Available online at www.sciencedirect.com

SCIENCE DIRECT®

Theoretical Computer Science 337 (2005) 51–104

Theoretical Computer Science

www.elsevier.com/locate/tcs

Fundamental study

Adding symbolic information to picture models: definitions and properties

Gennaro Costagliola*, Filomena Ferrucci, Carmine Gravino

Dipartimento di Matematica e Informatica, Universitá degli Studi di Salerno, Via Ponte Don Melillo, 84084 Fisciano (SA), Italy

Received 27 January 2004; accepted 2 March 2005

Communicated by G. Rozenberg

Abstract

In the paper we propose extensions of some picture models, such as colored, drawn and pixel pictures. Such extensions are conceived by observing that a picture may embed more information than the shape, such as colors, labels, etc., which can be represented by a symbol from an alphabet and can be associated to segments, points or pixels. New interesting issues derived from the introduction of symbols will be investigated together with some complexity and decidability questions for the proposed extensions.

Keywords: Chain code pictures; Picture languages; String descriptions; Decidability problems

1. Introduction

In the literature the term *picture* has been used for several different formalisms. One of the most interesting is the definition of picture as given in [16]. Essentially, a picture, named *drawn picture*, consists of unit lines drawn on the Cartesian plane considered as a square grid. For such a model an elegant description of picture languages in terms of string languages is also provided. Indeed, a drawn picture is described by a string over the alphabet

* Corresponding author. Tel.: +39 089 963319; fax: +39 089 963303.
E-mail addresses: gcostagliola@unisa.it (G. Costagliola), fferrucci@unisa.it (F. Ferrucci), gravino@unisa.it (C. Gravino).

0304-3975/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2005.03.028

$\Pi = \{\mathbf{u}, \mathbf{d}, \mathbf{r}, \mathbf{l}\}$ encoding a walk through the picture. The symbol $\mathbf{r}$($\mathbf{l}$, $\mathbf{u}$, $\mathbf{d}$, respectively) is interpreted as 'draw one unit line in the Cartesian plane by moving the pen right (left, up, and down, respectively) from the current position'.

In [15] the model of *pixel pictures* is introduced as direct extension of drawn pictures by changing the interpretation of the symbols in Π. In particular, a string over Π is such that the letters induce still a unit move but it is not the segment under the move that is plotted but the pixel (i.e., unit square) in the right of the move.

Hinz and Welzl in [8] introduced another extension of drawn pictures, named *colored pictures*, obtained by associating colors to segments of a drawn picture, including the "invisible" one (by the "pen up"). A colored picture is described by a word on Π and an alphabet of colors, such that a color letter in the word specifies the color of the segments described by the subsequent move letters. Although such a model turns out to be very interesting, Hinz and Welzl have exploited it mainly to study the *non-connected pictures* which are characterized by the use of only two colors describing 'pen up' and 'pen down', respectively.

In this paper, starting from the approach proposed by Hinz and Welzl we will introduce some extensions of the above picture models, namely *extended colored pictures*, *drawn symbolic pictures* and *symbolic pixel pictures*. Such extensions are conceived to specify further picture's information, such as colors, labels, icons, which is represented by symbols from an alphabet Σ and is associated to segments, points or pixels, respectively.

Another characteristic of the extensions is concerned with the string representations for the proposed models, and allows us to address a specific issue deriving from the symbols introduction. Indeed, it is worth noting that, by following the walk specified by a word, a segment (point or pixel, resp.) may be traversed more than once and different symbols can be associated to such a segment (point or pixel, resp.). Obviously, the same problem also arose in the setting of colored pictures. To solve it Hinz and Welzl proposed to adopt the "set union" approach of coloring, so that if a line is drawn once or more with color red and once or more with color blue, this results in a line with colors red and blue [8]. Moreover, whenever a word contains a subsequence of two or more consecutive colors then always the last color of the subsequence is selected and associated to next moves.

We will generalize the approach proposed by Hinz and Welz by defining suitable mappings for each picture model. This mapping will be parametric with respect to two functions, named *merging functions*, that will be used to establish symbols to be associated to segments, points or pixels, respectively. Many different merging functions can be considered, making the mechanism for symbolic picture description and construction to be very general.

In the paper a deep analysis of the proposed models will be carried out by determining how the choice of *merging functions* may affect some properties of the models. In particular, we will analyze the conditions ensuring that the set of *extended colored pictures*, the set of *drawn symbolic pictures*, and the set of *symbolic pixel pictures* are monoids, finitely generated monoids, and finitely generated inverse monoids.

Moreover, several decidability and complexity properties will be established. As a matter of fact, the analysis of the proposed string representation will allow us to formalize the property of *merging-independency* and we will prove that it is always possible to decide whether or not a context-free grammar generates only *merging-independent* descriptions for pictures. Then, the theoretical analysis of the proposed picture languages will proceed by determining the conditions which ensure the preservation of decidability and complexity

properties of the original models in the setting of the introduced extensions. We will provide two characterizations, one involves merging functions and the other is concerned with the *merging-independency* of string descriptions.

The paper is structured as follows. In Section 2 we settle some preliminary notations from Formal Language Theory and give the basic notions concerning with picture languages and their string descriptions. Sections 3 and 4 formally define *extended colored pictures*, *drawn symbolic pictures* and *symbolic pixel pictures*, their string representations and the grammar model which generates such descriptions. Moreover, a characterization of such pictures according to the finitely generated monoid theory is provided. In Section 5 the notion of *merging-independency* is introduced and the decidability of the *merging-independency* problem for context-free picture grammars is proved. Section 6 focuses on decidability and complexity problems for *extended colored picture languages*, *drawn symbolic picture languages*, and *symbolic pixel picture languages*. Final remarks conclude the paper.

2. Preliminaries

We assume that the reader is familiar with the basic definitions of the Formal Language Theory [6,9,21]. We just recall several standard notations.

Let A be a finite set of symbols called an alphabet. For a string w over A, $|w|$ 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 $pref(w)$ and $suff(w)$ denotes the set of suffixes of w. Analogously for a language L, $pref(L) = \bigcup_{w\in L} pref(L)$ and $suff(L) = \bigcup_{w\in L} suff(L)$.

Given a finite set P, 2^P denotes the set of its subsets. As usual, $\mathbb{Z}$ denotes the set of integers. Given a set S and a binary operation $*$, $M = (S, *)$ is a monoid if (1) $*$ is associative and (2) has a neutral element u (i.e. $u * a = a * u = a$ for all $a \in S$). Let M be a monoid, a generating system of M is a pair (Σ, h) where Σ is an alphabet e, h is a homomorphism on M such that $h(\Sigma^*) = M$ [20].

In the next subsections, we provide some basic notions on drawn pictures [16], colored pictures [8] and pixel pictures [15]. Now, we recall several basic concepts and notations.

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

2.1. Drawn pictures

A *drawn picture* q is a triple $q = \langle p, s, e\rangle$, where p is a connected finite subset of M_1, and the points $s = (0, 0)$ and e are called *start* and *end point* of q, respectively. If p is nonempty then s and e are points in $W(q) = \{v \in M_0 | \{v, v'\} \text{ is in } q, \text{ for some } v' \in M_0\}$. If p is empty then $s = e = (0, 0)$, and $W(q) = \{(0, 0)\}$. Thus, the empty drawn picture is denoted by $\langle \emptyset, (0, 0), (0, 0)\rangle$.

Fig. 1. A drawn picture.

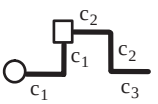


Fig. 2. A colored picture.

As an example, let us consider the drawn picture $q = \langle\{\{(0,0),(1,0)\},\{(1,0),(1,1)\},\{(1,1),(2,1)\},\{(2,1),(2,0)\},\{(2,0),(3,0)\}\},(0,0),(1,1)\rangle$ which is depicted in Fig. 1. A circle is used to denote the *start point* and a square is used to denote the *end point*.

A *drawn picture language* is a set of drawn pictures.

The translational mapping is used to move pictures in the plane preserving their shape, size and relative symbols. For integers m and n, the translational mapping $t_{m,n}$ is a function from M_0 to M_0, which is defined as $t_{m,n}(i,j) = (i+m, j+n)$. Several mappings are induced by $t_{m,n}$. In particular, if P is a subset of M_0, $t_{m,n}(\mathrm{P}) = \{t_{m,n}(v) \mid v \in P\}$. Similarly, if A is a subset of M_1, $t_{m,n}(\mathrm{A}) = \{\{t_{m,n}(v), t_{m,n}(v')\} \mid \{v, v'\} \in A\}$. Let $q_1 = \langle p_1, s_1, e_1\rangle$ and $q_2 = \langle p_2, s_2, e_2\rangle$ be two drawn pictures. q_1 is *translational equivalent* to q_2 if there exist integers m and n such that $q_1 = \langle t_{m,n}(p_2), t_{m,n}(s_2), t_{m,n}(e_2)\rangle$. Moreover, let $t_{m,n}(s_2) = e_1$ for some $m, n \in \mathbb{Z}$, the *concatenation* of q_1 and q_2, denoted by $q_1 \bullet q_2$, is defined as $q_1 \bullet q_2 = \langle p_1 \cup t_{m,n}(p_2), s_1, t_{m,n}(e_2)\rangle$.

Since any point v in the plane has four neighbors (namely, $u(v)$, $d(v)$, $l(v)$, and $r(v)$), a natural way to describe a drawn picture $q = \langle p, s, e\rangle$ is to describe a walk through the picture. Such a walk starts from the start point s, touches at least once each line in p, and ends at the end point e. Each move in the walk from a point v to its neighbor v' is represented in a string by a single symbol: "u" if $v' = u(v)$, "d" if $v' = d(v)$, "l" if $v' = l(v)$, "r" if $v' = \mathrm{r}(v)$. Thus, a walk is described by a string on the alphabet $\Pi = \{u, d, l, r\}$. The symbol u (d, r, and l, respectively) is interpreted as "draw one unit line in the Cartesian plane by moving the pen up (down, right, and left, respectively) from the current position". For example, the string $w = rurdrlul$ describes the drawn picture of Fig. 1. The empty drawn picture $\langle\emptyset, (0,0), (0,0)\rangle$ is described by the empty string ε.

2.2. Colored pictures

A *colored picture* is a triple $q = \langle p, s, e\rangle$ where p is a set of *unit lines* in M_1 that have associated colors from a set $C = (c_1, c_2, \ldots, c_n)$, and it is denoted as $base(q)$, $s = (0,0)$ and e are the *start* and the *end* points [8]. An example of colored picture is depicted in Fig. 2, where segments $\{(0,0),(1,0)\}$ and $\{(1,0),(1,1)\}$ have associated color c_1, segments $\{(1,1),(2,1)\}$ and $\{(2,1),(2,0)\}$ have associated color c_2, and segment $\{(2,0),(3,0)\}$ has associated color c_3.

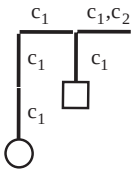


Fig. 3. A colored picture where a segment is drawn twice.

Let us observe that if the set of colors contains only one symbol then we can consider colored pictures as uncolored pictures, i.e. the drawn pictures.

As usual, a *colored picture language* is a set of colored pictures.

The translational mapping $t_{m,n}$ defined in [16] can be used to move colored pictures in the plane preserving their shape, size and colors. Let $q_1 = \langle p_1, s_1, e_1\rangle$ and $q_2 = \langle p_2, s_2, e_2\rangle$ be two colored pictures. q_1 is *translational equivalent* to q_2 if there exist integers m and n such that $q_1 = \langle t_{m,n}(p_2), t_{m,n}(s_2), t_{m,n}(e_2)\rangle$ and $\{t_{m,n}(v), t_{m,n}(v')\}$ in q_2 has associated the same color as $\{v, v'\}$ for each $\{v, v'\}$ in q_1 [8].

A natural way to describe a colored picture $q = \langle p, s, e\rangle$ is to describe a walk through the picture as in the case of drawn pictures. In particular, a walk is described by a string in the set $(C \cup \Pi)^*$, denoted as $C \cup \Pi$-word, where a color letter occurring in the string specifies the color of the subsequent move letters. Symbols from the two sets are shown using different styles for sake of clarity.

Let us consider the following example.

Example 1. Let $w = c_1\mathbf{uu}c_2\mathbf{rr}c_3c_1\mathbf{ld}$ be a $C \cup \Pi$-word. The colored picture described by w is depicted in Fig. 3. In particular, c_1 indicates the color of the moves "**uu**" and "**ld**", and c_2 of the moves "**rr**". Let us observe that there is a segment that is drawn more than once. The string description specifies for such segment first color c_2 and then color c_1. According to the approach proposed in [8] which refers to the "set union concept" of coloring, in the resulting picture both color c_1 and c_2 are associated to this segment. Moreover, c_3 does not specify a color of a move since it is followed by color c_1.

2.3. Pixel pictures

A *pixel picture* is considered as a finite set of pixels [15], where a pixel can be seen as a *unit square* described by $[i, i+1] \times [j, j+1]$ with $(i, j) \in \mathbb{Z}^2$ and denoted as $pix(i, j)$.

More formally, a "*pixel picture*" is a triple $q = \langle p, d, a\rangle$ where p is a set of pixels and is denoted as $base(q)$, $d \in \mathbb{Z}^2$ is *the start point* of q and it is denoted as $start(q)$, and $a \in \mathbb{Z}^2$ is the *end point* of q and it is denoted as $end(q)$. The start and the end point allow to concatenate two pixel pictures. In particular, given two pixel pictures q_1 and q_2, the concatenation of q_1 and q_2, denoted as $q_1 \bullet q_2$ is defined if $start(q_2) = end(q_1)$, and $q_1 \bullet q_2 = (base(q_1) \cup base(q_2), start(q_1), end(q_2))$.

In order to formally define the notion of *connected pixel pictures*, two concepts have been defined [15]: the *armor* of a pixel picture, and the *neighbors* of a pixel in analogy with the concepts of neighbors of a point for the drawn pictures.

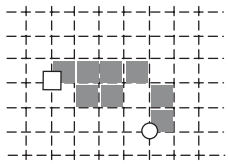


Fig. 4. A pixel picture.

Given a pixel picture $q = \langle p, d, a\rangle$ the armor of p is defined as:

$$arm(q) = \bigcup_{pix(i,j)\in p} \{(i, j), (i + 1, j), (i, j + 1), (i + 1, j + 1)\}.$$

Given a pixel $pix(i, j)$, its neighbors are defined as:

$$neigh(pix(i, j)) = \{pix(k, l) \mid i - 1 \leqslant k \leqslant i + 1 \quad \text{and } j - 1 \leqslant l \leqslant j + 1, \text{ with } (k, l) \neq (i, j)\}.$$

Given a pixel picture $q = (p, d, a)$,

- p is connected if $\forall a, b \in p\ \exists c_1, c_2, \ldots, c_n \in p$ such that $c_1 = a$, $c_n = b$ and $\forall 1 \leqslant i < n\ c_{i+1} \in neigh(c_i)$;
- q is connected if (a) p is connected; (b) $start(q) \in arm(q)$; (c) $end(q) \in arm(q)$.

In other words a pixel picture is connected if for each pixel $pix(i, j)$ it is possible to go from $pix(i, j)$ to any other pixel by following the neighbors of $pix(i, j)$.

Observe that the empty pixel $\langle \emptyset, (0, 0), (0, 0)\rangle$ is considered connected.

A connected pixel picture is shown in Fig. 4, where the pixels of the picture are in gray, and the circle represents the *start point* and the square represents the *end point*.

A *pixel picture language* is a set of pixel pictures.

Two pictures are equivalent if one can be translated into the other by applying *translational mapping* $t_{m,n}$ on pixels which is defined as follows. Given a pixel $pix(x, y)$ and $(m, n) \in \mathbb{Z}^2$, $t_{m,n}(pix(x, y)) = (pix(x + m, y + n))$. Moreover, given a set of pixels P, $t_{m,n}(P) = \{t_{m,n}(p_x) \mid p_x \in P\}$. As a consequence, given two pixel pictures q_1 and q_2, q_1 is *translational equivalent* to q_2, if there exist two integers i and j such that: $t_{i,j}(base(q_1)) = base(q_2)$, $t_{i,j}(start(q_1)) = start(q_2)$, $t_{i,j}(end(q_1)) = end(q_2)$.

A string on alphabet Π can be used to describe a pixel picture. Indeed, in this setting a letter of Π induces a unit move and the plot of the pixel in the right of the move [15]. Thus, the pixel picture of Fig. 4 can be described by the Π-word "**uulldull**", as shown in Fig. 5.

3. Extended colored picture languages

In this section we introduce an extension of colored pictures, named *extended colored pictures*, and theoretically analyze such picture model. According to the extension not only colors can be associated to segments but any kind of information denoted by symbols. However, the more distinguishing characteristic of the extension is concerned with the string representation proposed for *extended colored pictures*. Indeed, we will use a more

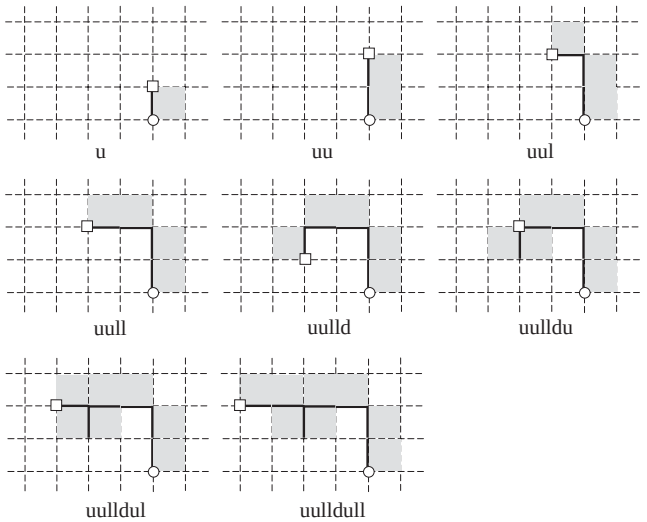


Fig. 5. A pixel picture.

general interpretation to associate symbols to segments w.r.t. the approach based on the set union concept which is exploited by Hinz and Welzl in [8].

We start by introducing a formal definition of *extended colored picture* which makes use of a function δ to explicitly associate symbols to segments. We use ϕ to denote the invisible symbol, i.e. a special symbol that has no visual representation when it is associated to a segment in a picture. In the sequel, we will indicate with Σ an arbitrary alphabet of symbols which does not contain ϕ and with $\Sigma_\phi = \Sigma \cup \{\phi\}$ the set which contains ϕ.

Definition 1. An *extended colored picture* is a triple $q = \langle\langle p, s, e\rangle, \Sigma, \delta\rangle$, where $\langle p, s, e\rangle$ is a drawn picture, $\Sigma = \{c_1, c_2, \ldots, c_n\}$ is an alphabet of colors, $\delta : p \to \Sigma_\phi$ is the function that specifies the symbols associated to the segments in p. The *start* and *end* points of q are s and e, respectively. The *empty extended colored picture* is denoted by $\underline{\varepsilon} = \langle\langle\emptyset, (0,0), (0,0)\rangle, \Sigma, \delta\rangle$ where δ is not defined.

Let us consider the following examples.

Example 2. Let $\Sigma = \{s_1, s_2, s_3, s_4, s_5, s_6\}$ be a set where symbol s_1 denotes a thin line, s_2 denotes a thick line, s_3 denotes a thin dashed line, s_4 denotes a thin dotted line, s_5 denotes a thick dashed line, and s_6 denotes a thick dotted line. These symbols can be associated to the segments of *extended colored picture* to indicate the type of drawing as shown in Fig. 6(i). Thus, let $q = \langle sc, \Sigma, \delta\rangle$ be an *extended colored picture*, where $sc = \langle\{\{(0,0),(0,1)\}, \{(0,1),(0,2)\}, \{(0,2),(1,2)\}, \{(1,2),(2,2)\}, \{(1,2),(1,1)\}, \{(1,1),(2,1)\}\}, (0,0), (2,1)\rangle$, and δ is the function such that $\delta(\{(0,0),(0,1)\}) = s_1$, $\delta(\{(0,1),(0,2)\}) = s_4$, $\delta(\{(0,2),(1,2)\}) = s_2$, $\delta(\{(1,2),(2,2)\}) = s_5$, $\delta(\{(1,2),(1,1)\}) = s_3$, $\delta(\{(1,1),(2,1)\}) = s_6$. The visual representation of this picture is depicted in Fig. 6(ii).

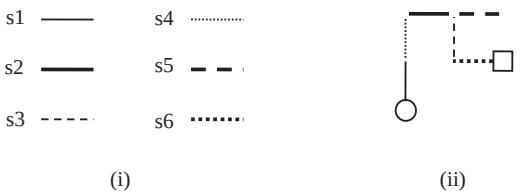


Fig. 6. The set of symbols Σ (i) and the *extended colored picture* (ii) of Example 2.

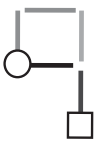


Fig. 7. The *extended colored picture* of Example 3.

Example 3. Let $q = \langle sc, \Sigma, \delta\rangle$ be an *extended colored picture*, where $sc = \langle\{\{(0,0),(1,0)\},\{(1,0),(1,1)\},\{(1,0),(1,-1)\},\{(1,1),(0,1)\},\{(0,1),(0,0)\}\},(0,0),(1,-1)\rangle$, $\Sigma = \{(r,b,g) \mid 0 \leqslant r,b,g < 255\}$, and δ is the function such that $\delta(\{(0,0),(1,0)\}) = (60,60,60)$, $\delta(\{(1,0),(1,1)\}) = (150,150,150)$, $\delta(\{(1,0),(1,-1)\}) = (87,87,87)$, $\delta(\{(1,1),(0,1)\}) = (128,128,128)$, $\delta(\{(0,1),(0,0)\}) = (128,128,128)$. In other words, the symbols associated to the segments represent colors in the *RGB* representation. Let us observe that symbol ϕ can correspond to (255, 255, 255), i.e. the white color. The visual representation of this picture is depicted in Fig. 7.

An *extended colored picture language* is a set of *extended colored pictures*.

Let us observe that if Σ is a set of colors then ϕ can represent the empty color denoted by $\uparrow$ in [8], and non-connected colored pictures can be described.

In Section 3.1 we will present a string-based representation of *extended colored pictures*, and in Section 3.2 a characterization of *extended colored pictures* according to the finitely generated monoid theory will be provided.

3.1. A string based representation of extended colored pictures

In order to introduce a string representation of *extended colored pictures*, let us consider Example 1. It shows that the approach proposed by Hinz and Welzl is characterized by the following behavior in the interpretation of a string description w. First, observe that c_3 has been ignored since it is followed by another color in the string. Thus, whenever w contains a subsequence of two or more consecutive colors then always the last color of the subsequence is selected and associated to next moves. Moreover, the string interpretation exhibits a different behavior whenever the string describes a segment more than once possibly with different colors. Indeed, in this case the set union approach is applied. As a matter of fact, in Example 1 both c_2 and c_1 are associated to the same segment since such a segment has been traversed twice, first with color c_2 and then with c_1.

The approach proposed by Hinz and Welz can be generalized by introducing two functions, named *merging functions*, which can be used to establish the actual symbols to be associated to segments whenever the above situations hold. As an example, let us consider the set of symbols $\Sigma = \{s_1, s_2, s_3, s_4, s_5, s_6\}$ given in Example 2 and two functions f and g such that $f(s_2, s_3) = s_5$ and $g(s_4, s_3) = s_3$. If f is used whenever a segment is traversed two times and g whenever the string description contains a subsequence of two or more consecutive symbols in Σ, then $\Sigma \cup \Pi$-word "$s_1\mathbf{u}s_4\mathbf{u}s_2\mathbf{rr}s_4s_3\mathbf{ld}s_6\mathbf{r}$" describes the picture depicted in Fig. 6(ii).

The notion of merging function is formally defined as follows.

Definition 2. Let Σ_ϕ be an alphabet of the symbols. A *merging function* f on Σ_ϕ is a mapping $\Sigma_\phi \times \Sigma_\phi \to \Sigma_\phi$ such that:
- f is a total function, and
- $f(\phi, a) = f(a, \phi) = a$ for each $a \in \Sigma_\phi$.

In the sequel, Φ_{Σ_ϕ} will denote the family of merging functions on Σ_ϕ.

Now, we provide the definition of *extended colored picture* described by a $\Sigma \cup \Pi$-word w with respect to merging functions f and g, denoted by $ecp = ecpic_f^g(w)$. Let us observe that f (i.e., the function at the pedex) will be used whenever segments are traversed more than once and g (i.e., the function at the apex) will be used whenever a word contains consecutive symbols in Σ. So, we will refer to f as a *merging function for overlapping* and to g as a *merging function for consecutive symbols*.

