



Using Alternating Words to Describe Symbolic Pictures

GENNARO COSTAGLIOLA, FILOMENA FERRUCCI,
RITA FRANCESE, CARMINE GRAVINO
Department of Mathematics and Computer Science
Università di Salerno, Salerno, Italy
84081 Baronissi (SA), Italy

E-mail: {gcostagliola, fferrucci, francese, gravino}@unisa.it

Abstract

In this paper we present the concepts of *drawn symbolic picture* and *symbolic picture*. Such notions have been conceived as an extension of the concept of drawn pictures which was introduced by Maurer, Rozenberg and Welzl [31]. We provide a string description of such pictures. It consists of *alternating words*, i.e. strings whose letters are in alternation from an alphabet of symbols and an alphabet of moves. We analyze the complexity of the *description language* of a (drawn) symbolic picture and show how these yield context-sensitive, context-free and regular languages. Then, we characterize the grammars for alternating words and define the generative model for (drawn) symbolic pictures. Moreover, we investigate the different types of ambiguity that occur when dealing with symbolic grammars. Finally, we propose a classification for symbolic picture grammars and languages.

1 Introduction

Several different models have been provided to describe *pictures*. According to the definition of Maurer, Rozenberg and Welzl, a picture (called *drawn picture*) consists of unit lines drawn on the Cartesian plane considered as a square grid. They also provide an elegant description of picture languages in terms of string languages. As matter of fact, a picture is specified by a string over the alphabet $\{\mathbf{u}, \mathbf{d}, \mathbf{r}, \mathbf{l}\}$ encoding a walk through the picture. The symbol \mathbf{u} (\mathbf{d} , \mathbf{r} , \mathbf{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’.

Interesting extensions of drawn pictures, named *drawn symbolic pictures* and *symbolic pictures* can be obtained. Such extensions are based on the observation that a picture may embed more information than the shape, such as colors, labels, icons etc., and are obtained by associating a symbol from an alphabet to each point of the

picture. Thus, 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. Whereas a *symbolic picture* is given by a set of symbols disposed on a two-dimensional plane, i.e. it can be considered as a *drawn symbolic picture* in which the unit lines are not visualized. Although the extensions are simple and intuitive, they will turn out to be useful to describe more complex pictures. However, the introduction of symbols requires new theoretical issues to be addressed.

In the paper we theoretically investigate the above classes of picture languages and provide solutions for their description, generation and recognition. In particular, we show how both a *symbolic picture* and a *drawn symbolic picture* may be obtained by evaluating a string alternating symbols from an alphabet Σ , with symbols from an alphabet Π indicating the moves *right*, *left*, *down* and *up*. This definition of *symbolic pictures* extends both the concept of the traditional string and that of picture as defined by Maurer, Rozenberg and Welzl in [31]. In order to define a *drawn symbolic picture* or a *symbolic picture* language we define a *symbolic picture grammar* as a pair composed of a grammar for alternating words and an evaluation function. A grammar for alternating words is a string grammar producing strings whose letters are in alternation from an alphabet of symbols Σ and an alphabet of moves Π . The evaluation function generates the two-dimensional layout corresponding to each alternated string to form a *drawn symbolic picture* or a *symbolic picture*. We provide the definition of a canonical form of a grammar for alternating words and investigate the different types of ambiguity that occur when dealing with symbolic grammars. The main goal of the paper is to formalize the classes of the *drawn symbolic* and *symbolic picture* languages by comparing the expressive power of both in order to provide a formal theory of such languages. In addition, some decidability results involving ambiguity are addressed. In particular, we study some basic decision problems on symbolic picture and drawn symbolic picture languages: the membership problem, the problem of determining whether or not an arbitrary picture has a finite number of descriptions in a picture language and the problem of determining whether or not a picture has an ambiguous description in a picture language. We show that these problems are decidable for drawn symbolic picture languages while for symbolic picture languages this is true only under certain conditions. Finally, we propose a classification for symbolic picture grammars and languages based on the traditional Chomsky hierarchy.

The paper is organized as follows. We briefly recall notions on drawn pictures languages in section 2. Section 3 provides the definition of the notions of drawn symbolic pictures and symbolic pictures. Then, in Section 4 a string based description of these picture languages is proposed. In Section 5 the grammar model which generates such descriptions is presented and a discussion on the grammar ambiguity problem is presented. Several kinds of ambiguity are analyzed and decidability results for context-free grammars are provided. Section 6 classifies the symbolic picture grammars in terms of an extension of the Chomsky hierarchy. A discussion on related work is contained in Section 7 and final remarks conclude the paper.

2 Preliminaries

We assume that the reader is familiar with the basic definitions and the properties of the theory of string languages as found for example in [19], and [36]. We just recall several standard notations before reporting the basic notions concerning picture languages from [31]. Let A be a finite set of symbols called an alphabet. A^* denotes the monoid generated by A with the operation of concatenation (the neutral element is the empty string ϵ). For a string w over A , $|w|$ denotes its length and if x is a symbol in A , $\#_x(w)$ indicates the number of occurrences of x in w . As usual, \mathbb{Z} denotes the set of integers. With regard to drawn pictures we recall the following 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, $M_1 = \{ \{v, v'\} \mid v, v' \in M_0 \text{ and } v' \in N(v) \}$.

A *drawn picture* q is a triple $q = \langle b, s, e \rangle$, where b 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 b is nonempty then s and e are points in $W(q) = \{v \in M_0 \mid \{v, v'\} \text{ is in } b, \text{ for some } v' \in M_0\}$. If b is empty then $s = e = (0, 0)$, and $W(q) = \{(0, 0)\}$. Thus, the empty drawn picture is denoted by $\langle \emptyset, (0, 0), (0, 0) \rangle$.

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 *drawn picture language* is a set of drawn pictures where the start and the

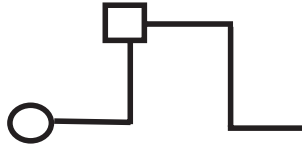


Figure 1: A drawn picture

end point are marked with a circle and a square, respectively.

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 $q = \langle b, s, e \rangle$ is to describe a walk through the picture. Such a walk starts from the start point s , touches at least once each line in b , 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 $v' = r(v)$. Thus, a walk is described by a string on the alphabet $\Pi = \{\mathbf{u}, \mathbf{d}, \mathbf{l}, \mathbf{r}\}$. For example, the string $w = \mathbf{rurdrul}$ 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 .

The drawn picture described by a string w over Π , denoted by $dpic(w)$, is defined inductively as follows: if $w = \epsilon$, then $dpic(w) = \langle \emptyset, (0,0), (0,0) \rangle$; if $w = z\pi$ for some $z \in \Pi^*$ and $\pi \in \Pi$, with $dpic(z) = \langle b, s, e \rangle$, then $dpic(w) = \langle b \cup \{e, \pi(e)\}, s, \pi(e) \rangle$.

Every string over Π is called a *picture description*, and every language over Π is called a *picture description language*. If G is a context-free string grammar which generates a picture description language, then we say that G is a *context-free picture description grammar*. Thus, the notion of picture grammar and language can be defined as follows.

A *picture grammar* \mathbf{G} can be seen as a pair $\langle G, dpic() \rangle$, where G is a picture description grammar and $dpic()$ is as above. Given a picture grammar $\mathbf{G} = \langle G, dpic() \rangle$, the picture language \mathbf{L} generated by \mathbf{G} , denoted by $\mathbf{L}(\mathbf{G})$, is $\mathbf{L}(\mathbf{G}) = dpic(L(G)) = \{dpic(x) \mid x \in L(G)\}$.

3 Drawn Symbolic Picture and Symbolic Picture Languages

In this section we formalize the notions of *drawn symbolic pictures* and *symbolic pictures* as an extension of the drawn pictures. The extension is based on the fact that a drawn picture can be seen as a set of points connected by unit lines. A drawn symbolic picture is then defined as a drawn picture where each of its points has associated a symbol from an alphabet Σ . In turn, a symbolic picture can be considered as a drawn symbolic picture where the unit lines are not visualized. In other words, a symbolic picture is a set of symbols disposed on the Cartesian plane.

In order to provide the formal definition we introduce the notion of invisible symbol ϕ as a special symbol that has no visual representation when used in a picture. In the sequel, we indicate with Σ an arbitrary alphabet which does not contain ϕ , with Σ_ϕ an alphabet which contains ϕ , and with δ_ϕ a function which may contain ϕ in its range. Thus, the definition of drawn symbolic picture can be formalized as follows.

Definition 3.1 *A drawn symbolic picture is a triple $dsp = \langle dp, \Sigma_\phi, \delta_\phi \rangle$, where dp is a drawn picture, Σ_ϕ is an alphabet of symbols, and δ_ϕ is a function $\delta_\phi: W(dp) \rightarrow \Sigma_\phi$. The start and end points of dsp are the start and end points of dp , respectively. An empty drawn symbolic picture is denoted by $\langle \langle \emptyset, (0,0), (0,0) \rangle, \Sigma_\phi, \delta_\phi \rangle$, where $\delta_\phi(0,0) = \phi$.*

Example 3.1 *Let $dsp = \langle dp, \Sigma_\phi, \delta_\phi \rangle$, be a drawn symbolic picture, where $dp = \langle \{(0,0), (1,0)\}, \{(1,0), (1,1)\}, \{(1,1), (2,1)\}, \{(2,1), (2,0)\}, \{(2,0), (3,0)\}, (0,0), (1,1) \rangle$, $\Sigma_\phi = \{a, b, c, \phi\}$, and δ_ϕ is the function such that $\delta_\phi(0,0) = a$, $\delta_\phi(1,0) = \phi$, $\delta_\phi(1,1) = b$, $\delta_\phi(2,1) = b$, $\delta_\phi(2,0) = \phi$, $\delta_\phi(3,0) = a$. The visual representation of this picture is depicted in Fig. 2, where the start and end points are marked with a circle and a square, respectively.*

In order to illustrate the descriptive capability of drawn symbolic pictures, let us note that they can be used to appropriately describe many graphical features of

Definition 3.2 A symbolic picture is a quadruple $sp = \langle P, s, \Sigma, \delta_\phi \rangle$, where $P \in M_0$ is a finite set of points, Σ is an alphabet of symbols, δ_ϕ is a function $\delta_\phi: P \rightarrow \Sigma_\phi$, and $s = (0,0)$ is the start point. The start point belongs to P if P is nonempty. An empty symbolic picture is denoted by $\langle \emptyset, (0,0), \Sigma, \delta_\phi \rangle$, where $\delta_\phi(0,0) = \phi$

Example 3.3 In Fig. 4 the symbolic picture $sp = \langle \{(0,0), (1,0), (1,1), (2,2)\}, (0,0), \Sigma, \delta_\phi \rangle$ is depicted, where $\Sigma = \{a, b, c\}$, and δ_ϕ is the function such that $\delta_\phi((0,0)) = a$, $\delta_\phi((1,0)) = a$, $\delta_\phi((1,1)) = c$, $\delta_\phi((2,2)) = b$. The start point is marked with a circle.

