Discrete Mathematics and Theoretical Computer Science DMTCS vol. **11:2**, 2009, 1–14

Directed figure codes are decidable

Michał Kolarz and Włodzimierz Moczurad

Institute of Computer Science, Jagiellonian University, ul. Łojasiewicza 6, 30-348 Kraków, Poland

received December 20, 2008, *revised May 5, 2009*, *accepted May 22, 2009*.

Two-dimensional structures of various kinds can be viewed as generalizations of words. Codicity verification and the defect effect, important properties related to word codes, are studied also in this context. Unfortunately, both are lost in the case of two common structures, polyominoes and figures. We consider directed figures defined as labelled polyominoes with designated start and end points, equipped with catenation operation that uses a merging function to resolve possible conflicts. We prove that in this setting verification whether a given finite set of directed figures is a code is decidable and we give a constructive algorithm. We also clarify the status of the defect effect for directed figures.

Keywords: directed figures, variable-length codes, codicity verification, Sardinas-Patterson algorithm, defect effect

1 Introduction

Codes, *i.e.*, subsets X of a monoid such that every product of the elements decomposes uniquely over X, are a common object of study. Standard variable-length word codes (subsets of a free monoid Σ^*, where Σ is an alphabet) are the best-known among them (see *e.g.* the classic monograph Berstel and Perrin (1985)), with properties that have become folklore. This includes the defect effect and the decidability of codicity. Various authors have extended codes to other structures like trees in Mantaci and Restivo (2001); Karhumäki and Mantaci (1999) and two-dimensional structures of various kinds like polyominoes in Aigrain and Beauquier (1995); Beauquier and Nivat (2003). Whilst the extension to trees preserves the above properties, both are lost in the case of polyominoes. Similar problems arise when considering figures defined as labelled polyominoes. Only a limited version of the defect theorem holds in this case and codicity is decidable for very restricted classes of figures; see a comprehensive study in Harju and Karhumäki (2004) and also Moczurad (2000, 2007).

Many authors, like Costagliola et al. (2003, 2005), study other kinds of two-dimensional structures. This interest is driven by obvious applications of picture models in diverse disciplines. Moreover, pictures can be seen as a further natural extension of words. Thus, in the present paper we are interested in codicity testing and the defect effect for yet another kind of pictures.

We consider directed figures defined as labelled polyominoes with designated start and end points. This setting is similar to one of the models, symbolic pixel pictures, described in Costagliola et al. (2005) and admits a natural definition of catenation. We use the attribute "directed" to emphasize the way figures are catenated; this should not be confused with the meaning of "directed" in *e.g.* directed polyominoes.

1365–8050 © 2009 Discrete Mathematics and Theoretical Computer Science (DMTCS), Nancy, France

We prove that verification whether a given finite set of directed figures is a code is decidable and we give a constructive algorithm (constructive in the sense that it finds a double factorization of a figure for a non-code). This is a significant change in comparison to previously mentioned picture models. On the other hand, we give several examples to disprove the defect theorem in this case.

Section 2 of the paper defines directed figures and related operations. Then, in Section 3, we give the necessary condition for a set of figures to be a code. We then proceed to the main result of the paper, the decidability of codicity verification, in Section 4. Proof of the main theorem leads to an algorithm which is presented in Section 5. Finally, Section 6 clarifies the status of the defect effect for directed figures. We conclude with some observations and directions for possible further research.

2 Preliminaries

Throughout the paper Σ is a finite, non-empty alphabet.