Informally, the picture *ecp* is obtained by scanning the string w from left to right and by associating to w a triple (ecp, σ, ρ) where $\sigma, \rho \in \Sigma_\phi$, in the following inductive way. At the beginning when the initial position in the plane is set to (0,0), and a pointer p_t points to the first symbol in w, the triple is set to $(\underline{\varepsilon}, \phi, \phi)$. Now, let us suppose that (q, l_σ, l_π) is the triple associated to the just scanned subsentence w' of w, where q is the picture described by w', l_σ is the merging of the last scanned consecutive symbols in Σ, and l_π is the symbol associated to the last move, and that p_t points to the next symbol τ (see Fig. 8 for example). If $\tau \in \Sigma$ then triple $(q, g(l_\sigma, \tau), l_\pi)$ is associated to the string $w'\tau$ (see Fig. 8(i)). If $\tau \in \Pi$ then the current position on the plane is updated in agreement with τ and (q', ϕ, l_σ) ((q', l_σ, l_π), resp.) is the triple associated to $w'\tau$ where picture q' is obtained as described in the sequel. If $l_\sigma \neq \phi$ ($l_\sigma = \phi$, resp.), i.e. the previous scanned symbol was a symbol in Σ (a move in Π, resp.), and the current segment is new then l_σ (l_π, resp.) is associated to it (see Fig. 8(ii) (see Fig. 8(iii), resp.)). Otherwise, if the current segment is not new then the symbol associated to such a segment is updated with $f(\gamma, l_\sigma)$ (see Fig. 8(iv)) ($f(\gamma, l_\pi)$, resp. (see Fig. 8(v))) where γ is the previously associated symbol.

Formally, we have the following definition where c_f^g denotes the function used to construct the triple.

Definition 3. Let w be a $\Sigma \cup \Pi$-word and f and g be two merging functions on Σ_ϕ. A triple is associated with w in the following inductive way:
- if $w = \varepsilon$, then $c_f^g(w) = (\underline{\varepsilon}, \phi, \phi)$;

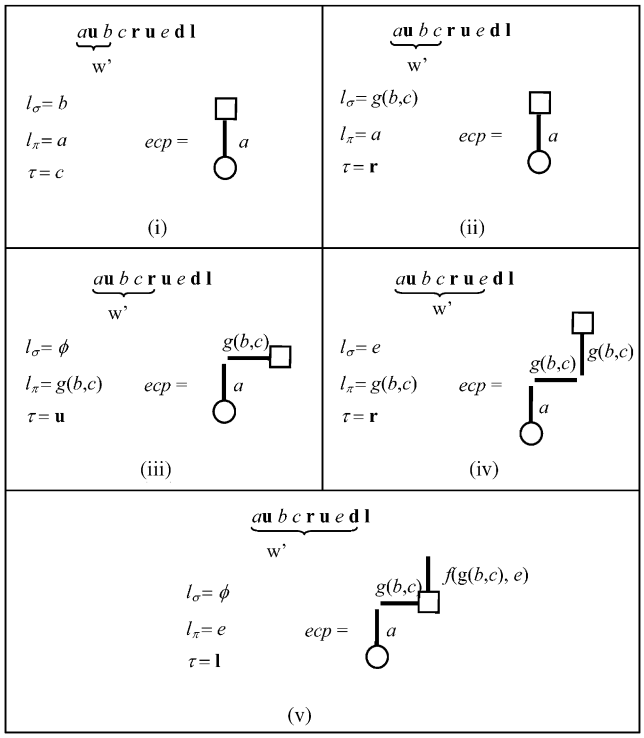


Fig. 8. Some of the triples to obtain from $\Sigma \cup \Pi$-word "a**u**bc**r**$u$$e$**d**$l$" the corresponding extended colored picture.

- if $w = w'\tau$ for some $w' \in (\Sigma \cup \Pi)^*$, with $c_f^g(w') = (\langle sc, \Sigma, \delta\rangle, l_\sigma, l_\pi)$, $sc = \langle p, s, e\rangle$, $l_\sigma, l_\pi \in \Sigma_\phi$, then

$$c_f^g(w) = \begin{cases} (\langle sc, \Sigma, \delta\rangle, g(l_\sigma, \tau), l_\pi) & \text{if } \tau \in \Sigma, \\ (\langle\langle p \cup \{e, \tau(e)\}, s, \tau(e)\rangle, \Sigma, \delta_1\rangle, \phi, l_\sigma) & \text{if } \tau \in \Pi \text{ and } l_\sigma \neq \phi, \\ (\langle\langle p \cup \{e, \tau(e)\}, s, \tau(e)\rangle, \Sigma, \delta_2\rangle, l_\sigma, l_\pi) & \text{if } \tau \in \Pi \text{ and } l_\sigma = \phi, \end{cases}$$

  where $\delta_1 : (p \cup \{e, \tau(e)\}) \to \Sigma_\phi$ is the function such that

$$\delta_1(\{v, v'\}) = \begin{cases} l_\sigma & \text{if } \{v, v'\} \notin p, \\ f(\delta(\{v, v'\}), l_\sigma) & \text{otherwise,} \end{cases}$$

  where $\delta_2 : (p \cup \{e, \tau(e)\}) \to \Sigma_\phi$ is the function such that:

$$\delta_2(\{v, v'\}) = \begin{cases} l_\pi & \text{if } \{v, v'\} \notin p, \\ f(\delta(\{v, v'\}), l_\pi) & \text{otherwise.} \end{cases}$$

The *extended colored picture* described by w w.r.t. merging functions f and g, denoted by $ecpic_f^g(w)$, is *ecp* if $c_f^g(w) = (ecp, \sigma, \rho)$.

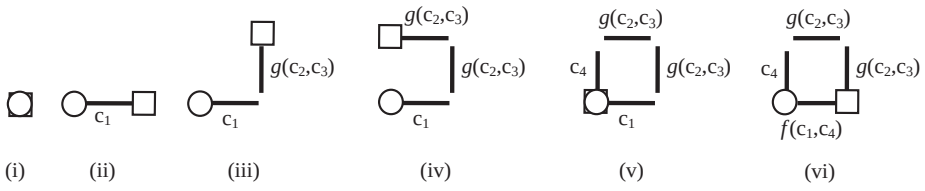


Fig. 9. The application of c_f^g on the prefixes of "$c_1\mathbf{r}c_2c_3\mathbf{ul}c_4\mathbf{dr}$".

In order to illustrate the use of the function $ecpic_f^g$ let us consider the following example.

Example 4. Let $\Sigma = \{c_1, c_2, c_3, c_4\}$, f and g be two merging functions, and $w = c_1\mathbf{r}c_2c_3\mathbf{ul}$ $c_4\mathbf{dr}$ be a $\Sigma \cup \Pi$-word. The *extended colored picture* described by w w.r.t. f and g is $ecpic_f^g(w)$=$\langle\langle\{\{(0,0),(1,0)\},\{(1,0),(1,1)\},\{(1,1),(0,1)\},\{(0,1),(0,0)\}\},(0,0),(1,0)\rangle, \Sigma, \delta\rangle$ which is constructed by applying c_f^g on the prefixes of w as illustrated in Fig. 9 where each row in the table describes the obtained triple and the corresponding δ function.

As usual, the definition of $ecpic_f^g$ can be easily extended to the languages of $\Sigma \cup \Pi$-words in a natural manner, i.e., given a $\Sigma \cup \Pi$-language S, $ecpic_f^g(S) = \{ecpic_f^g(w) \mid w \in S\}$.

Now, let us observe that many different *merging functions* can be considered (independently of the question whether they are for overlapping or for consecutive symbols), making the mechanism for picture description and construction very general. To illustrate such an aspect, let us consider Example 2 where symbols associated to segments represent information on their drawing. Whenever two or more conflicting symbols (i.e., symbols representing conflicting drawing) are associated to segments, only one of them should be considered. Let us consider the $\Sigma \cup \Pi$-word "$s_4\mathbf{uu}s_2\mathbf{rr}s_4s_3\mathbf{ld}$" which describes symbols s_4 and s_3 for segment {(1,2),(1,1)}, and symbols s_4, s_3 and s_2 for segment {(1,2),(2,2)}. In this case, a merging function which selects one of the input symbols should be suitable. To this aim, we might use a merging function, named *sfirst* in the following, which always selects the first symbol associated to a segment during scanning and ignores other subsequent symbols (see Fig. 10(i)).

Analogously, we might use a merging function, named g_{HW} in the sequel, that always selects the second input, thus producing an overwriting effect (see Fig. 10(ii)). This function is denoted by g_{HW} since it corresponds to the approach for consecutive colors described by Hinz and Welzl. As another example we might consider a merging function *ord* that uses a order relation on Σ_ϕ, such that $ord(a,b) = max(a,b)$. Thus, given the set of symbols of Example 2, if $s_1 < s_2 < s_3 < s_4 < s_5 < s_6$ then $ecpic_{ord}^{ord}(s_4\mathbf{uu}s_2\mathbf{rr}s_4s_3\mathbf{ld})$ denotes the picture in Fig. 10(iii).

In some cases a selection function could not be the most suitable choice. As an example, if symbols in Σ represent colors, we could associate to each segment the combination of the colors specified for it. Thus, suppose that colors are represented by the *RGB* representation, i.e. $\Sigma = \{(r,b,g) \mid 0 \leqslant r, b, g < 255\}$, and consider the additive function *addRGB* defined as follows: given two colors $c_1 = (r_1, g_1, b_1)$ and $c_2 = (r_2,$

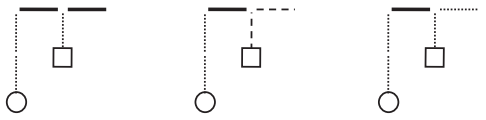


Fig. 10. The *extended colored pictures* denoted by (i) $ecpic^{sfirst}_{sfirst}(s_4\mathbf{uu}s_2\mathbf{rr}s_4s_3\mathbf{ld})$, (ii) $ecpic^{g_{HW}}_{g_{HW}}(s_4\mathbf{uu}s_2\mathbf{rr}s_4s_3\mathbf{ld})$, and (iii) $ecpic^{ord}_{ord}(s_4\mathbf{uu}s_2\mathbf{rr}s_4s_3\mathbf{ld})$.

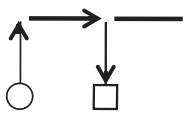


Fig. 11. The *extended colored picture* $ecpic^{union}_{union}(\rightarrow s_1\mathbf{u}s_2\mathbf{rr}s_4\mathbf{l}s_2\mathbf{l}\rightarrow \mathbf{rd})$.

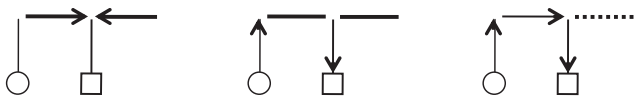


Fig. 12. The *extended colored pictures* denoted by (i) $ecpic^{g_{HW}}_{union}(\rightarrow s_1\mathbf{u}s_2\mathbf{rr}s_4 \leftarrow \mathbf{l}s_2\mathbf{l} \rightarrow \mathbf{r} \rightarrow s_1\mathbf{d})$, (ii) $ecpic^{union}_{sfirst}(\rightarrow s_1\mathbf{u}s_2\mathbf{rr}s_4 \leftarrow \mathbf{l}s_2\mathbf{l} \rightarrow \mathbf{r} \rightarrow s_1\mathbf{d})$, and (iii) $ecpic^{sfirst}_{g_{HW}}(\rightarrow s_1\mathbf{u}s_2\mathbf{rr}s_4 \leftarrow \mathbf{l}s_2\mathbf{l} \rightarrow \mathbf{r} \rightarrow s_1\mathbf{d})$.

g_2, b_2) *addRGB*$(c_1, c_2) = ((r_1{*}r_2)/255, (g_1{*}g_2)/255, (b_1{*}b_2)/255)$. Then, given word $w =$ (100, 100, 100)**r**(150, 150, 150)**u**(128, 128, 128)**ld**(153, 153, 153)**r**(87, 87, 87)**d**, which specifies the two colors (100, 100, 100) and (153, 153, 153) for the segment $\{(0, 0), (1, 0)\}$, $ecpic^{addRGB}_{addRGB}(w)$ denotes the picture of Fig. 7.

According to our definition of $ecpic^{g}_{f}$, the merging functions make a binary choice at each occurring overlap or subsequent colors, and associate a unique symbol to each segment. It is obvious that this is not the only possible solution. For instance, a merging function which allows us to collect for each segment all the symbols associated to it could be considered. This is the approach proposed by Hinz and Welzl for overlapping segments. Such a merging function can be formalized as follows. Let $\Sigma = 2^{\Delta} \setminus \emptyset$ with Δ a finite set of symbols. Merging function *union* : $\Sigma_\phi \times \Sigma_\phi \rightarrow \Sigma_\phi$ is such that $union(A, B) = A \cup B$. Thus, let $\Sigma = \{s_1, s_2, s_3, s_4, s_5, s_6, \rightarrow, \leftarrow\}$, where symbols s_1, s_2, s_3, s_4, s_5 and s_6 are those of Example 2, and $\leftarrow$ and $\rightarrow$ are symbols that indicate lines with direction. Then, $ecpic^{union}_{union}(\rightarrow s_1\mathbf{u}s_2\mathbf{rr}s_4\mathbf{l}s_2\mathbf{l}\rightarrow \mathbf{rd})$ denotes the picture of Fig. 11.

It is worth noting that f and g could be merging functions which exibit a different behavior. So, let us consider again set $\Sigma = \{s_1, s_2, s_3, s_4, s_5, s_6, \rightarrow, \leftarrow\}$, and let $w = \rightarrow s_1\mathbf{u}s_2\mathbf{rr}s_4 \leftarrow \mathbf{l}s_2\mathbf{l} \rightarrow \mathbf{r} \rightarrow s_1\mathbf{d}$. Then, $ecpic^{g_{HW}}_{union}(w)$, $ecpic^{union}_{sfirst}(w)$ and $ecpic^{sfirst}_{g_{HW}}(w)$ denote the pictures of Figs. 12(i), (ii) and (iii), respectively. Please note that for the picture of Fig. 12(ii) the substrings $s_4 \leftarrow \mathbf{l}$ and $\rightarrow \mathbf{r}$ have no effects on the whole picture since the merging function *sfirst* associates the first symbol and ignores other subsequent symbols.

In agreement with the definition of merging function, a grammar generating *extended colored picture descriptions* must be able to generate strings of symbols and moves. Given two alphabets Γ and Λ, a $\Gamma \cup \Lambda$-*grammar* is a string grammar that generates a $\Gamma \cup \Lambda$-*language*. More formally, we have the following definitions:

Definition 4. Let Γ and Λ be two disjoint finite sets of symbols, a $\Gamma \cup \Lambda$-*grammar* is specified by a 5-tuple $\langle \Gamma, \Lambda, \mathrm{N}, \mathrm{P}, \mathrm{S} \rangle$, and is defined as a grammar $\mathrm{G} = \langle \Gamma \cup \Lambda, \mathrm{N}, \mathrm{P}, \mathrm{S} \rangle$ for which $\mathrm{L(G)} \subseteq (\Gamma \cup \Lambda)^*$.

In the sequel, an *extended colored picture description grammars* will be a $\Sigma \cup \Pi$-grammar that will be used to generate *extended colored picture* descriptions and will be denoted by $G_{\Sigma \cup \Pi}$. Thus, the *extended colored picture grammars* and *languages* are defined as follows.

Definition 5. An *extended colored picture grammar* $\mathbf{G}$ is a pair $\langle G_{\Sigma \cup \Pi}, ecpic_f^g \rangle$ where $G_{\Sigma \cup \Pi}$ is a $\Sigma \cup \Pi$-grammar and $ecpic_f^g$ is the mapping that translates a $\Sigma \cup \Pi$-word into an *extended colored picture* in agreement with merging functions f and g.

Given an *extended colored picture grammar* $\mathbf{G} = \langle G_{\Sigma \cup \Pi}, ecpic_f^g \rangle$, the *extended colored picture language* $\mathbf{L}$ generated by $\mathbf{G}$, denoted by $\mathbf{L}(\mathbf{G})$, is

$$\mathbf{L}(\mathbf{G}) = ecpic_f^g(L(G_{\Sigma \cup \Pi})) = \{ecpic_f^g(x) \mid x \in L(G_{\Sigma \cup \Pi})\}.$$

Example 5. Let $\mathbf{G} = \langle G_{\Sigma \cup \Pi}, ecpic_f^g \rangle$ be an *extended colored picture grammar*, where $G_{\Sigma \cup \Pi}$ is the $\Sigma \cup \Pi$-grammar specified as follows: $G_{\Sigma \cup \Pi} = \langle \{c_1, c_2, c_3, c_4\}, \Pi, \{S, B, C\}, P, S \rangle$ where S is the start symbol and P contains the productions: $S \to c_1 \mathbf{r} c_2 B$, $B \to c_3 \mathbf{u} C \mathbf{dr}$, $C \to \mathbf{l} c_4$, $C \to \mathbf{d} C$. $\mathbf{L}(\mathbf{G})$ contains the *extended colored picture* denoted with (vi) in Fig. 9, indeed $G_{\Sigma \cup \Pi}$ generates $\Sigma \cup \Pi$-word "$c_1 \mathbf{r} c_2 c_3 \mathbf{ul} c_4 \mathbf{dr}$".

An *extended colored picture language* $\mathbf{L}$ is *context-free* if there exists a *context-free extended colored picture grammar* $\mathbf{G}$ that generates $\mathbf{L}$.

An *extended colored picture grammar* $\mathbf{G} = \langle G_{\Sigma \cup \Pi}, ecpic_f^g \rangle$ is *context-free* if $G_{\Sigma \cup \Pi}$ is a *context-free extended colored picture description grammar*.

3.2. The set of extended colored pictures as a finitely generated monoid

It is worth noting that all picture semigroups used in "picture language theory", such as drawn pictures [16], non-connected pictures [8] and connected pixel pictures [15] have been characterized as finitely generated monoids. In this subsection we provide an analogous characterization for the set of *extended colored pictures* presented in this paper. In particular, we will identify the properties of merging functions ensuring that the set of *extended colored pictures* together with the concatenation operator is a monoid and a finitely generated monoid.

The *concatenation* of two *extended colored pictures* can be obtained by overlapping the end point of the first picture with the start point of the second picture. However, the presence of symbols requires a suitable extension of such a procedure to determine the symbol to be associated to each overlapping segment. In order to clarify such need, let us consider

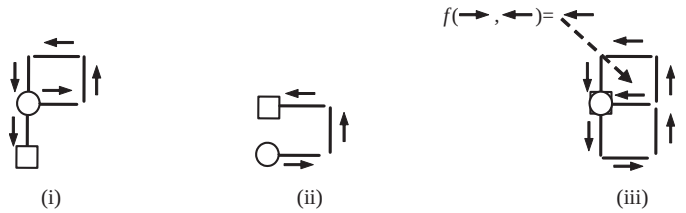


Fig. 13. Two *extended colored pictures* (i) and (ii), and their concatenation (iii).

the two *extended colored pictures* $q_1 = \langle\langle p_1, s_1, e_1\rangle, \Sigma, \delta_1\rangle$ and $q_2 = \langle\langle p_2, s_2, e_2\rangle, \Sigma, \delta_2\rangle$ depicted in Figs. 13(i) and (ii), respectively. Note that the symbols associated to the segments represent the directions. The *extended colored picture* obtained by concatenating q_1 and q_2 presents an overlapping segment. As a matter of fact, $\{(0, 0), (1, 0)\}$ does not only belong to p_1 but also to $t_{0,-1}(p_2)$ because there exists segment $\{(0, 1), (1, 1)\}$ in p_2. Since both δ_1 and δ_2 provide a symbol for this segment we can exploit a merging function for overlapping to determine the symbol to be associated to this segment. As an example, if we use the merging function f such that $f(\rightarrow, \leftarrow) = \leftarrow$ then we obtain the *extended colored picture* depicted in Fig. 13(iii).

Thus, the concatenation operator is defined below w.r.t. a merging function.

Definition 6. Let $q_1 = \langle\langle p_1, s_1, e_1\rangle, \Sigma, \delta_1\rangle$ and $q_2 = \langle\langle p_2, s_2, e_2\rangle, \Sigma, \delta_2\rangle$ be two *extended colored pictures*. Let $m, n \in \mathbb{Z}$ be such that $t_{m,n}(s_2) = e_1$, and let f be a merging function on Σ_ϕ. The *concatenation* of q_1 and q_2 w.r.t. f, denoted by $q_1 \bullet_f q_2$, is defined as

$$q_1 \bullet_f q_2 = \langle\langle p_1 \cup t_{m,n}(p_2), s_1, t_{m,n}(e_2)\rangle, \Sigma, \delta\rangle$$

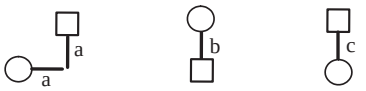
where $\delta : (p_1 \cup t_{m,n}(p_2)) \rightarrow \Sigma_\phi$ is such that

$$\delta(\{v, v'\}) = \begin{cases} \delta_1(\{v, v'\}) \\ \qquad \text{if } (\{v, v'\} \in p_1) \text{ and } (\{v, v'\} \notin t_{m,n}(p_2)), \\ \delta_2(t_{-m,-n}(\{v, v'\})) \\ \qquad \text{if } (\{v, v'\} \notin p_1) \text{ and } (\{v, v'\} \in t_{m,n}(p_2)), \\ f(\delta_1(\{v, v'\}), \delta_2(t_{-m,-n}(\{v, v'\}))) \\ \qquad (\{v, v'\} \in p_1) \text{ and } (\{v, v'\} \in t_{m,n}(p_2)). \end{cases}$$

It is easy to verify that the concatenation of two *extended colored pictures* is always defined and it is unique with respect to a given merging function f.

In the sequel we provide the properties of merging functions ensuring that the set of *extended colored pictures* equipped with the concatenation operator $\bullet_f$ is a monoid and a finitely generated monoid. To this aim, *ECP* will denote the set of *extended colored pictures* on a set Σ_ϕ and $(ECP, \bullet_f)$ the monoid of pictures.

Proposition 1. *$(ECP, \bullet_f)$ and (Σ_ϕ, f) are monoids iff f is associative.*



q_1 q_2 q_3

Fig. 14. Three *extended colored pictures* q_1, q_2, and q_3.

Proof. Let us observe that by definition of merging function it can be easily verified that if f is associative then (Σ_ϕ, f) is a monoid. Moreover, the associative property of $\bullet_f$ directly follows from the associative property of f, and $\underline{\varepsilon} = \langle\langle\emptyset, (0,0), (0,0)\rangle, \Sigma, \delta\rangle$ is the neutral element of $\bullet_f$.

In contrast, if f is not associative then (Σ_ϕ, f) is not a monoid and $\bullet_f$ is not associative. Indeed, let us consider a merging function f such that $f(a, f(b, c)) \neq f(f(a, b), c)$ and pictures q_1, q_2, q_3 depicted in Fig. 14. It is easy to verify that $(q_1 \bullet_f q_2) \bullet_f q_3$ is not equal to $q_1 \bullet_f (q_2 \bullet_f q_3)$. □

Proposition 2 states that (*ECP*, $\bullet_f$) is a finitely generated monoid.

Proposition 2. *Let $\bullet_f$ be the concatenation operator w.r.t. associative f.* (*ECP*, $\bullet_f$) *is a finitely generated monoid.*

Proof. Since (Σ_ϕ, f) is a monoid, from Proposition 1 it follows that (*ECP*, $\bullet_f$) is a monoid. We prove that it is finitely generated by induction on the length of pictures. The basis of induction is given by the *extended colored pictures* of length 1. In this case the thesis can be easily proved. Indeed, the *extended colored pictures* $\langle\langle\{(0, 0), \pi(0, 0)\}, (0, 0), \pi(0, 0)\rangle, \Sigma, \delta\rangle$ with $\pi \in \Pi$ are finitely generated since Σ and Π are finite sets. Let us suppose that the thesis holds for *extended colored pictures* of length n and prove that it also holds for *extended colored pictures* of length $n + 1$. It can be easily verified from the definition of concatenation, that an *extended colored picture* q, with $|q| = n + 1$, can be obtained as the concatenation of two *extended colored pictures* q_1 and q_2 such that $q = q_1 \bullet_f q_2$ with $|q_1| \leqslant n$ and $|q_2| \leqslant n$. By inductive hypothesis q_1 and q_2 are finitely generated so q is finitely generated. □

Let us observe that all picture semigroups used in picture language theory are inverse monoids. In the following we characterize merging functions for which the set of *extended colored pictures* is an inverse monoid.

Let us recall that a monoid M=(S, $*$) is inverse if for each $x \in$ S, there exists a unique element, denoted x^{-1}, such that $x*x^{-1}*x = x$ and $x^{-1}*x*x^{-1} = x^{-1}$. The element x^{-1} is named *inverse* of x [18].

Proposition 3 provides the conditions ensuring that monoid (*ECP*, $\bullet_f$) is inverse.

Proposition 3. (*ECP*, $\bullet_f$) *is an inverse monoid iff* (Σ_ϕ, f) *is an inverse monoid.*

Proof. Let us suppose that (Σ_ϕ, f) is an inverse monoid. By definition of inverse monoid we have that for each $x \in \Sigma_\phi$ there exists a unique symbol x^{-1} such that $f(f(x, x^{-1}), x) = x$

(i) (ii)

Fig. 15. $ecpic_f^g(c_1\mathbf{rr}c_2c_3) \bullet_f ecpic_f^g(\mathbf{u}c_4)$ (i), and $ecpic_f^g(c_1\mathbf{rr}c_2c_3\mathbf{u}c_4)$ (ii).

and $f(f(x^{-1}, x), x^{-1}) = x^{-1}$. From Proposition 1 $(ECP, \bullet_f)$ is a monoid. In order to prove that $(ECP, \bullet_f)$ is inverse, let us consider an *extended colored picture* $q = \langle\langle p, s, e\rangle, \Sigma, \delta_1\rangle$ in *ECP* and show that there exists a unique inverse q^{-1}. Let $q^{-1} = \langle\langle p, e, s\rangle, \Sigma, \delta_2\rangle$ where δ_2 is such that $f(f(\delta_1(\{v, v'\}), \delta_2(\{v, v'\})), \delta_1(\{v, v'\})) = \delta_1(\{v, v'\})$ and $f(f(\delta_2(\{v, v'\}), \delta_1(\{v, v'\})), \delta_2(\{v, v'\})) = \delta_2(\{v, v'\})$ for each $\{v, v'\} \in p$. Since (Σ_ϕ, f) is an inverse monoid $\delta_2(\{v, v'\})$ exists and is unique for each $\delta_1(\{v, v'\})$. This ensures that $\delta_2(\{v, v'\})$ is well-defined and q^{-1} exists and is unique for each $q \in ECP$.

