text{bpic}(w)$, is defined by $\text{bpic}(w) = [p]$. We sometimes use shift directly for π-words, meaning $\text{sh}(w) = \text{sh}(\text{dpic}(w))$. In Fig. 1.1 we give some examples. For $w = (ur)^2dlr$, $p_1 = \text{bpic}(w)$, $q_1 = \text{dpic}(w)$, $q_2 = \text{dpic}(wu)$, $q_3 = \text{dpic}(wr)$. The circle in the illustration of a drawn picture indicates the startpoint, a little square indicates the endpoint. Note that $\text{bpic}(w) = \text{bpic}(wu)$, while $\text{dpic}(w) \neq \text{dpic}(wu)$; $\text{sh}(w) = \text{sh}(q_1) = (2, 1)$, and $\text{sh}(wu) = \text{sh}(q_2) = (2, 2)$.

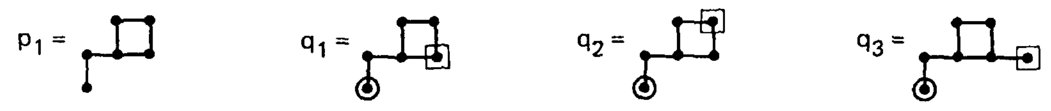


Fig. 1.1. Examples of basic and drawn pictures.

For a π-language L, the *drawn picture language described by* L, denoted $\text{dpic}(L)$, is defined by $\text{dpic}(L) = \{\text{dpic}(w) \mid w \in L\}$ and the *basic picture language described by* L, denoted $\text{bpic}(L)$, is defined by $\text{bpic}(L) = \{\text{bpic}(w) \mid w \in L\}$.

A drawn picture language D is a *regular* (*linear*, *context-free*) *picture language* if there is a regular (linear, context-free) π-language L, such that $D = \text{dpic}(L)$. Similarly, we define basic picture languages of different types.

In [9] it has been shown that every picture language described by a recursively enumerable π-language can be described by a context-sensitive π-language. Consequently, we do not consider these classes, because decidability results (or better: undecidability results) can be obtained by straightforward arguments.

Whenever we formulate a decidability result for a regular (linear, context-free) picture (or π-) language, then we assume that this language is specified by a regular (linear, context-free, respectively) grammar!

2. Optimality of picture description languages

In [9] the length of the shortest π-word describing a given picture has been analyzed. Thus it has been shown that such a 'minimal' word has length less than twice the size of the basic picture it describes. Moreover, this bound has been proved to be tight.

Here we consider the length of a shortest description in L of a picture p, provided the picture p is in $\text{bpic}(L)$. Let f be a computable function from the set of basic pictures into the set of positive integers. Then a π-language L is *$f(p)$-optimal* if for every picture p in $\text{bpic}(L)$, there is a word w in L, such that $p = \text{bpic}(w)$ and $|w| \leq f(p)$. For example, $L_1 = u^+r^+$ is $|p|$-optimal, $L_2 = r^+d^+u^+r^+$ is $2|p|$-optimal and $L_3 = (ru^*rld^*l)^+$ is $|p|^2$-optimal. (So $f(p)$-optimal means that the picture p has an optimal description in L with length not exceeding $f(p)$.)

Of course, whenever a decidable π-language L is $f(p)$-optimal for a computable function f, then for bpic(L) the picture membership problem is decidable. Moreover, if we can prove suitable bounds on $f(p)$ for a class of π-languages, then this has implications for the picture membership complexity of the class of picture languages described by these π-languages.

In this section we give asymptotically tight polynomial bounds for $f(p)$ of regular and linear π-languages and we show that $f(p)$ can be necessarily exponential in the size of the pictures for context-free π-languages.

The following observations will be helpful in the proofs of this section

2.1. Observation. *Let v and w be two π-words, such that* $\mathrm{dpic}(v) = \mathrm{dpic}(w)$. *Then for every pair of π-words x and y, we have* $\mathrm{dpic}(xvy) = \mathrm{dpic}(xwy)$.

2.2. Observation. *Let p be a nonempty basic picture. Then*

$$|p|/2 < |V(p)| \leq |p| + 1.$$

Note that in this observation it is essential that p is a connected picture (which we assume tacitly, anyhow).

2.3. Theorem. (i) *For every regular π-language L there is a constant c, $c > 0$, such that L is $c|p|^2$-optimal.*

(ii) *There is a regular π-language L and a constant C, $C > 0$, such that: For every positive integer n there is a picture p_n in* bpic(L) *with $|p_n| \geq n$, for which every description w of p_n in L has length $|w| \geq C|p_n|^2$.*

Proof. (i) Let $G = (N, \pi, P, S)$ be a regular π-grammar in $(A \to aB, A \to a)$-NF, such that $L = L(G)$. Consider now a nonempty picture p in bpic(L) and a π-word w of minimal length in L with $\mathrm{bpic}(w) = p$. Let $q = \langle r, (0,0), e\rangle = \mathrm{dpic}(w)$. We show that a derivation of w cannot be longer than $|N| \cdot |V(r)| \cdot |r|$. This gives the same upper bound for the size of w, i.e.,

$$|w| \leq |N| \cdot |V(r)| \cdot |r| \leq |N| \cdot (|p| + 1) \cdot |p| \leq 2|N| \cdot |p|^2.$$

To this end let

$$S = A_0 \Rightarrow w_1A_1 \Rightarrow w_2A_2 \Rightarrow \cdots \Rightarrow w_{n-1}A_{n-1} \Rightarrow w_n = w$$

be a derivation D of w. Let $w_0 = \lambda$.

We consider now the sequence

$$\mathrm{dpic}(w_0), \mathrm{dpic}(w_1), \ldots, \mathrm{dpic}(w_{n-1})$$

of drawn pictures induced by the derivation D. Let $\mathrm{dpic}(w_i) = \langle r_i, (0,0), e_i\rangle$, $1 \leq i \leq n-1$.

Claim. If, for $1 \leq i < j \leq n-1$, $\mathrm{dpic}(w_i) = \mathrm{dpic}(w_j)$, *then* $A_i \neq A_j$.

Proof of the Claim. Assume that $A = A_i = A_j$. Then the derivation D can be written as follows for $w_j = w_i\bar{w}_j$ and $w = w_j\bar{w}$:

$$S \overset{*}{\Rightarrow} w_iA \overset{*}{\Rightarrow} w_i\bar{w}_jA \overset{*}{\Rightarrow} w_i\bar{w}_j\bar{w}$$

with $\bar{w}_j \neq \lambda$. Since $\text{dpic}(w_i) = \text{dpic}(w_i\bar{w}_j)$, we have $\text{dpic}(w_i\bar{w}) = \text{dpic}(w_i\bar{w}_j\bar{w}) = \text{dpic}(w)$ (see Observation 2.1). But

$$S \overset{*}{\Rightarrow} w_iA \overset{*}{\Rightarrow} w_i\bar{w}$$

is now a derivation in G of a word $w_i\bar{w}$ shorter than w which also describes p, a contradiction.

This claim implies that in the sequence

$$(r_0, e_0, A_0), (r_1, e_1, A_1), (r_2, e_2, A_2), \ldots, (r_{n-1}, e_{n-1}, A_{n-1})$$

no two triples are the same. It is easily seen that at most $|N| \cdot |V(r)| \cdot |r|$ different triples are possible: $|N|$ for the number of possible variables A_i; $|V(r)|$ for the number of possible endpoints e_i; $|r|$ for the number of possible edge sets r_i. Note that for $0 \leq i < j \leq n-1$, we have $r_i \subseteq r_j$. This is why we have bound $|r|$ instead of $2^{|r|}$. Consequently, we have shown that $n \leq |N| \cdot |V(r)| \cdot |r|$ which proves part (i) of the theorem for $c = 2|N|$ (as mentioned above).

(ii) Let $L = (ru^*rld^*l)^+$. For $n \geq 1$, we consider the words

$$w_n = ru^1rld^1lru^2rld^2l \ldots ru^nrld^nl$$

in L which describe the pictures $p_n = \text{bpic}(w_n)$ (see Fig. 2.1).

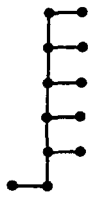


Fig. 2.1. The picture p_5.

It is straightforward to verify that $|p_n| = 2n+1$ and $|w_n| = n^2+5n$. We give an intuitive argument that there is no word shorter than w_n in L which describes p_n. A π-word in L draws an arbitrarily long vertical line with horizontal lines of length one, so-called '*thorns*', on both sides. After drawing a left thorn it draws somewhere a right thorn and vice versa. If a π-word in L draws a right thorn in p_n, then it must draw the left thorn before drawing a new right thorn. Hence, a minimal word describing p_n must have the shape of w_n, with permutations of the ru^irld^il parts possible. This implies that $|w| \geq \frac{1}{4}|p_n|^2$ for every description w of p_n in L. □

2.4. Theorem. (i) *For every linear π-language L there is a constant c, $c > 0$, such that L is $c|p|^3$-optimal.*

(ii) *There is a linear π-language L and a constant C, $C>0$, such that: For every positive integer n there is a picture p_n in* bpic(L) *with $|p_n|\geq n$, for which every description w of p_n in L has length $|w|\geq C|p_n|^3$.*

Proof. (i) Let $G=(N,\pi,P,S)$ be a linear π-grammar in $(A\to aB,\ A\to Ba,\ A\to a)$-NF, such that $L=L(G)$. We take an arbitrary nonempty picture p in bpic(L) and a π-word w of minimal length in L which describes p, i.e., bpic(w) $=p$. Let $q=\langle r,(0,0),e\rangle=\text{dpic}(w)$. Using a similar approach as in the proof of Theorem 2.3, we show that a derivation of w cannot have more than $|N|\cdot|V(r)|^2\cdot|r|$ steps. This implies directly that $|w|\leq 4|N|\cdot|p|^3$. To be precise, consider a derivation D in G:

$$S=A_0\Rightarrow v_1A_1w_1\Rightarrow v_2A_2w_2\Rightarrow\cdots\Rightarrow v_{n-1}A_{n-1}w_{n-1}\Rightarrow w$$

of the word w. Let $v_0=w_0=\lambda$ and $\langle p_i,(0,0),e_i\rangle=\text{dpic}(v_i)$, $\langle r_i,s_i,e\rangle=\text{dpic}(w_i)$—with e being the endpoint in $q=\langle r,(0,0),e\rangle$—for $0\leq i\leq n-1$.

Claim. If for $0\leq i<j\leq n-1$, $p_i\cup r_i=p_j\cup r_j$, $e_i=e_j$ and $s_i=s_j$, then $A_i\neq A_j$.

Proof of the Claim. Assume that $A=A_i=A_j$. Then the derivation D can be written as follows for $v_j=v_i\bar{v}_j$, $w_j=\bar{w}_jw_i$, and $w=v_j\bar{w}w_j$:

$$S\overset{*}{\Rightarrow}v_iAw_i\overset{*}{\Rightarrow}v_i\bar{v}_jA\bar{w}_jw_i\overset{*}{\Rightarrow}v_i\bar{v}_j\bar{w}\bar{w}_jw_i=w$$

with $\bar{v}_j\bar{w}_j\neq\lambda$. The assumptions of the Claim imply that dpic($v_i\bar{w}w_i$) $=$ dpic($v_i\bar{v}_j\bar{w}\bar{w}_jw_i$). In this case,

$$S\overset{*}{\Rightarrow}v_iAw_i\overset{*}{\Rightarrow}v_i\bar{w}w_i$$

is a derivation in G of a word $v_i\bar{w}w_i$ shorter than w which describes p, a contradiction.

We know that in the sequence

$$(p_0\cup r_0,e_0,s_0,A_0)(p_1\cup r_1,e_1,s_1,A_1)(p_{n-1}\cup r_{n-1},e_{n-1},s_{n-1},A_{n-1})$$

no tuple occurs twice which gives us an upper bound of $|N|\cdot|V(r)|^2\cdot|r|$ for the length n of the derivation D. This proves part (i) of the theorem for $c=4|N|$.