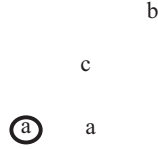


Figure 4: A symbolic picture

Example 3.4 The arithmetic expression in Fig. 5 can be described by the symbolic picture $sp = \langle \{(0,0), (1,0), (2,1), (2,0), (2,-1), (3,0), (4,1), (4,0), (4,-1)\}, (0,0), \Sigma, \delta_\phi \rangle$, where $\Sigma = \{a, b, c, e, f, +, -\}$, and δ_ϕ is the function such that $\delta_\phi(0,0) = a$, $\delta_\phi(1,0) = +$, $\delta_\phi(2,1) = b$, $\delta_\phi(2,0) = -$, $\delta_\phi(2,-1) = c$, $\delta_\phi(3,0) = +$, $\delta_\phi(4,1) = e$, $\delta_\phi(4,0) = -$, $\delta_\phi(4,-1) = f$.

$$a + \frac{b}{c} + \frac{e}{f}$$

Figure 5: An arithmetic expression

A *symbolic picture language* is a set of symbolic pictures. It can be easily shown that symbolic picture languages strictly include the class of string languages. Moreover, they allow us to describe two-dimensional languages such as arithmetic expressions and colored bitmaps.

4 String Representations of the Proposed Pictures

In this section, we define the string representations for both drawn symbolic pictures and symbolic pictures. To this aim, we need the following notations. Given two alphabets Γ and Λ such that $\Gamma \cap \Lambda = \emptyset$, with $\Gamma \approx \Lambda$ (read as *gamma alternated lambda*) we denote the set of strings whose symbols are taken from Γ and Λ in an alternate way. Thus, $\Gamma \approx \Lambda = \{\epsilon\} \cup \Gamma(\Lambda\Gamma)^*$. A string in the set $\Gamma \approx \Lambda$ is called a $\Gamma \approx \Lambda$ -word. For example, the strings ‘*a*’ and ‘*arbdclarbdau*’ are words in the set $\Gamma \approx \Lambda = \{a, b, c\} \approx \{\mathbf{u}, \mathbf{d}, \mathbf{l}, \mathbf{r}\}$, where symbols from the two sets are shown using different fonts for sake of clarity. A set of $\Gamma \approx \Lambda$ -words is called a $\Gamma \approx \Lambda$ -language.

4.1 Drawn Symbolic Picture Descriptions

The description of a drawn symbolic picture can be given in analogy with the description of a drawn picture by describing a walk through the picture. Such a walk starts from the start point, touches at least once each line and symbol in the drawn symbolic picture, and ends at the end point. Thus, such a walk is described by a $\Sigma_\phi \approx \Pi$ -word w , where the symbols from Σ_ϕ are alternated with the moves from Π . As an example, the $\Sigma_\phi \approx \Pi$ -word ‘ $ar\phi u\phi r\phi b\phi r\phi l\phi b\phi u\phi l\phi b$ ’ describes the drawn symbolic picture depicted in Fig. 2. The drawn symbolic picture described by a $\Sigma_\phi \approx \Pi$ -word w is denoted by $dspic(w)$ and is defined inductively in agreement with the following definition.

Definition 4.1 *Let w be a $\Sigma_\phi \approx \Pi$ -word. The drawn symbolic picture described by w , denoted by $dspic(w)$, is defined inductively as follows:*

- if $w = \epsilon$, then $dspic(w)$ is the empty drawn symbolic picture;
- if $w = \sigma \in \Sigma_\phi$, then $dspic(w) = \langle \langle \emptyset, (0,0), (0,0) \rangle, \Sigma_\phi, \delta_\phi \rangle$, where δ_ϕ is only defined in $(0,0)$ and $\delta_\phi((0,0)) = \sigma$;
- if $w = w' \tau$ for some $w' \in (\Sigma_\phi \approx \Pi \setminus \{\epsilon\})$, with $dspic(w') = \langle sc, \Sigma_\phi, \delta_\phi \rangle$, $sc = \langle b, s, e \rangle$, and $\tau = \pi\sigma$ with $\pi \in \Pi$ and $\sigma \in \Sigma_\phi$, then $dspic(w) = \langle \langle b \cup \{e, \pi(e)\}, s, \pi(e) \rangle, \Sigma_\phi, \underline{\delta}_\phi \rangle$, where $\underline{\delta}_\phi : W(sc) \cup \{\pi(e)\} \rightarrow \Sigma_\phi$ is the labeling function such that:

$$\underline{\delta}_\phi(v) = \begin{cases} \sigma & \text{if } (v = \pi(e) \text{ and } (v \notin W(sc) \text{ or } \delta_\phi(v) = \phi)) \\ \delta_\phi(v) & \text{otherwise} \end{cases}$$

Informally, $dspic(w)$ constructs a drawn symbolic picture by scanning the string w from left to right. The initial position in the plane is set to $(0,0)$ and a pointer p points to the first symbol in w . Whenever p points to a symbol $\sigma \in \Sigma_\phi$, and no visible symbol has been written on the current position on the plane then σ is written on that position and p is moved to the next symbol in w . Whenever a visible symbol has already been written on the current position then p is moved to the next symbol in w . If p points to a symbol $\pi \in \Pi$, then the current position on the plane is updated in agreement with π and p points to the next symbol in w .

Observe that a conflict arises whenever p points to a symbol $\sigma \in \Sigma_\phi$, and a symbol $\rho \in \Sigma_\phi$ with $\rho \neq \sigma$ has already been written on the current position on the plane. In agreement with the above definition, $dspic()$ solves this type of conflict by overwriting ρ with σ in the case $\rho = \phi$ and by ignoring σ otherwise. A drawn symbolic picture description without conflicts is called a *consistent* description.

Example 4.1 *The $\Sigma_\phi \approx \Pi$ -word $w = 'ar\phi r\phi r\phi r\phi l\phi c'$ describing the drawn symbolic picture of Fig. 6 is not consistent. The same picture can be described by the consistent $\Sigma_\phi \approx \Pi$ -word $w' = 'ar\phi r\phi r\phi r\phi l\phi b'$.*



Figure 6: The drawn symbolic picture of the Example 4.1

The importance of using consistent description for pictures can be deduced by considering the case of altimetrical information associated to buildings, proposed in the example 3.2. In fact, a drawn symbolic picture representing a building must not have two different height values associated to a given building vertex. More formally, we provide the following definition of consistent $\Sigma_\phi \approx \Pi$ -words.

Definition 4.2 A consistent $\Sigma_\phi \approx \Pi$ -word w is defined inductively as follows:

- if $w = \epsilon$, or $w = \sigma \in \Sigma_\phi$, then w is consistent;
- if $w = w' \tau$ for some consistent $w' \in (\Sigma_\phi \approx \Pi \setminus \{\epsilon\})$, with $\text{dspic}(w') = \langle \text{sc}, \Sigma_\phi, \delta_\phi \rangle$, $\text{sc} = \langle b, s, e \rangle$, and $\tau = \pi\sigma$ with $\pi \in \Pi$ and $\sigma \in \Sigma_\phi$, then w is consistent if the following implication holds:
if $\pi(e) \in W(\text{sc})$ then $\delta_\phi(\pi(e)) = \sigma$.

Given a drawn symbolic picture, there may exist many different $\Sigma_\phi \approx \Pi$ -words describing it. The set of words that describe a drawn symbolic picture dsp , defined as:

$$\text{dspdes}(\text{dsp}) = \{w \in \Sigma_\phi \approx \Pi \mid \text{dspic}(w) = \text{dsp}\}$$

is the *description language* of dsp . Similarly, the set of consistent words which describe a drawn symbolic picture dsp , defined as:

$$\text{Cdspdes}(\text{dsp}) = \{w \in \Sigma_\phi \approx \Pi \mid w \text{ is consistent and } \text{dspic}(w) = \text{dsp}\}$$

is the *consistent description language* of dsp . The following theorem states the regularity of the sets $\text{dspdes}(\text{dsp})$ and $\text{Cdspdes}(\text{dsp})$.

Theorem 4.1 Let dsp be a drawn symbolic picture. Then, the sets $\text{dspdes}(\text{dsp})$ and $\text{Cdspdes}(\text{dsp})$ are regular $\Sigma_\phi \approx \Pi$ -languages.