Let $(ECP, \bullet_f)$ be an inverse monoid. By definition of inverse monoid we have that for each $q \in ECP$ there exists a unique picture q^{-1} such that $q \bullet_f q^{-1} \bullet_f q = q$ and $q^{-1} \bullet_f q \bullet_f q^{-1} = q^{-1}$. By definition of $\bullet_f$, the *inverse extended colored picture* $q^{-1} = \langle\langle p, e, s\rangle, \Sigma, \delta_2\rangle$ is such that for each $\{v, v'\} \in p$ if $\delta_1(\{v, v'\}) = x$ then $\delta_2(\{v, v'\}) = y$, where y is such that $f(f(x, y), x) = x$ and $f(f(y, x), y) = y$. Since $(ECP, \bullet_f)$ is an inverse monoid then the above y exists and is unique for each $x \in \Sigma_\phi$. This ensures that (Σ_ϕ, f) is an inverse monoid. □

From Propositions 2 and 3, we have the following theorem.

Theorem 1. *$(ECP, \bullet_f)$ is a finitely generated inverse monoid iff (Σ_ϕ, f) is an inverse monoid.*

Another interesting characteristic of picture models is that the mapping function which allows us to translate a string description into a picture is a morphism from the monoid of the string descriptions into the monoid of pictures. In the following we analyze the problem in the context of the monoid of *extended colored pictures* $(ECP, \bullet_f)$ and provide some conditions ensuring the above property for $ecpic_f^g$ in our context.

We start by observing that in general function $ecpic_f^g$ is not a morphism from the free monoid $(\Sigma \cup \Pi)^*$ into the monoid $(ECP, \bullet_f)$. As a matter of fact, let us consider the $\Sigma \cup \Pi$-word "$c_1\mathbf{rr}c_2c_3\mathbf{u}c_4$" and the subwords "$c_1\mathbf{rr}c_2c_3$" and "$\mathbf{u}c_4$". The picture obtained $ecpic_f^g(c_1\mathbf{rr}c_2c_3) \bullet_f ecpic_f^g(\mathbf{u}c_4)$ depicted in Fig. 15(i) is not equal to $ecpic_f^g(c_1\mathbf{rr}c_2c_3\mathbf{u}c_4)$ depicted in Fig. 15(ii). Informally, $g(c_2, c_3)$ cannot be associated to move **u** in the concatenation $ecpic_f^g(c_1\mathbf{rr}c_2\ c_3) \bullet_f ecpic_f^g(\mathbf{u}c_4)$.

Now, let $S = (\Sigma(\Pi \cup \Sigma)^*\Pi \cup \{\varepsilon\})$, it is easy to verify that S equipped with the string concatenation is a monoid. As a matter of fact, for $a, b, c \in S$, $(a \cdot b) \cdot c = a \cdot (b \cdot c)$ and ε is the neutral element of concatenation operator.

In the following we will show that function $ecpic_f^g$ is a morphism from the monoid $(S, \cdot)$ into the monoid $(ECP, \bullet_f)$.

Proposition 4. *Let ECP be the set of* extended colored pictures *on a set* Σ_ϕ *and* $\bullet_f$ *be the concatenation operator w.r.t. f. If* (Σ_ϕ, f) *is a monoid then function* $ecpic_f^g$ *is a morphism from the monoid* $(S, \cdot)$ *into the monoid* $(ECP, \bullet_f)$.

Proof. By the definition of $ecpic_f^g$ it turns out that: $ecpic_f^g(\varepsilon) = \underline{\varepsilon}$. Moreover, it is easy to verify that the associativity of f ensures that $ecpic_f^g(w_1w_2) = ecpic_f^g(w_1) \bullet_f ecpic_f^g(w_2)$ for all $w_1, w_2 \in S$ and $g \in \Phi_{\Sigma_\phi}$. □

As a consequence of the above theorem, according to finitely generated monoid representation theory [17], each word in S represents an *extended colored picture* and $(S, ecpic_f^g)$ is the generating system of $(ECP, \bullet_f)$.

It is worth noting that given an *extended colored picture* q and two merging functions f and g, there may exist many different words w in S describing q (i.e., such that $ecpic_f^g(w) = q$). As described in [15] we can define the congruence associated to the generating system $(S, ecpic_f^g)$, denoted by $\equiv_{f,g}$, such that:

$$w \equiv_{f,g} w' \quad \text{if and only if } ecpic_f^g(w) = ecpic_f^g(w') \quad \text{for } w, w' \in S.$$

Thus, let q be an *extended colored picture* and $w \in S$ such that $ecpic_f^g(w) = q$. The *description language* of q w.r.t. f and g is the set

$$[w]_{\equiv_{f,g}} = \{w' \in S \mid w \equiv_{f,g} w'\}$$

It can be easily proved that each equivalence class of $\equiv_{f,g}$ is a rational language of S [1].

It is worth noting that this result does not hold for extended colored non-connected pictures. Informally, extended colored non-connected pictures can be considered as drawn pictures where some spatial relations (i.e., some segments) are not visualized. Even though this aspect may appear trivial, it introduces another degree of freedom in picture string descriptions which determines different complexity properties. As a matter of fact, it can be easily proved that the string description language of an extended colored non-connected picture is not a regular language [8].

Now, let $S2 = (\Sigma(\Pi \cup \Sigma)^* \cup \{\varepsilon\})$. In general $ecpic_f^g$ is not a morphism from the monoid $(S2, \cdot)$ into the monoid $(ECP, \bullet_f)$. Indeed, let us consider word "$c_1\mathbf{r}c_2c_3\mathbf{ul}c_4\mathbf{dr}$" and the subwords "$c_1\mathbf{r}c_2$" and "$c_3\mathbf{ul}c_4\mathbf{dr}$". The picture obtained $ecpic_f^g(c_1\mathbf{r}c_2) \bullet_f ecpic_f^g(c_3\mathbf{ul}c_4\mathbf{dr})$ depicted in Fig. 16(i) is not equal to $ecpic_f^g(c_1\mathbf{r}c_2c_3\mathbf{ul}c_4\mathbf{dr})$ depicted in Fig. 16(ii). Nevertheless, if we consider only g_{HW} as merging function for consecutive symbols, we can easily verify that $ecpic_f^{g_{HW}}$ is a morphism from the monoid $(S2, \cdot)$ into the monoid $(ECP, \bullet_f)$.

The above results are summarized in Table 1.

4. Extensions of drawn pictures and pixel pictures

Other extensions of picture models can be provided on the base of the observation that a picture may embed more information than the simple shape. In particular, another extension of drawn pictures, named *drawn symbolic pictures*, can be obtained by associating symbols

(i) (ii)

Fig. 16. $ecpic^{g}_{f}(c_1\mathbf{r}c_2) \bullet_f ecpic^{g}_{f}(c_3\mathbf{ul}c_4\mathbf{dr})$ (i), and $ecpic^{g}_{f}(c_1\mathbf{r}c_2c_3\mathbf{ul}c_4\mathbf{dr})$ (ii).

Table 1
Morphisms from the monoid of strings into the monoid of extended colored pictures

	$(\Sigma \cup \Pi)^*$	$\Sigma(\Sigma \cup \Pi)^* \cup \{\varepsilon\}$	$\Sigma(\Sigma \cup \Pi)^*\Pi \cup \{\varepsilon\}$
$ecpic^{g}_{f}$	No	No	Yes
$ecpic^{gHW}_{f}$	No	Yes	Yes

from an alphabet to the points of a drawn picture. Moreover, the notion of *symbolic pixel picture* can be defined by associating symbols to each pixel of a pixel picture.

Section 4.1 will be devoted to the formalization of *drawn symbolic pictures* and in Section 4.2 we will present *symbolic pixel pictures*.

4.1. Drawn symbolic picture languages

A *drawn symbolic picture* can be seen as a set of symbols arranged on a two-dimensional plane such that pairs of symbols on the plane are connected by unit lines to form a connected graph.

The definition of *drawn symbolic picture* can be formalized as follows.

Definition 7. A *drawn symbolic picture* is a triple $q = \langle sc, \Sigma, \delta\rangle$, where *sc* is a drawn picture, Σ is an alphabet of symbols, and $\delta : W(sc) \to \Sigma_\phi$. The *start* and *end points* of *q* are the start and end points of *sc*, respectively. The *empty drawn symbolic picture* is denoted by $\underline{\varepsilon_D} = \langle\langle\emptyset, (0,0), (0,0)\rangle, \Sigma, \delta\rangle$, where $\delta((0,0)) = \phi$.

Let us consider the following examples.

Example 6. Let $q = \langle sc, \Sigma, \delta\rangle$ be a *drawn symbolic picture*, where $sc = \langle\{\{(0,0),(0,1)\}, \{(0,1),(0,2)\}, \{(0,2),(1,2)\}, \{(1,2),(2,2)\}, \{(1,2),(1,1)\}\}, (0,0), (1,1)\rangle$, $\Sigma = \{a, b\}$, and δ is the function such that $\delta((0,0)) = \phi$, $\delta((0,1)) = b$, $\delta((0,2)) = a$, $\delta((1,2)) = \phi$, $\delta((2,2)) = a$, $\delta((1,1)) = \phi$. The visual representation of this picture is depicted in Fig. 17.

Example 7. Let $q = \langle sc, \Sigma, \delta\rangle$ be a *drawn symbolic picture*, where $sc = \langle\{\{(0,0),(0,1)\}, \{(0,1),(1,1)\}\}, (0,0), (0,1)\rangle$, $\Sigma = \{(r,b,g) \mid 0 \leqslant r, b, g < 255\}$, and δ is the function such that $\delta((0,0)) = (165,165,165)$, $\delta((0,1)) = (120,120,120)$, $\delta((1,1)) = (188,188,188)$.

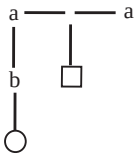


Fig. 17. A *drawn symbolic picture*.

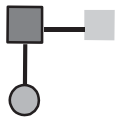


Fig. 18. A *drawn symbolic picture* where points have associated colors.

In other words, the symbols associated to points represent colors in *RGB* representation. The visual representation of this picture is depicted in Fig. 18.

As usual a *drawn symbolic picture language* is a set of *drawn symbolic pictures*.

Let us observe that a string description of a *drawn symbolic picture* can be given in analogy with the description of drawn pictures by describing a walk through the picture. Such a walk is described by a word on the alphabets of the symbols and moves. As an example, given the *drawn symbolic picture* depicted in Fig. 17, a possible walk is described by $\Sigma \cup \Pi$-word "**u***b***u***a***rr***a***ld**".

Analogously to *extended colored picture* descriptions, a $\Sigma \cup \Pi$-word w might associate several symbols to a point since w might traverse a point more than once or specify consecutive symbols in Σ before a move. The problem is addressed as in the case of *extended colored picture* by using a merging function f for point overlapping and a merging function g for consecutive symbols. So, given a $\Sigma \cup \Pi$-word $w = \mathbf{u}b\mathbf{u}ac\mathbf{rr}ba\mathbf{ll}o\mathbf{rd}$ and merging functions f and g such that $f(e, o) = a$, $g(a, c) = e$ and $g(b, a) = a$, w describes the *drawn symbolic picture* depicted in Fig. 17.

Now, we introduce the definition of *drawn symbolic picture* described by a $\Sigma \cup \Pi$-word w with respect to *merging functions* f and g, denoted by $dsp = dspic_f^g(w)$.

The picture *dsp* is obtained by scanning w from left to right and by associating to w a couple (*dsp*, σ) where $\sigma \in \Sigma_\phi$, in the following inductive way. At the beginning when the initial position in the plane is set to (0,0) and a pointer p_t points to the first symbol in w, the couple is set to ($\underline{\varepsilon_D}$, ϕ). Now, let us suppose that (q, l_σ) is the couple associated to the just scanned subsentence w' of w, where q is the picture described by w', and l_σ is the symbol to be associated to the current position (i.e. the end point of q), and that p_t points to the next symbol τ. If $\tau \in \Pi$ and ρ is the symbol associated to the current position then $f(\rho, l_\sigma)$ is written on that position. Moreover, the current position on the plane is updated in agreement with τ and ϕ is associated to that position if it is new. A picture q' is obtained and couple (q', ϕ) is associated to $w'\tau$. If $\tau \in \Sigma$ then couple (q, $g(l_\sigma, \tau)$) is associated to $w'\tau$.

At the end of scanning of w, if (dsp, σ) is the couple obtained and $\sigma \neq \phi$ then *dsp* is updated by associating the symbol σ to the end point.

Formally, we have the following definition where d_f^g denotes the function used to construct the triple.

Definition 8. Let w be a $\Sigma \cup \Pi$-word and f and g be two merging functions on Σ_ϕ. A couple is associated with w in the following inductive way:

- if $w = \varepsilon$, then $d_f^g(w) = (\underline{\varepsilon_D}, \phi)$;
- if $w = w'\tau$ for some $w' \in (\Sigma \cup \Pi)^*$, with $d_f^g(w') = (\langle sc, \Sigma, \delta\rangle, l_\sigma)$, $sc = \langle p, s, e\rangle$, then

$$d_f^g(w) = \begin{cases} (\langle\langle p \cup \{e, \tau(e)\}, s, \tau(e)\rangle, \Sigma, \delta_1\rangle, \phi) & \text{if } \tau \in \Pi, \\ (\langle\langle p, s, e\rangle, \Sigma, \delta\rangle, g(l_\sigma, \tau)) & \text{if } \tau \in \Sigma, \end{cases}$$

where δ_1: $W(sc) \cup \{\pi(e)\} \to \Sigma_\phi$ is the labeling function such that

$$\delta_1(v) = \begin{cases} f(\delta(v), l_\sigma) & \text{if } v = e, \\ \phi & \text{if } v \notin W(sc), \\ \delta(v) & \text{if } v \in W(sc) \text{ and } v \neq e. \end{cases}$$

The *drawn symbolic picture* described by w w.r.t. merging functions f and g, denoted as $dspic_f^g(w)$ is

- dsp if $d_f^g(w) = (dsp, \sigma)$ with $\sigma = \phi$,
- $\langle\langle p, s, e\rangle, \Sigma, \underline{\delta}\rangle$ if $d_f^g(w) = (\langle\langle p, s, e\rangle, \Sigma, \delta\rangle, \sigma)$ with $\sigma \neq \phi$,

where

$$\underline{\delta}(v) = \begin{cases} f(\delta(v), \sigma) & \text{if } v = e, \\ \delta(v) & \text{otherwise.} \end{cases}$$

In order to illustrate the use of the function $dspic_f^g$ let us consider the following example.

Example 8. Let $\Sigma = \{c_1, c_2, c_4, c_5\}$, f and g be two merging functions, and $w = \mathbf{r}c_1c_2\mathbf{uld}\,\mathbf{r}c_4c_5$ be a $\Sigma \cup \Pi$-word. The *drawn symbolic picture* described by w w.r.t. f and g is $dspic_f^g(w) = \langle\langle\{\{(0,0),(1,0)\},\{(1,0),(1,1)\},\{(1,1),(0,1)\},\{(0,1),(0,0)\}\},(0,0),(1,0)\rangle, \Sigma, \delta\rangle$ where δ is such that $\delta(0,0) = \phi$, $\delta(0,1) = \phi$, $\delta(1,1) = \phi$, $\delta(1,0) = f(g(c_1,c_2), g(c_4,c_5))$. $dspic_f^g(w)$ is constructed by applying d_f^g on the prefixes of w as illustrated in Fig. 19.

As usual, the definition of $dspic_f^g$ can be easily extended to the languages of $\Sigma \cup \Pi$-words in a natural manner, i.e., given a $\Sigma \cup \Pi$-language S, $dspic_f^g(S) = \{dspic_f^g(w) \mid w \in S\}$.

Thus, a *drawn symbolic picture grammar* $\mathbf{G}$ is a pair $\langle G_{\Sigma\cup\Pi}, dspic_f^g\rangle$ and the *drawn symbolic picture language* $\mathbf{L}$ generated by $\mathbf{G}$ is $\mathbf{L}(\mathbf{G}) = dspic_f^g(\mathrm{L}(G_{\Sigma\cup\Pi})) = \{dspic_f^g(x) \mid x \in \mathrm{L}(G_{\Sigma\cup\Pi})\}$.

Let us note that the properties of merging functions which have allowed us to prove that the set of *extended colored pictures* is a monoid, a finitely generated monoid and a finitely generated inverse monoid can be exploited to provide an analogous characterization for *drawn symbolic pictures*. To this aim, the concatenation operator between *drawn symbolic pictures* can be defined w.r.t. a merging function as shown in the following.

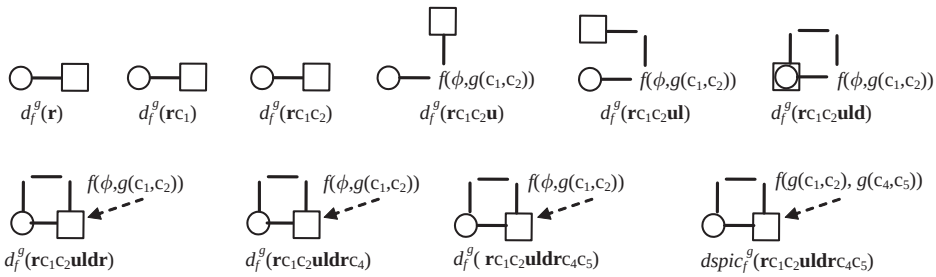


Fig. 19. The application of d_f^g on the prefixes of "$\mathbf{r}c_1c_2\mathbf{uldr}c_4c_5$", and $dspic_f^g(\mathbf{r}\,c_1c_2\mathbf{uldr}c_4c_5)$.

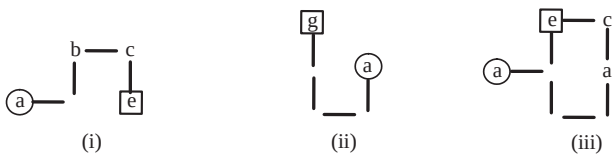


Fig. 20. Two *drawn symbolic pictures* (i) and (ii), and their concatenation (iii).

Definition 9. Let $q_1 = \langle\langle p_1, s_1, e_1\rangle, \Sigma, \delta_1\rangle$ and $q_2 = \langle\langle p_2, s_2, e_2\rangle, \Sigma, \delta_2\rangle$ be two *drawn symbolic pictures*. Let $m, n \in \mathbb{Z}$ be such that $t_{m,n}(s_2) = e_1$, and let f be a merging function on Σ_ϕ. The *concatenation* of q_1 and q_2 w.r.t. f, denoted by $q_1 \bullet_f q_2$, is defined as

$$q_1 \bullet_f q_2 = \langle\langle p_1 \cup t_{m,n}(p_2), s_1, t_{m,n}(e_2)\rangle, \Sigma, \delta\rangle,$$

where $\delta : W(\langle p_1 \cup t_{m,n}(p_2), s_1, t_{m,n}(e_2)\rangle) \to \Sigma_\phi$ is such that

$$\delta(v) = \begin{cases} \delta_1(v) \\ \qquad \text{if } (v \in W(\langle p_1, s_1, e_1\rangle)) \text{ and } (v \notin W(t_{m,n}(\langle p_2, s_2, e_2\rangle))), \\ \delta_2(t_{-m,-n}(v)) \\ \qquad \text{if } (v \notin W(\langle p_1, s_1, e_1\rangle)) \text{ and } (v \in W(t_{m,n}(\langle p_2, s_2, e_2\rangle))), \\ f(\delta_1(v), \delta_2(t_{-m,-n}(v))) \\ \qquad \text{if } (v \in W(\langle p_1, s_1, e_1\rangle)) \text{ and } (v \in W(t_{m,n}(\langle p_2, s_2, e_2\rangle))). \end{cases}$$

As an example, let q_1 and q_2 be the two pictures depicted in Figs. 20(i) and (ii), respectively. By using a merging function f such that $f(e, a) = a$ and $f(b, g) = e$, $q_1 \bullet_f q_2$ produces the *drawn symbolic picture* of Fig. 20(iii).

It is easy to verify that the concatenation of two *drawn symbolic pictures* is always defined and it is unique with respect to a given merging function *f*.

Moreover, let *DSP* denote the set of *drawn symbolic pictures* on a set Σ_ϕ and (*DSP*, $\bullet_f$) denote the set of *drawn symbolic pictures* equipped with the concatenation operator w.r.t. merging function *f*. The following proposition can be stated.

Proposition 5. *Let* $\bullet_f$ *be the concatenation operator w.r.t. associative f.*
(i) (*DSP*, $\bullet_f$) *and* (Σ_ϕ, f) *are monoids*;

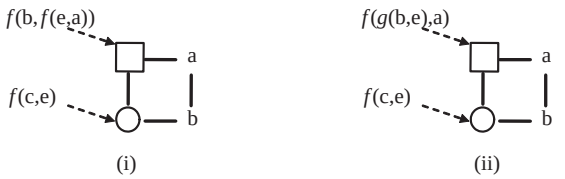


Fig. 21. $dspic^g_f(\mathbf{u}b) \bullet_f dspic^g_f(e\mathbf{r}a\mathbf{dl}\ c\mathbf{r}b\mathbf{l}e\mathbf{u}a)$ (i) and $dspic^g_f(\mathbf{u}be\mathbf{r}a\mathbf{dl}c\mathbf{r}b\mathbf{l}e\mathbf{u}a)$ (ii).

Table 2
Morphisms from the monoid of strings into the monoid of drawn symbolic pictures

	$(\Sigma \cup \Pi)^*$	$\Pi(\Sigma \cup \Pi)^* \cup \{\varepsilon\}$
$dspic^g_f$	No	Yes
$dspic^f_f$	Yes	Yes

(ii) $(DSP, \bullet_f)$ *is a finitely generated monoid*;
(iii) $(DSP, \bullet_f)$ *is a finitely generated inverse monoid iff* (Σ_ϕ, f) *is an inverse monoid*.

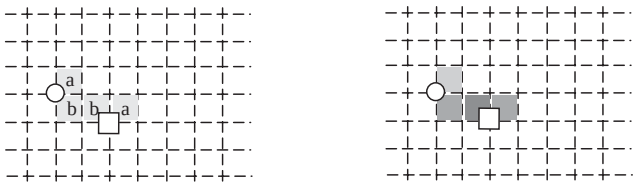
We omit the proofs for statements of the proposition since they are similar to those provided in Section 3.2 for *extended colored pictures*.

Now, let us observe that $dspic^g_f$ is not a morphism from the free monoid $(\Sigma \cup \Pi)^*$ into the monoid $(DSP, \bullet_f)$ since the condition $dspic^g_f(w_1w_2) = dspic^g_f(w_1) \bullet_f dspic^g_f(w_2)$ does not hold. As an example, let us consider $\Sigma \cup \Pi$-word "$\mathbf{u}be\mathbf{r}a\mathbf{dl}c\mathbf{r}b\mathbf{l}e\mathbf{u}a$" and subwords "$\mathbf{u}b$" and "$e\mathbf{r}a\mathbf{dl}c\mathbf{r}b\mathbf{l}e\mathbf{u}a$". The picture obtained $dspic^g_f(\mathbf{u}b) \bullet_f dspic^g_f(e\mathbf{r}a\mathbf{dl}c\mathbf{r}b\mathbf{l}e\mathbf{u}a)$ depicted in Fig. 21(i) is not equal to $dspic^g_f(\mathbf{u}be\mathbf{r}a\mathbf{dl}c\mathbf{r}b\mathbf{l}e\mathbf{u}a)$ depicted in Fig. 21(ii). Nevertheless, when we consider a unique merging function (i.e., a merging function for overlapping equal to the merging function for consecutive symbols) we have that function $dspic^f_f$ is a morphism from the free monoid $(\Sigma \cup \Pi)^*$ into the monoid $(DSP, \bullet_f)$.

As a consequence, according to finitely generated monoid representation theory [17], each $\Sigma \cup \Pi$-word represents a *drawn symbolic picture* and $(\Sigma \cup \Pi, dspic^f_f)$ is the generating system of $(DSP, \bullet_f)$. Moreover, it is easy to verify that the *description language* of a *drawn symbolic picture* q is a regular language.

In the case we would consider two different merging functions we could restrict the set of possible string descriptions to obtain an analogous result. Indeed, let $S = (\Pi(\Pi \cup \Sigma)^* \cup \{\varepsilon\})$, it is easy to verify that S equipped with the string concatenation is a monoid. As a matter of fact, for $a, b, c \in S$ $(a \cdot b) \cdot c = a \cdot (b \cdot c)$ and ε is the neutral element of the concatenation operator. Furthermore, if (Σ_ϕ, f) is a monoid then function $dspic^g_f$ is a morphism from the monoid $(S, \cdot)$ into the monoid $(DSP, \bullet_f)$. Indeed, by the definition of $dspic^g_f$ it turns out that: $dspic^g_f(\varepsilon) = \underline{\varepsilon_D}$ and the associativity of f ensures that $dspic^g_f(w_1w_2) = dspic^g_f(w_1) \bullet_f dspic^g_f(w_2)$ for all $w_1, w_2 \in S$, and $g \in \Phi_{\Sigma_\phi}$.