(ii) Consider the linear π-language

$$L=\{z\in x(ru^+l^2urdrd^+l)^{|x|}\,|\,x\in(ru^*rld^*l)^+\}.$$

Let p_n be the picture shown in Fig. 2.2. It is straightforward to verify that (1) $|p_n|=2n+7$, and (2) if $p_n=\text{bpic}(xy_1y_2\ldots y_m)$, where $x\in(ru^*rld^*l)^+$, $y_i\in ru^*l^2urdrd^*l$, for all i, $l\leq i\leq m$, and $xy_1y_2\ldots y_m\in L$, then $|x|\geq c|p_n|^2$ for some $c>0$ (see also the proof of part (ii) of Theorem 2.3). Since m must be equal to $|x|$ and each y_i draws the unit square on top and the horizontal line at the bottom, it follows that, for all i, $l\leq i\leq n$, $|y_i|\geq n$ and, consequently, $|xy_1y_2\ldots y_m|\geq C|p_n|^3$ for some constant $C>0$. □

2.5. Theorem. (i) *For every context-free π-language L there is a constant c, $c>0$, such that L is $c^{|p|^3}$-optimal.*

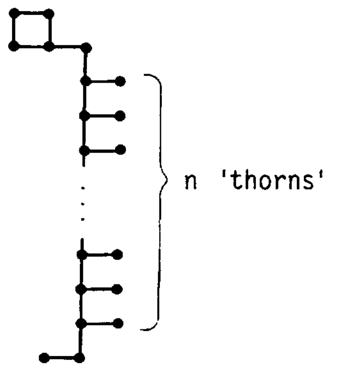


Fig. 2.2. The picture p_n.

(ii) *There is a context-free π-language L and a constant C, $C > 1$, such that: For every positive integer n there is a picture p_n in* bpic(L), *with $|p_n| \geq n$, for which every description w of p_n in L has length $|w| \geq C^{|p_n|}$.*

Proof (sketch). (i) Let $G = (N, \pi, P, S)$ be a context-free π-grammar in Chomsky-NF with $L = L(G)$. For some nonempty picture $p \in \text{bpic}(L)$, let w be a π-word of minimal length in L with $p = \text{bpic}(w)$. Moreover, $q = \langle r, (0,0), e\rangle = \text{dpic}(w)$. Assume that a derivation of w in G can be written as

$$S \overset{*}{\Rightarrow} v_1 A w_1 \overset{+}{\Rightarrow} v_2 A w_2 \overset{+}{\Rightarrow} v_2 \bar{w} w_2 = w$$

with $v_2 w_2 \neq \lambda$ and $A \in N$. Let $\langle p_1, (0,0), e_1\rangle = \text{dpic}(v_1)$, $\langle p_2, (0,0), e_2\rangle = \text{dpic}(v_2)$, $\langle r_1, s_1, e\rangle = \text{dpic}(w_1)$, and $\langle r_2, s_2, e\rangle = \text{dpic}(w_2)$ (with e being the endpoint of q). If the tuple $(p_1 \cup r_1, e_1, s_1)$ equals $(p_2 \cup r_2, e_2, s_2)$, then the derivation

$$S \overset{*}{\Rightarrow} v_1 A w_1 \overset{+}{\Rightarrow} v_1 \bar{w} w_1$$

in G gives a word $v_1 \bar{w} w_1$ in L with $\text{bpic}(v_1 \bar{w} w_1) = p$ which is shorter than w, a contradiction. This shows that in a derivation tree of w in G, every path from the root to a leaf has length at most $|N| \cdot |V(r)|^2 \cdot |r| \leq 4|N| \cdot |p|^3$. It is now an easy observation that $|w| \leq (2^{4|N|})^{|p|^3}$, which proves part (i) of the theorem for $c = 16^{|N|}$.

(ii) We show next that there are context-free π-languages L and pictures $p \in \text{bpic}(L)$ such that a word w in L which satisfies $p = \text{bpic}(w)$ has at least length $2^{|p|}$. For this purpose we recall the fact, explicitly recorded in [1], that a two-way deterministic pushdown automaton (2D-PDA) can be used to count from 0 to $2^n - 1$ using its input tape of length n and two endmarkers together with its pushdown store. That is, the pushdown store begins with the string 0^n representing, of course, the value zero in binary notation. (The low order bit is assumed to be on top.) In general, the pushdown operates by adding one to the binary encoded number in its pushdown store. It does so by deleting as many occurrences of the symbol 1 as occur on top of the store while simultaneously moving its input head one more cell away from the right endmarker. When an occurrence of the symbol 0 is encountered the 2D-PDA replaces this symbol with 1 and adds back as many 0's to the pushdown store as there are cells between the input head and the right endmarker. Fig. 2.3 depicts the situation, when, e.g., the 2D-PDA adds one to 19.

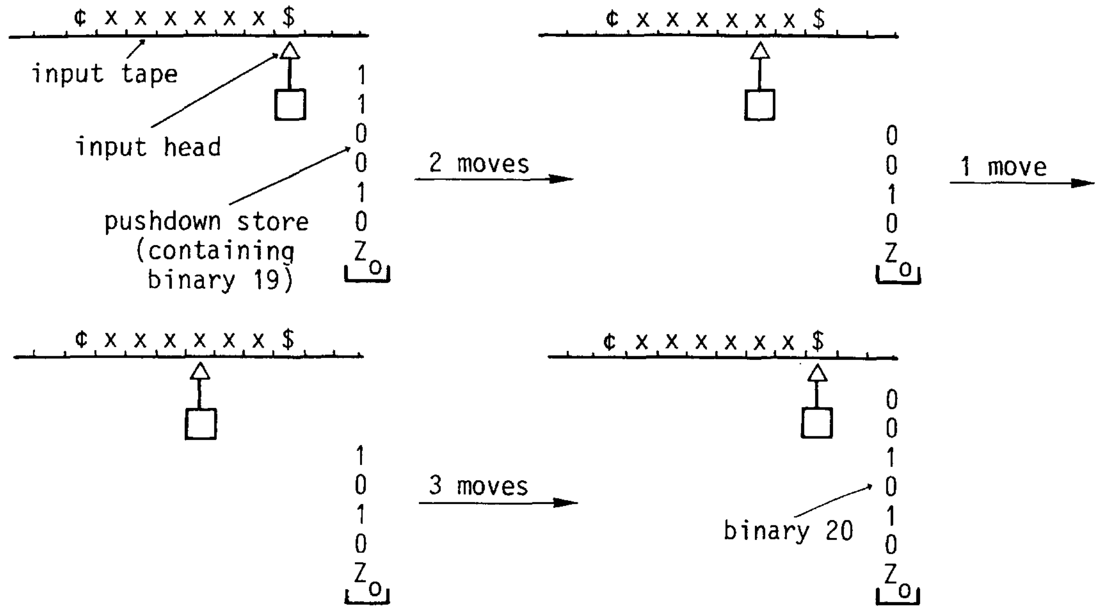


Fig. 2.3. The 2D-PDA adds 1 to binary 19 in its pushdown store ($n=6$).

We observe that the same type of computation can be simulated by a context-free picture language and the picture p_n shown in Fig. 2.4. That is, let P be the one-way pushdown automaton (1-PDA) specified in Fig. 2.5. Intuitively speaking, the 1-PDA can always guess that the input head of the simulated 2D-PDA is under the right endmarker. In this case, it accepts the substring *drul*, which means drawing a unit square. If the guesses were always correctly done, then the accepted word describes the picture p_n. If this is not the case, then we get additional unit squares in the picture described by the accepted word. (Note that P is actually a deterministic acceptor, so our 'lower bound' holds even for deterministic context-free languages.)

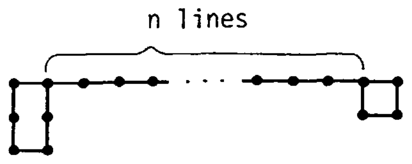


Fig. 2.4. The picture p_n.

Let L be the context-free π-language accepted by P. Then $p_n \in \text{bpic}(L)$ and any drawing of p_n by a π-word w in L simulates a computation of the 2D-PDA counting from 0 to 2^n-1. An elementary calculation shows now that $|p_n| = 10+n$ and that the length of the shortest word w describing p_n is $4+3\times 2^{n+1}$ (actually there is exactly one such word w). This shows that, for $c=\frac{1}{10}$, $|w| \geqslant 2^{c|p_n|}$. That is, any string w in L with $p_n = \text{bpic}(w)$ has length exponential in the size of p_n. □

We conclude this section by pointing out a problem related to the one we considered here. Let $L=(ru^*rld^*l)^+$ be the language in the proof of Theorem 2.3(ii). We have shown that it is—in some sense—no better than $c|p|^2$-optimal (for a suitable

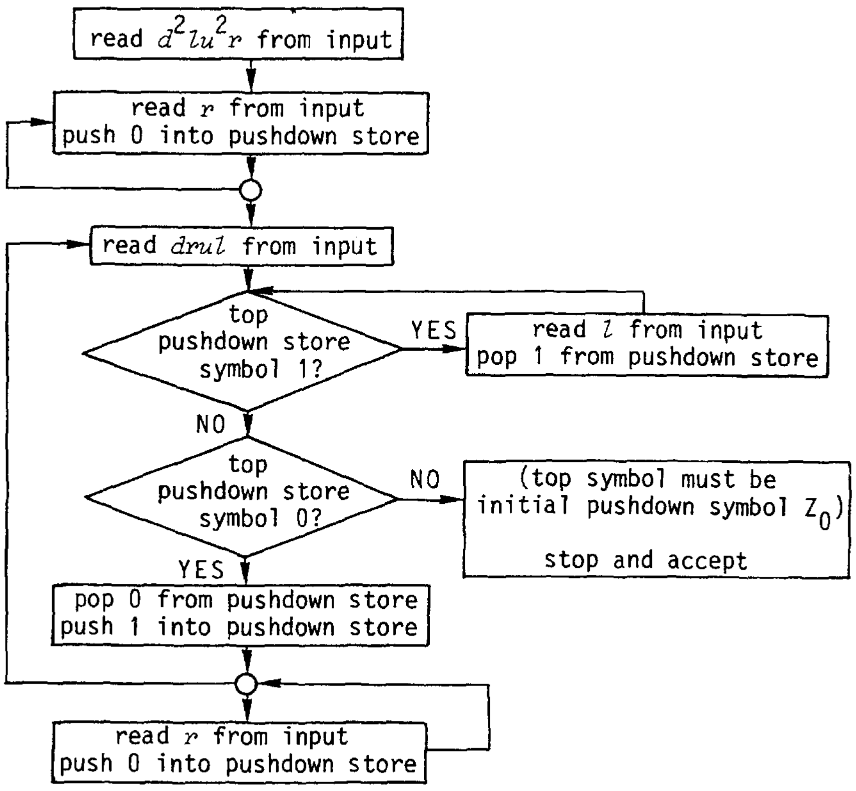

read d^2lu^2r from input
read r from input
push 0 into pushdown store
read $drul$ from input
top pushdown store symbol 1?
YES
read l from input
pop 1 from pushdown store
NO
top pushdown store symbol 0?
NO
(top symbol must be initial pushdown symbol Z_0)
stop and accept
YES
pop 0 from pushdown store
push 1 into pushdown store
read r from input
push 0 into pushdown store

Fig. 2.5. The 1-PDA *P*.

constant c). But for $L' = r\{u, rl, lr\}^*r$ we have $\text{bpic}(L) = \text{bpic}(L')$ and L' is $2|p|$-optimal. Hence we ask whether there is always a $c|p|$-optimal regular π-language describing a given regular picture language or whether there are regular picture languages which are only describable by 'inherently' $c|p|^2$-optimal π-languages. The same problems arise for linear and context-free picture languages.