Proof. Let $\text{dsp} = \langle dp, \Sigma_\phi, \delta_\phi \rangle$ be a drawn symbolic picture, with $dp = \langle b, s, e \rangle$ and $b \neq \emptyset$. (The case $b = \emptyset$ is trivial.) We show only the regularity of the set $\text{dspdes}(\text{dsp})$. The regularity of the set $\text{Cdspdes}(\text{dsp})$ can be proved in a similar way. In order to construct a deterministic finite automaton accepting $\text{dspdes}(\text{dsp})$ we define the following sets:

$$F_\Sigma = \{[v, N, P] \mid v \in W(dp), N \in 2^b, \text{ and } P \subseteq W(dp)\},$$

where v indicates the current position in dsp , N indicates the already visited lines in dsp , P indicates the points of dsp which are assigned the right symbol and

$$F_\Pi = \{[[v], N, P] \mid v \in W(dp), N \in 2^b, \text{ and } P \subseteq W(dp)\}.$$

A deterministic finite automaton accepting $\text{dspdes}(\text{dsp})$ is

$$H = (Q, \Sigma_\phi \cup \Pi, \tau, [(0, 0), \emptyset, \emptyset], \{[[e], b, W(dp)]\})$$

where:

$Q = F_\Sigma \cup F_\Pi \cup \{\$\}$ is the set of states ($\$$ is a new symbol which is used to signal an error), and $\tau : Q \times (\Sigma_\phi \cup \Pi) \rightarrow Q$ is the transition function which is defined as follows

$$\tau([v, N, P], x) = \begin{cases} [[v], N, P] & \text{if } x \in \Sigma_\phi \text{ and } v \in P \\ [[v], N, P \cup \{v\}] & \text{if } x \in \Sigma_\phi, \delta_\phi(v) = x \text{ and } v \notin P \\ [[v], N, P] & \text{if } x = \phi \text{ and } \delta_\phi(v) \neq x \\ \$ & \text{otherwise} \end{cases}$$

$$\tau([[v], N, P], x) = \begin{cases} [x(v), N \cup \{v, x(v)\}, P] & \text{if } x \in \Pi \text{ and } (v, x(v)) \in b \\ \$ & \text{otherwise} \end{cases}$$

$$\tau(\$, x) = \$$$

The reader can easily verify that $dspdes(dsp)$ is equal to the language accepted by H . Let us observe that, when the automaton is in a state $[v, N, P] \in F_\Sigma$, the input symbol x must be in Σ_ϕ . If the point v has been already visited, or the input symbol $x = \phi \neq \delta_\phi(v)$ then it enters state $[[v], N, P] \in F_\Pi$. Similarly, if the input symbol $x \in \Sigma_\phi$ is equal to the symbol associated with the point v , i.e., $\delta_\phi(v) = x$ it enters the state $[[v], N, P \cup \{v\}] \in F_\Pi$. When the automaton is in a state $[[v], N, P]$ of F_Π and the input symbol x represents a move such that the newly constructed line $(v, x(v))$ belongs to b , it enters state $[x(v), N \cup \{v, x(v)\}, P]$ of F_Σ . Such a state keeps track of the new point $x(v)$ and of the set of lines so far visited. A $\Sigma_\phi \approx \Pi$ -word w is accepted iff every line of b is visited, each point of b is assigned the right symbol, and the last visited point described by w is e .

4.2 Symbolic Picture Descriptions

In order to give the string description of symbolic pictures, we recall the notion of invisible lines which was introduced in [39]. These lines correspond to moves which are executed by raising the pen. The set of invisible moves is denoted by $\Pi_b = \{r_b, l_b, u_b, d_b\}$. According to this definition each move m in $\Pi_b = \{r_b, l_b, u_b, d_b\}$ is performed with the pen up so that no line is actually drawn on the plane. The description of a symbolic picture can be given by describing a walk (with the pen up) through the picture. Such a walk starts from the start point and touches at least once each point in the symbolic picture. The walk between two non-contiguous points is described by exploiting invisible symbols. Thus, a symbolic picture can be described by a $\Sigma_\phi \approx \Pi_b$ -word w , where the symbols from Σ_ϕ are alternated with the moves from Π_b . As an example, the $\Sigma_\phi \approx \Pi_b$ -word “ $ar_b au_b cr_b \phi u_b b$ ” describes the picture depicted in Fig. 4. In other words a string-based representation of a symbolic picture sp can be given by applying the following steps:

1. consider a drawn picture dp whose vertices “touch” all the symbols of the desired sp ,
2. consider a drawn symbolic picture dsp that could be obtained from the “superimposition” of sp with dp and let w a $\Sigma_\phi \approx \Pi$ -word representing dsp ,

3. substitute each symbol from Π in w with the corresponding symbol from $\Pi_b = \{r_b, l_b, u_b, d_b\}$ to obtain a $\Sigma_\phi \approx \Pi_b$ -word.

The symbolic picture described by a $\Sigma_\phi \approx \Pi_b$ -word w is denoted by $spic(w)$ and it is defined inductively in agreement with Definition 4.3.

In order to give the formal definition of $spic(w)$, we make use of the technical notation $\theta(w)$ which indicates the position of the last symbol in w . In other words, $\theta(w)$ captures the notion of end point which is lost in a symbolic picture. $\theta(w)$ is defined as $\theta(w) = (\#_{r_b}(w) - \#_{l_b}(w), \#_{u_b}(w) - \#_{d_b}(w))$.

Definition 4.3 *Let w be a $\Sigma_\phi \approx \Pi_b$ -word. The symbolic picture described by w , denoted by $spic(w)$, is defined inductively as follows:*

- if $w = \epsilon$, then $spic(w)$ is the empty symbolic picture;
- if $w = \sigma \in \Sigma_\phi$, then $spic(w) = \langle \{(0,0)\}, (0,0), \Sigma, \delta_\phi \rangle$, where δ_ϕ is only defined in $(0,0)$ and $\delta_\phi((0,0)) = \sigma$;
- if $w = w' \tau$ for some $w' \in (\Sigma_\phi \approx \Pi_b - \{\epsilon\})$, with $spic(w') = \langle P, (0,0), \Sigma, \delta_\phi \rangle$, and $\tau = \pi\sigma$ with $\pi \in \Pi_b$, then

if $\sigma = \phi$ then $spic(w) = spic(w')$, else

if $\sigma \in \Sigma$, then $spic(w) = \langle P \cup \{\pi(\theta(w'))\}, (0,0), \Sigma, \underline{\delta}_\phi \rangle$,

where $\underline{\delta}_\phi: P \cup \{\pi(\theta(w'))\} \rightarrow \Sigma$ is the labeling function such that:

$$\underline{\delta}_\phi(v) = \begin{cases} \delta_\phi(v) & \text{if } v \in P \\ \sigma & \text{if } v \notin P \end{cases}$$

Informally, a symbolic picture is derived by scanning the string w from left to right similarly to the construction of a drawn symbolic picture. Starting from the $(0,0)$ point, whenever the pointer p points to a symbol $\sigma \in \Sigma$, and the current position on the plane is free then σ is written on that position and p is moved to the next symbol in w . If p points to the symbol ϕ , then p points to the next symbol in w . If p points to a symbol $\pi \in \Pi_b$, then the current position on the plane is updated in agreement with π and p points to the next symbol in w . According to Definition 4.3 a position is free only if this position does not have a visible symbol assigned yet. The set of words which describe a symbolic picture sp , defined as:

$$spdes(sp) = \{w \in \Sigma_\phi \approx \Pi_b \mid spic(w) = sp\}$$

is the description language of sp . The notion of consistency can be extended to the descriptions for the symbolic pictures in the natural way. The set of consistent words which describe a symbolic picture sp , defined as:

$$Cspdes(sp) = \{w \in \Sigma_\phi \approx \Pi_b \mid w \text{ is consistent and } spic(w) = (sp)\}$$

is the consistent description language of sp .

Informally, symbolic picture languages can be considered as drawn symbolic picture languages where the spatial relations 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, Theorem 4.3 establishes that the string description language and the consistent string description language of a symbolic picture are context-sensitive languages. On the other hand, it is possible to consider subsets of string description languages that are regular languages. In particular, the *k-description language* of a symbolic picture is a regular description language. It consists of a subset of the description language obtained by allowing to be described only a limited number of points not belonging to the given symbolic picture. In order to give the formal definition of k-description language, we introduce the following notation. Let w_b be a $\Sigma_\phi \approx \Pi_b$ -word and h_b the homomorphism $h_b : \Pi_b^* \rightarrow \Pi^*$, which replaces any symbol $\pi_b \in \Pi_b$ with the corresponding element $\pi \in \Pi$.

Definition 4.4 Let $sp = \langle P, s, \Sigma, \delta_\phi \rangle$ be a symbolic picture and k a natural number. The k-description language of sp is the set:

$$spdes^k(sp) = \{w_b \in \Sigma_\phi \approx \Pi_b \mid spic(w_b) = sp \text{ and (if } dspic(h_b(w_b)) = \langle sc, \Sigma_\phi, \delta_\phi \rangle \text{ then } |W(sc) - P| \leq k)\}$$

The consistent k-description language of sp is the set:

$$Cspdes^k(sp) = \{w_b \in spdes^k(sp) \mid w_b \text{ is consistent}\}$$

In other words, $w_b \in spdes^k(sp)$ if the $\Sigma_\phi \approx \Pi$ -word $w =_b w_b$, with $dspic(w) = \langle sc, \Sigma_\phi, \delta_\phi \rangle$ is such that $|\{v \in W(sc) \mid \delta_\phi(v) = \phi\}| \leq k$.

Theorem 4.2 Let $sp = (P, s, \Sigma, \delta_\phi)$ be a symbolic picture, and k be a natural number. Then the sets $spdes^k(sp)$ and $Cspdes^k(sp)$ are regular $\Sigma_\phi \approx \Pi_b$ -languages.

Proof. Let SS_{sp}^k be the set of the drawn symbolic pictures $dsp = \langle sc, \Sigma_\phi, \delta_\phi \rangle$ such that $P \cup \{s\} \subset W(sc)$ for sp and $\delta_\phi(v) = \phi$ for not more than k vertices $v \in W(sc)$. It is easily seen that the set is finite for any given natural number k and that $spdes^k(sp) = \bigcup_{dsp \in SS_{sp}^k} dspdes(dsp)$ once each move π is substituted with π_b . Similarly, $Cspdes^k(sp) = \bigcup_{dsp \in SS_{sp}^k} Cdspdes(dsp)$ once each move π is substituted with π_b . Since each $dspdes(dsp)$ and $Cdspdes(dsp)$ is a regular language (see Theorem 4.1) and the regular languages are closed under finite union, then $spdes^k(sp)$ and $Cspdes^k(sp)$ are regular languages.

In order to prove the context-sensitiveness of the set $spdes(sp)$ containing all the possible string descriptions for a symbolic picture sp , we introduce the concept of invisible path.

Definition 4.5 An invisible path between two points p_1 and p_2 at distance $|p_2 - p_1| = (i, j)$, is a $\Pi_b \approx \{\phi\}$ -word w such that $|\#_{r_b}(w) - \#_{l_b}(w)| = i$ and $|\#_{u_b}(w) - \#_{d_b}(w)| = j$. The set of all the invisible paths from p_1 to p_2 is denoted by $I_{p_1 > p_2}$.