The above results are summarized in Table 2.



(i) (ii)

Fig. 22. Two *symbolic pixel pictures.*

4.2. Symbolic pixel picture languages

A *symbolic pixel picture* can be seen as a set of symbols associated to a set of pixels arranged on a two-dimensional plane. The formal definition of *symbolic pixel picture* follows.

Definition 10. A *symbolic pixel picture* is a triple $q = \langle sc, \Sigma, \delta \rangle$, where $sc = \langle p, s, e \rangle$ is a connected pixel picture, Σ is an alphabet of symbols, and $\delta : p \to \Sigma_\phi$ is a function that associates a symbol to each pixel. *s* and *e* are the start and end points of *q*, respectively. The *empty symbolic pixel picture* is denoted by $\underline{\varepsilon_P} = \langle \langle \emptyset, (0,0), (0,0) \rangle, \Sigma, \delta \rangle$ where δ is not defined.

Let us consider the following example.

Example 9. Let $q_1 = \langle sc, \Sigma_1, \delta_1 \rangle$ be a *symbolic pixel picture,* where $sc = \langle \{ pix(0,0), pix(0,-1), pix(1,-1), pix(2,-1) \}, (0,0), (2,-1) \rangle$, $\Sigma_1 = \{a, b\}$, and $\delta_1(pix(0,0)) = a$, $\delta_1(pix(0,-1)) = b$, $\delta_1(pix(1,-1)) = b$, and $\delta_1(pix(2,-1)) = a$.

Let $q_2 = \langle sc, \Sigma_2, \delta_2 \rangle$ be a *symbolic pixel picture,* where $\Sigma_2 = \{(r,b,g) \mid 0 \leqslant r, b, g < 255\}$ (i.e., colors in *RGB* representation), and $\delta_2(pix(0,0)) = (192,192,192)$, $\delta_2(pix(0,-1)) = (150,150,150)$, $\delta_2(pix(1,-1)) = (128,128,128)$, $\delta_2(pix(2,-1)) = (150,150,150)$.

The visual representations of q_1 and q_2 are depicted in Figs. 22(i) and (ii), respectively.

A *symbolic pixel picture language* is a set of *symbolic pixel pictures*.

Let us observe that a string based representation of *symbolic pixel pictures* can be given by considering words on alphabets Π and Σ. Since, a $\Sigma \cup \Pi$-word can specify several symbols for a pixel we still exploit merging functions to determine the actual symbols to be associated to pixels. Moreover, function $dspic_f^g$ introduced for *drawn symbolic picture descriptions* can be easily adapted to work on *symbolic pixel picture descriptions*. Indeed, such a function has to differ only in the interpretation of the move letters. A letter of Π induces still a unit move but we do not plot the segment under the move but the pixel in the right of the move [15]. In particular, given a point $(i, j) \in \mathbb{Z}$ we have

- $pix(\mathbf{u}(i, j)) = pix(i, j)$,
- $pix(\mathbf{d}(i, j)) = pix(i-1, j-1)$,

- $pix(\mathbf{r}(i, j)) = pix(i, j-1)$,
- $pix(\mathbf{l}(i, j)) = pix(i-1, j)$.

We define a function $sppic_f^g$ which maps a $\Sigma \cup \Pi$-word w into the corresponding picture $spp = sppic_f^g(w)$.

The picture *spp* is obtained by scanning w from left to right and by associating to w a triple (spp, σ, p_x) where $\sigma \in \Sigma_\phi$ and p_x is a pixel, in the following inductive way. At the beginning when the initial position in the plane is set to (0,0) and a pointer p_t points to the first symbol in w, the triple is set to $(\underline{\varepsilon_P}, \phi, pix(0, 0))$. Now, let us suppose that (q, l_σ, l_{px}) is the triple associated to the just scanned subsentence w' of w, where q is the picture described by w', l_σ and l_{px} are the last symbol and the last pixel examined, respectively, and that p_t points to the next symbol τ. If $\tau \in \Pi$ and ρ is the symbol associated to l_{px} then $f(\rho, l_\sigma)$ is written on that pixel. Moreover, the current pixel on the plane is updated in agreement with τ, obtaining pixel l'_{px}, and ϕ is associated to l'_{px} if it is new. Thus, a picture q' is obtained and triple (q', ϕ, l'_{px}) is associated to $w'\tau$. If $\tau \in \Sigma$ then triple $(q, g(l_\sigma, \tau), l_{px})$ is associated to $w'\tau$.

At the end of scanning of w, if (spp, σ, p_x) is the triple obtained and $\sigma \neq \phi$ then *spp* is updated by associating the symbol σ to p_x.

Formally, we have the following definition where s_f^g denotes the function used to construct the triple.

Definition 11. Let w be a $\Sigma \cup \Pi$-word and f and g be two merging functions on Σ_ϕ. A triple is associated with w in the following inductive way:

- if $w = \varepsilon$, then $s_f^g(w) = (\underline{\varepsilon_P}, \phi, pix(0, 0))$;
- if $w = w'\tau$ for some $w' \in (\Sigma \cup \Pi)^*$, with $s_f^g(w') = (\langle sc, \Sigma, \delta\rangle, l_\sigma, l_{px})$, $sc = \langle p, s, e\rangle$, $l_\sigma \in \Sigma_\phi$, $l_{px} \in p$, then

$$s_f^g(w) = \begin{cases} (\langle\langle p \cup \{pix(\tau(e))\}, s, \tau(e)\rangle, \Sigma, \delta_1\rangle, \phi, pix(\tau(e))) & \text{if } \tau \in \Pi, \\ (\langle sc, \Sigma, \delta\rangle, g(l_\sigma, \tau), l_{px}) & \text{if } \tau \in \Sigma, \end{cases}$$

where $\delta_1 : (p \cup \{pix(\tau(e))\}) \to \Sigma_\phi$ is the labeling function such that

$$\delta_1(pix(i, j)) = \begin{cases} f(\delta(pix(i, j)), l_\sigma) & \text{if } pix(i, j) = l_{px}, \\ \phi & \text{if } pix(i, j) \notin p, \\ \delta(pix(i, j)) & pix(i, j) \in p \text{ and } pix(i, j) \neq l_{px}. \end{cases}$$

The symbolic pixel picture described by w w.r.t. merging functions f and g, denoted by $sppic_f^g(w)$, is

- *spp* if $s_f^g(w) = (spp, \sigma, p_x)$ with $\sigma = \phi$;
- $\langle sc, \Sigma, \underline{\delta}\rangle$ if $s_f^g(w) = (\langle sc, \Sigma, \delta\rangle, \sigma, p_x)$ with $\sigma \neq \phi$,

where

$$\underline{\delta}(pix(i, j)) = \begin{cases} f(\delta(pix(i, j)), \sigma) & \text{if } pix(i, j) = p_x, \\ \delta(pix(i, j)) & \text{otherwise.} \end{cases}$$

In order to illustrate the use of the function $sppic_f^g$ let us consider the following example.

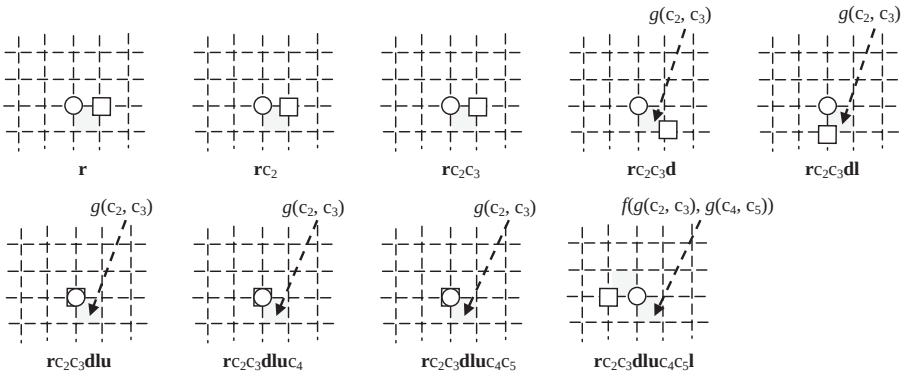


Fig. 23. The application of s_f^g on the prefixes of "$\mathbf{r}c_2c_3\mathbf{dlu}c_4c_5\mathbf{l}$".

Example 10. Let $\Sigma = \{c_2, c_3, c_4, c_5\}$, f and g be two merging functions, and $w = \mathbf{r}c_2c_3\,\mathbf{dlu}c_4c_5\mathbf{l}$ be a $\Sigma \cup \Pi$-word. The *symbolic pixel picture* described by w w.r.t. f and g is $sppic_f^g(w) = \langle\langle\{pix(-1,0), pix(0,-1)\}, (0,0), (-1,0)\rangle, \Sigma, \delta\rangle$ where δ is such that $\delta(pix(0,-1)) = f(g(c_2,c_3), g(c_4,c_5))$ and $\delta(pix(-1,0)) = \phi$. $sppic_f^g(w)$ is constructed by applying s_f^g on the prefixes of w as illustrated in Fig. 23.

As usual, the definition of $sppic_f^g$ can be easily extended to the languages of $\Sigma \cup \Pi$-words in a natural manner, i.e., given a $\Sigma \cup \Pi$-language S, $sppic_f^g(S) = \{sppic_f^g(w) \mid w \in S\}$.

Thus, a *symbolic pixel picture grammar* $\mathbf{G}$ is a pair $\langle G_{\Sigma\cup\Pi}, sppic_f^g\rangle$ and the *symbolic pixel picture language* $\mathbf{L}$ generated by $\mathbf{G}$ is $\mathbf{L}(\mathbf{G}) = sppic_f^g(\mathrm{L}(G_{\Sigma\cup\Pi})) = \{sppic_f^g(x) \mid x \in \mathrm{L}(G_{\Sigma\cup\Pi})\}$.

Now, by exploiting the properties of merging functions which have allowed us to prove that the set of *extended colored pictures* is a monoid and a finitely generated monoid, we will provide an analogous characterization for the set of *symbolic pixel pictures*.

We start by defining the concatenation operator between *symbolic pixel pictures* w.r.t. a merging function, as shown in the following.

Definition 12. Let $q_1 = \langle\langle p_1, s_1, e_1\rangle, \Sigma, \delta_1\rangle$ and $q_2 = \langle\langle p_2, s_2, e_2\rangle, \Sigma, \delta_2\rangle$ be two *symbolic pixel pictures*. Let $m, n \in \mathbb{Z}$ be such that $t_{m,n}(s_2) = e_1$, and let f be a merging function on Σ_ϕ. The *concatenation* of q_1 and q_2 w.r.t. f, denoted by $q_1 \bullet_f q_2$, is defined as

$$q_1 \bullet_f q_2 = \langle\langle p_1 \cup t_{m,n}(p_2)^1, s_1,\ t_{m,n}(e_2)\rangle, \Sigma, \delta\rangle,$$

where $\delta : (p_1 \cup t_{m,n}(p_2)) \to \Sigma_\phi$ is such that for each pixel $p_x \in (p_1 \cup t_{m,n}(p_2))$

$$\delta(p_x) = \begin{cases} \delta_1(p_x) & \text{if } p_x \in p_1 \text{ and } p_x \notin t_{m,n}(p_2), \\ \delta_2(t_{-m,-n}(p_x)) & \text{if } p_x \notin p_1 \text{ and } p_x \in t_{m,n}(p_2), \\ f(\delta_1(p_x), \delta_2(t_{-m,-n}(p_x))) & \text{if } p_x \in p_1 \text{ and } p_x \in t_{m,n}(p_2). \end{cases}$$

[1] Let us remember that given a pixel $pix(x,y)$ and $(m,n) \in \mathbb{Z}^2$, $t_{m,n}(pix(x,y)) = (pix(x+m, y+n))$. Moreover, given a set of pixels P, $t_{m,n}(P) = \{t_{m,n}(p_x) \mid p_x \in P\}$.

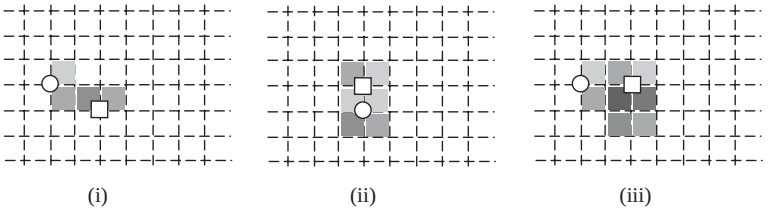


Fig. 24. Two symbolic pixel pictures (i) and (ii), and their concatenation (iii).

Fig. 25. $sppic_f^g(\mathbf{r}c_1\mathbf{d}c_2c_3\mathbf{u}c_4)$ (i) and $sppic_f^g(\mathbf{r}c_1\mathbf{d}c_2) \bullet_f sppic_f^g(c_3\mathbf{u}c_4)$ (ii).

As an example, let q_1 and q_2 be the two *symbolic pixel pictures* depicted in Figs. 24(i) and (ii), respectively. By using function *addRGB* introduced in Section 3, the concatenation of q_1 and q_2 produces the picture of Fig. 24(iii).

It is easy to verify that the concatenation of two *symbolic pixel pictures* is always defined and it is unique with respect to a given merging function *f*.

Moreover, let *SPP* denote the set of *symbolic pixel pictures* on a set Σ_ϕ and (*SSP*, $\bullet_f$) denote the set of *symbolic pixel pictures* equipped with the concatenation operator w.r.t. merging function *f*. The following proposition can be stated.

Proposition 6. *Let $\bullet_f$ be the concatenation operator w.r.t. associative f.*
(i) *(SPP, $\bullet_f$) and (Σ_ϕ, f) are monoids*;
(ii) *(SPP, $\bullet_f$) is a finitely generated monoid*;
(iii) *(SPP, $\bullet_f$) is a finitely generated inverse monoid iff (Σ_ϕ, f) is an inverse monoid*.

Let us observe that $sppic_f^g$ is not a morphism from the free monoid $(\Sigma \cup \Pi)^*$ into the monoid (*SSP*, $\bullet_f$) since the condition $sppic_f^g(w_1w_2) = sppic_f^g(w_1) \bullet_f sppic_f^g(w_2)$ does not hold. Indeed, let us consider the $\Sigma \cup \Pi$-word "$\mathbf{r}c_1\mathbf{d}c_2c_3\mathbf{u}c_4$" and the subwords "$\mathbf{r}c_1\mathbf{d}c_2$" and "$c_3\mathbf{u}c_4$". The picture obtained $sppic_f^g(\mathbf{r}c_1\mathbf{d}c_2\ c_3\mathbf{u}c_4)$ depicted in Fig. 25(i) is not equal to $sppic_f^g(\mathbf{r}c_1\mathbf{d}c_2) \bullet_f sppic_f^g(c_3\mathbf{u}c_4)$ depicted in Fig. 25(ii).

Now, let $S = (\Pi(\Pi \cup \Sigma)^* \cup \{\varepsilon\})$. If (Σ_ϕ, f) is a monoid then the function $sppic_f^g$ is a morphism from the monoid $(S, \cdot)$ into the monoid (*SPP*, $\bullet_f$). Indeed, by the definition of $sppic_f^g$ it turns out that: $sppic_f^g(\varepsilon) = \underline{\varepsilon_P}$ and the associativity of *f* ensures that $sppic_f^g(w_1w_2) = sppic_f^g(w_1) \bullet_f sppic_f^g(w_2)$ for all $w_1, w_2 \in S$ and $g \in \Phi_{\Sigma_\phi}$.

As a consequence, according to finitely generated monoid representation theory [17], each word in S represents a *symbolic pixel picture* and $(S, sppic_f^g)$ is the generating system of $(SSP, \bullet_f)$. Moreover, as in the case of *extended colored pictures*, the *description language* of a *symbolic pixel picture* q is a regular language.

5. The merging-independency problem for symbolic picture description languages

As we have shown in the previous sections, merging functions play a crucial role in the interpretation of a string description of symbolic pictures. Nevertheless, in this section we will characterize a subclass of string descriptions whose interpretation is not affected by the specific merging functions which are adopted. Then, we say that a word w is *merging-independent* whenever it always describes the same picture independently from the applied merging functions.

The decidability of the *merging-independency* for symbolic picture grammars is an interesting problem. Indeed, in Section 6 some decidability and complexity properties of symbolic picture languages will be established as an extension of analogous properties for drawn picture languages, and the proofs exploit the hypothesis of merging-independency for symbolic picture grammars. Thus, in the next subsections we address the decidability of the merging independency for symbolic picture grammars and in particular we show that it is always possible to decide whether or not a context-free grammar generates only merging-independent descriptions for symbolic pictures.

In Section 5.1 we will give the proof of decidability of the merging independency problem for *drawn symbolic picture description grammars*, then in Sections 5.2 we will extend such a proof to the models of *extended colored pictures* and *symbolic pixel pictures*.

5.1. The decidability of the merging-independency for drawn symbolic picture description grammars

In the following we provide the formal definition of merging independent string description for *drawn symbolic pictures*.

Definition 13. Let Φ_{Σ_ϕ} be the family of merging functions on Σ_ϕ and w be a $\Sigma \cup \Pi$-word. w is a *merging-independent drawn symbolic picture description* whenever the following condition holds:

$$dspic_f^g(w) = dspic_h^k(w), \quad \forall\, f, g, h, k \in \Phi_{\Sigma_\phi}.$$

A $\Sigma \cup \Pi$-language L is a *merging-independent drawn symbolic picture description language* if $\forall w \in L$, w is a *merging-independent drawn symbolic picture description*.

Taking into account the definition of merging functions, it is easy to verify that a $\Sigma \cup \Pi$-word w is *merging-independent* if and only if it specifies at most one symbol in Σ for each point of the described picture. Otherwise we say that w presents a "conflict" or w is *merging-dependent*. As an example, the $\Sigma \cup \Pi$-word $w = a\mathbf{r}b\mathbf{r}c\mathbf{l}$ is *merging-independent*. In contrast, the $\Sigma \cup \Pi$-words $w' = a\mathbf{r}b\mathbf{r}c\mathbf{l}e$ and $w'' = \mathbf{r}aa\mathbf{r}c\mathbf{l}e$ are *merging-dependent*

because two symbols (b and e in w', and two occurrences of a and one of e in w'') are specified for the same position.

The concept of *merging-independency* can be extended to picture grammars in a natural way.

Definition 14. A $\Sigma \cup \Pi$-grammar $G_{\Sigma\cup\Pi}$ is a *merging-independent drawn symbolic picture description grammar* if any $\Sigma \cup \Pi$-word $w \in L(G_{\Sigma\cup\Pi})$ is a *merging-independent drawn symbolic picture description*.

A grammar $\mathbf{G} = \langle G_{\Sigma\cup\Pi}, dspic_f^g \rangle$ is a *merging-independent drawn symbolic picture grammar* if $G_{\Sigma\cup\Pi}$ is a *merging-independent drawn symbolic picture description grammar*.

Now, we introduce some preliminary notions, which will be exploited to prove that it is decidable to establish whether or not a given context-free *drawn symbolic picture description grammar* is *merging-independent*.

Let $\mathrm{G} = \langle \Gamma, \Delta, N, P, S \rangle$ be a $\Gamma \cup \Delta$-grammar and $A \subseteq (\Gamma \cup \Delta)$ a subset of the terminal symbols. The grammar $\mathrm{G}_A = (A, N, P_A, S)$ is defined as the projection of the grammar G on the set A, where P_A is the set of productions obtained from the productions in P by removing all the terminals not in A. Moreover, if L is the language generated by G, L_A denotes the language generated by the grammar G_A, i.e. the set of words obtained by eliminating from the words of L the terminal symbols not in A. As an example if $\mathrm{L} = \{a\mathbf{r}b\mathbf{d}g, c\mathbf{luu}g\mathbf{dd}\}$ then $\mathrm{L}_\Pi = \{\mathbf{rd}, \mathbf{luudd}\}$. Let us observe that if L is a context-free language then also L_Π is a context-free language. Indeed, the projection defined above is a homomorphism $h : (\Gamma \cup \Delta)^* \to A^*$ and the context-free languages are closed under homomorphism [9].

Given $\Sigma = \{a_i \mid 1 \leqslant i \leqslant n\}$, if L is a context-free language and if with each word w in L we associate the n-tuple whose ith coordinate, $1 \leqslant i \leqslant n$, is the number of occurrences of a_i in w, then the resulting set of n-tuples, i.e., the set of Parikh vectors of L, is semilinear. In particular, given the function $\Psi : \Sigma^* \to \mathbb{N}^n$ defined by $\Psi(z) = (\#_{a_1}(z), \ldots, \#_{a_n}(z))$, it has been proved that $\Psi(M)$ is semilinear for each context-free language M (see Theorem 5.2.1 in [6]).

In analogy to the concept of shift for a drawn picture as defined in [16], the notion of shift of a $\Sigma \cup \Pi$-word can be introduced. Let w be a $\Sigma \cup \Pi$-word and $\langle \langle p, s, e \rangle, \Sigma, \delta \rangle$ be the corresponding picture. The shift of w is defined by $\mathit{shift}(w) = (x(e) - x(s), y(e) - y(s))$, where for a point $v = (m, n)$, $x(v) = m$ and $y(v) = n$. As an example, given the $\Sigma \cup \Pi$-word $w = \mathbf{r}b\mathbf{r}c\mathbf{l}a$, the *drawn symbolic picture* resulting from $dspic_f^g(w)$ has start point (0,0) and end point (1,0), so $shift(w) = (1, 0)$. In contrast, the $\Sigma \cup \Pi$-word "$\mathbf{r}b\mathbf{r}c\mathbf{l}a\mathbf{l}c$" has shift (0,0). In other words, a $\Sigma \cup \Pi$-word w has shift (0,0) if $\#_{\mathbf{r}}(w) = \#_{\mathbf{l}}(w)$ and $\#_{\mathbf{u}}(w) = \#_{\mathbf{d}}(w)$.

Now, let us observe that word $w = \mathbf{r}b\mathbf{r}c\mathbf{l}a$ is a *merging dependent drawn symbolic picture description* since the application of $dspic_f^g$ to w produces a conflict between symbols b and a that are associated to the same point (1,0). Such a conflict can also be detected by considering the subsentence $w_1 = b\mathbf{r}c\mathbf{l}a$ (which starts from symbol b and ends to symbol a) and by observing that $\mathit{shift}(w_1) = (0, 0)$. So, the *merging-independency* problem for a *drawn symbolic picture description grammar* G can be reduced to verifying whether or not G generates a subsentence $w = a_i \alpha a_j$ with $\alpha \in (\Pi \cup \Sigma)^*, a_i, a_j \in \Sigma$ such that $shift(w) = (0, 0)$.

Thus, we can state the following theorem.

Theorem 2. *Given a context-free* drawn symbolic picture description grammar $G_{\Sigma\cup\Pi}$, *it is decidable whether or not* $G_{\Sigma\cup\Pi}$ *is* merging-independent.

Proof. We start by observing that $G_{\Sigma\cup\Pi}$ is *merging-independent* if any $\Sigma\cup\Pi$-word $w\in \mathrm{L}(G_{\Sigma\cup\Pi})$ is *merging-independent*. In order to prove that $\mathrm{L}(G_{\Sigma\cup\Pi})$ is merging independent we consider the sets $L'=\mathit{suff}(\mathit{pref}(\mathrm{L}(G_{\Sigma\cup\Pi})))$ and $L''=\{w\in L' \mid w=axb$ with $a,b\in\Sigma$ and $x\in(\Sigma\cup\Pi)^*\}$. It can be easily proved that since $\mathrm{L}(G_{\Sigma\cup\Pi})$ is context-free also L', L'' and $L_{\Pi}{}''$ are context-free. Now, let P_1 denote the Parikh vectors of the words in $L_{\Pi}{}''$, i.e. $P_1=\Psi(L_{\Pi}{}'')$. Moreover, let K be the set of all $\Sigma\cup\Pi$-words w such that $\mathit{shift}(w)=(0,0)$ and P_2 be the set of Parikh vectors of K, i.e. $P_2=\Psi(K)$. Then $\mathrm{L}(G_{\Sigma\cup\Pi})$ is *merging-independent* if and only if $P_1\cap P_2=\emptyset$. Since $\Psi(X)$ is semilinear for each context-free language X and since P_2 is semilinear even though K is not context-free it is well known that the intersection problem is decidable for semilinear languages (see Theorems 5.2.1 and 5.6.1 in [6]), then it is decidable whether or not $P_1\cap P_2=\emptyset$. Hence the theorem holds. □

It is worth noting that the notion of *merging-independent* description for *drawn symbolic picture* is strongly related to the concept of self-avoiding introduced in [19]. In particular, a picture word is *self-avoiding* if all its nonempty factors are not loop. A word represents a loop if its shift is (0,0). In other words we can say that a picture description is self-avoiding if all its nonempty subpictures have shift different from (0,0).