3. Membership complexity for picture languages

In this section we treat the complexity of deciding whether a given picture is in a picture language. We show that the problem is NP-complete for regular and linear picture languages. (NP denotes the class of problems, which can be solved in polynomial time by a nondeterministic Turing machine. A problem is NP-complete if it is complete for the class NP. For further and detailed definitions we refer to Garey and Johnson [4].)

3.1. Lemma. *The membership problem for basic and drawn linear picture languages is in* NP.

Proof. Let G be a linear π-grammar and p be a basic picture. Then Theorem 2.4(i) shows that for a suitable constant c it suffices to guess a word w with $|w| \leq c|p|^3$

and to check whether $w \in L(G)$ and $p = \text{bpic}(w)$. Both tasks can be done in polynomial time. It is easily seen that the proof of Theorem 2.4(i) works also for drawn pictures, so that we can proceed equivalently for drawn pictures. □

Let $G = (N, \pi, P, S)$ be the regular π-grammar with $N = \{S, S_0, T, U, X_1, X_2, X_3, F, G, H, Y\}$ and the set of productions

$$\begin{aligned}
&S \to d^2r^2u^2S_0,\\
&S_0 \to T \mid F \mid rd^3ru^3,\\
&T \to uU \mid rF \mid d^2ru^2S_0,\\
&\left.\begin{aligned}
&U \to uU \mid ru^iluX_i\\
&X_i \to rX_i \mid lX_i \mid drd^ilY\\
&G \to ru^iludrd^ilH \mid uG
\end{aligned}\right\} \text{ for } i = 1, 2, 3,\\
&Y \to dY \mid rulrF,\\
&F \to uG \mid rT \mid d^2ru^2S_0,\\
&H \to dH \mid rulrT.
\end{aligned}$$

Let R be the regular set generated by the grammar G. We can express the set of strings generated by the nonterminal T as

$$\begin{aligned}
&[u^+(rulu(r+l)^*drdl + ru^2lu(r+l)^*drd^2l +\\
&\quad ru^3lu(r+l)^*drd^3l)d^*rulr + r]F + d^2ru^2S_0
\end{aligned}$$

and the set of strings generated from the nonterminal F as

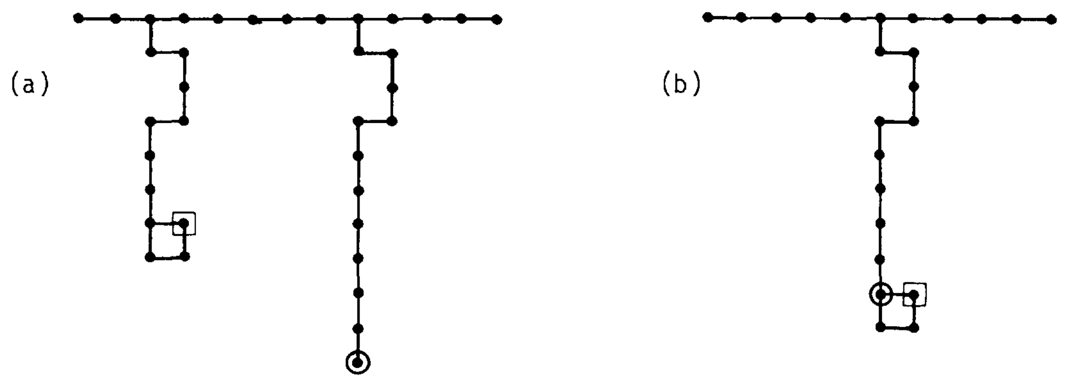
$$[u^+(ruludrdl + ru^2ludrd^2l + ru^3ludrd^3l)d^*rulr + r]T + d^2ru^2S_0.$$

We observe that the principal difference between the nonterminals T and F is that (1) from T we can go arbitrarily far up, make one of the 'bends' $rulu$, ru^2lu, or ru^3lu, then make an arbitrarily long horizontal line segment, make the corresponding reverse 'bend' $drdl$, drd^2l, or drd^3L, go arbitrarily far down and then make a unit square, and (2) from F we can do the same except that the horizontal line segment cannot be drawn after going up (consult Fig. 3.1 for illustration).

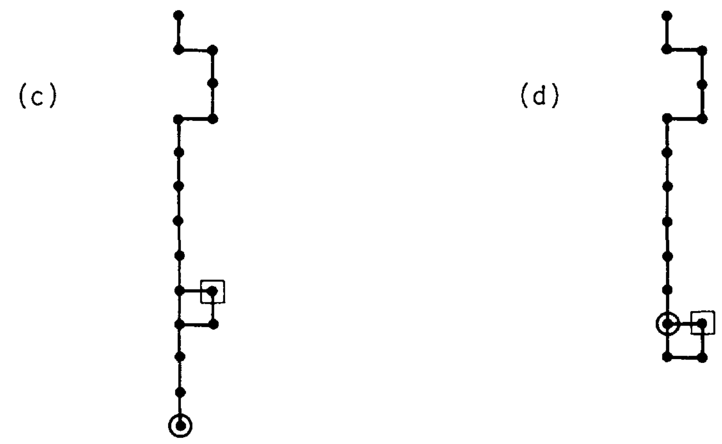
We show that it is NP-hard to decide if a given picture p is in dpic(R) by describing a reduction from the problem 3SAT (see Garey and Johnson [4]). The NP-hardness of deciding whether a given picture p is in bpic(R) follows as an easy corollary.

3.2. Lemma. *The membership problem for* dpic(R) *is* NP-*hard.*

Proof. Let $w = C_1C_2 \ldots C_m$ be a well-formed formula (wff) in 3CNF with the variables $x_1, x_2, \ldots, x_n$. We construct a drawn picture $p(w)$ such that $p(w)$ is in



Pictures which can be drawn 'starting up' from nonterminal T (for $i=2$); (a) example for general case; (b) example, if both 'bends' are in the same position and start and end point are on the same 'level'.



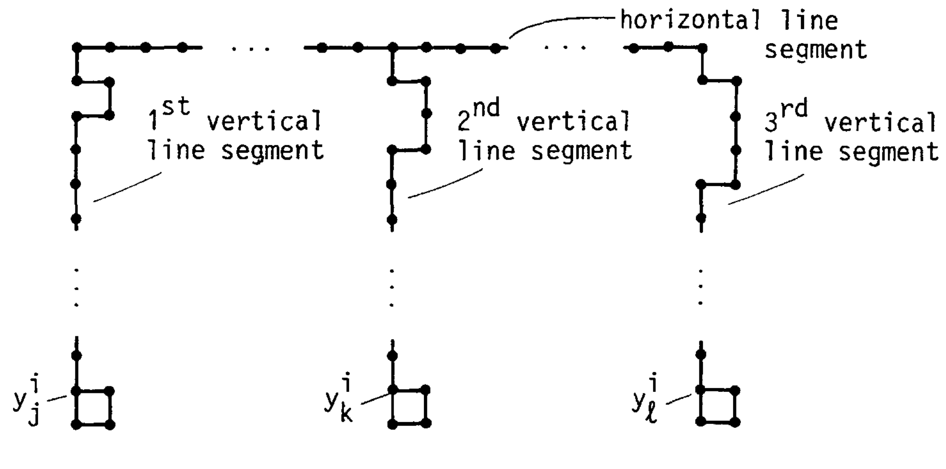
Pictures which can be drawn 'starting up' from nonterminal F (again for $i=2$); (c) example for general case; (d) example, if start and end point are on the same 'level'.

Fig. 3.1. Pictures drawn starting from T or F.

dpic(R) if and only if w is satisfiable. The picture $p(w)$ is constructed from component pictures corresponding to the various clauses and variables of w.

Let $C_i = (y_j + y_k + y_l)$ be the ith clause, where y_j, y_k, and y_l are positive or negative instances of variables. Construct a component $p(C_i)$ as shown in Fig. 3.2 where (1) the 'bend' in the sth vertical line segment, is made by ru^sl, rd^sl, or some equivalent sequence, for $1 \leq s \leq 3$, (2) the unit squares containing the nodes labeled y_j^i, y_k^i, and y_l^i, shown in Fig. 3.2, are equally far below the horizontal line segment, and (3) the exact size of the horizontal line segment and the vertical line segments is as follows.

For each variable x_i there is a picture component $p(x_i)$, which is shown in Fig. 3.3. The component $p(x_i)$ contains $4m+6$ nodes. A principal feature of this component is that (1) each node labeled x_i^j, which corresponds to the variable x_i and the clause C_j, is at a distance $d(x_i^j) \equiv 0 \pmod 4$ from the left end of the component, and (2) each node labeled $\bar{x}_i^j$, which corresponds to the negation of the variable x_i and the clause C_j, is at a distance $d(\bar{x}_i^j) \equiv 1 \pmod 4$ from the left end of the component. Thus, if one imagines starting out with T (F) at the left end and



horizontal line segment
1st vertical line segment
2nd vertical line segment
3rd vertical line segment
y_j^i
y_k^i
y_ℓ^i

Fig. 3.2. A component $p(C_i)$.

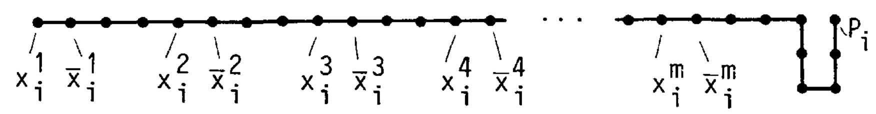


Fig. 3.3. The component $p(x_i)$.

alternating between these two values between successive vertices of the component, then one has the same value T (F) at each vertex x_i^j, $1 \leq j \leq m$, and at each vertex $\bar{x}_i^j$, $1 \leq j \leq m$. That is, choosing T (F) at the left end of the component corresponds to choosing the truth value true (false) and this choice is held consecutively across the component.

The entire picture $p(w)$ is formed by (1) coalescing the nodes labeled y_j^i, y_k^i, and y_l^i in each component $p(C_i)$ with the correspondingly labeled nodes in the components constructed for the variables, (2) for all i, $1 \leq i \leq n$, coalescing node P_i of the component $p(x_i)$ with node x_{i+1}^1 of the component $p(x_{i+1})$, (3) separating horizontal line segments in distinct clause components by five vertical positions by adding a sufficient number of nodes in their vertical line segments, (4) attaching the end of the drawn picture $d^2r^2u^2$ to the node x_1^1, and (5) attaching the beginning of the drawn picture rd^3ru^3 to the node P_n of the component $p(x_n)$. An example of the picture $p(w)$ is shown in Fig. 3.4. (It should perhaps be noted that the size of the horizontal line segments in the clause components is determined by the literals the clause contains and the manner in which the variable components have been catenated together. Also, many nodes in vertical line segments of clause components will be coalesced with nodes in horizontal line segments of other clause components in forming the picture $p(w)$ as specified by the above rules.)