As an example, let p_1 and p_2 be two points at distance $|p_2 - p_1| = (1, 1)$, an invisible path in $I_{p_1 > p_2}$ is the $\Pi_b \approx \{\phi\}$ -word $\mathbf{d}_b \phi \mathbf{r}_b \phi \mathbf{u}_b \phi \mathbf{u}_b$. It can be noted that, for any given pair of points, the set of all the invisible paths on the set of moves $\Pi_b = \{r_b, d_b, l_b, u_b\}$ is a context-sensitive language. An intuition of this is obtained by noticing that

an invisible path between two points at distance (i,j) is described by a $\Pi_b \approx \{\phi\}$ -word containing $m+i$ occurrences of r_b , $n+j$ occurrences of u_b , m occurrences of l_b , and n occurrences of d_b . Any automaton should then remember both m and n for recognizing an invisible path. On the other hand, it is not difficult to build a context-sensitive grammar for such paths.

Theorem 4.3 *Let $sp = \langle P, s, \Sigma, \delta_\phi \rangle$ be a symbolic picture. Then, $spdes(sp)$ and $Cspdes(sp)$ are context-sensitive $\Sigma_\phi \approx \Pi_b$ -languages.*

Proof. See the Appendix.

5 Grammars for the Proposed Chain Code Models

As shown in the previous section, a drawn symbolic picture is the result of applying the function $dspic$ to a $\Sigma_\phi \approx \Pi$ -word, while a symbolic picture is obtained by applying the function $spic$ to a $\Sigma_\phi \approx \Pi_b$ -word. By following this approach, a picture grammar generating drawn symbolic pictures and symbolic pictures must be based on a string grammar able to generate strings whose letters are in alternation from two alphabets which denote symbols and moves. The *grammars for alternating words* (or *alternating grammars*, for short) are described in next subsection.

5.1 Grammars for Alternating Words

In this subsection, we formalize the concept of grammar for alternating words on two sets Γ and Λ ($\Gamma \approx \Lambda$ -grammar for short) and provide a canonical form for context-free alternating grammars. Informally, a $\Gamma \approx \Lambda$ -grammar is a string grammar, with terminals in $\Gamma \cup \Lambda$, whose corresponding language is a subset of $\Gamma \approx \Lambda$.

Definition 5.1 *Let Γ and Λ be two disjoint sets of symbols, an alternating grammar on the sets Γ and Λ , denoted by $\Gamma \approx \Lambda$ -grammar, is specified by the 5-tuple $\langle \Gamma, \Lambda, N, P, S \rangle$, and is defined as a $\Gamma \cup \Lambda$ -grammar $G = \langle \Gamma \cup \Lambda, N, P, S \rangle$ for which $L(G) \subseteq \Gamma \approx \Lambda$.*

In particular we will focus our attention on *context-free* $\Gamma \approx \Lambda$ -grammars which are characterized by context-free $\Gamma \cup \Lambda$ -grammars.

An important form for context-free $\Gamma \approx \Lambda$ -grammar is the *canonical form* which is defined in agreement with the following definition:

Definition 5.2 *Let $G = \langle \Gamma, \Lambda, N, P, S \rangle$ be a $\Gamma \approx \Lambda$ -grammar. We say that G is in canonical form if each production in P is of type $A \rightarrow \alpha$ where $A \in N$ and $\alpha \in ((\Gamma \cup \Lambda) \approx \Lambda) - \{\epsilon\}$.*

As an example, let $G = \langle \Sigma_\phi \cup \Pi, N, P, S \rangle$ be a $\Sigma_\phi \cup \Pi$ -grammar, where $\Sigma_\phi = \{a, b, \phi\}$, $N = \{S, A, B\}$ and P is the set of productions $\{S \rightarrow adbrA, A \rightarrow uB|aB|a, B \rightarrow radS|\epsilon\}$. G is not in canonical form, since it contains the two offending productions $B \rightarrow radS$ and $B \rightarrow \epsilon$.

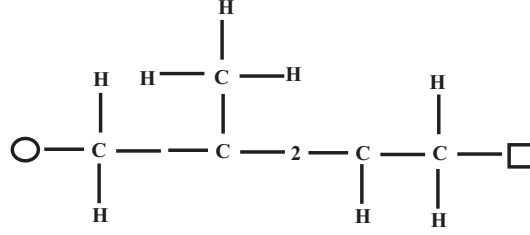


Figure 7: A drawn symbolic picture describing a section of a natural rubber molecule

Now, let us observe that any alternating grammar can be translated into an equivalent alternating grammar in canonical form (but without the ϵ -word). Thus, the following theorem holds.

Theorem 5.1 For each $\Gamma \approx \Lambda$ -grammar G there exists a canonical $\Gamma \approx \Lambda$ -grammar G' such that $L(G') = L(G) \setminus \{\epsilon\}$.

5.2 DSP and SP Picture Grammars

In the sequel, a $\Sigma_\phi \approx \Pi$ -grammar, denoted by $G_{\Sigma_\phi \approx \Pi}$, will be used to generate drawn symbolic picture descriptions. Thus, the drawn symbolic picture grammars and languages are defined as follows.

Definition 5.3 A drawn symbolic picture grammar (DSP grammar, for short) \mathbf{G} is a pair $\langle G_{\Sigma_\phi \approx \Pi}, \text{dspic} \rangle$, where $G_{\Sigma_\phi \approx \Pi}$ is a $\Sigma_\phi \approx \Pi$ -grammar and dspic is the function that translates a $\Sigma_\phi \approx \Pi$ -word into a drawn symbolic picture. Given a DSP grammar $\mathbf{G} = \langle G_{\Sigma_\phi \approx \Pi}, \text{dspic} \rangle$, the DSP language \mathbf{L} generated by \mathbf{G} , denoted by $\mathbf{L}(\mathbf{G})$, is:

$$\mathbf{L}(\mathbf{G}) = \text{dspic}(L(G_{\Sigma_\phi \approx \Pi})) = \{\text{dspic}(x) \mid x \in L(G_{\Sigma_\phi \approx \Pi})\}.$$

Example 5.1 Let us consider the drawn symbolic picture depicted in Fig. 7. It describes a section of a natural rubber molecule [8]. The symbols 'C' and 'H', represent the carbon atom, and the hydrogen atom, resp., while symbol '2' is used to denote a double link between two carbon atoms.

The repetitive chemical structure of a natural rubber molecule is described by the drawn symbolic picture language generated by the DSP grammar $\mathbf{RM} = \langle \text{RM}_{\Sigma_\phi \approx \Pi}, \text{dspic} \rangle$, where $\text{RM}_{\Sigma_\phi \approx \Pi}$ is the $\Sigma_\phi \approx \Pi$ -grammar specified as follows.

$\text{RM}_{\Sigma_\phi \approx \Pi} = \langle \{C, H, 2, \phi\}, \Pi, \{S, \text{Section}, C_2-H_3, C-H_2, C-H\}, P, S \rangle$

where S is the initial symbol and P contains the productions:

$$\begin{aligned} S &\rightarrow \phi \mathbf{r} \text{Section} \mathbf{r} \phi \\ S &\rightarrow \phi \mathbf{r} \text{Section} \mathbf{r} S \\ \text{Section} &\rightarrow C-H_2 \mathbf{r} \phi \mathbf{r} C_2-H_3 \mathbf{r} 2 \mathbf{r} C-H \mathbf{r} C-H_2 \\ C-H_2 &\rightarrow C \mathbf{u} H \mathbf{d} C-H \\ C_2-H_3 &\rightarrow C \mathbf{u} C \mathbf{r} H \mathbf{l} C \mathbf{u} H \mathbf{d} C \mathbf{l} H \mathbf{r} C \mathbf{d} C \\ C-H &\rightarrow C \mathbf{d} H \mathbf{u} C \end{aligned}$$

The concept of consistency can be extended to DSP grammars in the natural way.

Definition 5.4 *A DSP grammar $\mathbf{G} = \langle G_{\Sigma_\phi \approx \Pi}, \text{dspic} \rangle$, is consistent if any $\Sigma_\phi \approx \Pi$ -word $w \in L(G_{\Sigma_\phi \approx \Pi})$ is consistent.*

The DSP grammar **RM** of Example 5.1 is a consistent DSP grammar. Indeed, any word $w \in L(RM_{\Sigma_\phi \approx \Pi})$ is consistent as can be easily verified by observing that any string on the right-hand side of any production is consistent and the subpictures generated by any nonterminal can never overlap. The decidability of the consistency of a drawn symbolic picture grammar is an interesting open problem. The problem will be partially addressed in the following. In particular, it is always possible to decide whether or not a $\Sigma_\phi \approx \Pi$ -grammar generates only consistent descriptions for a given drawn symbolic picture. The proof of such results exploits the following decidability properties which are straightforward consequences of the regularity of the set of consistent descriptions of a drawn symbolic picture:

Corollary 5.1 *Let G be a $\Sigma_\phi \approx \Pi$ -grammar, and dsp a drawn symbolic picture. The following problems are decidable:*

1. $L(G) \cap C\text{dspdes}(\text{dsp}) \neq \emptyset$
i.e., $L(G)$ contains a consistent description for the drawn symbolic picture dsp ;
2. $|L(G) \cap C\text{dspdes}(\text{dsp})| < \infty$
i.e., $L(G)$ contains a finite number of consistent descriptions for the drawn symbolic picture dsp ;
3. $|L(G) \cap C\text{dspdes}(\text{dsp})| = 1$
i.e., $L(G)$ contains only one consistent description for the drawn symbolic picture dsp .

Proof. From Theorem 4.1 and from the closure property of context-free languages with regular sets we have that the set $L(G) \cap C\text{dspdes}(\text{dsp})$ is a context-free language. Thus, the decidability results follow from the decidability of the emptiness, the finiteness, and the singleton problems for context-free string languages [19].

As a consequence of Theorem 4.1 and Corollary 5.1, we have the following result which states that the problem of determining whether or not a drawn symbolic picture has only consistent descriptions in a language is decidable for context-free languages.

Theorem 5.2 *Let G be a $\Sigma_\phi \approx \Pi$ -grammar, and dsp a drawn symbolic picture. It is decidable whether $L(G)$ contains only consistent descriptions for dsp .*

Proof. By Corollary 5.1 we have that it is decidable whether $L(G)$ contains a consistent description for dsp . Moreover, we show that it is decidable whether $L(G)$ contains a nonconsistent description for dsp . To this aim, we notice that the nonconsistent description language of dsp , defined as:

$$\text{NCdspdes}(\text{dsp}) = \text{dspdes}(\text{dsp}) - C\text{dspdes}(\text{dsp})$$

is a regular $\Sigma_\phi \approx \Pi$ -language (by the closure properties of regular sets). Thus, by the closure properties of context-free languages, the set

$$L(\mathbf{G}) \cap \text{NCdspdes}(\text{dsp})$$

is a context-free language. Then, the claim follows from the decidability of the emptiness problem of context-free languages.