Let us consider the *merging-dependent* $\Sigma\cup\Pi$-word $w=\mathbf{r}b\mathbf{r}c\mathbf{l}e$ in which two symbols b and e are described for the position (1,0) in the corresponding *drawn symbolic picture*. The Π-word $w'=\mathbf{rrl}$, obtained from w by not considering the symbols in Σ, is not self-avoiding. So, we can easily adapt the proof of Theorem 2 to the case of self-avoiding, by considering $L''=suff(pref(\mathrm{L}(G_{\Sigma\cup\Pi})))$. In other words, we take all the $\Sigma\cup\Pi$-words, not only those that traverse a position more than once by assigning it only one symbol. Thus, given a context-free $\Sigma\cup\Pi$-grammar $G_{\Sigma\cup\Pi}$ generating a string description language $\mathrm{L}(G_{\Sigma\cup\Pi})$, it is decidable whether or not $\mathrm{L}(G_{\Sigma\cup\Pi})$ contains a non self-avoiding string description. As an immediate consequence of Theorem 2, we can state the following Corollary.

Corollary 1. *Given a context-free* Π*-grammar* G *that generates a string description language* $L(G)$, *it is decidable whether or not* $L(G)$ *contains a non-self-avoiding string description.*

The problem to decide whether or not a Π-language L contains a self-avoiding Π-word is more difficult. As a matter of fact Robilliard and Simplot have proved that, given a rational language L over Π, it is undecidable to know whether or not L contains a self-avoiding word [19].

5.2. The decidability of the merging-independency for extended colored picture description grammars and symbolic pixel picture description grammars

In this subsection, we extend the decidability result of Theorem 2 to the *extended colored picture description grammars* and to the *symbolic pixel picture description grammars*.

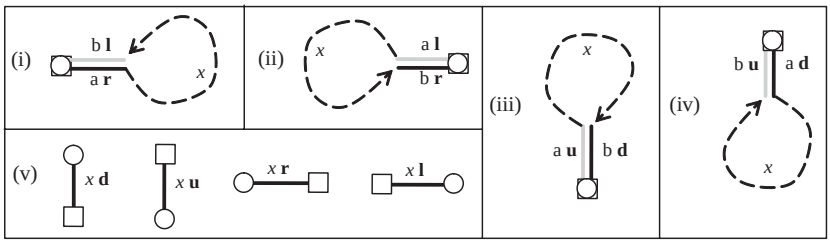


Fig. 26. The possible conflicts for *extended colored pictures*.

First, we analyze the model of the *extended colored pictures*. A $\Sigma \cup \Pi$-word w is *merging-independent* whenever the following condition holds:

$$ecpic_f^g(w) = ecpic_h^k(w), \quad \forall\, f, g, h, k \in \Phi_{\Sigma_\phi}.$$

It can be easily verified that a $\Sigma \cup \Pi$-word w is *merging-independent* if and only if it specifies at most one symbol in Σ for each segment of the described *extended colored picture*.

In other words, a $\Sigma \cup \Pi$-word is *merging-dependent* if it contains one of the following conflicting (sub)sentences:

(i) "$a\mathbf{r}xb\mathbf{l}$" with $x \in (\Sigma \cup \Pi)^*$ and $shift(x) = (0, 0)$ and $a, b \in \Sigma$;
(ii) "$a\mathbf{l}xb\mathbf{r}$" with $x \in (\Sigma \cup \Pi)^*$ and $shift(x) = (0, 0)$ and $a, b \in \Sigma$;
(iii) "$a\mathbf{u}xb\mathbf{d}$" with $x \in (\Sigma \cup \Pi)^*$ and $shift(x) = (0, 0)$ and $a, b \in \Sigma$;
(iv) "$a\mathbf{d}xb\mathbf{u}$" with $x \in (\Sigma \cup \Pi)^*$ and $shift(x) = (0, 0)$ and $a, b \in \Sigma$;
(v) "$\alpha\pi$" with $\alpha \in \Sigma^*$, $\pi \in \Pi$ and $|\alpha| > 1$

which are visualized in Fig. 26. Cases (i)–(iv) are concerned with segment overlapping and (v) is concerned with consecutive symbols that precede a move. As a consequence, the merging-independency problem for an *extended colored picture description grammar* G can be reduced to verifying whether or not G generates one of the above (sub)sentences. To this aim, we can exploit the proof of Theorem 2, by considering the following set L'', where the set $L' = suff(pref(L(G)))$.

$$L'' = \{x \in (\Sigma \cup \Pi)^* \mid (\text{"}a\mathbf{r}xb\mathbf{l}\text{"} \in L' \text{ or "}a\mathbf{l}xb\mathbf{r}\text{"} \in L' \text{ or "}a\mathbf{u}xb\mathbf{d}\text{"} \in L' \text{ or "}a\mathbf{d}xb\mathbf{u}\text{"} \in L') \text{ and } a, b \in \Sigma\} \cup \{\alpha \in \Sigma^* \mid \text{"}\alpha\pi\text{"} \in L',\ \pi \in \Pi,\ |\alpha| > 1\}.$$

Thus, the following theorem can be stated.

Theorem 3. *Given a context-free* extended colored picture description grammar G, *it is decidable whether or not* G *is* merging-independent.

Now, let us consider the model of *symbolic pixel pictures*. A $\Sigma \cup \Pi$-word w is *merging-independent* whenever the following condition holds:

$$sppic_f^g(w) = sppic_h^k(w), \quad \forall\, f, g, h, k \in \Phi_{\Sigma_\phi}.$$

It can be easily verified that a $\Sigma \cup \Pi$-word w is *merging-independent* if and only if it specifies at most one symbol in Σ for each pixel of the described *symbolic pixel picture*.

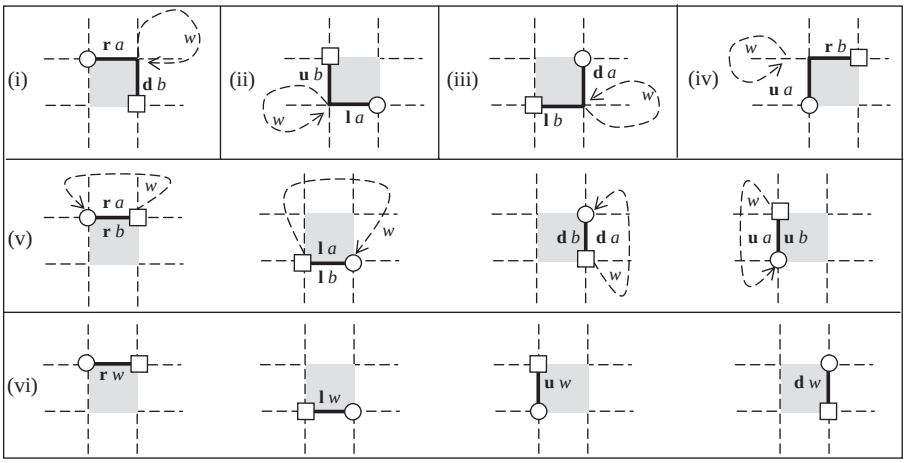


Fig. 27. The possible conflicts for *symbolic pixel pictures*.

In other words, a $\Sigma \cup \Pi$-word is *merging-dependent* if it contains one of the following conflicting (sub)sentences:

(i) "$\mathbf{r}ax\mathbf{d}b$" with $x \in (\Sigma \cup \Pi)^*$ and $shift(x) = (0, 0)$ and $a, b \in \Sigma$;
(ii) "$\mathbf{l}ax\mathbf{u}b$" with $x \in (\Sigma \cup \Pi)^*$ and $shift(x) = (0, 0)$ and $a, b \in \Sigma$;
(iii) "$\mathbf{d}ax\mathbf{l}b$" with $x \in (\Sigma \cup \Pi)^*$ and $shift(x) = (0, 0)$ and $a, b \in \Sigma$;
(iv) "$\mathbf{u}ax\mathbf{r}b$" with $x \in (\Sigma \cup \Pi)^*$ and $shift(x) = (0, 0)$ and $a, b \in \Sigma$;
(v) "$\boldsymbol{\pi}ax\boldsymbol{\pi}b$" with $x \in (\Sigma \cup \Pi)^*$ and $shift(\boldsymbol{\pi}ax) = (0, 0)$ and $a, b \in \Sigma$ and $\pi \in \Pi$;
(vi) "$\pi\alpha$" with $\alpha \in \Sigma^*$, $\pi \in \Pi$ and $|\alpha| > 1$

which are depicted in Fig. 27. Cases (i)–(v) describe picture descriptions specifying two symbols for a pixel, and (vi) represents picture descriptions containing consecutive symbols that follow a move. As a consequence, in order to decide the merging-independency problem for *symbolic pixel picture description grammars* we can exploit the proof of Theorem 2 by considering the following language L'', where the set $L' = suff(pref(L(G_{\Sigma\cup\Pi})))$.

$$L'' = \{x \in (\Sigma \cup \Pi)^* \mid (\text{"}\mathbf{r}ax\mathbf{d}b\text{"} \in L' \text{ or "}\mathbf{l}ax\mathbf{u}b\text{"} \in L' \text{ or "}\mathbf{d}ax\mathbf{l}b\text{"} \in L' \text{ or "}\mathbf{u}ax\mathbf{r}b\text{"} \in L') \text{ and } a, b \in \Sigma\} \cup \{\pi ax \mid \text{"}\pi ax\pi b\text{"} \in L',\ x \in (\Sigma \cup \Pi)^*, a, b \in \Sigma,\ \pi \in \Pi\} \cup \{\alpha \in \Sigma^* \mid \text{"}\pi\alpha\text{"} \in L',\ \pi \in \Pi,\ |\alpha| > 1\}.$$

Finally, the following result can be provided.

Theorem 4. *Given a context-free* symbolic pixel picture description grammar $G_{\Sigma\cup\Pi}$, *it is decidable whether or not* $G_{\Sigma\cup\Pi}$ *is* merging-independent.

6. Complexity and decidability problems of symbolic picture languages

In the last decades, picture languages have been deeply analyzed and the intractability of several important decision problems has been proved [2,3,5,7,8,10–14,16,19,22].

In particular, numerous results have been provided for drawn picture languages. Indeed, the *membership problem* has been proved to be undecidable for context-sensitive picture languages [16] and NP-Complete for regular picture languages [22] and for context-free picture languages [11], while the *equivalence*, the *containment* and the *ambiguity problems* are undecidable for regular picture languages [11,13]. Moreover, the *intersection emptiness problem* is not partially decidable for regular picture languages. The considerations about the intractability of the above problems motivated the investigation of subclasses of drawn pictures with nice properties from the decidability and complexity point of view [11,12,22]. Some of these subclasses are the *stripe*, the *three-way* and the *k-retreat-bounded picture languages*.

Some complexity and decidability results have also been provided for colored non-connected picture languages. In particular, in [5,7,8] the results established for drawn pictures have been extended to the colored picture model except for the NP-completeness of context-free languages. Indeed, it has been proved that the membership problem for linear context-free languages is also undecidable [7].

In contrast, to the best of our knowledge, results concerning these decidability and complexity problems have not been provided in the literature for pixel picture languages.

In this section we analyze the symbolic picture models introduced in the paper with respect to the above decidability and complexity problems.

It is worth noting that the problems, which were intractable for a class of picture languages, turn out to be obviously intractable also for the corresponding symbolic class. On the other hand, the nice properties (from the decidability and complexity point of view) determined for restricted subclasses of picture languages cannot always be inherited by the analogous subclasses of symbolic picture languages. Thus, in the following we determine the conditions, in terms of properties of both string descriptions and merging functions, which allow us to preserve such properties in the setting of the introduced picture models. Moreover, we prove that such properties also hold for symbolic pixel picture languages (and so for the original pixel picture model).

In Section 6.1 we consider the class of *symbolic stripe picture languages*, in Section 6.2 the class of *symbolic three-way picture languages*, and in Section 6.3 the class of *symbolic k-retreat-bounded picture languages*.

6.1. Stripe picture languages

In this section we analyze decidability problems for regular symbolic stripe picture languages. Let us recall that in the setting of drawn pictures, it has been proved that it is decidable whether or not a context-free picture language is a stripe picture language. Moreover, the membership problem for regular drawn stripe picture languages is decidable in linear time, and it is decidable whether two regular drawn stripe picture languages coincide or whether they have a nonempty intersection [22]. Thus, in the following we first investigate the conditions ensuring that such decidability and complexity results can also be provided for *extended colored pictures*. Then, we show how such conditions also work in the case of *drawn symbolic pictures* and *symbolic pixel pictures*. Two kinds of conditions have been identified, one concerns with properties of merging functions and the other is related to picture descriptions.

An *extended colored stripe picture language* is an *extended colored picture language* whose pictures fit into a stripe defined by two parallel lines. In particular, let $y = kx + d_1$, and $y = kx + d_2$ be the equations of two parallel lines, where k, d_1, d_2, are real numbers such that $d_1 < d_2$. The (k, d_1, d_2)-stripe consists of the integer grid points which are between these lines and is defined by

$$\mathrm{M}_0^{(k,d_1,d_2)} = \{(i, j) \in M_0 \mid ki + d_1 \leqslant j \leqslant ki + d_2\}.$$

The special case of the vertical stripe (i.e., $k = \infty$) can be defined by

$$\mathrm{M}_0^{(\infty,d_1,d_2)} = \{(i, j) \in M_0 \mid d_1 \leqslant j \leqslant d_2\}.$$

In the sequel we only consider non vertical stripes. Nevertheless, the provided results can be easily extended to the case of vertical stripes.

We are ready to give the formal definition of *extended colored stripe picture language*.

Definition 15. An *extended colored picture* $q = \langle\langle p, s, e\rangle, \Sigma, \delta\rangle$ is an *extended colored* (k, d_1, d_2)*-stripe picture* if $W(\langle p, s, e\rangle) \subseteq M_0^{(k,d_1,d_2)}$. An *extended colored picture language* L is an *extended colored* (k, d_1, d_2)*-stripe picture language* if every *extended colored picture* in L is an *extended colored* (k, d_1, d_2)*-stripe picture*.

Let us consider a (k, d_1, d_2)-stripe with $k = n/m$, such that n and m have no common divisor and $m > 0$ (the case with $k = 0$, can be similarly analyzed), the (m, n, d_1, d_2)*-unit point field*, denoted by F_0, is defined as follows:

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

It is worth noting that $M_0^{(k,d_1,d_2)} = \bigcup_{i\in\mathbb{Z}} t_{im,in}(F_0)$.[2] Thus, any point in the stripe can be mapped in a corresponding point in the set F_0, and the set F_0 can be used to represent any position of an *extended colored* (k, d_1, d_2)*-stripe picture*. As an example, let us consider the $(3/2, -3, 2)$-stripe shown in Fig. 28. The shaded region represents the $(2, 3, -3, 2)$-unit point field F_0. Given the point $(3/2, 1)$ belonging to F_0, we have that $(7/2, 4) = t_{2,3}((3/2, 1))$, and $(-1/2, -2) = t_{-2,-3}((3/2, 1))$.

The next theorem states that we can decide whether a context-free *extended colored picture grammar* generates an *extended colored stripe picture language* or not. The proof can be obtained by a simple extension of Theorem 5.3 in [22].

Theorem 5. *Let* **G** *be a* context-free extended colored picture grammar. *It is decidable whether or not* $\mathbf{L}(\mathbf{G})$ *is an* extended colored stripe picture language *or not*.

Now, we prove that the membership problem can be decided deterministically in linear time for some subclasses of regular *extended colored stripe picture languages*. First, we will analyze the class of languages generated by *extended colored picture grammars* $\mathbf{G} = \langle G_{ECPD}, ecpic_f^g\rangle$ where f is commutative and associative. Then, we extend the result to the

[2] Let us remember that $t_{m,n}(v) = (i + m, j + n)$, where $v = (i, j)$, and $t_{m,n}(\mathrm{A}) = \{\{t_{m,n}(v),\ t_{m,n}(v')\} \mid \{v, v'\} \in \mathrm{A}\}$.

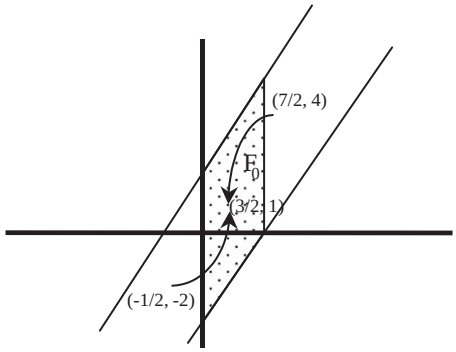


Fig. 28. The set F_0 is representative of any point in the (3/2, −3, 2)-stripe.

class of *merging independent* languages. The proof is based on the following Lemma 1 which establishes a correspondence between regular string languages and the above regular picture languages.

Lemma 1. *Let k be a rational number and* d_1 *and* d_2 *real numbers* ($d_1 \leqslant 0 \leqslant d_2$). *There exists an alphabet* Γ *and an encoding* μ *from the set of* extended colored (k, d_1, d_2)-stripe pictures *into* Γ^* *with the following properties*:

(1) *For two* extended colored (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 an* extended colored (k, d_1, d_2)-stripe picture q, *we can compute* $\mu(q)$ *in linear time*.
(3) *If* D *is a regular* extended colored (k, d_1, d_2)-stripe picture *language generated by an* extended colored picture grammar $\mathbf{G} = \langle G_{ECPD}, ecpic_f^g \rangle$ [3] where f is commutative and associative merging function then

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

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

In the following we informally justify the correspondence stated by Lemma 1 whose complete proof is given in Appendix A.

A stripe can be divided into vertical stripes of equal width (named *sliced portions*) so that any picture is divided into a sequence of subpictures. Then, the set of such possible different subpictures forms the alphabet of the string language, which is finite since in any stripe picture language only a finite number of different subpictures can occur. Thus, any subpicture in a sliced portion can be encoded by a subset of $LF_1 = \{[\{v, v'\}, a] \mid a \in \Sigma, \{v, v'\} \in M_1, v \in F_0, v' \in (F_0 \cup t_{m,n}(F_0))\}$. As an example, the string $\mu(q) = \sigma_{-1}\sigma_0\sigma_1$ in Fig. 29(b) represents the encoding of the *extended colored* (1/3, −1, 2)*-stripe picture q*

[3] $ECPD = (\Sigma(\Sigma \cup \Pi)^*\Pi \cup \{\varepsilon\})$ will denote the set of picture descriptions and a picture description grammar generating words in *ECPD* will be denoted by G_{ECPD}.

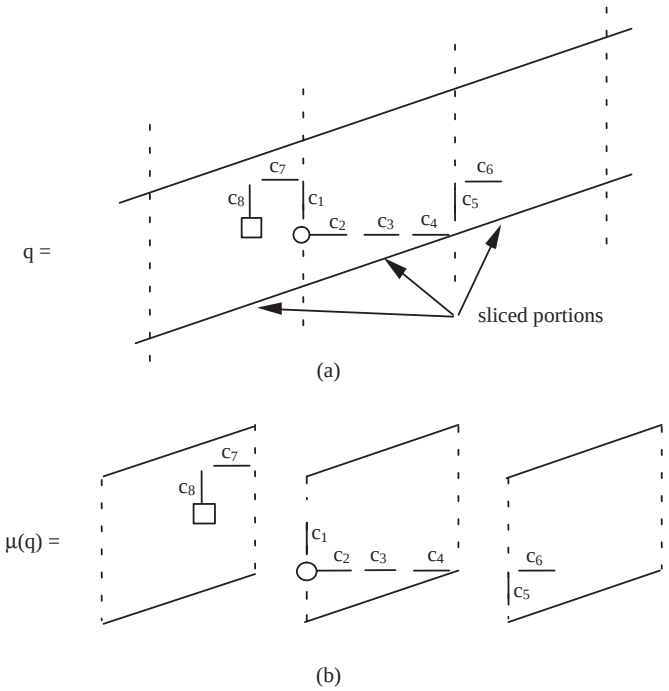


Fig. 29. (a) An *extended colored* (1/3, −1, 2)-*stripe picture* q, and (b) its corresponding sequence μ(q).

depicted in Fig. 29(a), where

$F_0 = \{(i, j) \mid (i, j) \in M_0,\ 0 \leqslant i < 3\}$,

$\sigma_0 = \{[\{(0, 0), (0, 1)\}, c_1], [\{(0, 0), (1, 0)\}, c_2], [\{(1, 0), (2, 0)\}, c_3], [\{(2, 0), (3, 0)\}, c_4], ¢\}$,

$\sigma_{-1} = \{[\{(3, 2), (2, 2)\}, c_7], [\{(2, 2), (2, 1)\}, c_8], \$, (2, 1)\}$,

$\sigma_1 = \{[\{(0, -1), (0, 0)\}, c_5], [\{(0, 0), (1, 0)\}, c_6]\}$,

and ¢ and \$ denote the presence of the start point and the end point in the sliced portion, respectively.

In order to prove the correspondence between regular string languages and regular *extended colored stripe picture languages* generated by grammar $\mathbf{G} = \langle G_{ECPD}, ecpic_f^g \rangle$ where *f* is commutative and associative, we have constructed a two-way finite state generator (*2FSG*) *H* which generates the regular string language *lang*(*H*). Then, we show that an image of *lang(H)* through a suitable homomorphism is equivalent to μ(D).

A *2FSG* is a device having a writing head with a finite control and a two-way infinite working tape. Every cell of the tape contains an initially empty set of symbols from a given alphabet. At each step it adds a symbol to the set under the writing head, moves in either direction and changes state. In [4] it has been proved that the language generated by a *2FSG* is a regular language.

The constructed *2FSG H* is characterized by states of this form: $\langle$A, v_1, v_2, *a*, *i*$\rangle$ which specifies that a segment $\{v_1, v_2\}$ is drawn in the cell under the writing head and symbol *a*

is assigned to the segment. In this way, the cells of the working tape of H correspond to the fields σ_i of the encoded *extended colored stripe picture* except for the case where two different symbols are associated to the same segment of the picture. In order to overcome such a problem, we have introduced a homomorphism h_1 to be applied to $lang(H)$. h_1 substitutes in the fields σ_i of the encoded *extended colored stripe picture* the symbols associated to the same segment with the symbol obtained by applying merging function f on such symbols. The language L obtained is equivalent to μ(D) where D is a regular language. The correspondence follows from the regularity of L since $lang(H)$ is regular and the regular languages are closed under homomorphism [9].

In order to provide an intuitive justification of the need for the associative and commutative properties of the merging functions f to ensure the equivalence between L and μ(D), let us observe that the $2FSG$ is not able to keep track of the symbol insertion order during the construction of the slices. Moreover, if a segment is traversed more than once the homomorphism exploits the merging function f to establish the actual symbol to associate to such segment. As a consequence, if f is not associative or commutative then different symbols could be associated depending on the application order of f. Obviously, this could determine erroneous results because we have already lost such ordering. On the other hand, to keep track of the symbol insertion order we should need a device enhanced with memory (and hence we would have different complexity results).

Thus, the following membership result can be provided.

Theorem 6. *The membership problem for a regular* extended colored stripe picture language *generated by an* extended colored picture grammar $\mathbf{G} = \langle \mathrm{G}_{ECPD}, ecpic_f^g \rangle$ *with f commutative and associative can be decided deterministically in linear time*.

Examples of commutative and associative merging functions are *addRGB* and *ord* presented in Section 3. The conditions on merging functions required in the previous theorem can be linked to the results provided in Section 3. As a matter of fact, the associative property of merging functions has turned out to be crucial also to ensure that the set $(ECP, \bullet_f)$ is a monoid. Thus, we can restate Theorem 6 as follows:

"Let (Σ_ϕ, f) be an abelian monoid. The membership problem for a regular *extended colored stripe picture language* generated by an *extended colored picture grammar* $\mathbf{G} = \langle \mathrm{G}_{ECPD}, ecpic_f^g \rangle$ can be decided deterministically in linear time".

Now, the previous membership result can be extended to the case of merging-independent grammars. Let us observe that if $\mathbf{G}$ is *merging-independent* then each segment in σ_i has associated at most one symbol in Σ and/or a certain number of occurrences of ϕ. As a consequence, merging functions always produce a unique symbol when applied by homomorphism h_1 independently from its application order. Thus, the associative and commutative hypothesis are no longer needed in order to ensure that L is equivalent to μ(D). As an immediate consequence, Lemma 1 can be easily extended to *merging-independent extended colored* (k, d_1, d_2)*-stripe picture languages*, and the following complexity result can be stated.

Corollary 2. *The membership problem for a regular* extended colored stripe picture language *generated by a* merging-independent extended colored picture grammar $\mathbf{G} = \langle \mathrm{G}_{ECPD}, ecpic_f^g \rangle$ *can be decided deterministically in linear time*.

Moreover, the above membership results can be exploited to prove other decidability results. As a matter of fact, the following theorem provides the conditions ensuring the decidability of the equivalence and intersection emptiness problems for regular stripe picture languages.

Theorem 7. *Given two regular languages of words in ECPD, L_1 and L_2, and merging functions f, g, h and k on Σ_ϕ such that $ecpic_f^g(L_1)$ is an* extended colored stripe picture language. *It is decidable whether $ecpic_f^g(L_1) = ecpic_h^k(L_2)$, and $ecpic_f^g(L_1) \cap ecpic_h^k(L_2) = \emptyset$ in the following cases*:
(1) *f and h are* commutative *and* associative, *or*
(2) *L_1 and L_2 are* merging-independent.