Let w be a satisfiable well-formed formula. We show that $p(w)$ is in dpic(R). Let A be an assignment of truth values that makes w true. In drawing $p(w)$ we must, of course, pass through each of the vertices $x_1^1, x_2^1, \ldots, x_n^n$. At each such vertex we enter the nonterminal S_0 via the sequence d^2ru^2 from either the nonterminal T or the nonterminal F. Then from S_0 we enter either the nonterminal T or the nonterminal F. We enter the nonterminal T at node x_i^1 if and only if $A(x_i) = \text{true}$. In passing

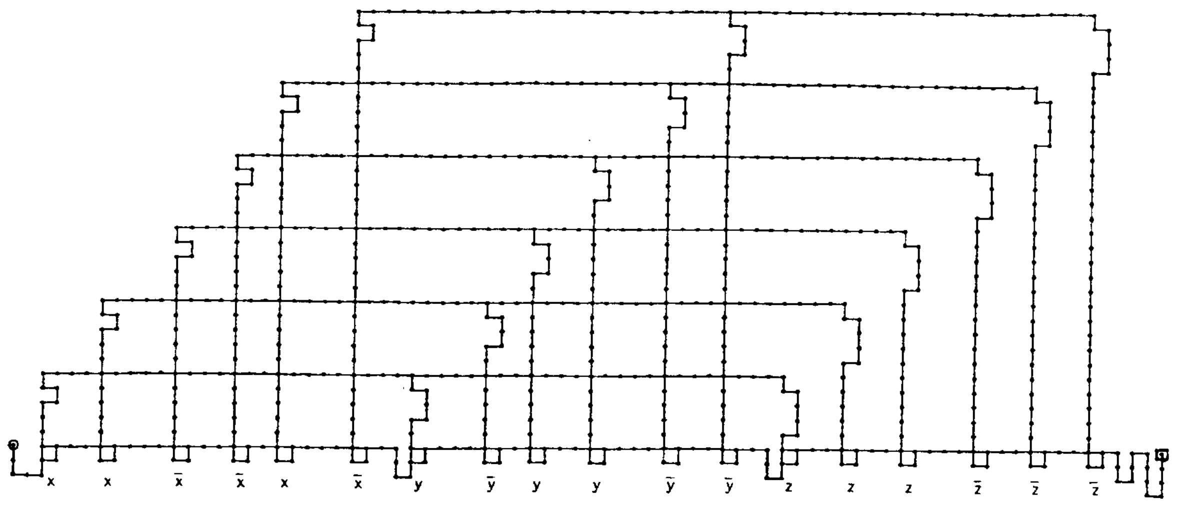


Fig. 3.4. The picture $p(w)$ corresponding to the wff $w=(x+y+z)(x+\bar{y}+z)(\bar{x}+y+z)(\bar{x}+y+\bar{z})\cdot(x+\bar{y}+\bar{z})(\bar{x}+\bar{y}+\bar{z})$.

from x_i^1 to P_i in the component $p(x_i)$ we draw all of the vertical line segments in any of the clause components that are attached. In addition, each time a vertical line segment is encountered in the nonterminal T the entire horizontal line segment of that clause component is drawn. The fact that all of $p(w)$ can be drawn follows from the fact that each clause contains a true literal under the assignment A and, consequently, all of the horizontal line segments can be drawn in this way. Therefore, $p(w)$ is in dpic(R).

Let $p(w)$ be in dpic(R). From the description of the regular set R it should be apparent that if the horizontal line segment in the component $p(C_i)$ is drawn by coming up from the sth vertical line segment of that component (for $1 \leq s \leq 3$), then one must return down the same sth vertical line segment. That is, each vertical line segment is coded with a special 'bend' and the regular set R requires the code used to go through this unique bend going up to agree with the code to pass through the bend going down. Furthermore, the horizontal line segment in the component $p(C_i)$ can only be entered from a vertical line segment belonging to this component. That is, the regular set R ensures that a horizontal line segment can be drawn after going up a vertical line segment only after having just passed through a 'bend'. Since the 'bends' are located in $p(w)$ on the vertical line segments just before the horizontal line segments of the same component, it is only possible to enter this horizontal line segment when coming up the correct vertical line segments. Furthermore, one cannot enter one of these horizontal line segments going down, since the regular set R ensures that the path going down stops at a unit square and unit squares appear only in the components corresponding to variables.

Since $p(w)$ is in dpic(R), there is a string z in R such that dpic(z) $= p(w)$. Define the truth assignment A by: $A(x_i)=$ true if and only if a horizontal line segment of a clause component is drawn in going up from a vertex x_i^j, $1 \leq j \leq m$, in the component $p(x_i)$. It should be clear from our previous discussion and the construction of $p(w)$

that each horizontal line segment can be drawn only by going up from and returning to some vertex either of the form x_i^j or $\bar{x}_i^j$, for some $1 \leq j \leq m$.

We show that the well-formed formula w is true under the assignment A. We need only show that each clause contains one true literal under the assignment A. By the construction of $p(w)$ and the regular set R, if a horizontal line segment is drawn going up from a vertex x_i^j, then no horizontal line segment in $p(w)$ can be drawn going up from a vertex $\bar{x}_i^k$. That is, each such vertex x_i^j is at an odd numbered distance from each such vertex $\bar{x}_i^k$ and it is not possible by any substring of the regular set R to go up on vertical line segments which are at an odd numbered distance apart in a component and draw horizontal line segments. Consequently, since each horizontal line segment must be drawn, it follows that each clause contains a true literal under the assignment A and, therefore, w is true under this assignment. □

3.3. Corollary. *The membership problem for* bpic(R) *is* NP-*hard.*

The corollary easily follows from the observation that the only way the picture $p(w)$ described in the proof of Lemma 3.2 can be drawn from a string in R is starting at the beginning of the unique portion of this picture drawn by $d^2r^2u^2$ and ending in the unique portion of this picture drawn by d^3ru^3.

We summarize the results of this section in the following theorem.

3.4. Theorem. *The membership problem for drawn and basic regular and linear picture languages is* NP-*complete.*

Of course, the membership problem for context-free picture languages is NP-hard, too, but we are not able to show that the problem is in NP. Recall the result in Theorem 2.5(ii). It shows that the direct approach as used for regular and linear picture languages in Lemma 3.1 does not work for context-free picture languages.

4. An undecidability result for regular picture languages

We can give simple examples which show that, in general, the intersection of two regular basic picture languages is not a regular basic picture language. Consider $B_1 = \text{bpic}(r^+d^+l^+u^+)$ and $B_2 = \text{bpic}(l^+u^+r^+d^+)$. It is easily seen that $B_1 \cap B_2$ is exactly the set of rectangles and it can be shown that this is not a regular picture language. (We refer to Maurer et al. [9] for proof techniques for showing that a picture language is not regular.) It is even conceivable that this is not a context-free picture language. In this section we show that the emptiness of the intersection of two regular basic picture languages is undecidable. This will be done by a reduction to the halting problem of so-called two-counter automata, which have been investigated by Minsky [11] (see also Hopcroft and Ullman [7]). A *two-counter automaton* is a triple $M=(Q, P, A_0)$, where Q is a finite *set of states*, A_0 in Q is an *initial state*, and P is a finite set of instructions, being tuples in

$$(Q\times\{\text{add1}, \text{add2}, \text{sub1}, \text{sub2}, \text{if1}, \text{if2}\}\times Q)\cup Q\times\{\text{halt}\}.$$

The automaton is working on two nonnegative counters, whose values can be increased, decreased (if nonzero) and checked for zero. The meanings of the tuples in P are listed in Fig. 4.1. (We neglect here the input tape of M!)

Instruction	Meaning
$(A, \text{add1}, B)$	Move from state A to state B, while adding one to counter 1
$(A, \text{add2}, B)$	Move from state A to state B, while adding one to counter 2
$(A, \text{sub1}, B)$	If counter 1 is nonzero, then move from state A to state B, while subtracting one from counter 1
$(A, \text{sub2}, B)$	If counter 2 is nonzero, then move from state A to state B, while subtracting one from counter 2
$(A, \text{if1}, B)$	Move from state A to state B if counter 1 is zero
$(A, \text{if2}, B)$	Move from state A to state B if counter 2 is zero
(A, halt)	The computation terminates in state A

Fig. 4.1. Meaning of two-counter instructions.

A *configuration* of a two-counter automaton is a triple (A, n_1, n_2), A in Q, n_1 and n_2 two nonnegative integers where A describes the current state of the automaton, n_1 the value of counter 1, and n_2 the value of counter 2. The *move relation* $\vdash$ of M is defined on configurations in the obvious way. E.g., $(A, n_1, n_2)\vdash(B, n_1-1, n_2)$ if $(A, \text{sub1}, B)\in P$ and $n_1\neq 0$, or $(A, n_1, 0)\vdash(B, n_1, 0)$ if $(A, \text{if2}, B)\in P$. $\vdash^*$ is the reflexive transitive closure of $\vdash$. Since two-counter automata can simulate every Turing machine [11], we have the following proposition.

4.1. Proposition. *For a two-counter automaton $M=(Q, P, A_0)$ it is undecidable whether there is an integer n such that $(A_0, n, 0)\vdash^*(B, n_1, n_2)$ and $(B, \text{halt})\in P$, for some n_1, n_2, and B, i.e., whether there is an integer n such that there is a computation of M which halts on input $(n, 0)$.*

4.2. Theorem. *It is undecidable whether or not, for two regular π-languages L_1 and L_2, $\text{bpic}(L_1)\cap\text{bpic}(L_2)\neq\emptyset$ holds.*

Proof. The intuitive idea of the proof is as follows. We simulate a two-counter automaton by a walk (drawing) in the plane starting from some point $(4n, 0)$, $n\geqslant 0$.

That is, for $i, j \geq 0$, the point $(4i, 4j)$ stands for "counter 1 contains i" and "counter 2 contains j". Thus, e.g., instead of an instruction $(A, \text{add1}, B)$ of M, we will have a production $A \to r^4B$ in a corresponding regular π-grammar (describing the so-called *M-simulating picture language*). Of course, we cannot test for zero in a regular grammar. Instead of this we will simply guess the appropriate counter to be zero. This will be done by drawing little squares starting from the points $(4i, 4j)$, namely, *rdlu* for testing counter 1 and, *uldr* for testing counter 2. E.g., we take $A \to rdluB$ for an instruction $(A, \text{if1}, B)$ of M. A drawing of a picture is now a correct simulation of a computation of A, (i) if all the squares *rdlu* are attached to points $(4i, 0)$ and all the squares *uldr* are attached to points $(0, 4j)$, and (ii) if the drawing never leaves the area of points $(4i, 4j)$, $i, j \geq 0$. In order to specify the origin $(0, 0)$ in an unattached picture, we will draw a 3×3 square and define its upper right corner to be the origin $(0, 0)$.

A second regular π-language (so-called *test picture language*) will describe all pictures with one 3×3 square, little squares in the appropriate positions (on the axes) and a drawing between $(4i, 4j)$-points, $i, j \geq 0$. We show that there is a picture in the intersection of the M-simulating picture language and the test picture language if and only if there is an integer n such that M halts on input $(n, 0)$.

Let $M = (Q, P, A_0)$ be a two-counter automaton. We construct a regular π-grammar $G_M = (N_M, \pi, P_M, B_0)$ which has nonterminals $N_M = Q \cup \{B_0, B_1, B_2, B_3\}$ and productions P_M as follows:

$B_0 \to uldrd^4B_0 \mid uldrl^3d^3r^3u^3B_1,$

$B_1 \to rdlur^4B_1 \mid rdluB_2,$

$B_2 \to l^4B_2 \mid A_0$ (A_0 is the initial state of M),

$X \to r^4Y$	if $(X, \text{add1}, Y)$ in P,
$X \to u^4Y$	if $(X, \text{add2}, Y)$ in P,
$X \to l^4Y$	if $(X, \text{sub1}, Y)$ in P,
$X \to d^4Y$	if $(X, \text{sub2}, Y)$ in P,
$X \to rduY$	if $(X, \text{if1}, Y)$ in P,
$X \to uldrY$	if $(X, \text{if2}, Y)$ in P,
$X \to B_3$	if (X, halt) in P,

$B_3 \to u^4B_3 \mid d^4B_3 \mid r^4B_3 \mid l^4B_3 \mid \lambda.$

Let $L_M = L(G_M)$. Secondly, we define a regular π-language L_T by the following expression:

$$L_T = u^3r^3((rdul)u^{4*}d^{4*}(drlu)r^4)^*l^{4*}d^3l^3r^3u^3((ulrd)r^{4*}l^{4*}(ludr)u^4)^*.$$

We start the analysis of $\text{bpic}(L_M) \cap \text{bpic}(L_T)$ by some simple observations:

(O1) $L_M \subseteq R = \{r^4, l^4, d^4, u^4, uldr, rdlu, l^3d^3r^3u^3\}^*$.