A DSP grammar $\mathbf{G} = \langle G_{\Sigma_\phi \approx \Pi}, \text{dspic} \rangle$ is in canonical form iff the $\Sigma_\phi \approx \Pi$ -grammar $G_{\Sigma_\phi \approx \Pi}$ is in canonical form. It can be easily verified that when $G_{\Sigma_\phi \approx \Pi}$ is in canonical form, any sentential form sf of $G_{\Sigma_\phi \approx \Pi}$ is a sentence in $(\Sigma_\phi \cup N) \approx \Pi$. As a consequence, by applying the function dspic to sf , this is translated into a drawn symbolic picture on $(\Sigma_\phi \cup N)$ where N is the set of non-terminals of $G_{\Sigma_\phi \approx \Pi}$. Such a translation will be referred to as a *pictorial sentential form*.

Example 5.2 Let $\mathbf{G} = \langle \langle \Sigma_\phi = \{a, b, c, \phi\}, \Pi, N = \{S, A, B\}, P, S \rangle, \text{dspic} \rangle$ be a DSP grammar, where P is the set of productions $\{ S \rightarrow \text{adbr}A, A \rightarrow \text{cu}B|a, B \rightarrow \text{ar}S \}$. Since \mathbf{G} is in canonical form, a derivation of a drawn symbolic picture from \mathbf{G} is given by a sequence of pictorial sentential forms. Thus, $\text{adbr}A$ is a sentential form and adbra is a sentence of the grammar \mathbf{G} and their evaluations produce the pictures in Fig. 8(a) and Fig. 8(b), respectively.



Figure 8: A pictorial sentential form (a), and a sentence (b)

Similarly to the case of DSP, the definition of an SP grammar and SP language can be given.

Definition 5.5 A symbolic picture grammar (SP grammar, for short) \mathbf{G} is a pair $\langle G_{\Sigma_\phi \approx \Pi_b}, \text{spic} \rangle$, where $G_{\Sigma_\phi \approx \Pi_b}$ is a $\Sigma_\phi \approx \Pi_b$ -grammar and spic is the function that translates a $\Sigma_\phi \approx \Pi_b$ -word into a symbolic picture. Given an SP grammar $\mathbf{G} = \langle G_{\Sigma_\phi \approx \Pi_b}, \text{spic} \rangle$, the SP language \mathbf{L} generated by \mathbf{G} , denoted by $\mathbf{L}(\mathbf{G})$, is:

$$\mathbf{L}(\mathbf{G}) = \text{spic}(L(G_{\Sigma_\phi \approx \Pi_b})) = \{\text{spic}(x) \mid x \in L(G_{\Sigma_\phi \approx \Pi_b})\}.$$

In the following example we provide a symbolic picture grammar describing a set of text layouts with left justified lines. The whole text is ended with an end-of-text marker.

Example 5.3 Let $\mathbf{TLayout} = \langle \langle T \cup \Pi_b, N, S, P \rangle, \text{spic} \rangle$ be an SP grammar with:

$$\begin{aligned} S &= \{\text{Text}\} \\ N &= \{\text{Text}, \text{Line}\} \\ T &= \{\phi, \text{char}, \text{eot}\} \\ P &= \{ \text{Text} \rightarrow \text{Line } \mathbf{d}_b \text{ Text} \mid \text{eot} \\ &\quad \text{Line} \rightarrow \text{char } \mathbf{r}_b \text{ Line } \mathbf{l}_b \phi \mid \text{char} \}. \end{aligned}$$

A sentence from this grammar is:

$$\text{char } \mathbf{r}_b \text{ char } \mathbf{r}_b \text{ char } \mathbf{r}_b \text{ char } \mathbf{l}_b \phi \mathbf{l}_b \phi \mathbf{l}_b \phi \mathbf{d}_b \text{ char } \mathbf{d}_b \text{ char } \mathbf{r}_b \text{ char } \mathbf{l}_b \phi \mathbf{d}_b \text{ eot}$$

whose evaluation produces the symbolic picture shown in Fig. 9.

```

char char char char
char
char char
eot

```

Figure 9: A left justified text

An SP grammar $\mathbf{G} = \langle G_{\Sigma_\phi \approx \Pi_b}, spic \rangle$ is in canonical form iff the $G_{\Sigma_\phi \approx \Pi_b}$ -grammar is in canonical form. It can be easily verified that when $G_{\Sigma_\phi \approx \Pi_b}$ is in canonical form, any sentential form sf of $G_{\Sigma_\phi \approx \Pi_b}$ is a sentence in $(\Sigma_\phi \cup N) \approx \Pi_b$. Similarly to the case of canonical DSP grammars, the sentential forms of a $\Sigma_\phi \approx \Pi_b$ -grammar in canonical form can be translated into symbolic pictures on the alphabet $\Sigma \cup N$ where N is the set of non-terminals of $G_{\Sigma_\phi \approx \Pi_b}$ by the function $spic$. Such a symbolic picture will be referred to as a *pictorial sentential form*.

Example 5.4 Let $\mathbf{G} = \langle \langle \Sigma_\phi = \{n, +, *, -, \phi\}, \Pi_b, N = \{E, T, F\}, P, S \rangle, spic \rangle$, be the symbolic picture grammar in canonical form with:

$$\begin{aligned}
P = \{ & E \rightarrow E \mathbf{r}_b + \mathbf{r}_b T, \\
& E \rightarrow T, \\
& T \rightarrow T \mathbf{r}_b * \mathbf{r}_b F, \\
& T \rightarrow - \mathbf{u}_b n \mathbf{d}_b \phi \mathbf{d}_b n \mathbf{u}_b \phi, \\
& T \rightarrow F, \\
& F \rightarrow n \}
\end{aligned}$$

describing some simple arithmetic expressions. A rightmost pictorial derivation of \mathbf{G} is:

$$E \rightarrow E + T \rightarrow E + F \rightarrow E + n \rightarrow T + n \rightarrow \frac{n}{n} + n$$

It is worth noting that the SP Grammar given in Example 5.3 is another example of a symbolic picture grammar in canonical form.

5.3 Ambiguous DSP and SP Grammars

In the field of formal language theory the grammar ambiguity problem has been extensively investigated. As a matter of fact, for compiling applications we need to design unambiguous grammars, or to use ambiguous grammars with additional rules to resolve the ambiguities. Thus, the problem of determining whether or not a grammar is ambiguous is crucial. In the former case, we cannot uniquely determine which parse tree to select for a sentence. In the following, we will focus our attention on this aspect and provide some decidability results.

A string grammar G is *ambiguous* if there exists a sentence in $L(G)$ for which G produces two different derivation trees. In the case of DSP and SP grammars, this definition implies two distinct cases: a grammar \mathbf{G} is ambiguous if there exists a picture p in $\mathbf{L}(\mathbf{G})$ such that, either there are two string descriptions in \mathbf{G} for p , or the only string description of p in \mathbf{G} may be derived in two different ways. In both cases, p would have two different syntactic interpretations. In the former we have *visual*

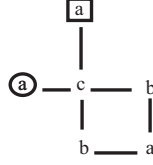


Figure 10: An ambiguous picture under the DSP grammar of Example 5.5

ambiguity in the latter we have *structural ambiguity*. In this section we will focus our attention on visual ambiguity.

The general visual ambiguity for picture languages was addressed in [23]. There, it has been proved that the problem of determining whether or not a language contains two distinct words that describe the same picture is undecidable for regular languages and for linear languages which describe three-way stripe picture languages. Visual ambiguity was previously treated in [31], where the authors show that the problem of determining whether or not an arbitrary picture p can be drawn by two distinct words in a language L is undecidable for context-sensitive Π -languages and is decidable for context-free Π -languages. In this section, we will show that this problem, extended to DSP languages and SP languages, is decidable for a context-free grammar \mathbf{G} . For the SP languages, we refer to k -description languages.

In a DSP grammar we distinguish two reasons for which $dspic$ produces the same drawn symbolic picture on different sentences:

1. the sentences describe different walks on the drawn symbolic picture;
2. the sentences describe the same walk on the drawn symbolic picture, and the walk intersects itself in a point. Due to the definition of $dspic$, the sentences may have different symbols associated to that point yet produce the same picture.

In the first case, we talk about *visual ambiguity on draw*, in the second case, we talk about *visual ambiguity on intersection*. It is intuitive that a consistent DSP grammar cannot exhibit visual ambiguity on intersection.

Example 5.5 Let $\mathbf{G} = \langle G_{\Sigma_{\phi} \approx \Pi}, dspic \rangle$ be a DSP grammar, where $G_{\Sigma_{\phi} \approx \Pi} = \langle \{r, l, u, d\}, \{a, b, c, k\}, \{S, A, B, C\}, P, S \rangle$ and P is the set of productions $\{S \rightarrow arcAaB, A \rightarrow rbd|dbr, B \rightarrow lbuC|ublC, C \rightarrow cua|kua\}$. This grammar is both visually ambiguous on draw, and visually ambiguous on intersection. It is visually ambiguous on draw since there exist two different sentences $w = arcrbdalbuca$ and $w_1 = arcdbraublca$ in $L(G_{\Sigma_{\phi} \approx \Pi})$, that are the description of different walks on the same drawn symbolic picture. It is visually ambiguous on intersection since there exist two different sentences $w = arcrbdalbuca$ and $w_2 = arcrbdalbukua$ in $L(G_{\Sigma_{\phi} \approx \Pi})$, that describe a walk on the drawn symbolic picture intersecting itself in a point in which they have different symbols associated. Fig. 10 shows the ambiguous drawn symbolic picture described by \mathbf{G} which is the result of applying the function $dspic$ on any of w , w_1 and w_2 .

The decidability of the visual ambiguity problem for a drawn symbolic picture derives from the following decidability properties which are straightforward consequences of the regularity of the set of consistent description language of a drawn symbolic picture (see Theorem 4.1) and from the closure property of context-free languages with regular sets:

Corollary 5.2 *Let G be a $\Sigma_\phi \approx \Pi$ -grammar, and dsp a drawn symbolic picture. The following problems are decidable:*

1. $L(G) \cap dspdes(dsp) \neq \emptyset$
i.e., $L(G)$ contains a description for the drawn symbolic picture dsp ;
2. $|L(G) \cap dspdes(dsp)| < \infty$
i.e., $L(G)$ contains a finite number of descriptions for the drawn symbolic picture dsp ;
3. $|L(G) \cap dspdes(dsp)| = 1$
i.e., $L(G)$ contains only one description for the drawn symbolic picture dsp .