Proof. It follows from Theorem 5, Lemma 1, Corollary 2 and the properties of regular string languages. □

The above decidability and complexity results can also be provided for *regular symbolic pixel stripe picture languages* and *regular drawn symbolic stripe picture languages*. To this aim, Lemma 1 can be restated by considering a *symbolic pixel picture grammar* $\mathbf{G} = \langle G_{SPPD}, sppic_f^g \rangle$, and a *drawn symbolic picture grammar* $\mathbf{G} = \langle G_{SPPD}, dspic_f^g \rangle$ respectively, where $SPPD = (\Pi(\Sigma \cup \Pi)^* \cup \{\varepsilon\})$ denotes the set of string descriptions and G_{SPPD} denotes the grammar generating words in *SPPD*. The details of the two extensions are provided in the Appendix.

Thus, the following theorems can be given.

Theorem 8. *The membership problem for a regular* symbolic pixel stripe picture language *generated by a* symbolic pixel picture grammar $\mathbf{G} = \langle G_{SPPD}, sppic_f^g \rangle$ *with f commutative and associative can be decided deterministically in linear time.*

Theorem 9. *Given two regular languages of words in SPPD, L_1 and L_2, and merging functions f, g, h and k on Σ_ϕ such that $sppic_f^g(L_1)$ is a* symbolic pixel stripe picture language. *It is decidable whether $sppic_f^g(L_1) = sppic_h^k(L_2)$, and $sppic_f^g(L_1) \cap sppic_h^k(L_2) = \emptyset$ in the following cases*:
(1) *f and h are* commutative *and* associative, *or*
(2) *L_1 and L_2 are* merging-independent.

Theorem 10. *The membership problem for a regular* drawn symbolic stripe picture language *generated by a* drawn symbolic picture grammar $\mathbf{G} = \langle G_{SPPD}, dspic_f^g \rangle$ *with f commutative and associative can be decided deterministically in linear time.*

Theorem 11. *Given two regular languages of words in SPPD, L_1 and L_2, and merging functions f, g, h and k on Σ_ϕ such that $dspic_f^g(L_1)$ is a* drawn symbolic stripe picture language. *It is decidable whether $dspic_f^g(L_1) = dspic_h^k(L_2)$, and $dspic_f^g(L_1) \cap dspic_h^k(L_2) = \emptyset$ in the following cases*:
(1) *f and h are* commutative *and* associative, *or*
(2) *L_1 and L_2 are* merging-independent.

6.2. Three-way picture languages

In the setting of drawn pictures, it has been proved that the membership problem for context-free three-way picture languages is decidable in polynomial time [11]. However, other important problems are undecidable for three-way drawn pictures. In particular, the equivalence and containment problems are undecidable for a regular language L_1 and a linear language L_2, where both L_1 and L_2 describe three-way stripe picture languages. The intersection emptiness problem is undecidable for two three-way stripe linear picture languages. The intersection emptiness problem is also undecidable for a regular language L_1 and a linear language L_2, where L_1 describes a three-way stripe picture language and L_2 describes a stripe picture language [11].

It is obvious that the intractability of the previous problems also holds in the case of the symbolic picture languages introduced in this paper. Thus, in the sequel we provide the conditions ensuring that the nice membership decidability result given in [11] for three-way drawn pictures can be extended to the three-way versions of our picture models. In particular, we prove the decidability of the membership for *three-way extended colored picture languages* and we show how the proof can be easily extended to the languages of *three-way symbolic pixel pictures* and the languages of *three-way drawn symbolic pictures*.

Let Π' be a three-letter subset of Π. An *extended colored picture* is called a *three-way extended colored picture* if it can be described by a word in the set $ECPD' = \Sigma(\Sigma \cup \Pi')^*\Pi' \cup \{\varepsilon\}$. An *extended colored picture language* is called a *three-way extended colored picture language* if it can be described by words in $ECPD'$. As usual, $G_{ECPD'}$ will denote an *extended colored picture description grammar* generating words in $ECPD'$.

We start by considering *extended colored picture grammars* $\mathbf{G} = \langle G_{ECPD'}, ecpic_f^g \rangle$ such that f is commutative and associative, and proving that the membership problem is solvable deterministically in polynomial time by exploiting the approach provided in [11]. Then, we show the same result for *merging-independent extended colored picture grammars*.

Let us consider a push down automaton (PDA) $M = (Q, (\Sigma \cup \Pi' \cup \{\varepsilon\}), \Gamma, \gamma, q_0, Z_0, q_f)$ and picture $q = \langle\langle p, s, e\rangle, \Sigma, \delta^p\rangle$, where Q is a finite set of states of the finite control, $\Sigma \cup \Pi' \cup \{\varepsilon\}$ is a finite set of input symbols, Γ is a finite set of pushdown store symbols not including Z_0, γ is a transition function that maps elements of the set $Q \times (\Sigma^*\Pi' \cup \{\varepsilon\}) \times (\Gamma \cup Z_0)$ into finite subsets of $Q \times \{0, 1\} \times ((\Gamma \cup \{pop\}))$, q_0 is the initial state, Z_0 is the bottom marker for the pushdown store, and q_f is the final state. We want to determine whether or not $q \in ecpic_f^g(M)$.

A PDA works as described in the following. The value given by a transition function γ is a triple (q_i, X, d) where q_i is a state in Q, X is in {0,1}, and d in Γ. This means that the possible next actions of PDA are (1) entering next state q_i, (2) moving the input head to the right one cell or leaving it stationary, according to whether X is 1 or 0, and (3) adding the symbol d to the top of the pushdown store. Moreover if (q_i, X, pop) is the value of the transition function then in step (3) M removes the top pushdown store symbol instead of adding a symbol to the pushdown store. We can assume that M accepts by entering the final state q_f with the empty pushdown store, scanning the right endmarker of the input tape. A configuration of M is a 6-tuple $(q_s, w, p', v, v', \delta^{p'})$ which means that the PDA M is in

state q_s, the current content of the pushdown store is w (where the first symbol of w is the symbol at the top of the pushdown store), and M has read some portion of the input tape, say x where $x \in ECPD'$, such that $ecpic_f^g(x) = \langle\langle p', v, v'\rangle, \Sigma, \delta^{p'}\rangle$, for each segment p_s in $p'\ \delta^{p'}(p_s) \in \Sigma_\phi$.

Given two configurations I and J, I $\vdash$ J if and only if the PDA M in configuration I can move in one step to configuration J. Moreover, I $\vdash^k$ J establishes that M can move from configuration I to configuration J after exactly k moves. $\vdash^*$ denotes the transitive reflexive closure of the binary relation $\vdash$.

Now, a 7-*tuple* $(q_s, A, q_r, p', v, v', \delta^{p'})$ is *q-valid* if the PDA M can go from state q_s with symbol A alone on the pushdown store to state q_r with an empty store, drawing the *extended colored picture* $\langle\langle p', v, v'\rangle, \Sigma, \delta^{p'}\rangle \subseteq q$, starting from $v \in M_0$ and ending at $v' \in M_0$, i.e., if $(q_s, A, \emptyset, v, v, \delta)\vdash^*(q_r, \emptyset, p', v, v', \delta^{p'})$. Moreover, let $W(\langle p, s, e\rangle) = \{v_1(= s), v_2, \ldots, v_m(= e)\}$ and X be a matrix such that, for each i and j with $1 \leqslant i,\ j \leqslant m$, $X[i,\ j] = \{(q_s, A, q_r, p', v_i, v_j, \delta^{p'}) \mid q_s, q_r \in Q, A \in (\Gamma \cup \{Z_0\})$ and $(q_s, A, q_r, p', v_i, v_j, \delta^{p'})$ is a *q-valid* 7-*tuple*$\}$.

From the definition of *q-valid* 7-*tuples*, $q = \langle\langle p, s, e\rangle, \Sigma, \delta\rangle \in ecpic_f^g$ if and only if $X[1, M]$ contains the element $(q_0, Z_0, q_f, p, s, e, \delta)$. As a consequence, in order to test the membership of *three-way extended colored pictures* we need only to construct the matrix X. To this aim we can use a dynamic programming method defined by the following function $\text{YIELD}_M : 2^C \times 2^C \to 2^C$, where C is the set of all 7-*tuples*.

$\text{YIELD}_M(S_1, S_2) = \{(q_s, B, q_t, p', v, v_3, \delta^{p'}) \mid (q_s, B, \emptyset, v, v, \delta)\vdash(q_r, AB, p_1, v, v_1, \delta^{p1}),\ (q_r, A, q_z, p_2, v_1, v_2, \delta^{p2}) \in S_1, (q_z, B, q_t, p_3, v_2, v_3, \delta^{p3}) \in S_2, \langle\langle p', v, v_3\rangle, \Sigma, \delta^{p'}\rangle = \langle\langle p_1, v, v_1\rangle, \Sigma, \delta^{p1}\rangle \bullet_f \langle\langle p_2, v_1, v_2\rangle, \Sigma, \delta^{p2}\rangle \bullet_f \langle\langle p_3, v_2, v_3\rangle, \Sigma, \delta^{p3}\rangle\}$.

The idea behind the definition of function YIELD_M is that, from the partial computations of M corresponding to the drawing of the *extended colored picture* $\langle\langle p_1, v, v_1\rangle, \Sigma, \delta^{p1}\rangle$, $\langle\langle p_2, v_1, v_2\rangle, \Sigma, \delta^{p2}\rangle$ and $\langle\langle p_3, v_2, v_3\rangle, \Sigma, \delta^{p3}\rangle$, we deduce $(q_s, B, \emptyset, v, v, \delta)\vdash^+(q_t, \emptyset, p', v, v_3, \delta^{p'})$.

Thus, given the *extended colored picture* $q = \langle\langle p, s, e\rangle, \Sigma, \delta^p\rangle$, the dynamic programming method works as described in the following:

(1) puts in the table X all *q-valid* 7-*tuples* corresponding to single pop transitions of M,
(2) by using function YIELD_M, combines repeatedly the information in the table X that corresponds to partial computations of M until no new information can be added to X, and
(3) checks whether or not the element $(q_0, Z_0, q_f, p, s, e, \delta^p)$ has been added to $X[1, m]$.

It is worth noting that the associative and commutative properties of merging functions play a crucial role in the construction of the matrix X. As a matter of fact, the above technique is not able to keep track of the symbol insertion order in the subpictures corresponding to the partial computations. Moreover, if a segment is traversed more than once, YIELD_M exploits the merging function f to establish the actual symbol to associate to it. In particular, in the application of YIELD_M, for each p_s in p' if p_s is in p_1, p_2 and p_3, the symbol associated to p_s is given by $\delta^{p'}(p_s) = f(f(\delta^{p1}(p_s), \delta^{p2}(p_s)), \delta^{p3}(p_s))$.

Thus, if the merging function f is not associative or commutative then different symbols could be associated to the segments depending on the application order of f. Obviously, this

could determine erroneous results during the combination of the partial computations in the construction of X because we have already lost such ordering.

In order to decide the membership problem for the *three-way extended colored picture languages*, we can use Algorithm 3.5 provided by Kim in [11] for three-way drawn picture languages. Indeed, such an algorithm can be easily adapted to work according to the above definitions of matrix X and function YIELD_M. Moreover, we can easily prove that the algorithm works in polynomial time. Due to the presence of symbols associated to the segments we have to add a factor of $|\Sigma|$ to the time required for each operation in the steps of the algorithm [11]. But given a description grammar $G_{ECPD'}$, $|\Sigma|$ is a constant, so the total time required does not change asymptotically with respect to the result provided for drawn pictures.

As a consequence we have the following theorem.

Theorem 12. *Let (Σ_ϕ, f) be an abelian monoid. The membership problem for* context-free three-way extended colored picture languages *generated by* extended colored picture grammars $\mathbf{G} = \langle G_{ECPD'}, ecpic_f^g \rangle$ *can be decided deterministically in polynomial time*.

Let us observe that if $\mathbf{G}$ is *merging-independent* then each segment in the extended colored pictures has associated at most one symbol in Σ and/or a certain number of occurrences of ϕ. Thus the associative and commutative hypothesis of merging function f are no longer needed for the function YIELD_M in order to determine $(q_r, A, q_z, p_2, v_1, v_2, \delta^{p2}) \in S_1$, $(q_z, B, q_t, p_3, v_2, v_3, \delta^{p3}) \in S_2$, such that $(q_s, B, \emptyset, v, v, \delta) \vdash (q_r, AB, p_1, v, v_1, \delta^{p1})$ and $\langle\langle p', v, v_3\rangle, \Sigma, \delta^{p'}\rangle = \langle\langle p_1, v, v_1\rangle, \Sigma, \delta^{p1}\rangle \bullet_f \langle\langle p_2, v_1, v_2\rangle, \Sigma, \delta^{p2}\rangle \bullet_f \langle\langle p_3, v_2, v_3\rangle, \Sigma, \delta^{p3}\rangle$. As a matter of fact, for each p_s in p', if p_s is in p_1, p_2 and p_3, functions δ^{p1}, δ^{p2} and δ^{p3} associate to p_s the same symbol in Σ or the symbol ϕ.

As an immediate consequence, Theorem 12 can be easily extended to *merging-independent extended colored picture languages*, and the following complexity result can be stated.

Corollary 3. *The membership problem for* context-free three-way extended colored picture languages *generated by* merging-independent extended colored picture grammars $\mathbf{G} = \langle G_{ECPD'}, ecpic_f^g \rangle$ *can be decided deterministically in polynomial time*.

Now, we show that the membership result given in Theorem 12 for *extended colored pictures* can also be provided for *drawn symbolic pictures* and *symbolic pixel pictures* by slightly changing the proof presented above.

In particular, for *drawn symbolic picture languages* we consider a grammar $\mathbf{G} = \langle G_{SPPD'}, dspic_f^g \rangle$ and a PDA $M = (Q, (\Sigma \cup \Pi \cup \{\varepsilon\}), \Gamma, \gamma, q_0, Z_0, q_f)$, where γ is a transition function that maps elements of the set $Q \times (\Pi'\Sigma^* \cup \{\varepsilon\}) \times (\Gamma \cup Z_0)$ into finite subsets of $Q \times \{0, 1\} \times ((\Gamma \cup \{pop\}))$, so that the membership problem is reduced to determining whether or not a *drawn symbolic picture* $q = \langle\langle p, s, e\rangle, \Sigma, \delta^p\rangle$ is in $dspic_f^g(M)$. To this aim, we can use again a matrix X which is indexed by points of $\text{W}(\langle p, s, e\rangle) = \{v_1(= s), v_2, \ldots, v_m(= e)\}$. The entries of X are constructed by determining the *q-valid tuples* from configuration of M. Here, a 7-*tuple* $(q_s, A, q_r, p', v, v', \delta^{p'})$ is *q-valid* if the PDA M can go from state q_s with symbol A alone on the pushdown store to state q_r with

an empty store, drawing the *drawn symbolic picture* $\langle\langle p', v, v'\rangle, \Sigma, \delta^{p'}\rangle \subseteq q$, starting from $v \in M_0$ and ending at $v' \in M_0$. In other words, we can use the function YIELD_M defined above, which is able to combine the information in the table X that corresponds to partial computations of M. When no new information can be added to X, we check whether or not the element $(q_0, Z_0, q_f, p, s, e, \delta^p)$ has been added to $X[1, m]$.

For *symbolic pixel picture languages* we consider a grammar $\mathbf{G} = \langle G_{SPPD'}, sppic_f^g\rangle$ and a PDA $M = (Q, (\Sigma \cup \Pi' \cup \{\varepsilon\}), \Gamma, \gamma, q_0, Z_0, q_f)$, where γ is a transition function that maps elements of the set $Q \times (\Pi'\Sigma^* \cup \{\varepsilon\}) \times (\Gamma \cup Z_0)$ into finite subsets of $Q \times \{0, 1\} \times ((\Gamma \cup \{pop\}$. Now, in order to determine whether or not a *symbolic pixel picture* $q = \langle\langle p, s, e\rangle, \Sigma, \delta^p\rangle$ is in $sppic_f^g(M)$, by using function YIELD_M, we can construct a matrix X which is now indexed by points of the armor $arm\langle p, s, e\rangle) = \{v_1(= s), v_2, \ldots, v_m(= e)\}$. The entries of X are obtained by analyzing the *q-valid tuples*, where a 7-*tuple* $(q_s, A, q_r, p', v, v', \delta^{p'})$ is *q-valid* if the PDA M can go from state q_s with symbol A alone on the pushdown store to state q_r with an empty store, drawing the *symbolic pixel picture* $\langle\langle p', v, v'\rangle, \Sigma, \delta^{p'}\rangle \subseteq q$, starting from $v \in M_0$ and ending at $v' \in M_0$. When matrix X has been constructed, we check whether or not $X[1, m]$ contains the element $(q_0, Z_0, q_f, p, s, e, \delta^p)$.

Thus, the following theorems can be provided.

Theorem 13. *Let (Σ_ϕ, f) be an abelian monoid. The membership problem for* context-free three-way drawn symbolic picture languages *generated by* drawn symbolic picture grammars $\mathbf{G} = \langle G_{SPPD'}, dspic_f^g\rangle$ *can be decided deterministically in polynomial time.*

Theorem 14. *Let (Σ_ϕ, f) be an abelian monoid. The membership problem for* context-free three-way symbolic pixel picture languages *generated by* symbolic pixel picture grammars $\mathbf{G} = \langle G_{SPPD'}, sppic_f^g\rangle$ *can be decided deterministically in polynomial time.*

6.3. K-retreat bounded picture languages

Other interesting subclasses of drawn pictures have been introduced to solve the intractability of major decision problems. In particular, given an integer $k \geqslant 0$ and a $\Sigma \cup \Pi$-word x, x is *k-retreat-bounded* if it draws a symbolic picture in such a manner that the maximum distance of **l** moves, ignoring **u** and **d** moves, from a rightmost point of any partial picture is bounded by k [12].

Let us observe that this subclass is an extension of three-way picture languages. By extending the result given in [12] for Π-grammars, it can be easily proved that it is decidable whether or not a context-free $\Sigma \cup \Pi$-language L is a *k-retreat-bounded language*. Moreover, in [12] it has been shown that there is a 1-retreat-bounded regular drawn picture language (1-retreat-bounded vertical-stripe linear drawn picture language) for which the membership problem is NP-complete. The inclusion and intersection-emptiness problems are undecidable for a 1-retreat-bounded regular drawn picture language and a 2-retreat-bounded regular drawn picture language. The equivalence and ambiguity problems are undecidable for 2-retreat-bounded regular drawn picture languages. It is obvious that the intractability of previous problems also holds in the case of symbolic picture languages introduced in this paper.

Fortunately, *k*-retreat-bounded picture languages have also nice complexity properties. As a matter of fact, it has been proved that the membership problem is decidable in polynomial time for each k-retreat-bounded nonvertical-stripe context-free drawn picture language [12]. Such a result can be extended to our picture models by exploiting the same characterizations (associativity and commutativity of merging functions, and merging-independency of string descriptions) which have allowed us to generalize the results concerning three-way languages. In particular, we will first provide the result for *k-retreat-bounded extended colored picture languages* and then we will show how to extend the proof to the languages of *k-retreat-bounded drawn symbolic pictures* and the languages of *k-retreat-bounded symbolic pixel pictures*.

Let μ, d_1, d_2 be real numbers such that $\mu \neq \infty$, $d_1 < d_2$ and $M = (Q, (\Sigma \cup \Pi \cup \{\varepsilon\}), \Gamma, \gamma, q_0, Z_0, q_f)$ be a *k-retreat-bounded* PDA, such that $ecpic_f^g(M)$ is a (μ, d_1, d_2)-*nonvertical stripe extended colored picture language*. The transition function is defined as $\gamma : Q \times (\Sigma^* \Pi \cup \{\varepsilon\}) \times \Gamma \to 2^{Q \times \Gamma}$, and we shall assume that M either pushes or pops one symbol in each move and it accepts by entering the unique final state q_f with the empty stack, after scanning the input word. Let us observe that it is decidable whether or not M is a *k-retreat-bounded* PDA (by extending Theorem 3.1 in [12]) and whether or not $ecpic_f^g(M)$ is a (μ, d_1, d_2)-*nonvertical stripe extended colored picture language* (Theorem 5).

Let $q = \langle\langle p, (0,0), e\rangle, \Sigma, \delta^p\rangle$ be an *extended colored picture* given as input and $M_1^{(\mu,d_1,d_2)} = \{\{v, v'\} \in M_1 \mid v, v' \in M_0^{(\mu,d_1,d_2)}\}$. For $v, v' \in W(\langle p, (0,0), e\rangle)$ let $U(v, v') = M_1^{(\mu,d_1,d_2)} \cap M_1^{(\infty, x(v)-k, x(v')+k)}$ and define: $J(v, v') = U(v, v') - U(v, v) - U(v', v')$.

An *extended colored picture* $q' = \langle\langle p', s', e'\rangle, \Sigma, \delta^{p'}\rangle$ is *q-valid* if $p' \subseteq p$, $\delta^{p'}(p_s) = \delta^p(p_s)$ for each segment p_s in p' and $p' \cap J(s', e') = p \cap J(s', e')$. (If $q = ecpic_f^g(x)$ for some $x \in L(M)$ and $x = yzw$, where $ecpic_f^g(y) = \langle\langle p_y, (0,0), s'\rangle, \Sigma, \delta^{p_y}\rangle$ and $ecpic_f^g(z) = langle\langle p', s', e'\rangle, \Sigma, \delta^{p'}\rangle$, then $\langle\langle p', s', e'\rangle, \Sigma, \delta^{p'}\rangle$ should be a *q-valid* picture since q is a (μ, d_1, d_2)-stripe picture and M is a *k-retreat bounded* PDA.) For $q_s, q_t \in Q$ and $A \in \Gamma$, (q', q_r, A, q_t) is a *q-valid tuple* if q' is a *q-valid extended colored picture* and M can go from state q_r with A alone in its stack to state q_t, emptying the stack and consuming a word x from the input tape with $ecpic_f^g(x) = q'$.

Let X be the set of q-valid tuples. X is a finite set and it is easy to see that $q \in ecpic_f^g(M)$ if and only if $(q, q_0, Z_0, q_f) \in X$. As a consequence, in order to test if $q \in ecpic_f^g(M)$ we need only to construct X. We can use a dynamic programming method defined by the following function Y_M such that, for all $\tau \in X$ (with $\tau = (\langle\langle p', s', e'\rangle, \Sigma, \delta^{p'}\rangle, q_r, A, q_t)$) and $E \in 2^X$, $Y_M(\tau, E)$ consists of all *q-valid* tuples $\underline{\tau}$ defined as:

(1) $\underline{\tau} = (\langle\langle\{v, s'\} \cup p' \cup p'', v, v'\rangle, \Sigma, \delta^p\rangle, q_{r'}, B, q_{t'})$ if $(q_r, AB) \in \gamma(q_{r'}, \sigma, B)$, $pic_{T,\{f,g\}}(\sigma) = \langle\langle\{v, s'\}, v, s'\rangle, \Sigma, \delta'\rangle$, $(\langle\langle p'', e', v'\rangle, \Sigma, \delta^{p''}\rangle, q_t, B, q_{t'}) \in E$ and $\langle\langle\{v, s'\}, v, s'\rangle, \Sigma, \delta'\rangle \bullet_f \langle\langle p', s', e'\rangle, \Sigma, \delta^{p'}\rangle \bullet_f \langle\langle p'', e', v'\rangle, \Sigma, \delta^{p''}\rangle = \langle\langle\{v, s'\} \cup p' \cup p'', v, v'\rangle, \Sigma, \delta^p\rangle$; and

(2) $\underline{\tau} = (\langle\langle\{v, v'\} \cup p'' \cup p', v, e'\rangle, \Sigma, \delta^p\rangle, q_{r'}, A, q_t)$ if $(q_{t'}, BA) \in \gamma(q_{r'}, \sigma, A)$, $pic_{T,\{f,g\}}(\sigma) = \langle\langle\{v, v'\}, v, v'\rangle, \Sigma, \delta'\rangle$, $(\langle\langle p'', v', s'\rangle, \Sigma, \delta^{p''}\rangle, q_{t'}, B, q_r) \in E$ and $\langle\langle\{v, v'\}, v, v'\rangle, \Sigma, \delta'\rangle \bullet_f \langle\langle p'', v', s'\rangle, \Sigma, \delta^{p''}\rangle \bullet_f \langle\langle p', s', e'\rangle, \Sigma, \delta^{p'}\rangle = \langle\langle\{v, s'\} \cup p'' \cup p', v, e\rangle, \Sigma, \delta^p\rangle$.

It is worth noting that the hypothesis of associativity and commutativity of merging function f is exploited in order to associate a unique symbol to each segment in each subpicture obtained from partial computations and to glue consecutive valid computations, as it is the case for three-way extended colored picture languages.

The algorithm for the membership test can be easily obtained by adapting the algorithm provided by Kim in [12] for k-retreat-bounded drawn pictures, which is similar to the one for three-way context-free drawn picture languages. Moreover, we can easily prove that the algorithm works in polynomial time. Indeed, the presence of symbols requires that only a constant factor has to be added to the time required for each operation.

Now, we are ready to provide the following theorem.

Theorem 15. *Let (Σ_ϕ, f) be an abelian monoid and k be a non negative integer. The membership problem for k*-retreat-bounded nonvertical-stripe context-free extended colored picture languages *generated by* extended colored picture grammars $\mathbf{G} = \langle G_{ECPD}, ecpic_f^g \rangle$ *can be decided deterministically in polynomial time.*

Let us observe that if $\mathbf{G}$ is *merging-independent* then each segment in the extended colored pictures has associated at most one symbol in Σ and/or a certain number of occurrences of ϕ. Thus, the associative and commutative hypothesis of merging function f are no longer needed for the function Y_M in order to determine the valid tuples $\underline{\tau}$.