(O2) A picture p is in bpic$(L_T) \cap$ bpic(R) if and only if it has the shape as, e.g., described in Fig. 4.2. That is: (1) We have a single 3×3 square (whose topright corner we define as origin $(0,0)$). (2) The horizontal axis contains unit squares *rdlu* attached to positions $(4i, 0)$, $0 \leqslant i \leqslant i_0$, for some $i_0 \geqslant 0$. (3) The vertical axis contains unit squares *uldr* attached to positions $(0, 4j)$, $0 \leqslant j \leqslant j_0$, for some $j_0 \geqslant 0$. (4) There are upgoing vertical (rightgoing horizontal) lines of arbitrary length $4n$, $n \geqslant 0$, starting from positions $(4i, 0)$, $0 \leqslant i \leqslant i_0$ (respectively $(0, 4j)$, $0 \leqslant j \leqslant j_0$).

Let M halt on input $(n_0, 0)$ for some $n_0 \geqslant 0$. Let

$$C : (A_0, n_0, 0) = (A_0, n_0, m_0) \vdash (A_1, n_1, m_1) \vdash \cdots \vdash (A_k, n_k, m_k)$$

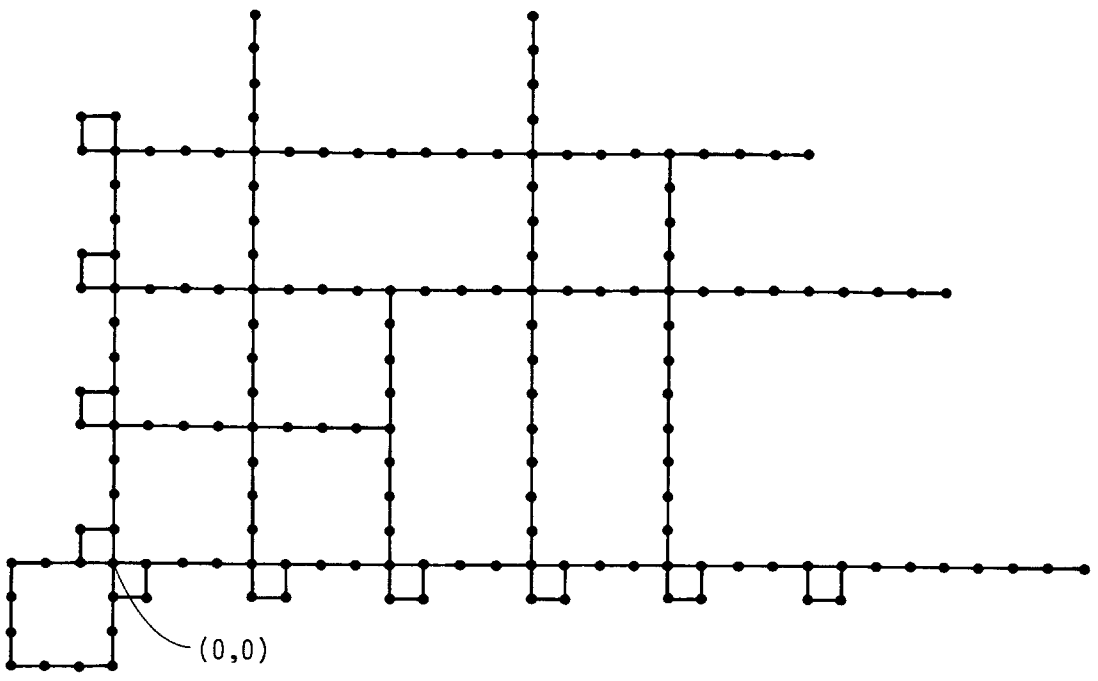


Fig. 4.2. An example of a picture in bpic$(L_T) \cap$ bpic(R).

be a computation of M such that $(A_k$, halt) is in P. Let $I_0, I_1, \ldots, I_{k-1}$ be the sequence of instructions in P such that for all i, $0 \leqslant i \leqslant k-1$, I_i is an instruction that takes M from configuration (A_i, n_i, m_i) to the configuration $(A_{i+1}, n_{i+1}, m_{i+1})$. For all i, $0 \leqslant i < k-1$, let prod(I_i) be the production of G_M that corresponds to I_i. (The correspondence is described in the construction of the grammar G_M.) Let $b = \max\{n_i \mid 0 \leqslant i \leqslant k\}$ and $a = \max\{m_i \mid 0 \leqslant i \leqslant k\}$, i.e., a and b are the maximal values of counter two and one during the computation C. We observe that G_M can generate strings of the form

$$w(a, b, n_0) = (uldrd^4)^a uldrl^3 d^3 r^3 u^3 (rdlur^4)^b rdlul^{4(b-n_0)} A_0,$$

using its initial rules. The string $w(a, b, n_0)$ corresponds to a picture as shown in Fig. 4.3. Let $w(C)$ be the string generated by the sequence of productions

$$\text{prod}(I_0), \text{prod}(I_1), \ldots, \text{prod}(I_{k-1}).$$

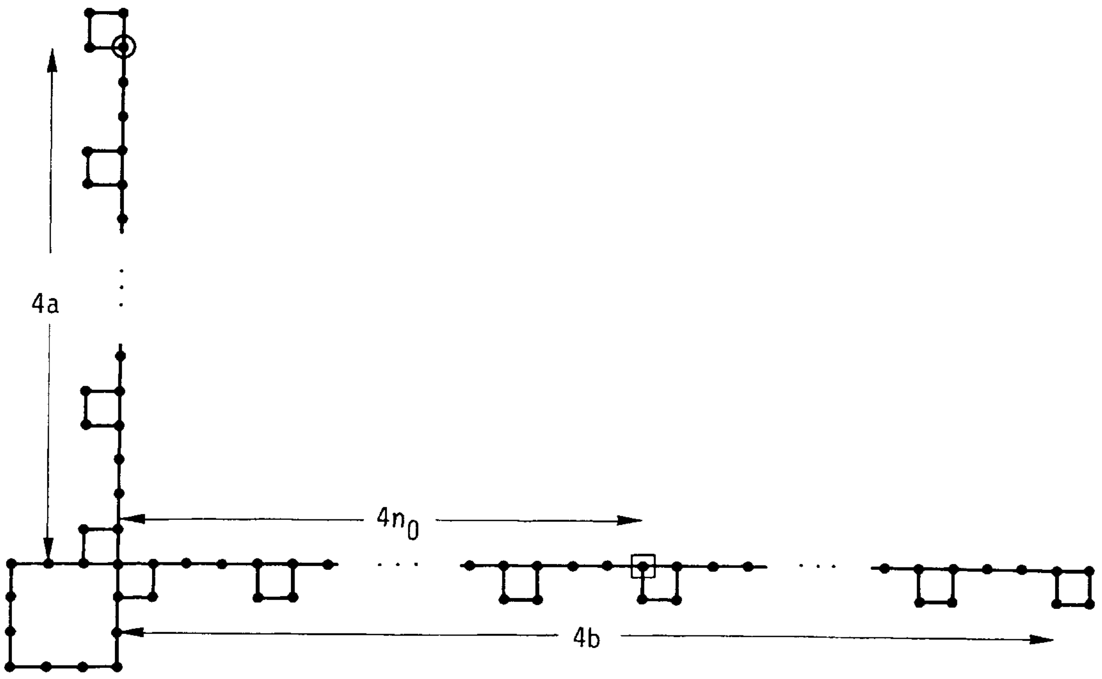


Fig. 4.3. The picture produced by $w(a, b, n_0)$.

We observe that the rightmost symbol of this string is the nonterminal A_k, since the computation terminates in state A_k. Since (A_k, halt) is an instruction of M, we see that G_M can generate terminal strings of the form

$$w = w(a, b, n_0)w(C)w_h$$

for any $w_h \in \{u^4, d^4, r^4, l^4\}^*$.

The picture produced by the string $w(a, b, n_0)w(C)$ represents the computation C in the following way: If after the sequence of instructions $I_0, I_1, \ldots, I_{j-1}, j \geq 1$, M, started with counter values $(n_0, 0)$ in state A_0, has the counter values (n_j, m_j) and is in state A_j, then the string generated by the sequence of productions $\text{prod}(I_0), \text{prod}(I_1), \ldots, \text{prod}(I_{j-1})$ has the nonterminal A_j and produces a picture which terminates at position $(4n_j, 4m_j)$. (Recall that we took $(0, 0)$ to be the top right corner position of the 3×3 square.) Furthermore, there are no additional unit squares in the picture except for those which were already produced by the prefix $w(a, b, n_0)$.

Finally, we take a suffix w_h, such that w_h rasters the whole rectangle $\{(n, m) \mid 0 \leq n \leq b, 0 \leq m \leq a\}$ with line distance 4, so that we end up with a picture as shown in Fig. 4.4. This picture is in $\text{bpic}(L_T)$ as seen from observation (O2). That is, $\text{bpic}(L_G) \cap \text{bpic}(L_T) \neq \emptyset$.

Conversely, let $\text{bpic}(L_T) \cap \text{bpic}(L_M) \neq \emptyset$. As we have seen, $\text{bpic}(L_T) \cap \text{bpic}(R)$ is the set of all pictures of the form indicated in observation (O2)(1)-(4) above. We have also seen that pictures in $\text{bpic}(L_M)$ correspond to computations by the two-counter automaton M. If a picture p is in $\text{bpic}(L_M) \cap \text{bpic}(L_T)$, then it follows that all of the 'branch on zero' instructions simulated by the rules of the grammar G_M,

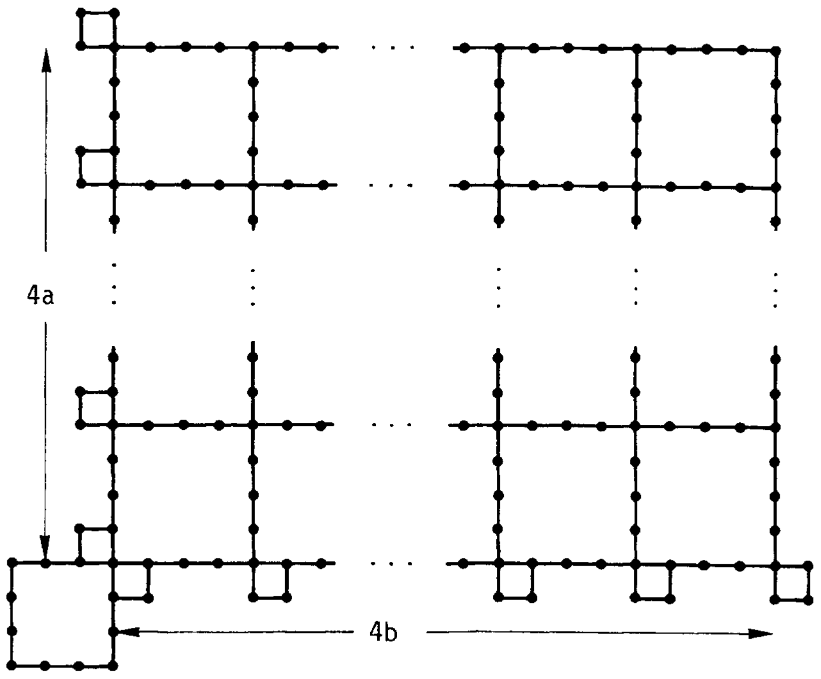


Fig. 4.4. The picture bpic($w(a, b, n_0)w(C)w_h$).

are valid, i.e., occur only when the appropriate counter is zero, since in pictures in bpic(L_T) unit squares occur only appropriately on the horizontal axis and on the vertical axis. Also, we note that only one 3×3 square exists in a picture of bpic(L_T) and, consequently, the positions represented in the computation simulated by the rules of G_M are correct with respect to both axes.