As an immediate consequence of this Corollary, given a drawn symbolic picture dsp and an arbitrary $\Sigma_\phi \approx \Pi$ -grammar G it is possible to decide if dsp is described by only one sentence of G , i.e., if the description of dsp in G is visually ambiguous.

In a symbolic picture grammar the concept of ambiguity is a little more complex than in a DSP grammar. In this case, we distinguish one more case of visual ambiguity besides the two individuated for drawn symbolic pictures.

In fact, in a symbolic picture grammar there are three reasons for which the evaluation of $spic$ on different $\Sigma_\phi \approx \Pi_b$ -words may produce the same symbolic picture:

1. the $\Sigma_\phi \approx \Pi_b$ -words describe different drawn symbolic pictures for the same symbolic picture;
2. the $\Sigma_\phi \approx \Pi_b$ -words describe different walks on the same drawn symbolic picture of the symbolic picture;
3. the $\Sigma_\phi \approx \Pi_b$ -words describe the same walk on the drawn symbolic picture of the symbolic picture, and the walk intersects itself in a point. Due to the definition of $spic$ the $\Sigma_\phi \approx \Pi_b$ -words may present different symbols associated to that point yet producing the same picture.

In the first case, we have *visual ambiguity on different draws*, in the second one, *visual ambiguity on the same draw*, in the third one, finally, *visual ambiguity on intersections*.

Example 5.6 *Let $\mathbf{G} = \langle G_{\Sigma_\phi \approx \Pi_b}, spic \rangle$ be an SP grammar where $G_{\Sigma_\phi \approx \Pi_b} = \langle \{r_b, l_b, u_b, d_b\}, \{a, b, c, k, \phi\}, \{S, A, B, C\}, P, S \rangle$ and P is the set of productions:
 $\{S \rightarrow ar_b c A a B | ar_b c D, A \rightarrow r_b b d_b | d d_b b r_b, B \rightarrow l_b b u_b C | u_b b l_b C,$
 $C \rightarrow c u_b a | k u_b a, D \rightarrow u_b a E, E \rightarrow r_b \phi d_b b d_b a l_b b\}$.*

This grammar is an example of symbolic picture grammar that is visually ambiguous on different draws, visually ambiguous on the same draw, and visually ambiguous on

intersection. Indeed, it is visually ambiguous on different draws since there exists two different sentences $w = \mathbf{ar}_b\mathbf{cr}_b\mathbf{bd}_b\mathbf{al}_b\mathbf{bu}_b\mathbf{cu}_ba$ and $w_1 = \mathbf{ar}_b\mathbf{cu}_b\mathbf{ar}_b\mathbf{cd}_b\mathbf{bd}_b\mathbf{al}_b\mathbf{bb}$ generated by $G_{\Sigma_\phi \approx \Pi_b}$ that are the "descriptions" of different drawn symbolic pictures of the same symbolic picture. It is visually ambiguous on the same draw since there exist two different sentences w and $w_2 = \mathbf{ar}_b\mathbf{cd}_b\mathbf{br}_b\mathbf{au}_b\mathbf{bl}_b\mathbf{cu}_ba$ generated by $G_{\Sigma_\phi \approx \Pi_b}$ that are the description of different walks on the drawn symbolic picture corresponding to the symbolic picture. It is visually ambiguous on intersection since there exist two different sentences w and $w_3 = \mathbf{ar}_b\mathbf{cr}_b\mathbf{bd}_b\mathbf{al}_b\mathbf{bu}_b\mathbf{ku}_ba$ generated by $G_{\Sigma_\phi \approx \Pi_b}$ describing the same walk on the drawn symbolic picture of the symbolic picture intersecting itself in a point in which they have different symbols associated. Fig. 11 shows the symbolic picture generated by G which is the result of applying the function $spic$ to the strings w , w_1 , w_2 and w_3 .

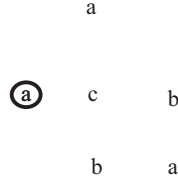


Figure 11: The symbolic picture under the SP grammar of Example 5.6

In a similar way to the case of drawn symbolic pictures, we can exploit the regularity of the k -description language of a symbolic picture (see Theorem 4.2) to prove some assertions about the ambiguity of symbolic picture grammars with respect to a given symbolic picture.

Corollary 5.3 *Let G be a $\Sigma_\phi \approx \Pi_b$ -grammar, and sp a symbolic picture. The following problems are decidable:*

1. $L(G) \cap spdes^k(sp) \neq \emptyset$ $(L(G) \cap Cspdes^k(sp) \neq \emptyset)$
i.e., $L(G)$ contains a (consistent) k -description for the symbolic picture sp ;
2. $|L(G) \cap spdes^k(sp)| < \infty$ $(|L(G) \cap Cspdes^k(sp)| < \infty)$
i.e., $L(G)$ contains a finite number of (consistent) k -descriptions for the symbolic picture sp ;
3. $|L(G) \cap spdes^k(sp)| = 1$ $(|L(G) \cap Cspdes^k(sp)| = 1)$
i.e., $L(G)$ contains only one (consistent) k -description for the symbolic picture sp .

6 Notes on the Classification of the Proposed Picture Models

Both the drawn symbolic picture and symbolic picture grammars are based on grammars whose sentences are strings alternating symbols with moves. The Chomsky

hierarchy can then be easily extended to these types of grammars. As an example, a DSP grammar $\langle G_{\Sigma_\phi \approx \Pi}, dspic \rangle$ is context-free iff $G_{\Sigma_\phi \approx \Pi}$ is context-free. A different type of classification can be devised if we look at a DSP (or SP) grammar as the interleaving of two string grammars: a Σ_ϕ -grammar and a Π -grammar (Π_b -grammar, resp.). In this case, we can characterize 16 types of DSP or SP grammars depending on whether the component grammars are regular, context-free, context-sensitive or type-0 grammars. To this aim we introduce the following definition:

Definition 6.1 *Let $G = \langle \Gamma, \Lambda, N, P, S \rangle$ be a $\Gamma \approx \Lambda$ -grammar and $A \subseteq (\Gamma \cup \Lambda)$ a subset of the terminal symbols. We define the string grammar $G_A = \langle A, N, P_A, S \rangle$ 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 .*

Example 6.1 *Let $G = \langle \Sigma_\phi = \{a, b, c, \phi\}, \Pi, N = \{S, A, B\}, P = \{^{[1]}S \rightarrow AdB, ^{[2]}A \rightarrow arA, ^{[3]}A \rightarrow c, ^{[4]}B \rightarrow Blb, ^{[5]}B \rightarrow c\}, S \rangle$ be a $\Sigma_\phi \approx \Pi$ -grammar. By projecting G on the sets Π and Σ_ϕ , respectively, we obtain the grammars G_Π and G_{Σ_ϕ} , where:*

- $G_\Pi = \langle \Pi = \{r, l, u, d\}, N = \{S, A, B\}, P = \{^{[1]}S \rightarrow AdB, ^{[2]}A \rightarrow rA, ^{[3]}A \rightarrow \epsilon, ^{[4]}B \rightarrow Bl, ^{[5]}B \rightarrow \epsilon\}, S \rangle;$
- $G_{\Sigma_\phi} = \langle \Sigma_\phi = \{a, b, c, f\}, N = \{S, A, B\}, P = \{^{[1]}S \rightarrow AB, ^{[2]}A \rightarrow aA, ^{[3]}A \rightarrow c, ^{[4]}B \rightarrow Bb, ^{[5]}B \rightarrow c\}, S \rangle.$

6.1 Classification of DSP and SP Grammars

The classification of DSP and SP grammars depends on the traditional Chomsky classification of the two string grammars obtained by projecting their corresponding alternating grammar on the set Π and Σ_ϕ . In this paragraph, without loss of generality, we ignore the difference between the sets Π and Π_b .

It can be shown that if a $\Sigma_\phi \approx \Pi$ -grammar is strictly of type i ($i = 0, 1, 2, 3$) according to the Chomsky hierarchy, then one of its two projections on Σ_ϕ and on Π must be strictly of type i , and the other of type j , with $i \leq j$.

Example 6.2 *Let $G = \langle \Sigma = \{a, b, c\}, \Pi, N = \{S, A, B\}, P = \{S \rightarrow brA\phi, A \rightarrow auB, B \rightarrow brA|\phi d\}, S \rangle$ be a context-free $\Sigma_\phi \approx \Pi$ -grammar. By projecting G on Σ_ϕ and on Π , we obtain:*

- $G_{\Sigma_\phi} = \langle \Sigma, N, P_{\Sigma_\phi} = \{S \rightarrow brA\phi, A \rightarrow aB, B \rightarrow brA|\phi\}, S \rangle.$
- $G_\Pi = \langle \Pi, N, P_\Pi = \{S \rightarrow rA, A \rightarrow uB, B \rightarrow rA|d\}, S \rangle;$

Note that G_{Σ_ϕ} is context-free but G_Π is also regular.

The classification of a string grammar is usually represented in a one-dimensional space. The classification of DSP and SP grammars may be represented in a two-dimensional Cartesian plane whose axes represent the Chomsky classification of the projections on the sets Σ_ϕ and Π , respectively.

Reg	E	E	E	E
Cs	S	S	S	E
Cf	F	F	S	E
Reg	R	F	S	E
	Reg	Cf	Cs	Re

Figure 12: Classification of DSP and SP grammars

Fig. 12 states that the projections of a regular DSP (or SP) grammar on the sets Σ_ϕ and Π (or Π_b) are both regular: so, the regular picture grammars are in the zone ‘R’ of the plane. The projections of strictly context-free picture grammars may be regular or context-free, with the constraint that at least one of them is strictly context-free: they are disposed in one of the three zones ‘F’ of the plane, corresponding to the type of its projections. Analogously, the context-sensitive picture grammars are in the five zones ‘S’, and the recursive enumerable ones are in the seven zones ‘E’. By knowing the type of the two projected grammars, then, it is easy to derive the type of the projecting grammar.