As an immediate consequence, Theorem 15 can be easily extended to *merging-independent* extended colored pictures languages, and the following complexity result can be stated.

Corollary 4. *Let k be a non-negative integer. The membership problem for k*-retreat-bounded nonvertical-stripe context-free extended colored picture languages *generated by* merging-independent extended colored picture grammars $\langle G_{ECPD}, ecpic_f^g \rangle$ *can be decided deterministically in polynomial time.*

Now, we show that the membership result given in Theorem 15 can be extended to *drawn symbolic picture languages* and *symbolic pixel picture languages* by slightly changing the proof presented above.

For *drawn symbolic picture languages* we consider a *drawn symbolic picture grammar* $\mathbf{G} = \langle G_{SPPD}, dspic_f^g \rangle$ and a PDA $M = (Q, (\Sigma \cup \Pi \cup \{\varepsilon\}), \Gamma, \gamma, q_0, Z_0, q_f)$, where the transition function is defined as $\gamma : Q \times (\Pi\Sigma^* \cup \{\varepsilon\}) \times \Gamma \to 2^{Q \times \Gamma}$. A *drawn symbolic picture* $q' = \langle \langle p', s', e' \rangle, \Sigma, \delta^{p'} \rangle$ is *q-valid* if $p' \subseteq p$, $\delta^{p'}(p_p) = \delta^p(p_p)$ for each point p_s in $\mathrm{W}(\langle p', s', e' \rangle)$, and $p' \cap J(s', e') = p \cap J(s', e')$. In order to determine whether or not a *drawn symbolic picture* $q = \langle \langle p, s, e \rangle, \Sigma, \delta^p \rangle$ is in $dspic_f^g(M)$, by using function Y_M we can construct the set of *q-valid tuples*. For $q_s, q_t \in Q$ and $A \in \Gamma$, (q', q_r, A, q_t) is a *q-valid tuple* if q' is a *q-valid drawn symbolic picture* and M can go from state q_r with A alone in its stack to state q_t, emptying the stack and consuming a word x from the input tape with $dspic_f^g(x) = q'$. When no new tuples can be added to X, we check whether or not X contains the element (q, q_0, Z_0, q_f).

Analogously, for *symbolic pixel pictures* we consider a *symbolic pixel picture grammar* $\mathbf{G} = \langle G_{SPPD}, sppic_f^g \rangle$ and a PDA $M = (Q, (\Sigma \cup \Pi \cup \{\varepsilon\}), \Gamma, \gamma, q_0, Z_0, q_f)$. In this case,

we define $J(v, v') = U(v, v') - U(v, v) - U(v', v')$, where $v, v' \in arm(\langle p, (0,0), e\rangle)$ and $U(v, v') = M_1^{(\mu,d_1,d_2)} \cap M_1^{(\infty,x(v)-k,x(v')+k)}$. Moreover, a *symbolic pixel picture* $q' = \langle\langle p', s', e'\rangle, \Sigma, \delta^{p'}\rangle$ is *q-valid* if $p' \subseteq p$, $\delta^{p'}(p_x) = \delta^p(p_x)$ for each pixel p_x in p', and $p' \cap J(s', e') = p \cap Js', e')$. So, in order to determine whether or not a *symbolic pixel picture* $q = \langle\langle p, s, e\rangle, \Sigma, \delta^p\rangle$ is in $sppic_f^g(M)$, we can construct the set of *q-valid tuples*. For $q_s, q_t \in Q$ and $A \in \Gamma$, (q', q_r, A, q_t) is a *q-valid tuple* if q' is a *q-valid symbolic pixel picture* and M can go from state q_r with A alone in its stack to state q_t, emptying the stack and consuming a word x from the input tape with $sppic_f^g(x) = q'$. Again the set of tuples X is obtained by using function Y_M, and we check whether or not $(q, q_0, Z_0, q_f) \in X$.

Thus, the following theorems can be provided.

Theorem 16. *Let (Σ_ϕ, f) be an abelian monoid and k be a non negative integer. The membership problem for k-*retreat-bounded nonvertical-stripe context-free drawn symbolic picture languages *generated by* drawn symbolic picture grammars $\mathbf{G} = \langle G_{SPPD}, dspic_f^g\rangle$ *can be decided deterministically in polynomial time*.

Theorem 17. *Let (Σ_ϕ, f) be an abelian monoid and k be a non negative integer. The membership problem for k-*retreat-bounded nonvertical-stripe context-free symbolic pixel picture languages *generated by* symbolic pixel picture grammars $\mathbf{G} = \langle G_{SPPD}, sppic_f^g\rangle$ *can be decided deterministically in polynomial time*.

7. Final remarks

In this paper, starting from the approach proposed by Hinz and Welzl [8] we have introduced some extensions of picture models, namely *extended colored pictures*, *drawn symbolic pictures* and *symbolic pixel pictures*. Such extensions are conceived to specify further information on the picture, such as colors, labels, icons, which is represented by symbols from an alphabet Σ and is associated to segments, points or pixels, respectively. A distinguishing characteristic of such extensions is concerned with the string descriptions of the pictures. Indeed, we have defined suitable mappings which are able to obtain from a string of symbols and moves the corresponding picture. These mappings are parametric with respect to merging functions that have a crucial role in establishing which symbol must be visualized whenever more than one symbol is associated to a segment, a point or a pixel. Moreover, we have identified the properties of merging functions ensuring that the set of *extended colored pictures*, the set of *drawn symbolic pictures* and the set of *symbolic pixel pictures* are monoids and finitely generated inverse monoids.

Several decidability and complexity properties have been determined for the introduced picture models. In particular, the notion of merging-independency has been provided for string descriptions, and the decidability of the merging-independency problem for context-free languages has been proved. Furthermore, we have investigated the conditions which allow us to extend the nice results proved for subclasses of drawn picture and colored picture languages in the setting of the analogous subclasses of symbolic picture languages. We have proved that such results also hold in the setting of symbolic pixel picture languages (and so for the original pixel picture model).

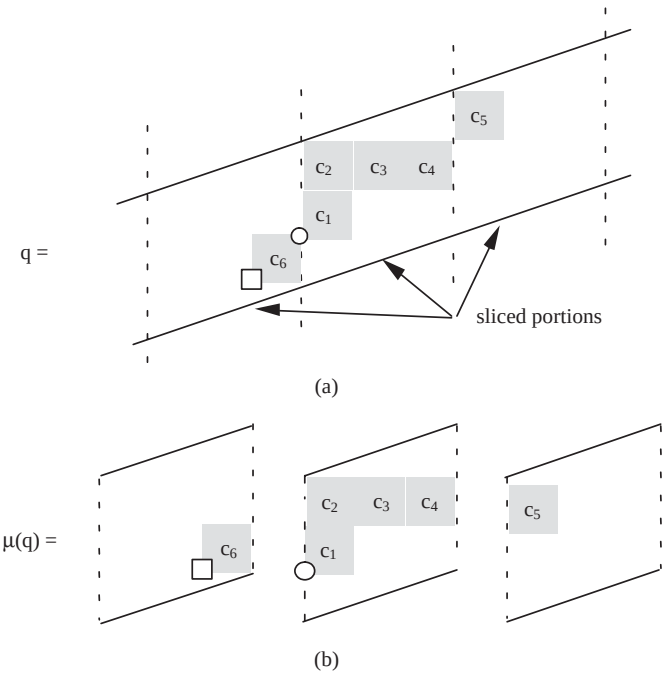


Fig. 30. (a) A *symbolic pixel* (1/3, −1, 2)-*stripe picture* q, and (b) its corresponding sequence μ(q).

As future work, we intend to analyze the application of the merging functions in the context of other picture models. Moreover, other subclasses of the proposed picture languages with nice decidability and complexity properties could be identified.

Appendix A.

In this appendix we first provide the proof of Lemma 1 introduced in Section 6.1 to establish a correspondence between regular string languages and a subclass of regular *extended colored stripe picture languages*, identified by providing properties (associativity and commutativity) of merging functions, is provided. Then, we show how such lemma can be extended to *regular symbolic pixel stripe picture languages* and *drawn symbolic stripe picture languages*.

The proof of Lemma 1 exploits a *two-way finite state generator* whose formal definition follows.

Definition A1 (*Culik and Welzl [4]*). A *two-way finite state generator* (*2FSG*, for short) is a tuple

$$H = (Q, \tau, d, A_0, A_f, \Delta, out),$$

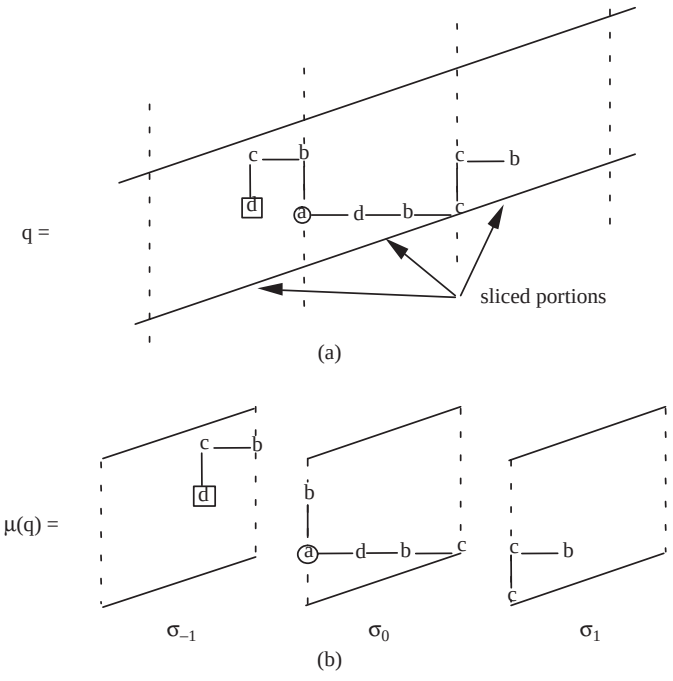


Fig. 31. (a) A drawn symbolic (1/3, −1, 2)-stripe picture; (b) its corresponding sequence μ(q).

where Q is a finite nonempty set of states, $\tau : Q \to 2^Q$ is a transition function, $d : Q \to 2^{\{-1,0,1\}}$ is a direction function, which indicates whether the writing head can move to the left (-1), to the right (1) or can remain in the same position, A_0 is the initial state, A_f is the accepting state, Δ is the output alphabet and, finally $out : Q^+ \to \Delta$ is an interpretation function. In particular, the interpretation function *out* can be defined by using the printing function $pr : Q \to \Lambda$, where Λ is the printing alphabet such that $\Delta = 2^\Lambda$.

In the sequel, we recall from [22] the concepts of *computations* and *generated language* related to two-way finite state generators. The computations of a *2FSG H* are described by strings over $(Q \times \mathbb{Z})$, where (A, j) means that "the current state is A and the current position on the tape is j". The transition function τ induces a move relation $\vdash$ associated to a *2FSG*. Let $c \in (Q \times \mathbb{Z})^*$, $(A, j) \in (Q \times \mathbb{Z})$. Then

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

if $A' \in \tau(A)$, $i \in d(A)$. The transitive closure of $\vdash$ is denoted by $\vdash^*$ and it allows us to define the set of valid computations of H, denoted by *comp(H)*, and defined as

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

For a prefix of a valid computation $c \in (Q \times \mathbb{Z})^*$, the leftmost (rightmost) position visited by c, denoted by $lm(c)$ ($rm(c)$, resp.), is defined as the minimal (maximal, resp.) j such that

c can be written as $c = c_1(A, j)c_2$, for some $c_1, c_2 \in (Q \times \mathbb{Z})^*$, $(A, j) \in (Q \times \mathbb{Z})$. Given a valid computation c, the *history-j-homomorphism* $h_j : (Q \times \mathbb{Z})^* \to Q^*$ is defined as

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

Such a mapping allows us to describe the sequence of states of the finite control, in which the writing head passed position j on the tape. Finally, the cell j will contain the element $out_j(c) = \{pr(A) \mid A \in alph(h_j(c))\}$, where $alph(h_j(c))$ denotes the set of symbols which occur in $h_j(c)$. Thus the word generated by a computation c is

$$word(c) = out_a(c)out_{a+1}(c) \dots out_{b-1}(c)out_b(c) \in \Lambda^*,$$

where $a = lm(c)$ and $b = rm(c)$. Finally, the language generated by H is defined as

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

As it has been proved in [4] the language generated by a *2FSG* is a regular language.

In order to prove statements (1) and (2) of Lemma 1, let $k = n/m$, such that n and m have no common divisor and $m > 0$ and consider the (m, n, d_1, d_2)-unit labeled line field:

$$LF_1 = \{[\{v, v'\}, a] \mid a \in \Sigma, \{v, v'\} \in M_1, v \in F_0, v' \in (F_0 \cup t_{m,n}(F_0))\}.$$

The alphabet Γ consists of the set of subsets of $F_0 \cup LF_1 \cup \{\not\in, \$\}$. Any subpicture in a sliced portion will be encoded by a subset of LF_1. Moreover, the presence of the endpoint within the sliced portion of the subpicture will be indicated by means of a \$. While $\not\in$ will denote the presence of the start point in the sliced portion. The position of the start point will be (0,0). More formally, let $q = \langle\langle r, (0, 0), e\rangle, \Sigma, \delta\rangle$ be *an extended colored* (k, d_1, d_2)*-stripe picture*. Moreover, let us define for all integers i,

$$\sigma_{-i} = \begin{cases} (\delta(r) \cap LF_1) \cup \{\not\in\} & \text{if } i = 0, e \notin F_0, \\ (\delta(r) \cap LF_1) \cup \{\not\in, \$, e\} & \text{if } i = 0, e \in F_0, \\ (t_{im,in}(\delta(r)) \cap LF_1) & \text{if } i \neq 0, t_{im,in}(e) \notin F_0, \\ (t_{im,in}(\delta(r)) \cap LF_1) \cup \{\$, t_{im,in}(e)\} & \text{if } i \neq 0, t_{im,in}(e) \in F_0 \end{cases}$$

where $\delta(r) = \{[\{v, v'\}, a] \mid a \in \Sigma,\ \{v, v'\} \in r,\ \delta(\{v, v'\}) = a\}$, and $t_{im,in}(\delta(r)) = \{[\{t_{im,in}(v), t_{im,in}(v')\}, a] \mid [\{v, v'\}, a] \in \delta(r)\}$.

Now, 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 $\mu(q) = \sigma_a\sigma_{a+1} \dots \sigma_0 \dots \sigma_{b-1}\sigma_b$ represents the encoding of the picture q. Thus, it is easy to verify that the first two statements of the lemma hold.

In order to prove statement (3), we construct a two-way finite state generator H which generates the regular string language $lang(H) = \mu(\mathrm{D})$.

Let $\mathbf{G} = \langle G_{ECPD}, ecpic_f^g\rangle$ be a regular *extended colored picture grammar* with f commutative and associative such that D$= ecpic_f^g(L(G_{ECPD}))$ is an *extended colored* (k, d_1, d_2)*-stripe picture language*. Without loss of generality, we can suppose that $G_{ECPD} = (\Sigma, \Pi, \mathrm{N}, \mathrm{P}, \mathrm{S})$ is in right linear form, i.e., G_{ECPD} has productions of the form S$\to a$A, S$\to\varepsilon$, A$\to\alpha\pi$B, A$\to\alpha\pi$, where the start symbol S does not occur on the right-hand side of the productions in P, A, B $\in$ N, $\alpha \in \Sigma^*$, $a \in \Sigma$ and $\pi \in \Pi$.

In the following, we show how to construct a $2FSG$ which simulates the drawing of a picture w in L(**G**). The cells of the working tape of the automaton will correspond to the fields σ_i of the encoded *extended colored stripe picture*. The $2FSG$ $H = (\mathrm{Q}, \tau, \mathrm{d}, \mathrm{A}_0, \mathrm{A}_f, \Lambda, out)$ is defined as follows:

(1) $\mathrm{Q} = \{\mathrm{A}_0, \mathrm{A}_f\}$
$\cup \{\langle \mathrm{A}, v_1, v_2, a, i\rangle \mid \mathrm{A} \in \mathrm{N}, [\{v_1, v_2\}, a] \in LF_1, i \in \{-1, 0, 1\}\}$
$\cup \{\langle \mathrm{S}, \notin, (0,0), \text{-}, i\rangle \mid \mathrm{S}$ is the start symbol of G, $i \in \{-1, 0, 1\}\}$
$\cup \{\langle \mathrm{S}, \notin, (0,0), a, i\rangle \mid a \in \Sigma, i \in \{-1, 0, 1\}\}$
$\cup \{\langle \varepsilon, v, \$, \text{-}, 0\rangle \mid v \in F_0 \cup \{(0,0)\}\}$
$\cup \{\langle \varepsilon, v_1, v_2, a, 0\rangle \mid [\{v_1, v_2\}, a] \in LF_1, a \in \Sigma\}$;

(2) $\tau(\mathrm{A}_0) = \{\langle \mathrm{S}, \notin, (0,0), \text{-}, i\rangle \mid i \in \{-1, 0, 1\}\}$;
$\tau(\mathrm{A}_f) = \emptyset$;
$\tau(\langle \varepsilon, v, \$, \text{-}, 0\rangle) = \{\mathrm{A}_f\}$ for $v \in F_0 \cup \{(0,0)\}$;
- $\langle \mathrm{A}, \notin, (0,0), a, i\rangle \in \tau(\langle \mathrm{S}, \notin, (0,0), \text{-}, i\rangle)$ if and only if $\mathrm{S} \to a\mathrm{A}$ is in P, where $a \in \Sigma$;
- $\langle \varepsilon, (0,0), \$, -, 0\rangle \in \tau(\langle \mathrm{S}, \notin, (0,0), \text{-}, i\rangle)$ if and only if $\mathrm{S} \to \varepsilon$ is in P;
- for $\langle \mathrm{B}, w_1, w_2, b, j\rangle, \langle \mathrm{A}, v_1, v_2, a, i\rangle \in \mathrm{Q}$ (v_1 possibly $\notin$)
$\langle \mathrm{B}, w_1, w_2, b, j\rangle \in \tau(\langle \mathrm{A}, v_1, v_2, a, i\rangle)$ if and only if $\mathrm{A} \to \alpha\pi\mathrm{B}$ is in P, where $\pi \in \Pi$, $\alpha \in \Sigma^*$, $w_1 = t_{-im,-in}(v_2)$, $w_2 = \pi(w_1)$, b is obtained from α by using merging function g and if $\alpha = \varepsilon$ then $b = a$;
- for $\langle \varepsilon, w_1, w_2, b, 0\rangle, \langle \varepsilon, w_2, \$, \text{-}, 0\rangle, \langle \mathrm{A}, v_1, v_2, a, i\rangle \in \mathrm{Q}$ (v_1 possibly $\notin$)
$\{\langle \varepsilon, w_1, w_2, b, 0\rangle, \langle \varepsilon, w_2, \$, \text{-}, 0\rangle\} \subseteq \tau(\langle \mathrm{A}, v_1, v_2, a, i\rangle)$
if and only if $\mathrm{A} \to \alpha\pi$ is in P, where $\pi \in \Pi$, $\alpha \in \Sigma^*$, $w_1 = \mathrm{t}_{-im,-in}(v_2)$, $w_2 = \pi(w_1)$, b is obtained from α by using merging function g and if $\alpha = \varepsilon$ then $b = a$;

(3) $\mathrm{d}(\mathrm{A}_0) = \mathrm{d}(\mathrm{A}_f) = \{0\}$;
$\mathrm{d}(\langle \mathrm{A}, v_1, v_2, a, i\rangle) = \{i\}$ with $\langle \mathrm{A}, v_1, v_2, a, i\rangle \in \mathrm{Q}$;

(4) $\Lambda = \mathrm{F}_0 \cup \mathrm{LF}_1 \cup \{\notin, \$, (0,0)\}$

(5) pr: $\mathrm{Q} \to \Lambda$, is such that:
$\mathrm{pr}(\mathrm{A}_0) = \notin$,
$\mathrm{pr}(\langle \mathrm{S}, \notin, (0,0), -, i\rangle) = \notin$,
$\mathrm{pr}(\langle \mathrm{A}, \notin, (0,0), a, i\rangle) = \notin$,
$\mathrm{pr}(\langle \varepsilon, v, \$, -, 0\rangle) = v$,
$\mathrm{pr}(\langle \varepsilon, v_1, v_2, a, 0\rangle) = [\{v_1, v_2\}, a]$, $v_1 \neq \notin$, $v_2 \neq \$$,
$\mathrm{pr}(\langle A, v_1, v_2, a, i\rangle) = [\{v_1, v_2\}, a] v_1 \neq \notin$, $v_2 \neq \$$,
$\mathrm{pr}(\mathrm{A}_f) = \$$.

Now, we give the interpretation of the states for a $2FSG$ which simulates the generation of a word w in $\mathrm{L}(G_{ECPD})$. The state $\langle \mathrm{S}, \notin, (0,0), \text{-}, i\rangle$ means that the generation starts, and (0,0) is the current position of the drawing head in the field, and i denotes the position of the next move (-1 for 'left', 0 for 'no move' and +1 for 'right'). $\langle \mathrm{A}, \notin, (0,0), a, i\rangle$ means that the generation is started and a will be the symbol associated to the next segments. A state $\langle \mathrm{A}, v_1, v_2, a, i\rangle$ means that: the derivation uses a nonterminal A, a line $\{v_1, v_2\}$ is drawn in the cell under the writing head and symbol a is assigned to the line. Finally, state $\langle \varepsilon, v, \$, \text{-}, 0\rangle$ means that the generation ended in position v in the cell under the writing head.

Let us observe that in general G_{ECPD} could describe a segment of the picture more than once possibly with different symbols. Thus, the cells of the working tape of H could not correspond to the fields σ_i of the encoded *extended colored stripe picture*. In order

to overcome such problem, we introduce the homomorphism h_1 that substitutes in the fields σ_i of the encoded *extended colored stripe picture* the symbols associated to the same segment with the symbol obtained by applying merging function f on such symbols. The homomorphism $h_1 : \Gamma \to \Gamma$ is defined as follows:

$$h_1(\sigma) = \begin{cases} \sigma & \text{if not } \exists\, [\{v_1, v_2\}, a_1], \ldots, [\{v_1, v_2\}, a_n] \in \sigma \text{ with } n > 1, \\ h_1(\sigma') & \text{otherwise,} \\ & \text{where } \sigma' = \rho \cup \{[\{v_1, v_2\},\ f(a_1, \ldots, a_n)],\ \text{with} \\ & \rho \cup \{[\{v_1, v_2\}, a_1], \ldots, [\{v_1, v_2\}, a_n]\} = \sigma. \end{cases}$$

Thus, $h_1(lang(H)) = \{h_1(word(c)) \mid c \in comp(H)\}$
$= \{h_1(out_a(c)out_{a+1}(c) \ldots out_{b-1}(c)out_b(c)) \mid c \in comp(H)\}$
$= \{h_1(out_a(c))h_1(out_{a+1}(c)) \ldots h_1(out_{b-1}(c))h_1(out_b(c)) \mid c \in comp(H)\}$.

Now, we provide an example of application of homomorphism h_1. Let us consider the $\langle G_{\Sigma\cup\Pi}, ecpic^g_f\rangle$ grammar where f is commutative and associative, and it is defined as: f(a,c)=c, f(e,c)=c, f(b,b)=b, f(b,e)=a, f(a,a)=a. Moreover, let H be the $2FSG$ constructed from the grammar $G_{\Sigma\cup\Pi}$ and $\rho_{-1}\rho_0\rho_1 \in lang(H)$ such that

$\rho_{-1} = \{[\{(3, 2), (2, 2)\}, a], [\{(2, 2), (2, 1)\}, c], \$, (2, 1)\}$
$\rho_0 = \{[\{(0, 0), (0, 1)\}, b], [\{(0, 0), (0, 1)\}, b], [\{(0, 0), (1, 0)\}, a],$
$[\{(1, 0), (2, 0)\}, b], [\{(2, 0), (3, 0)\}, e], [\{(0, 0), (1, 0)\}, a], [\{(1, 0), (2, 0)\}, e],$
$[\{(2, 0), (3, 0)\}, c], \not\in\}$
$\rho_1 = \{[\{(0, -1), (0, 0)\}, e], [\{(0, 0), (1, 0)\}, e], [\{(0, -1), (0, 0)\}, c], [\{(0, 0), (1, 0)\},$
$c]\}$.

By applying the homomorphism h_1 on ρ_{-1}, ρ_0, ρ_1 we obtain

$\rho'_{-1} = \{[\{(3, 2), (2, 2)\}, a], [\{(2, 2), (2, 1)\}, c], \$, (2, 1)\}$
$\rho'_0 = \{[\{(0, 0), (0, 1)\}, c], [\{(0, 0), (1, 0)\}, a], [\{(1, 0), (2, 0)\}, a], [\{(2, 0), (3, 0)\}, c],$
$\not\in\}$
$\rho'_1 = \{[\{(0, -1), (0, 0)\}, c], [\{(0, 0), (1, 0)\}, c]\}$.

Finally, from the associative and commutative properties of merging function f it follows that μ(D) corresponds to regular language L obtained by applying homomorphism h_1 to $lang(H)$ as described previously. The thesis of point (3) follows then from the regularity of L since $lang(H)$ is regular and the regular languages are closed under homomorphism [9].