It follows that there is a valid computation by M on some counter values $(n, 0)$ that halts. That is, if bpic(L_M) $\cap$ bpic(L_T) $\neq \emptyset$, then there is an n such that M halts on input $(n, 0)$. □

It is straightforward to verify that it is partially decidable for two regular π-languages L_1 and L_2 whether bpic(L_1) $\cap$ bpic(L_2) $\neq \emptyset$. That is, one can simply enumerate all pictures and test each for membership in bpic(L_1) $\cap$ bpic(L_2). Consequently, the complement problem must not be partially decidable. Thus we have the following corollary.

4.3. Corollary. *It is not partially decidable for two regular picture languages L_1 and L_2 whether* bpic(L_1) $\cap$ bpic(L_2) $= \emptyset$.

5. Stripe picture languages

A stripe picture language is a picture language whose pictures 'fit' into a stripe defined by two parallel lines. In this section we give decidability and complexity results for this restricted class of picture languages which differ significantly from the ones we learned in Section 3 and 4.

The main lemma states that for a given stripe there is an (injective) correspondence between regular picture languages within this stripe and regular string languages over some alphabet Σ. As straightforward implications we have, e.g., that picture membership is decidable in linear time for regular stripe picture languages and that the emptiness of the intersection problem of two regular stripe picture languages is decidable. This correspondence, of course, is only possible, because stripe picture languages contain—in some sense—only one-dimensional objects, i.e., pictures can 'extend' arbitrarily only in two colinear directions. But the last result mentioned in this section reveals that already linear stripe picture languages are more complex than linear string languages.

For real numbers k, d_1, d_2, $d_1 < d_2$, the *(k, d_1, d_2)-stripe* ($M_0^{(k, d_1, d_2)}$ for short) is the subset of points of the universal point set M_0 defined by

$$M_0^{(k,d_1,d_2)} = \{(i,j) \in M_0 \mid ki + d_1 \leq j \leq ki + d_2\}.$$

$M_0^{(k, d_1, d_2)}$ contains simply the integer grid points which are between the parallel lines defined by the equations $y = kx + d_1$ and $y = kx + d_2$. (The reader might miss here the case of the vertical stripe (i.e., $k = \infty$). Clearly, our results would hold also if we add the special case of $M_0^{(\infty, d_1, d_2)} = \{(i,j) \in M_0 \mid d_1 \leq i \leq d_2\}$. But this would cause the necessity of considering special cases in every proof which follows. The interested reader can easily check these special cases, but we omit them here.)

A drawn picture $q = \langle r, (0,0), e\rangle$ is a *drawn (k, d_1, d_2)-stripe picture* if $V(r) \subseteq M_0^{(k, d_1, d_2)}$. A basic picture p is a *basic (k, d_1, d_2)-stripe picture* if there is an attached basic picture p', such that $p = [p']$ and $V(p') \subseteq M_0^{(k, d_1, d_2)}$. Note that the drawn picture q in Fig. 5.1 is a drawn $(\frac{1}{3}, -\frac{5}{3}, 4)$-stripe picture. While q is not a drawn $(\frac{1}{3}, -\frac{2}{3}, 3)$-stripe picture, its basic picture $p = \text{base}(q)$ is a basic $(\frac{1}{3}, -\frac{2}{3}, 3)$-stripe picture.

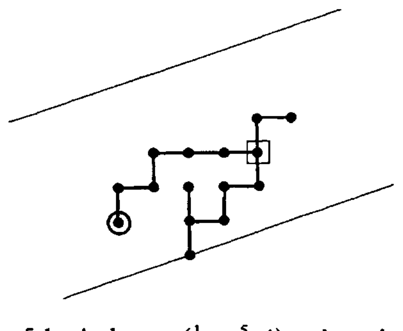


Fig. 5.1. A drawn $(\frac{1}{3}, -\frac{5}{3}, 4)$-stripe picture.

A drawn (basic) picture language B is a *drawn (basic) (k, d_1, d_2)-stripe picture language* if every picture in B is a drawn (basic) (k, d_1, d_2)-stripe picture. A drawn (basic) picture language D is called a *stripe picture language* if there are real numbers k, d_1, and d_2 such that D is a drawn (basic) (k, d_1, d_2)-stripe picture language. (As a consequence, every finite picture language is a stripe picture language.) The following proofs are always given for drawn stripe picture languages. The case of basic stripe picture languages can always be treated analogously with minor modifications.

First we show that for a context-free π-grammar G it is decidable whether dpic($L(G)$) (respectively, bpic($L(G)$)) is a stripe picture language or not. This generalizes the result of the decidability of picture finiteness in [9] and we also use some of the ideas of the proof given there.

5.1. Lemma. *Let $G=(N, \pi, P, S)$ be a reduced context-free π-grammar and x be a nonempty π-word such that $A\overset{*}{\Rightarrow}xAy$ or $A\overset{*}{\Rightarrow}yAx$ for some $A\in N$, $y\in\pi^*$. If* dpic($L(G)$) *is a drawn (k, d_1, d_2)-stripe picture language, then, for* $\text{sh}(x)=(m,n)$, *we have $m=n=0$ or $n/m=k$.*

Proof. We observe that if dpic($L(G)$) is a drawn (k, d_1, d_2)-stripe picture language, then, for all $w\in L(G)$, $\text{sh}(w)\in M_0^{(k,d_1,d_2)}$. Assume that $A\overset{*}{\Rightarrow}xAy$, $\text{sh}(x)=(m,n)\neq(0,0)$ and $m\neq 0$. (The case $n\neq 0$ and $m=0$ is left to the reader.) Then there are π-words z_1, z_2, z_3, such that $z_1x^iz_2y^iz_3$ is in $L(G)$ for all positive integers i. Note that if dpic($z_1x^iz_2y^iz_3$) is a (k, d_1, d_2)-stripe picture, then dpic(z_1x^i) is a (k, d_1, d_2)-stripe picture. Hence $\text{sh}(z_1x^i)=(m'+im, n'+in)$, for $(m',n')=\text{sh}(z_1)$ and we have

$$d_1\leq n'+in-k(m'+im)\leq d_2,$$

i.e.,

$$d_1\leq n'-km'+i(n-km)\leq d_2,$$

for all $i=1,2,\ldots$. This can be true for all i if and only if $n-km=0$, i.e., $n/m=k$. □

As an easy consequence we have the following corollary.

5.2. Corollary. *If k is a nonrational real number, then a context-free drawn (basic) (k, d_1, d_2)-stripe picture language is finite.*

Proof. Let $G=(N,\pi,P,S)$ be a reduced context-free π-grammar. Note that (i) if for all $A\in N$, $x, y\in\pi^*$, $A\overset{*}{\Rightarrow}xAy$ implies that $\text{sh}(x)=\text{sh}(y)=(0,0)$, then dpic($L(G)$) is finite (see [9, Lemma 5.8]), (ii) if for $A\in N$, y, $x\in\pi^*$, $A\overset{*}{\Rightarrow}xAy$ or $A\overset{*}{\Rightarrow}yAx$ and $\text{sh}(x)=(m,n)\neq(0,0)$, then dpic($L(G)$) is an $(n/m, d_1', d_2')$-stripe picture language for some real numbers d_1' and d_2' (if it is a stripe picture language), and (iii) $M_0^{(n/m,d_1',d_2')}\cap M_0^{(k,d_1,d_2)}$ is finite unless $k=n/m$. □

5.3. Theorem. *It is decidable whether a context-free drawn (basic) picture language D is a drawn (basic) stripe picture language or not.*

Proof. First we observe that, for a π-language L, $D=\text{dpic}(L)$ is a stripe picture language if and only if $D'=\text{dpic}(\text{pref}(L))$ is a stripe picture language. Second, if $L=\text{pref}(L)$, then $D=\text{dpic}(L)$ is a (k, d_1, d_2)-stripe picture language if and only if, for every word w in L, $\text{sh}(w)\in M_0^{(k,d_1,d_2)}$. Hence, w.l.o.g we assume that $D=\text{dpic}(L(G))$ for some reduced context-free π-grammar $G=(N,\pi,P,S)$ in Chomsky normal form such that $L(G)=\text{pref}(L(G))$.

Claim. If there is a real number k, such that for all words x in

$$T=\{x\in\pi^*\mid A\overset{*}{\Rightarrow}xAy \text{ or } A\overset{*}{\Rightarrow}yAx \text{ for some } y\in\pi^*,\ A\in N\}$$

we have $\mathrm{sh}(x)=(m,n)=(0,0)$ *or* $n/m=k$, *then D is a drawn* (k, d_1, d_2)*-stripe picture language for two real numbers* d_1 *and* d_2.

Proof of the Claim. Let w be a word in $L(G)$ whose derivation can be written as

$$S\overset{*}{\Rightarrow}z_1Az_2\overset{*}{\Rightarrow}z_1xAyz_2\overset{*}{\Rightarrow}z_1xz_3yz_2=w$$

for some variable $A\in N$ and $xy\neq\lambda$. Then $z_1z_3z_2$ is in $L(G)$ and for $\mathrm{sh}(w)=(m,n)$ and $\mathrm{sh}(z_1z_2z_3)=(m',n')$ we have $n-km=n'-km'$. (This can easily be seen by elementary calculation.) Consequently,

$$km+d_1\leqslant n\leqslant km+d_2$$

if and only if

$$km'+d_1\leqslant n'\leqslant km'+d_2.$$

We can apply a 'reduction' as used above to every word in L with $|w|>2^{|N|-1}$. This implies that if, for all words w with $|w|\leqslant 2^{|N|-1}$, $\mathrm{sh}(w)=(m,n)$, we have

$$km+d_1\leqslant n\leqslant km+d_2,$$

then this holds for all words in $L(G)$. Hence, we set $d_2=2^{|N|-1}+|k|\cdot 2^{|N|-1}$ and $d_1=-d_2$ and we are done.

Together with Lemma 5.1 this Claim implies that $\mathrm{dpic}(L(G))$ is a stripe picture language iff the assumptions of the claim hold for some k.

The language T described in the claim is obviously a context-free π-language. The property stated in the claim is fulfilled if and only if for all words x in T we have either

$$\#_u(x)-\#_d(x)=0 \quad\text{and}\quad \#_r(x)-\#_l(x)=0$$

or

$$\#_r(x)-\#_l(x)=k(\#_u(x)-\#_d(x)).$$

The set of Parikh vectors fulfilling this property, say P_1, is a semilinear set. Moreover, the set of Parikh vectors of words in T, say P_2, is a semilinear set. So, $\mathrm{dpic}(L(G))$ is a stripe picture language iff $P_2\subseteq P_1$. This is decidable for semilinear sets (see Ginsburg [5]). □

5.4. Theorem. *Let k be a rational number,* d_1 *and* d_2 *be real numbers (we assume that* $d_1\leqslant 0\leqslant d_2$*). The set of* π*-words describing drawn* (k, d_1, d_2)*-stripe pictures, denoted by*

$$\mathrm{ddes}(k,d_1,d_2)=\{w\in\pi^*\mid \mathrm{dpic}(w) \textit{ is a drawn } (k,d_1,d_2)\textit{-stripe picture}\}$$

is a regular π-language. Analogously,

$$\text{bdes}(k, d_1, d_2) = \{w \in \pi^* \mid \text{bpic}(w) \textit{ is a basic } (k, d_1, d_2)\textit{-stripe picture}\}$$

is a regular π-language.