6.2 Classification of Picture Languages

A *picture language* is a regular (resp. context-free, context-sensitive, recursive enumerable) picture language if there exists a regular (resp. context-free, context-sensitive, recursive enumerable) picture grammar generating it. The classification of picture languages is obtained analogously to picture grammars ones. It also depends on the following theorem proved in [31] and reported here for sake of completeness.

Theorem 6.1 *The class of context-sensitive drawn picture languages is equal to the class of recursive enumerable ones.*

By this theorem, the axis of classification of projection of a picture languages on Π is split in only three sections. This splitting produces twelve zones in the plane in which we pose regular, context-free, context-sensitive and recursive enumerable picture languages (See Fig. 13).

The projections of the regular picture languages (i.e. the languages generated by regular picture grammars) are both regular: so, the regular picture grammars are in the zone ‘R’ of the plane. The projection of context-free picture languages may be regular or context-free, with the constraint that at least one of them is context-free: they are disposed in one of three zones ‘F’ of the plane, correspondingly to the type of its projections. Analogously, the context-sensitive picture languages are in the five zones ‘S’. By theorem 6.1, the recursive enumerable ones are in the three zones ‘E’.

The classification proposed in this last section suggests the use of a generator on Π and a recognizer on Σ for the recognition of symbolic picture languages. Given

primitives which could be combined to form all ‘interesting’ patterns (e.g. Shaw). In the second case a grammar was used to specify the rules for combining the primitives.

Another important chain code model is the description method used in ‘turtle geometry’ [1], where the trace left by a turtle (or robot) while moving in the plane is interpreted as a picture.

Good surveys for picture description models, some of which are similar to the chain code scheme, can also be found in [35] for picture language automata and grammars, and in [12] for general syntactic pattern recognition methods.

Subsequently, some efforts toward classifying the complexity of picture languages [31, 40] have simplified the chain codes to strings over a four letters alphabet $\{\mathbf{r}, \mathbf{l}, \mathbf{u}, \mathbf{d}\}$ to represent the movements *right*, *left*, *up* and *down*, respectively. One of the most elegant and successful four-letters chain code methods for working with line-drawing was introduced in 1982 by Maurer, Rozenberg and Welzl [31]. They define a drawn picture as a connected set of axis parallel unit lines from the Cartesian plane considered as a square grid, with the specification of its start and end points. A picture description is a word over $\{\mathbf{r}, \mathbf{l}, \mathbf{u}, \mathbf{d}\}$ describing the picture by the graphical representation of the movement sequence.

Many authors have continued the study of chain code pictures (see for example [2], [3], [17], [18], [27], [38], [39]). By changing the semantics of the alphabet Π , in [27] a description of pictures made of *pixels* (*unit squares*) instead of segments has been proposed. Moreover, additional features allow to describe different *colours* for the move letters, including the “invisible” one (by the “pen up”). A more realistic description is achieved this way, but also a more difficult one. As a matter of fact, problems like membership that are decidable for context-free picture description with “pen down” turn out to be undecidable in the case of “pen up” [20].

Languages of chain-encoded pictures have been studied intensively also from a computational point of view [24],[25],[26],[31],[40].

More recently, new chain code models have been proposed for shapes composed of regular cells and for representing *3D curves* (see [5], [6]). In particular, in [5] the author represents *3D discrete curves* composed of constant *straight-line segments*. Two contiguous straight-line segments define a *direction change* and two direction changes define a *chain element*. Such discrete curves are isolated from the real world and are obtained by a pre-processing. Moreover, they are described by only five orthogonal direction changes. The use of relative direction changes allow us to consider curve descriptions invariant under translation and rotation, and optionally, under mirroring transformation (see [5]).

Other interesting approaches proposed for the descriptions of the pictures are: the *two dimensional string languages* [14, 15, 16, 21, 22, 41] and the *shape grammars* [4]. In the context of two dimensional languages pictures are conceived as sets of rectangular arrays of symbols and represent a generalization of the notion of string languages to picture languages. In [14] the notion of recognizability of a set of pictures in terms of *tiling systems* has been introduced. In particular, the authors define the notion of a local picture language by giving a set of authorized 2x2 tiles over $\Sigma \cup \{\#\}$ where $\#$ is a boundary symbol that surrounds the pictures. They define the class of recognizable picture languages (generated by finite state machines) which is of interest since

it has been proved that the following families of two dimensional languages coincide: languages recognized by on-line tessellation automata [21, 22], languages expressed by formulas of existential monadic second-order logic [16, 41], languages recognized by finite tiling systems [14, 15], languages corresponding to regular expressions of special type [15].

A *shape grammar* represents a *pictorial computability model* which operates on polygons consisting of unit square orthogonally connected. Pixels are the elementary objects and by *orthogonally adjoining* them without overlapping polyomonoes are obtained. Shape grammars combine the approach of deriving linear descriptions (strings) of line segments [11, 20, 31] and connected set of pixel [27, 37] (that are the typical atomic structures used) and the approach of characterize pictorial languages in terms of 2D [15, 35]. Moreover, in [4] the authors characterize the recursively enumerable languages in this framework.

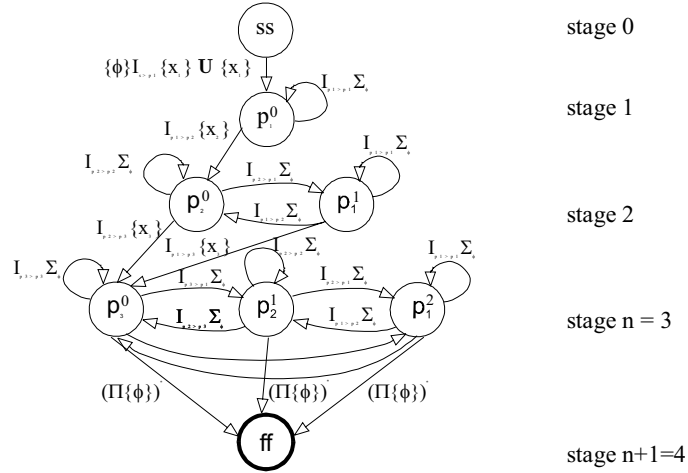
8 Final remarks

The models of drawn symbolic pictures and symbolic pictures have turned out to provide a simple, intuitive and useful formalism to describe more complex pictures. Nevertheless the introduction of symbols requires new issues to be addressed. In this paper we have investigated several of these issues by providing descriptive and generative models. In particular, we have focused on a string-based description for both symbolic pictures and drawn symbolic pictures. By using this kind of representation, we have described a (drawn) symbolic picture grammar in terms of a string grammar, the grammar for alternating words. A notion of consistency has been provided for the alternating words and we have given some decidability properties for symbolic picture and drawn symbolic picture languages. In the future, we intend to investigate other decidability problems such as the membership problem for picture languages. Moreover, we plan to devise a parsing algorithm for (drawn) symbolic picture languages. Such an algorithm could exploit a combination of top down and bottom up parsing for string grammars.

In this appendix we prove Theorem 4.3. In particular, we provide the proof only for $spdes(sp)$ (the proof for the set $Cspdes(sp)$ is analogous). Let $|P| = n$, $perm = \langle p_1, p_2, \dots, p_n \rangle$ be a permutation of the positions in P and $\delta_\phi(p_i) = x_i \in \Sigma$ for $i = 1, \dots, n$. Any word w describing sp and “touching” the symbols in sp in the order defined by $perm$ must begin with the symbol ϕ if $s \notin P$ or with x_1 , otherwise. In the first case, w may “reach” position p_1 and present the symbol x_1 only after an invisible path from s to p_1 ; in the second case p_1 must be s . Once in p_1 , w may proceed with zero or more invisible paths from p_1 to p_1 (each followed by any symbol in $\Sigma \cup \{\phi\}$), and a path from $I_{p_1 > p_1}$ followed by x_2 .

At this point w is any string described by the following set expression $\{x_1\} (I_{p_1 > p_1}(\Sigma \cup \{\phi\})) * I_{p_1 > p_2} \{x_2\}$ with prefix $\{\phi\} I_{s > p_1}$ in the case $s \notin P$. In order to better describe the complete composition of w we make use of a multi-stage labeled graph G_{sp}^{perm} defined as follows.

The set of nodes of G_{sp}^{perm} is partitioned in $n + 2$ stages:

Figure 14: An instance of graph G_{sp}^{perm} with $n = 3$

stage 0: contains the node ss ;

stage k ($1 \leq k \leq n$): contains the set of k nodes $\{p_k^0, p_{k-1}^1, \dots, p_1^{k-1}\}$;

stage $n + 1$: contains the node ff (the *final node*).

Each stage k ($1 \leq k \leq n$) is a clique on the nodes $p_k^0, p_{k-1}^1, \dots, p_1^{k-1}$. Each edge (p_t^v, p_x^y) in the clique is labeled with the set expression $I_{pt > px} \Sigma_\phi$ indicating the Cartesian product between the set of all the invisible paths from p_t to p_x and the set $\Sigma_\phi = \Sigma_\phi \cup \{\phi\}$. Each stage i ($1 \leq i \leq n$) is connected to stage $i + 1$ through labeled edges as follows:

- if $i = 0$ then the node ss is connected to node p_1^0 through the edge (ss, p_1^0) labeled with the set expression $\{\phi\}I_{s > p_1} \cup \{x_1\}$, indicating the set of strings containing the string "x₁" and all the strings starting with ϕ , continuing with an invisible path from s to p_1 and ending with x_1 .
- if $1 \leq i \leq n - 1$ then all the nodes p_t^v at stage i are connected to the node at stage $i + 1$ through an edge (p_t^v, p_{i+1}^0) with label $I_{pt > p(i+1)} \{x_{i+1}\}$ indicating the set of all the strings starting with an invisible path from p_t to p_{i+1} and ending with the symbol x_{i+1} .
- if $i = n$ then all the nodes p_t^v at stage n are connected to the node ff at stage $n + 1$ through an edge (p_t^v, ff) with label $(\Pi\{\phi\})^*$ indicating the set of all the invisible paths followed by a ϕ .

Fig. 14 shows an example of a graph G_{sp}^{perm} where $n = 3$. The node ff is in bold to indicate that it is the final node. For sake of clarity not all the edge labels are shown.