A.1. Symbolic pixel stripe pictures languages

Lemma 1 can be extended for *symbolic pixel stripe picture languages* by considering a grammar $\mathbf{G} = \langle G_{SPPD}, sppic^g_f\rangle$, where G_{SPPD} is a string description grammar generating words in $SPPD = (\Pi(\Sigma \cup \Pi)^* \cup \{\varepsilon\})$. Moreover, a *symbolic pixel picture* $q = \langle\langle p, s, e\rangle, \Sigma, \delta\rangle$ is a *symbolic pixel* (k, d_1, d_2)*-stripe picture* if $arm(\langle p, s, e\rangle) \subseteq M_0^{(k,d_1,d_2)}$.

The proof of the lemma changes as described in the following.

Statements (1) and (2) of the lemma are proved by defining the (m, n, d_1, d_2)-unit labeled line field

$LF_1 = \{(pix(i, j), a) \mid a \in \Sigma, (i, j) \in F_0, (i+1, j+1) \in (F_0 \cup t_{m,n}(F_0))\}$ and for all integers i

$$\sigma_{-i} = \begin{cases} (\delta(r) \cap LF_1) \cup \{\not\epsilon\} & \text{if } i = 0, e \notin F_0, \\ (\delta(r) \cap LF_1) \cup \{\not\epsilon, \$, e\} & \text{if } i = 0, e \in F_0, \\ (t_{im,in}(\delta(r)) \cap LF_1) & \text{if } i \neq 0, t_{im,in}(e) \notin F_0, \\ (t_{im,in}(\delta(r)) \cap LF_1) \cup \{\$, t_{im,in}(e)\} & \text{if } i \neq 0, t_{im,in}(e) \in F_0, \end{cases}$$

where $\delta(r) = \{(pix(h, k), a) \mid a \in \Sigma,\ pix(h, k) \in r,\ \delta(pix(h, k)) = a\}$ and $t_{im,in}(\delta(r)) = \{[t_{im,in}(pix(h, k)), a] \mid (pix(h, k), a) \in \delta(r)\}$.

For example, given the *symbolic pixel* (1/3, −1, 2)*-stripe picture* depicted in Fig. 30(a), $F_0 = \{(i, j) \mid (i, j) \in M_0,\ 0 \leqslant i < 3\}$, and

$\sigma_0 = \{((pix(0, 0),\ c_1)), ((pix(0, 1),\ c_2)), ((pix(1, 1),\ c_3)), ((pix(2, 1),\ c_4)),\ \not\epsilon\}$
$\sigma_{-1} = \{((pix(2, 0),\ c_6)), \$, (2, 0)\}$
$\sigma_1 = \{((pix(0, 1),\ c_5))\}$

and the string $\mu(q) = \sigma_{-1}\sigma_0\sigma_1$ representing the encoding of the picture q is shown in Fig. 30(b).

Then, to prove statement (3), we define grammar $G_{SPPD} = (\Sigma,\ \Pi, \text{N}, \text{P}, \text{S})$ such that the regular *symbolic pixel* (k, d_1, d_2)*-stripe picture* language D is generated by a *symbolic pixel picture grammar* $\mathbf{G} = \langle G_{SPPD}, sppic_f^g \rangle$ where f is a commutative and associative merging function. The productions of G_{SPPD} have the form $\text{S} \to \pi\alpha\text{A}$, $\text{S} \to \pi\alpha$, $\text{S} \to \varepsilon$, $\text{A} \to \pi\alpha\text{B}$, $\text{A} \to \pi\alpha$, where the start symbol S does not occur on the right-hand side of the productions in P, A, B $\in$ N, $\alpha \in \Sigma^*$, and $\pi \in \Pi$. The $2FSG\ H = (\text{Q}, \tau, d, \text{A}_0,\ \text{A}_f, \Delta, out)$ is defined as follows:

(1) $\text{Q} = \{\text{A}_0,\ \text{A}_f\}$
$\cup\ \{\langle \text{A}, v_1, v_2, (pix(h, k), a), i\rangle \mid \text{A} \in \text{N}, (pix(h, k), a) \in LF_1, v_1, v_2 \in arm(pix(h, k)), i \in \{-1, 0, 1\}\}$
$\cup\ \{\langle \text{S}, \not\epsilon, (0, 0), \text{-}, i\rangle \mid \text{S is the start symbol of G}, i \in \{-1, 0, 1\}\}$
$\cup\ \{\langle \varepsilon, v, \$, \text{-}, 0\rangle \mid v \in F_0 \cup \{(0,0)\}\}$
$\cup\ \{\langle \varepsilon, v_1, v_2, (pix(h, k), a), 0\rangle \mid (pix(h, k), a) \in LF_1, a \in \Sigma, v_1, v_2 \in arm(pix(h, k))\}$;

(2) $\tau(\text{A}_0) = \{\langle \text{S}, \not\epsilon, (0,0), \text{-}, i\rangle \mid i \in \{-1, 0, 1\}\}$;
$\tau(\langle \varepsilon, v, \$, \text{-}, 0\rangle) = \text{A}_f$ for $v \in F_0 \cup \{(0, 0)\}$
$\tau(\text{A}_f) = \emptyset$;
- $\langle S, \pi(0, 0), \$, (pix(\pi(0, 0)), a), 0\rangle \in \tau(\langle \text{S}, \not\epsilon, (0,0), \text{-}, i\rangle)$ if and only if $\text{S} \to \pi\alpha$ is in P, where $\alpha \in \Sigma^*$, a is obtained from α by using merging function g and if $\alpha = \varepsilon$ then $a = \phi$;
- $\langle \varepsilon, (0,0), \$, -, 0\rangle \in \tau(\langle \text{S}, \not\epsilon, (0,0), \text{-}, i\rangle)$ if and only if $\text{S} \to \varepsilon$ is in P;
- for $\langle \text{B}, w_1, w_2, (pix(\pi(w_1)), a), j\rangle, \langle \text{A}, v_1, v_2, (pix(h, k), b), i\rangle \in \text{Q}$ (v_1 possibly $\not\epsilon$) $\langle \text{B}, w_1, w_2, (pix(\pi(w_1)), a), j\rangle \in \tau(\langle \text{A}, v_1, v_2, (pix(h, k), b), i\rangle)$ if and only if $\text{A} \to \pi\alpha\text{B}$ is in P, where $\pi \in \Pi$, $\alpha \in \Sigma^*$, $w_2 = \pi(w_1)$, $w_1 = t_{-im,-in}(v_2)$, a is obtained from α by using merging function g and if $\alpha = \varepsilon$ then $a = \phi$;
- for $\langle \varepsilon, w_1, w_2, (pix(\pi(w_1)), a), 0\rangle, \langle \varepsilon, w_2, \$, \text{-}, 0\rangle, \langle \text{A}, v_1, v_2, (pix(h, k), b), i\rangle \in \text{Q}$ (v_1 possibly $\not\epsilon$)
$\{\langle \varepsilon, w_1, w_2, (pix(\pi(w_1)), a), 0\rangle, \langle \varepsilon, w_2, \$, \text{-}, 0\rangle\} \subseteq \tau(\langle \text{A}, v_1, v_2, (pix(h, k), b), i\rangle)$
if and only if $\text{A} \to \pi\alpha$ is in P, where $\pi \in \Pi$, $\alpha \in \Sigma^*$, $w_2 = \pi(w_1)$, $w_1 = t_{-im,-in}(v_2)$, a is obtained from α by using merging function g and if $\alpha = \varepsilon$ then $a = \phi$;

(3) $d(A_0) = d(A_f) = \{0\}$;
$d(\langle A, v_1, v_2, (pix(h,k), a), i\rangle) = \{i\}$ with $\langle A, v_1, v_2, (pix(h,k), a), i\rangle \in Q$;
(4) $\Lambda = F_0 \cup LF_1 \cup \{\not\epsilon, \$, (0,0)\}$
(5) pr: $Q \to \Lambda$, is such that:
$pr(A_0) = \not\epsilon$,
$pr(\langle S, \not\epsilon, (0,0), -, i\rangle) = \not\epsilon$,
$pr(\langle \varepsilon, v, \$, -, 0\rangle) = v$,
$pr(\langle \varepsilon, v_1, v_2, (pix(h,k), a), 0\rangle) = (pix(h,k), a), v_1 \neq \not\epsilon, v_2 \neq \$$
$pr(\langle A, v_1, v_2, (pix(h,k), a), i\rangle) = (pix(h,k), a), v_1 \neq \not\epsilon, v_2 \neq \$$,
$pr(A_f) = \$$,

where the generic state $\langle A, v_1, v_2, (pix(h,k), a), i\rangle$ means that the derivation uses a non-terminal A, a pixel $pix(h,k)$, with the symbol a associated, is drawn in the cell under the writing head by following the line $\{v_1, v_2\}$, and i denotes the position of the next move (-1 for 'left', 0 for 'no move' and +1 for 'right').

Moreover, since G_{SPPD} could describe a pixel of the picture more than once possibly with different symbols, homomorphism $h_1 : \Gamma \to \Gamma$, where Γ denotes the encoding alphabet, is defined as follows:

$$h_1(\sigma) = \begin{cases} \sigma & \text{if not } \exists (pix(i,j), a_1), \ldots, (pix(i,j), a_n) \in \sigma \text{ with } n > 1, \\ h_1(\sigma') & \text{otherwise,} \\ & \text{where } \sigma' = \rho \cup \{(pix(i,j), f(a_1, \ldots a_n))\} \\ & \text{with} \rho \cup \{(pix(i,j), a_1), \ldots, (pix(i,j), a_n)\} = \sigma. \end{cases}$$

A.2. Drawn symbolic stripe pictures languages

Lemma 1 can be extended for *drawn symbolic stripe picture languages* by considering a grammar $\mathbf{G} = \langle G_{SPPD}, dspic_f^g \rangle$ and by changing the proof of the lemma as described in the following.

Statements (1) and (2) of the lemma can be proved by defining the (m, n, d_1, d_2)-unit labeled point field

$$LF_0 = \{[a, (i,j)] \mid a \in \Sigma, (i,j) \in F_0\}$$

and the (m, n, d_1, d_2)-unit labeled line field
$LF_1 = \{\{[a, v], [b, v']\} \mid a, b \in \Sigma_\phi, \{v, v'\} \in M_1, v \in F_0, v' \in (F_0 \cup t_{m,n}(F_0))\}$ and for all integers i

$$\sigma_{-i} = \begin{cases} (\delta(r) \cap LF_1) \cup \{\not\epsilon\} & \text{if } i = 0, e \notin F_0, \\ (\delta(r) \cap LF_1) \cup \{\not\epsilon, \$, [\delta(e), e]\} & \text{if } i = 0, e \in F_0, \\ (t_{im,in}(\delta(r)) \cap LF_1) & \text{if } i \neq 0, t_{im,in}(e) \notin F_0, \\ (t_{im,in}(\delta(r)) \cap LF_1) \cup \{\$, [\delta(e), t_{im,in}(e)]\} & \text{if } i \neq 0, t_{im,in}(e) \in F_0, \end{cases}$$

where $\delta(r) = \{\{[a, v], [b, v']\} \mid a, b \in \Sigma_\phi, \{v, v'\} \in r, \delta(v) = a, \delta(v') = b\}$ and $t_{im,in}(\delta(r)) = \{\{[a, t_{im,in}(v)], [b, t_{im,in}(v')]\} \mid \{[a, v], [b, v']\} \in \delta(r)\}$.

For example, given the drawn symbolic $(1/3, -1, 2)$-stripe picture depicted in Fig. 31(a), $F_0 = \{(i,j) \mid (i,j) \in M_0,\ 0 \leqslant i < 3\}$ and
$\sigma_0 = \{\{[a, (0,0)], [b, (0,1)]\}, \{[a, (0,0)], [d, (1,0)]\}, \{[d, (1,0)], [b, (2,0)]\}, \{[b, (2,0)],$

$[c, (3, 0)]\}, \not\in\}$

$$\sigma_{-1} = \{\{[b, (3, 2)], [c, (2, 2)]\}, \{[c, (2, 2)], [d, (2, 1)]\}, \$, [d, (2, 1)]\}$$
$$\sigma_1 = \{\{[c, (0, -1)], [c, (0, 0)]\}, \{[c, (0, 0)], [b, (1, 0)]\}\},$$

and the string $\mu(q) = \sigma_{-1}\sigma_0\sigma_1$ representing the encoding of the picture q is shown in Fig. 31(b).

Statement (3) of the lemma is proved by defining grammar $G_{SPPD} = (\Sigma, \Pi, \mathrm{N}, \mathrm{P}, \mathrm{S})$ such that the regular *drawn symbolic* (k, d_1, d_2)*-stripe picture* language D is generated by a *drawn symbolic picture grammar* $\mathbf{G} = \langle G_{SPPD}, dspic_f^g \rangle$ where f is a commutative and associative merging function. The productions of G_{SPPD} have the form $\mathrm{S}\to\pi\alpha\mathrm{A}$, $\mathrm{S}\to\pi\alpha$, $\mathrm{S}\to\varepsilon$, $\mathrm{A}\to\pi\alpha\mathrm{B}$, $\mathrm{A}\to\pi\alpha$, where the start symbol S does not occur on the right-hand side of the productions in P, A, B ∈ N, $\alpha \in \Sigma^*$, and $\pi \in \Pi$. The $2FSG\mathrm{H} = (\mathrm{Q}, \tau, \mathrm{d}, \mathrm{A}_0, \mathrm{A}_f, \Delta,$ *out*) is defined as follows:

(1) $\mathrm{Q} = \{\mathrm{A}_0, \mathrm{A}_f\}$
$\cup\ \{\langle \mathrm{A}, v_1, v_2, i\rangle \mid \mathrm{A}\in\mathrm{N}, \{v_1, v_2\}\in LF_1, i \in \{-1, 0, 1\}\}$
$\cup\ \{\langle \mathrm{S}, \not\in, (0, 0), i\rangle \mid \mathrm{S}$ is the start symbol of G, $i \in \{-1, 0, 1\}\}$
$\cup\ \{\langle \varepsilon, v, \$, 0\rangle \mid v \in LF_0\cup\{(0,0)\}\}$
$\cup\ \{\langle \varepsilon, v_1, v_2, 0\rangle \mid \{v_1, v_2\} \in LF_1\}$;

(2) $\tau(\mathrm{A}_0) = \{\langle \mathrm{S}, \not\in, (0,0), i\rangle \mid i \in \{-1, 0, 1\}\}$;
$\tau(\mathrm{A}_f) = \emptyset$;
$\tau(\langle \varepsilon, v, \$, 0\rangle) = \{\mathrm{A}_f\}$ for $v \in \mathrm{LF}_0\cup\{(0,0)\}$;
- $\langle S, [a, \pi(0, 0)], \$, 0\rangle \in \tau(\langle \mathrm{S}, \not\in, (0,0), i\rangle)$ if and only if $\mathrm{S}\to\pi\alpha$ is in P, where $\alpha \in \Sigma^*$, a is obtained from α by using merging function g and if $\alpha = \varepsilon$ then $a = \phi$;
- $\langle \varepsilon, (0,0), \$, 0\rangle \in \tau(\langle \mathrm{S}, \not\in, (0,0), i\rangle)$ if and only if $\mathrm{S}\to\varepsilon$ is in P;
- for $\langle \mathrm{B}, w_1, w_2, j\rangle$, $\langle \mathrm{A}, v_1, v_2, i\rangle \in \mathrm{Q}$ (v_1 possibly $\not\in$)
$\langle \mathrm{B}, w_1, w_2, j\rangle \in \tau(\langle \mathrm{A}, v_1, v_2, i\rangle)$ if and only if $\mathrm{A}\to\pi\alpha\mathrm{B}$ is in P, where $\pi \in \Pi$, $\alpha \in \Sigma^*$, $w_1 = [\phi, t_{-im,-in}(\mathrm{x,y})]$, $v_2 = [b, (\mathrm{x,y})]$ and $w_2 = [a, \pi(w_1)]$, a is obtained from α by using merging function g and if $\alpha = \varepsilon$ then $a = \phi$;
- for $\langle \varepsilon, w_1, w_2, 0\rangle$, $\langle \varepsilon, w_2', \$, 0\rangle$, $\langle \mathrm{A}, v_1, v_2, i\rangle \in \mathrm{Q}$ (v_1 possibly $\not\in$)
$\{\langle \varepsilon, w_1, w_2, 0\rangle, \langle \varepsilon, w_2', \$, 0\rangle\} \subseteq \tau(\langle \mathrm{A}, v_1, v_2, \mathrm{i}\rangle)$
if and only if $\mathrm{A}\to\pi\alpha$ is in P, where $\pi \in \Pi$, $\alpha \in \Sigma^*$, $w_1 = [\phi, \mathrm{t}_{-im,-in}(\mathrm{x,y})]$, $v_2 = [b, (\mathrm{x,y})]$, $w_2 = [a, \pi(\mathrm{w}_1)]$, $w_2' = [\phi, \pi(\mathrm{w}_1)]$, a is obtained from α by using merging function g and if $\alpha = \varepsilon$ then $a = \phi$;

(3) $\mathrm{d}(\mathrm{A}_0) = \mathrm{d}(\mathrm{A}_f) = \{0\}$;
$\mathrm{d}(\langle \mathrm{A}, v_1, v_2, i\rangle) = \{i\}$ with $\langle \mathrm{A}, v_1, v_2, i\rangle \in \mathrm{Q}$;

(4) $\Lambda = \mathrm{LF}_0\cup\mathrm{LF}_1\cup\{\not\in, \$, (0,0)\}$.

(5) pr: $\mathrm{Q} \to \Lambda$, is such that:
$\mathrm{pr}(\mathrm{A}_0) = \not\in$,
$\mathrm{pr}(\langle \mathrm{S}, \not\in, (0,0), i\rangle) = \not\in$,
$\mathrm{pr}(\langle \varepsilon , v, \$, 0\rangle) = v$,
$\mathrm{pr}(\langle \varepsilon, v_1, v_2, 0\rangle) = \{v_1, v_2\}$, $v_1 \neq \not\in$, $v_2 \neq \$$,
$\mathrm{pr}(\langle \mathrm{A}, v_1, v_2, i\rangle) = \{v_1, v_2\}$, $v_1 \neq \not\in$, $v_2 \neq \$$,
$\mathrm{pr}(\mathrm{A}_f) = \$$,

where the generic state $\langle \mathrm{A}, v_1, v_2, i\rangle$ with $v_1 = [a, w_1]$, $v_2 = [b, w_2]$ means that the derivation uses a nonterminal A, a line w_1, w_2 is drawn in the cell under the writing head

and the labels a, b are assigned to the points w_1 and w_2 respectively, and i denotes the position of the next move (-1 for 'left', 0 for 'no move' and $+1$ for 'right').

Moreover, we define the homomorphism $h_1 : \Gamma \to \Gamma$, where Γ denotes the encoding alphabet, as shown in the following.

$$h_1(\sigma) = \begin{cases} h_1(\sigma') & \text{if } \exists\{[a_1,(x,y)],v_1\},\ldots,\{[a_n,(x,y)],v_n\},[a,(x,y)] \in \sigma \\ & \text{with } n \geqslant 1, \text{ where } \sigma' = \rho \cup \{\{[f(a_1,\ldots,a_n,a),(x,y)], \\ & v_1\},\ldots,\{[f(a_1,\ldots,a_n,a),(x,y)],v_n\},\{[f(a_1,\ldots,a_n,a), \\ & (x,y)]\} \text{ with } \rho \cup \{\{[a_1,(x,y)],v_1\},\ldots,\{[a_n,(x,y)],v_n\}, \\ & [a,(x,y)]\} = \sigma, \\ h_1(\sigma') & \text{if } \exists\{[a_1,(x,y)],v_1\},\ldots,\{[a_n,(x,y)],v_n\} \in \sigma \\ & \text{with } n \geqslant 2, \text{ where } \sigma' = \rho \cup \{\{[f(a_1,\ldots,a_n,a),(x,y)],v_1\}, \\ & \ldots,\{[f(a_1\ldots,a_n,a),(x,y)],v_n\}\} \text{ with } \rho \cup \{\{[a_1,(x,y)], \\ & v_1\},\ldots,\{[a_n,(x,y)],v_n\}\} = \sigma, \\ \sigma & \text{otherwise.} \end{cases}$$

Now, observe that the points on the border of two contiguous slices are described by both, so two different points in two contiguous slices can represent the same position in the plane. It can happen that such points have associated different symbols, even if the homomorphism h_1 has been applied. In order to resolve such ambiguity, we introduce a homomorphism h_2 that is applied to each contiguous couple of elements of the words in $lang(H)$. Let $\Theta = \{w \in \Gamma^* \mid |w| = 2\}$, the homomorphism $h_2 : \Theta \to \Theta$ is defined as follows:

$$h_2(\sigma_1\sigma_2) = \begin{cases} \sigma_1\sigma_2 & \text{if } \nexists\{[a,(x,y)],v_1\} \in \sigma_1 \text{ and } \{[c,t_{-m,-n}((x,y))], \\ & v_2\} \in \sigma_2, \\ h_2(\sigma_1'\sigma_2') & \text{otherwise,} \\ & \text{where } \sigma_1' = \rho_1 \cup \{[f(a,c),(x,y)],v_1\} \text{ and } \sigma_2' = \\ & \rho_2 \cup \{[f(a,c),t_{-m,-n}((x,y))],v_2\} \text{ with } \rho_1 \cup \{[a, \\ & (x,y)],v_1\} = \sigma_1 \text{ and } \rho_2 \cup \{[c,t_{-m,-n}(x,y)],v_2\} = \sigma_2. \end{cases}$$

Furthermore, we need these notations.

Given a language K, the following languages can be defined.

$K^1 = \{w \mid w \in K \text{ and } |w| = 1\}$

$K^e = \{w \mid w \in K \text{ and } |w| \text{ is even}\}$

$K^o = \{w \mid w \in K \text{ and } |w| \text{ is odd}\}$

The above notations allow us to specify languages L_1 and L_2.

$L_1 = h_1(lang(H))$

$L_2 = L_1^1 \cup h_2(L_1^e) \cup \{h_2(w) \mid wa \in L_1^o - L_1^1\}$

Thus, we are ready to define the regular language L.

$L = L_2^1 \cup \{h_2(w) \mid awb \in L_2^e\} \cup \{h_2(w) \mid aw \in L_2^o - L_2^1\}$.

Finally, from the associative and commutative properties of merging function f it follows that μ(D) corresponds to regular language L obtained by applying homomorphisms h_1 and h_2 to $lang(H)$ as described previously.

References

[1] J. Berstel, Transductions and Context-Free Languages, Teubner Studienbücher, Stuttgart, 1979.
[2] F.J. Brandenburg, On Minimal Picture Words, MIP 8903, Tech. Report, University of Passau, 1989.
[3] F.J. Brandenburg, J. Dassow, Reduction of Picture Words, RAIRO TCS 27 (1993) 49–56.
[4] K. Culik, E. Welzl, Two finite state generator, Lecture Notes on Computer Science, Vol. 158, Springer, Berlin, 1983, 1997, pp. 106–114.
[5] J. Dassow, F. Hinz, Decision problems and regular chain code picture languages, Discrete Math. 45 (1993) 29–49.
[6] S. Ginsburg, The Mathematical Theory of Context-Free Languages, McGraw-Hill, New York, 1966.
[7] F. Hinz, Questions of decidability for context-free chain code picture languages, in: Proc. IMYCS'88, Hungarian Academy of Sciences, 1988.
[8] F. Hinz, E. Welzl, Regular chain code picture languages with invisible lines, Tech. Report 252, IIG, Tech. Univ. Graz, Austria, 1988.
[9] J.E. Hopcroft, J.D. Ullman, Introduction to Automata Theory, Languages, and Computation, Addison-Wesley, Reading, MA, 1979.
[10] C. Kim, Picture iteration and picture ambiguity, J. Comput. System Sci. 40 (1990) 289–306.
[11] C. Kim, Complexity and decidability for restricted class of picture languages, Theoret. Comput. Sci. 73 (1990) 295–311.
[12] C. Kim, Retrat bounded picture languages, Theoret. Comput. Sci. 132 (1994) 85–112.
[13] C. Kim, I.H. Sudborough, The membership and equivalence problems for picture languages, Theoret. Comput. Sci. 52 (1987) 177–191.
[14] C. Kim, I.H. Sudborough, On reversal bounded picture languages, Theoret. Comput. Sci. 104 (1992) 185–206.
[15] M. Latteux, D. Robilliard, D. Simplot, Figures composés de pixels et monoide inversif, Bull. Belgian Math. Soc. 4 (1997) 89–111.
[16] H.A. Maurer, G. Rozenberg, E. Welzl, Using string languages to describe picture languages, Inform. Control 54 (1982) 155–185.
[17] M. Pelletier, J. Sakarovitch, Easy multiplication II—extensions of rational semigroups, Inform. Comput. 88 (1990) 18–59.
[18] M. Petrich, Inverse Semigroups, Wiley, New York, 1984.
[19] D. Robilliard, D. Simplot, Undecidability of existential properties in picture languages, Theoret. Comput. Sci. 233 (2000) 51–74.
[20] J. Sakarovitch, Description des monoides de type fini, J. Inform. Process. Cybernet. 17 (8) (1981) 417–434.
[21] A. Salomaa, Formal Languages, Academic Press, London, 1973.
[22] I.H. Sudborough, E. Welzl, Complexity and Decidability for Chain Code Picture Languages, Theoret. Comput. Sci. 36 (1985) 173–202.