Proof. Let $k = n/m$, such that m and n have no common divisor and $m > 0$. Then the *(m, n, d_1, d_2)-unit point field* is defined by

$$F_0 = \{(i, j) \in M_0^{(k, d_1, d_2)} \mid 0 \leq i < m\}.$$

A deterministic finite automaton accepting $\text{ddes}(k, d_1, d_2)$ is $A = (F_0 \cup \{\$\}, \pi, \delta, (0, 0), F_0)$, where $F_0 \cup \{\$\}$ is the set of states, $(0, 0) \in F_0$ the initial state, and F_0 the set of accepting states, and the transition function $\delta : (F_0 \cup \{\$\}) \times \pi \to F_0 \cup \{\$\}$ is defined by the following two cases:

$$v \in F_0,\ b \in \pi\colon\ \delta(v, b) = \begin{cases} b(v) & \text{if } b(v) \in F_0, \\ t_{m,n}(b(v)) & \text{if } t_{m,n}(b(v)) \in F_0, \\ t_{-m,-n}(b(v)) & \text{if } t_{-m,-n}(b(v)) \in F_0, \\ \$ & \text{else,} \end{cases}$$

and

$$b \in \pi\colon\ \delta(\$, b) = \$.$$

The correctness of the construction essentially stems from the fact that

$$M_0^{(k, d_1, d_2)} = \bigcup_{i \in \mathbb{Z}} t_{im, in}(F_0). \qquad \square$$

5.5. Corollary. *Let k be a rational number. The set of drawn (basic) (k, d_1, d_2)-stripe pictures in a context-free picture language is a context-free picture language. Analogously, the (k, d_1, d_2)-stripe pictures in a linear (regular) picture language form a linear (regular) picture language.*

Note that such a closure property does not hold for a nonrational number k (see Corollary 5.2).

5.6. Corollary. *Let k be a rational number, d_1 and d_2 be real numbers. It is decidable whether a drawn context-free picture language* (i) *contains drawn (k, d_1, d_2)-stripe pictures or not,* (ii) *contains finitely many (k, d_1, d_2)-stripe pictures or not, and* (iii) *is a drawn (k, d_1, d_2)-stripe picture language or not. Analogous statements hold for 'basic' instead of 'drawn'.*

For the next lemma we will use so-called two-way finite state generators, which have been introduced in [2]. The following definition is a modification of the one given in [2] and covers a special case sufficient for our purposes. (We use only two-way finite state generators with so-called set-interpretations.)

Intuitively speaking, this device has a two-way writing head with finite control and a two-way infinite working tape. Every cell of the tape contains an initially empty *set* of symbols from a finite printing alphabet Δ. In every step the writing head adds a symbol to the set currently under the writing head (if the symbol is already in the cell, then its contents does not change), moves in either direction and changes its state (without reading capacity). The output string is a sequence of sets, i.e., the output alphabet is $\Sigma = 2^{\Delta}$.

Formally, a *two-way finite state generator*, 2FSG for short, is a system

$$H = (Q, \delta, d, A_0, A_f; \Delta, \mathrm{pr}),$$

where (i) Q is a finite nonempty set of *states*, (ii) $\delta: Q \to 2^Q$ is a *transition function*, (iii) $d: Q \to 2^{\{-1,0,1\}}$ is a *direction function*, (iv) $A_0 \in Q$ is an *initial state*, (v) $A_f \in Q$ is an *accepting state*, (vi) Δ is a finite *printing alphabet*, and finally (vii) $\mathrm{pr}: Q \to \Delta$ is a *printing function*. The *output alphabet* is $\Sigma = 2^{\Delta}$.

Computations of a 2FSG H are described by strings over $(Q \times Z)$, where $(A, j) \in Q \times Z$ stands for "current state is A" and "current position on the tape is j". The move relation $\vdash$ associated with a 2FSG is defined as follows. Let $c \in (Q \times Z)^*$, $(A, j) \in Q \times Z$. Then

$$c(A,j) \vdash c(A,j)(A', j+i)$$

if $A' \in \delta(A)$, $i \in d(A)$. The reflexive transitive closure of $\vdash$ is denoted by $\vdash^*$.

The set of *valid computations* of H is

$$\mathrm{comp}(H) = \{c \in (Q \times Z)^*(\{A_f\} \times Z) \mid (A_0, 0) \vdash^* c\}.$$

For a prefix of a valid computation $c \in (Q \times Z)^*$ the *leftmost* (*rightmost*) *position visited by* c—$\mathrm{lm}(c)$ ($\mathrm{rm}(c)$) for short—is defined as the minimal (maximal) j such that c can be written as $c = c_1(A, j)c_2$ for some $c_1, c_2 \in (Q \times Z)^*$, $(A, j) \in Q \times Z$. In order to examine the final contents of a cell, the *history-j-homomorphism* $h_j: (Q \times Z)^* \to Q^*$ is defined for $(A, i) \in Q \times Z$:

$$h_j(A, i) = \begin{cases} A & \text{if } i = j, \\ \lambda & \text{if } i \neq j. \end{cases}$$

Intuitively, for a valid computation c, $h_j(c)$ describes the sequence of states of the finite control, in which the writing head passed position j on the tape. Finally, the cell will contain the element $\mathrm{out}_j(c) = \{\mathrm{pr}(A) \mid A \in \mathrm{alph}(h_j(c))\}$ of Σ. (For a word w, $\mathrm{alph}(w)$ denotes the set of symbols which occur in w.)

Thus the *word generated by a computation* c is

$$\mathrm{word}(c) = \mathrm{out}_a(c)\mathrm{out}_{a+1}(c) \ldots \mathrm{out}_{b-1}(c)\mathrm{out}_b(c) \in \Sigma^*,$$

where $a = \mathrm{lm}(c)$ and $b = \mathrm{rm}(c)$. The *language generated by H* is

$$\mathrm{lang}(H) = \{\mathrm{word}(c) \mid c \in \mathrm{comp}(H)\} \subseteq \Sigma^*.$$

5.7. Proposition (Culik and Welzl [2]). *Let H be a* 2FSG. *Then* lang(H) *is a regular language.*

Now we have prepared the notions to prove the following lemma.

5.8. Lemma (String Representation Lemma).

(I) *Let k be a rational number and d_1, d_2 be real numbers, $d_1 < d_2$. Then there is an alphabet Σ and an encoding μ from the set of drawn (k, d_1, d_2)-stripe pictures into Σ^* with the following properties*:

(1) *For two drawn (k, d_1, d_2)-stripe pictures q_1 and q_2 we have $\mu(q_1) = \mu(q_2)$ if and only if $q_1 = q_2$.*

(2) *For a drawn (k, d_1, d_2)-stripe picture q we can compute $\mu(q)$ in linear time (linear in the size of q).*

(3) *If D is a regular drawn (k, d_1, d_2)-stripe picture language, then*

$$\mu(D) = \{\mu(q) \mid q \in D\}$$

is a regular string language, which can be effectively constructed from D.

(II) *A corresponding statement holds for basic (k, d_1, d_2)-stripe pictures and picture languages.*

Proof. The idea of the proof is very simple. We divide every picture into vertical stripes of equal width. By this we get a sequence of subpictures. If we consider the subpictures as unattached objects (which are possibly disconnected), then in a stripe picture language there occur only finitely many different subpictures. We take this set of possible subpictures as alphabet. The encoded picture is now simply the sequence of unattached subpictures we obtained by 'slicing' the picture (with additional markers which give information about start and end point). See Fig. 5.2 for an example of such an encoding.

We will show that for a regular π-grammar G which describes a drawn stripe picture language $D = \text{dpic}(L(G))$ there is a 2FSG which generates exactly the strings of encoded pictures in D. This will prove the regularity of $\mu(D)$.

Let $k = n/m$, with m and n being two integers without a common divisor, and $m > 0$ (if $k = 0$, then $m = 1$, $n = 0$). We need again the (m, n, d_1, d_2)-*unit point field*,

$$F_0 = \{(i, j) \in M_0^{(k, d_1, d_2)} \mid 0 \leq i < m\}$$

and additionally we define a (m, n, d_1, d_2)-*unit line field*,

$$F_1 = \{\{v, v'\} \in M_1 \mid v \in F_0 \text{ and } v' \in (F_0 \cup t_{m,n}(F_0)\}$$

(see Fig. 5.3). The alphabet Σ is now the set of subsets of $F_1 \cup F_0 \cup \{¢, \$\}$ (where the subpicture in the sliced portion will be a subset of F_1, an element in F_0 and a \$ will indicate the position of the endpoint in a field—provided it is in the corresponding sliced portion—and ¢ indicates the startpoint in a sliced portion, whose position will be explicitly $(0, 0) \in M_0$). Clearly, Σ is a finite set.

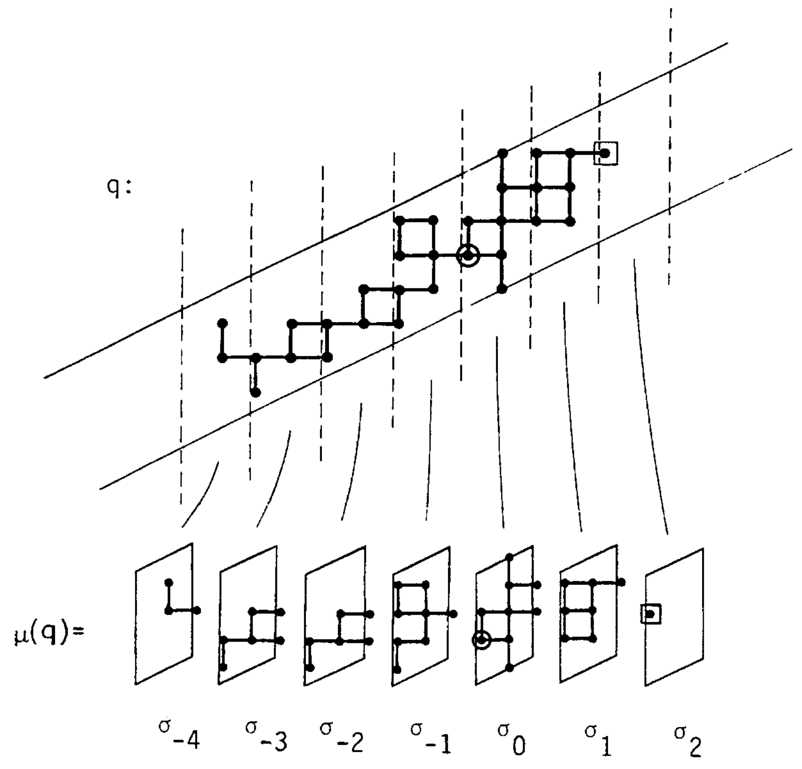


Fig. 5.2. A drawn $(\frac{1}{2}, -\frac{3}{2}, \frac{5}{2})$ stripe picture and its corresponding sequence $\mu(q)$.

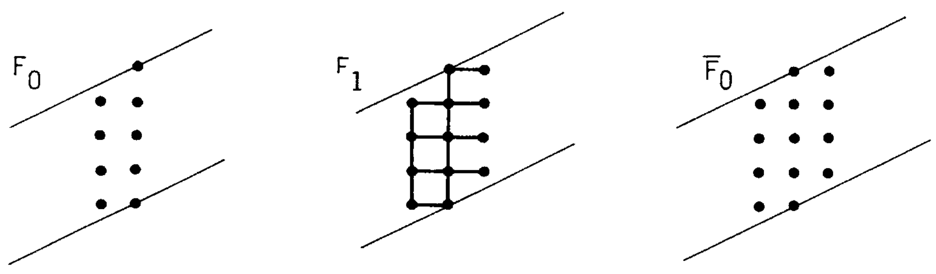


Fig. 5.3. The $(2, 1, -\frac{3}{2}, \frac{5}{2})$ unit point field F_0, the $(2, 1, -\frac{3}{2}, \frac{5}{2})$ unit line field F_1 and the extended $(2, 1, -\frac{3}{2}, \frac{5}{2})$ unit point field $\bar{F}_0$.