Given the following definitions:

- $gpath(G_{sp}^{perm})$: the set of all the graph paths in G_{sp}^{perm} starting from node ss and ending at the final node ff ,
- $label(\alpha)$: the set expression obtained by concatenating all the labels met during the visit of the graph path $\alpha \in gpath(G_{sp}^{perm})$,
- $L(\alpha)$: the set of words described by the set expression $label(\alpha)$,

it is easy to verify that $L_{sp}^{perm} = \bigcup_{\alpha \in gpath(G_{sp}^{perm})} L(\alpha)$ is the set of all the words w describing sp and “touching” the symbols of sp in the order defined by $perm$. Moreover, by considering all the possible permutations $perm$ we can provide a new equivalent definition for the description language of sp . In fact, we can state that $spdes(sp) = \bigcup_{perm \in Permut(P)} L_{sp}^{perm}$ where $permut(P)$ is

- - the set of all the permutations of P starting with s , if $s \in P$, or
- - the set of all the permutations of P preceded by s , if $s \notin P$.

By noticing that each expression $I_{pi>pj}$ describes a context-sensitive language and that each G_{sp}^{perm} contains a finite set of nodes, it is easy to prove that $spdes(sp)$ is a context-sensitive $\Sigma_\phi \approx \Pi_b$ -language for any symbolic picture sp .

Example .1 *Let us consider the following symbolic picture*

$$sp = \langle \{(0, 0), (1, 0), (2, -1)\}, s, \{a, b, c\}, \delta_\phi \rangle$$

with $\delta_\phi(0, 0) = a$, $\delta_\phi(1, 0) = b$ and $\delta_\phi(2, -1) = c$. Moreover, let

$$perm = \langle p_1, p_2, p_3 \rangle = \langle (0, 0), (2, -1), (1, 0) \rangle$$

and, hence, $\langle x_1, x_2, x_3 \rangle = \langle a, c, b \rangle$. The following $\Sigma_\phi \approx \Pi_b$ -word describing sp

$$w = a \underline{u}_b \phi \underline{l}_b \phi \underline{d}_b \phi \underline{r}_b \underline{b} \underline{d}_b \phi \underline{r}_b \phi \underline{r}_b \phi \underline{c} \underline{u}_b \phi \underline{l}_b \phi \underline{l}_b \underline{c} \underline{r}_b \underline{b} \underline{r}_b \phi$$

contains 5 invisible paths (shown as underlined in w) that are, respectively, from left to right, $I_{p1>p1}$, $I_{p1>p2}$, $I_{p2>p1}$, $I_{p1>p3}$ and a final invisible path.

Referring to the graph G_{sp}^{perm} of Fig. 14, it can be verified that w produces the graph path

$$\alpha = ssp_1^0 p_1^0 p_2^0 p_1^1 p_3^0 ff$$

and that it belongs to the set $L(\alpha)$ described by the set expression $label(\alpha)$:

$$(\{\phi\}_{I_{s>p1}} \cup \{a\})(I_{p1>p1} \Sigma_\phi)(I_{p1>p2}\{c\})(I_{p2>p1} \Sigma_\phi)(I_{p1>p3}\{b\})(\Pi\{\phi\}).$$

If each language described by an expression $I_{pi>pj}$ is made to be regular, then each labeled graph G_{sp}^{perm} is easily translated in a finite automaton and then, since the number of permutations $perm$ is finite, the whole language is regular. There are at least two ways to make each $I_{pi>pj}$ a regular set:

- the number of positions that each string described by $I_{pi>pj}$ can span on is limited by a pre-defined integer; this brings us back to Theorem 4.2, and
- the set Π_b does not contain two moves where one is the inverse of the other, i.e., for instance, $\Pi_b = \{r_b, d_b\}$ or $\Pi_b = \{r_b\}$.

References

- [1] H. Abelson, A. Di Sessa, “Turtle Geometry”, *The MIT Press, Cambridge, MA*, 1986.
- [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, pp. 49-56, 1993.
- [4] P. Bottoni, G. Mauri, P. Mussio, G. Păun, “Computing with Shapes”, *Journal of Visual Languages and Computing*, 12, 4, pp. 601-626, 2001.
- [5] E. Bribiesca, “A New Chain Code”, *Pattern Recognition*, 32, 2, pp. 235-251, 1999.
- [6] E. Bribiesca, “A Chain Code for Representing 3D Curves”, *Pattern Recognition*, 33, 5, pp. 755-765, 2000.
- [7] J. Feder, “Languages of Encoded Line Patterns”, *Information and Control*, 13, pp. 230-244, 1968.
- [8] J. Feder, “Plex Languages”, *Information Sciences*, 3, pp. 225-241, 1971.
- [9] H. Freeman, “On Encoding Arbitrary Geometric Configuration”, *IRE Trans Electroning Computation*, 10, pp. 260-268, 1961.
- [10] H. Freeman, “On the Digital Computer Classification of Geometric Line Patterns”, *Proc. 18th Nat. Elect. Conf.*, pp. 312-334, 1962.
- [11] H. Freeman, “Computer Processing of Line-Drawing Images”, *Computer Surveys*, 6, pp. 57-97, 1974.
- [12] K.S. Fu, “Syntactic Pattern Recognition and Applications”, *Prentice-Hall, Inc. Englewood Cliffs, N.J.*, 1982.
- [13] E.J. Golin, S.P. Reiss, “The Specification of Visual Language Syntax”, *Journal of Visual Languages and Computing* 1, pp. 141-157, 1990.
- [14] D. Giammaresi, A. Restivo, “Recognizable Picture Languages”, *Proc. 1st Internat. Colloq. on Parallel Image Processing, Internat. J. Pattern Recognition Artif. Intell.*, 6, M. Nivat, A. Saoudi and P.S.P. Wand, eds., pp. 231-256, 1992.
- [15] D. Giammaresi, A. Restivo, “Two dimensional languages”, Chapter 4 in Volumes 3 of G. Rozenberg, A. Salomaa, Eds., *Handbook of Formal Languages*, Springer-Verlag, Heidelberg, 1997.
- [16] D. Giammaresi, A. Restivo, S. Seibert, W. Tomas, “Monadic second-order logic over rectangular pictures and recognizability by tiling systems”, *Information and Computation*, 125, pp. 32-45, 1996.
- [17] R. Gutbrod, “A Transformation System for Generating Description Languages of Chain Code Pictures”, *Theoretical Computer Science*, 68, pp. 239-252, 1989.
- [18] F. Hinz, “Classes of Picture Languages that Cannot be Distinguished in the Chain Code Concept Deletion of Redundant Retreats”, *Procs. STACS’89, Lecture Notes in Computer Sciences*, 349, Springer Berlin, pp. 132-143, 1989.
- [19] J.E. Hopcroft, J.D. Ullman, *Introduction to Automata Theory, Languages, and Computation*, Addison-Wesley, Reading, MA, 1979.

- [20] F. Hinz, E. Welzl, "Regular chain code picture languages with invisible lines", *Tech. report*, 252, IIG, Tech. Univ. Graz, Austria, 1988.
- [21] K. Inoue, A. Nakamura, "Some properties of two-dimensional on-line tessellation acceptors", *Information Science*, 13, pp. 95-121, 1977.
- [22] K. Inoue, I. Takanami, "A characterization of recognizable picture languages", *Proc. 2nd Internat. Colloq. on Parallel Image Processing, Lecture Notes in Computer Science*, 654, Springer Berlin, pp. 133-143, 1992.
- [23] C. Kim, "Picture Iteration and Picture Ambiguity", *Journal of Computer and System Sciences*, 40, pp. 289-306, 1990.
- [24] C. Kim, "Complexity and Decidability for Restricted Class of Picture Languages", *Theoretical Computer Science*, 73, pp. 295-311, 1990.
- [25] C. Kim, I.H. Sudborough, "The Membership and Equivalence Problems for Picture Languages", *Theoretical Computer Science*, 52, pp. 177-191, 1987.
- [26] C. Kim, I.H. Sudborough, "On Reversal Bounded Picture Languages", *Theoretical Computer Science*, 104, pp. 185-206, 1992.
- [27] M. Latteux, D. Robilliard, D. Simplot, "Figures composés de pixels et monode inversif", *Bullettin of Belgian Mathematical Society*, 4, pp. 89-111, 1997.
- [28] K. Marriott, "Constraint Multiset Grammars", *Procs. of IEEE Symposium on Visual Languages*, St. Louis, Missouri, pp. 118-125, 1994.
- [29] G. Masini, R. Mohr, "MIRABELLE, a system for structural analysis of drawings", *Pattern Recognition*, 16, 4, pp. 363-372, 1982.
- [30] R. Mohr, "Precompilation of syntactical descriptions and knowledge directed analysis of patterns", *Pattern Recognition*, 19, 4, pp. 255-266, 1986.
- [31] H.A. Maurer, G. Rozenberg, E. Welzl, "Using String Languages to Describe Picture Languages", *Information and Control*, 54, pp. 155-185, 1982.
- [32] R. Narasimhan, "Syntax-Direct Interpretation of Classes of Pictures", *Comm. ACM*, 9, pp. 166-173, 1966.
- [33] T. Pavlidis, "Analysis of Set Patterns", *Pattern Recognition*, 1, pp. 165-178, 1968.
- [34] J. Rekers, A. Schurr, "A Graph Grammar Approach to Graphical Parsing", *Procs. of the 1995 IEEE Symposium on Visual Languages*, Darmstadt, Germany, 1995.
- [35] A. Rosenfeld, *Picture Languages: Formal Model of Picture Recognition*, Academic Press, New York, 1979.
- [36] A. Salomaa, *Formal Languages*, Academic Press, London, 1973.
- [37] A.C. Shaw, "A Formal Picture Description Scheme as a Basis for Picture Processing Systems", *Information and Control*, 14, pp. 9-52, 1969.
- [38] P. Séébold, K. Slowinski, "The Shortest Way to Draw a Connect Picture", *Computer Graphics Forum*, 10, pp. 319-327, 1991.
- [39] K. Slowinski, "Picture Words with Invisible Lines", *Theoretical Computer Science*, 108, pp. 357-363, 1993.
- [40] I.H. Sudborough, E. Welzl, "Complexity and Decidability for Chain Code Picture Languages", *Theoretical Computer Science*, 36, pp. 173-202, 1985.

- [41] W. Tomas, “On logics, tilings, and automata”, *Proc. 18th Internat. Colloq. on Automata, Languages and Programming, Lecture Notes in Computer Science*, 510, Springer Berlin, pp. 441-453, 1991.