Consider now a drawn (k, d_1, d_2)-stripe picture $q = \langle r, (0, 0), e\rangle$. For the encoding we define, for all integers i,

$$\sigma_{-i} = \begin{cases} (r \cap F_1) \cup \{¢\} & \text{if } i = 0, e \notin F_0, \\ (r \cap F_1) \cup \{¢, \$, e\} & \text{if } i = 0, e \in F_0, \\ t_{im,in}(r) \cap F_1 & \text{if } i \neq 0, t_{im,in}(e) \notin F_0, \\ (t_{im,in}(r) \cap F_1) \cup \{\$, t_{im,in}(e)\} & \\ & \text{if } i \neq 0, t_{im,in}(e) \in F_0. \end{cases}$$

For the drawn $(\frac{1}{2}, -\frac{3}{2}, \frac{5}{2})$-stripe picture in Fig. 5.2, we have, e.g., $\sigma_{-4} = \{\{(1, 1), (1, 2)\}, \{(1, 1), (2, 1)\}\}$, $\sigma_{-3} = \sigma_{-2}$, $\sigma_2 = \{(0, 1), \$\}$ and $\sigma_i = \emptyset$ for $i < -5$ or $i > 2$. Let a be the minimal i such that $\sigma_i \neq \emptyset$ and b be the maximal i such that $\sigma_i \neq \emptyset$. The string $\mu(q)$

encoding the picture q is now

$$\mu(q) = \sigma_a\sigma_{a+1}\ldots\sigma_0\ldots\sigma_{b-1}\sigma_b.$$

It is straightforward to see that $\mu(q)$ can be computed in linear time and that, for two drawn (k, d_1, d_2)-stripe pictures, $\mu(q_1) = \mu(q_2)$ holds if and only if $q_1 = q_2$.

Our final goal is to define a 2FSG which generates $\mu(D)$ for a regular drawn (k, d_1, d_2)-stripe picture language. For this purpose let $G = (N, \pi, P, S)$ be a reduced regular π-grammar in $(A \to aB, A \to \lambda)$-NF, such that $D = \text{dpic}(L(G))$ is a drawn (k, d_1, d_2)-stripe picture language.

The cells of the working tape of our 2FSG will correspond to the fields σ_i of the encoded stripe picture. As states we use tuples $\langle A, v_1, v_2, i\rangle$, where $A \in N$, $i \in \{-1, 0, +1\}$ and $v_1, v_2 \in \bar{F}_0 = \{(i, j) \in M_0^{(k, d_1, d_2)} \mid 0 \leq i \leq m\}$. (See Fig. 5.3 for an example of an *extended* (m, n, d_1, d_2)*-unit point field* $\bar{F}_0$.) Such a tuple means for the 2FSG simulating the derivation (respectively drawing) of a word w in $L(G)$: (i) The derivation just 'uses' the nonterminal A, (ii) just drew line $\{v_1, v_2\}$ in the field (respectively cell) under the writing head, (iii) v_2 is the current position of the drawing head in the field, and (iv) i indicates the direction of the next move (-1 for 'left', 0 for 'no move', and $+1$ for 'right'). The reader might realize that the assignment of points on the 'border' between two fields is not unique. This stems from the fact that for such a point v_2, the line $\{v_1, v_2\}$ could belong to each of the adjacent fields.

Additionally, the state $\langle S, ¢, (0, 0), i\rangle$ means that the drawing starts, and the states $\langle \lambda, v, \$, 0\rangle$ mean that the drawing ended in position v of the cell under the writing head. Formally, we define the 2FSG $H = (Q, \delta, d, A_0, A_f; \Delta, \text{pr})$ as follows:

(i) $Q = \{A_0, A_f\}$
$\cup \{\langle A, v_1, v_2, i\rangle \mid A \in N, \{v_1, v_2\} \in F_1, i \in \{-1, 0, 1\}\}$
$\cup \{\langle S, ¢, (0, 0), i\rangle \mid S$ the axiom of G, $i \in \{-1, 0, 1\}\}$
$\cup \{\langle \lambda, v, \$, 0\rangle \mid v \in F_0\}$.

(ii) $\delta(A_0) = \{\langle S, ¢, (0, 0), i\rangle \mid i \in \{-1, 0, 1\}\}$;
$\delta(A_f) = \emptyset$;
$\delta(\langle \lambda, v, \$, 0\rangle) = \{A_f\}$ for $v \in F_0$;

- for $\langle B, w_1, w_2, j\rangle$, $\langle A, v_1, v_2, i\rangle \in Q$ (v_1 possibly \$) $\langle B, w_1, w_2, j\rangle \in \delta(\langle A, v_1, v_2, i\rangle)$ if and only if, for $b \in \pi$, $w_2 = b(w_1)$, the production $A \to bB$ is in P and $w_1 = t_{-im,-in}(v_2)$;
- $\langle \lambda, v, \$, 0\rangle \in \delta(\langle A, v_1, v_2, i\rangle)$ if and only if $A \to \lambda \in P$ and $v = t_{-im,-in}(v_2)$.

(iii) The direction function for $\langle A, v_1, v_2, i\rangle \in Q$, $A \in N \cup \{\lambda\}$, $v_1 \in \bar{F}_0 \cup \{¢\}$, $v_2 \in \bar{F}_0 \cup \{\$\}$, $i \in \{-1, 0, 1\}$ is defined as $d(\langle A, v_1, v_2, i\rangle) = \{i\}$; $d(A_0) = \{0\}$; $d(A_f) = \{0\}$.

(iv) $\Delta = F_1 \cup F_0 \cup \{\$, ¢\}$.

(v) $\text{pr}: Q \to \Delta$, is defined as

$$\text{pr}(A_0) = ¢,$$
$$\text{pr}(\langle S, ¢, (0, 0), i\rangle) = ¢,$$

$\mathrm{pr}(\langle\lambda, v, \$, 0\rangle) = v,$
$\mathrm{pr}(\langle A, v_1, v_2, i\rangle) = \{v_1, v_2\}, \quad v_1 \neq ¢, v_2 \neq \$,$
$\mathrm{pr}(A_f) = \$.$

The equivalence of lang(H) and $\mu(D)$ should be clear from the intuitive arguments given above. □

5.9. Corollary. *For regular π-languages L_1 and L_2 where* dpic(L_2) *is a stripe picture language, it is decidable whether* (1) dpic(L_1) = dpic(L_2), (2) bpic(L_1) = bpic(L_2), (3) dpic(L_1) ∩ dpic(L_2) = ∅, *and* (4) bpic(L_1) ∩ bpic(L_2) = ∅.

Proof. Let k, d_1, d_2 be such that dpic(L_1) is a drawn (k, d_1, d_2)-stripe picture language. If dpic(L_1) is finite, then the decidability of the above problems (1) through (4) follows immediately. If dpic(L_1) is infinite, then k is a rational number (see Corollary 5.2). Hence, it is decidable whether dpic(L_2) is a (k, d_1, d_2)-stripe picture language (see Corollary 5.6(iii)).

If this is not the case, then dpic(L_1) = dpic(L_2) does not hold. If dpic(L_2) is a (k, d_1, d_2)-stripe picture language, then we apply the construction of Lemma 5.8 to obtain two 2FSG's H_1 and H_2 such that lang(H_1) = μ(dpic(L_1)) and lang(H_2) = μ(dpic(L_2)). (Note that the mapping μ in the construction is unique, as soon as k, d_1, and d_2 are chosen.) Then dpic(L_1) = dpic(L_2) holds if and only if lang(H_1) = lang(H_2). From [2, proof of Proposition 5.7] it can be seen that we can effectively find two regular grammars G_1 and G_2 with $L(G_1)$ = lang(H_1) and $L(G_2)$ = lang(H_2). Hence, problem (1) has been reduced to the equivalence problem for regular grammars, which is known to be decidable.

To show that (3) is decidable, we first construct a regular grammar $\bar{G}_2$ such that dpic($L(\bar{G}_2)$) consists of all drawn (k, d_1, d_2)-stripe pictures in dpic(L_2). $\bar{G}_2$ is simply the regular grammar generating L_2 ∩ ddes(k, d_1, d_2). Since a regular grammar generating ddes(k, d_1, d_2) can effectively be constructed (see proof of Theorem 5.4), the grammar $\bar{G}_2$ can be effectively constructed. Now, problem (3) can be reduced to the intersection emptiness problem for regular grammars, in a similar way as shown above for (1). □

The next result directly follows from the String Representation Lemma 5.8.

5.10. Corollary. *For a regular drawn (basic) stripe picture language the membership problem can be decided in linear time.*

We compare these results with the following theorem for linear π-grammars.

5.11. Theorem. *For a linear π-language L_1 and a regular π-language L_2, where both* bpic(L_1) *and* bpic(L_2) *are stripe picture languages, it is undecidable whether or not* bpic(L_1) ∩ bpic(L_2) = ∅.

Proof. We refer to [10, proof of Theorem 16]. There it is shown that for a linear π-language $L_1 \subseteq \{r, l, ud\}^*$ and a regular π-language $L_2 \subseteq \{r, l, ud\}^*$ it is undecidable whether $\mathrm{bpic}(L_1) \cap \mathrm{bpic}(L_2) = \emptyset$. Obviously, both $\mathrm{bpic}(L_1)$ and $\mathrm{bpic}(L_2)$ are basic $(0, 0, 1)$-stripe picture languages. $\square$

6. Discussion

We have continued the investigations of chain code picture languages, considering complexity and decidability problems. We conclude by stating some problems which are related to the results presented here:

(i) Is there always a regular $c|p|$-optimal π-language which describes a given regular picture language? (ii) Is it decidable whether two regular π-languages describe the same picture language? (So far we have a similar situation as in the case of deterministic context-free string languages, where intersection emptiness is known to be undecidable and the equivalence problem is still open.) (iii) What is the membership complexity of context-free picture languages? (iv) Find other restricted classes of picture languages, for which we get 'better' decidability and complexity results.

References

[1] A.V. Aho, J.E. Hopcroft and J.D. Ullman, Time and tape complexity of pushdown automaton languages, *Inform. Control* **13** (1968) 186-206.

[2] K. Culik II and E. Welzl, Two-way finite state generators, in: M. Karpinsky, ed., *Foundations of Computation Theory*, Lecture Notes in Computer Science **158** (Springer, Berlin, 1983) 106-114.

[3] H. Freeman, Computer processing of line-drawing images, *Computing Survey* **6** (1974) 57-97.

[4] M.R. Garey and D.S. Johnson, *Computers and Intractability—A Guide to the Theory of NP-Completeness* (Freeman, San Francisco, 1979).

[5] S. Ginsburg, *The Mathematical Theory of Context-Free Languages* (McGraw-Hill, New York, 1966).

[6] M.A. Harrison, *Introduction to Formal Language Theory* (Addison-Wesley, Reading, MA, 1978).

[7] J.E. Hopcroft and J.D. Ullman, *Formal Languages and Their Relation to Automata* (Addison-Wesley, Reading, MA, 1969).

[8] H.A. Maurer, Theoretische Grundlagen der Programmiersprachen, BI, Mannheim, 1969.

[9] H.A. Maurer, G. Rozenberg and E. Welzl, Using string languages to describe picture languages, *Inform. Control* **54** (1982) 155-185.

[10] H.A. Maurer, G. Rozenberg and E. Welzl, Chain code picture languages, in: H. Ehrig, M. Nagl and G. Rozenberg, eds., *Graph Grammars and Their Application to Computer Science*, Lecture Notes in Computer Science **153** (Springer, Berlin, 1983) 232-244.

[11] M.L. Minsky, Recursive unsolvability of Post's problem of "Tag" and other topics in the theory of Turing machines, *Ann. Math.* **74** (1961) 437-455.

[12] A. Rosenfeld, *Picture Languages—Formal Models of Picture Recognition* (Academic Press, New York, 1979).

[13] A. Salomaa, *Formal Languages* (Academic Press, London, 1973).