A *translation* in $\mathbb{Z}^2$ by vector $u = (u_x, u_y) \in \mathbb{Z}^2$ will be denoted by τ_u,

$$\tau_u : \mathbb{Z}^2 \ni (x, y) \mapsto (x + u_x, y + u_y) \in \mathbb{Z}^2.$$

For a set $V \subseteq \mathbb{Z}^2$ and an arbitrary function $f : V \to \Sigma$ it obviously induces

$$\begin{aligned} \tau_u : P(\mathbb{Z}^2) \ni V &\mapsto \{\tau_u(v) \mid v \in V\} \in P(\mathbb{Z}^2), \\ \tau_u : \Sigma^V \ni f &\mapsto f \circ \tau_{-u} \in \Sigma^{\tau_u(V)}. \end{aligned}$$

For two points $u = (u_x, u_y), v = (v_x, v_y) \in \mathbb{Z}^2$ and a subset $U \subseteq \mathbb{Z}^2, U \neq \emptyset$ we define

$$\begin{aligned} \text{dist}(u, v) &= |u_x - v_x| + |u_y - v_y|, \\ \text{dist}(U, v) &= \min_{u \in U} \{\text{dist}(u, v)\}, \\ \Gamma(U) &= \{v \in \mathbb{Z}^2 \mid \text{dist}(U, v) \leq 1\}. \end{aligned}$$

A set $U \subseteq \mathbb{Z}^2$ is *connected* if for any $x, y \in U$ there exists a sequence $x = x_1, x_2, \ldots, x_n, x_{n+1} = y$ contained in U such that $\text{dist}(x_i, x_{i+1}) = 1$ for $i \in \{1, \ldots, n\}$.

Definition 1 (Directed figure) *Let $D \subseteq \mathbb{Z}^2$ be finite and connected, $b \in D, e \in \Gamma(D)$ and $l : D \to \Sigma$. A quadruple $f = (D, b, e, l)$ is called a* directed figure *(over Σ) with*

domain	$\text{dom}(f)$	=	D,
start point	$\text{begin}(f)$	=	b,
end point	$\text{end}(f)$	=	e,
labelling function	$\text{label}(f)$	=	l.

Translation vector *of f is defined as* $\text{tran}(f) = \text{end}(f) - \text{begin}(f)$. *Additionally, we define the* empty directed figure *ε as $(\emptyset, (0, 0), (0, 0), \{\})$, where $\{\}$ denotes a function with an empty domain.*

The set of all directed figures over Σ is denoted by $\Sigma^\diamond$. Two directed figures x, y are *equal* if there exists $u \in \mathbb{Z}^2$ such that

$$y = (\tau_u(\text{dom}(x)), \tau_u(\text{begin}(x)), \tau_u(\text{end}(x)), \tau_u(\text{label}(x))).$$

Thus, we actually consider figures up to a translation.

Example 1 *A directed figure and its graphical representation. Each point of the domain, (x, y), is represented by a unit square in $\mathbb{R}^2$ with bottom left corner in (x, y). A circle marks the start point and a diamond marks the end point of the figure. However, since we consider figures up to a translation, we do not mark the coordinates.*

$$(\{(0,0),(1,0),(1,1)\},(0,0),(1,2),\{(0,0)\mapsto a,(1,0)\mapsto b,(1,1)\mapsto c\})$$

Definition 2 (Catenation) *Let $x = (D_x, b_x, e_x, l_x)$ and $y = (D_y, b_y, e_y, l_y)$ be directed figures.* Catenation *of x and y with respect to a* merging function $m : \Sigma \times \Sigma \to \Sigma$ *is defined as*

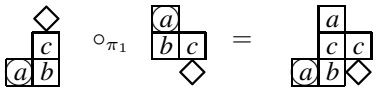
$$x \circ_m y = (D_x \cup \tau_{x_e - y_b}(D_y), b_x, \tau_{x_e - y_b}(e_y), l),$$

where

$$l(z) = \begin{cases} l_x(z) & \text{for } z \in D_x \setminus \tau_{x_e-y_b}(D_y), \\ \tau_{x_e-y_b}(l_y)(z) & \text{for } z \in \tau_{x_e-y_b}(D_y) \setminus D_x, \\ m(l_x(z), \tau_{x_e-y_b}(l_y)(z)) & \text{for } z \in D_x \cap \tau_{x_e-y_b}(D_y). \end{cases}$$

When m is fixed, we will write $x \circ y$, $x \cdot y$ or simply xy instead of $x \circ_m y$.

Example 2



where π_1 denotes projection on the first argument.

Observation 1 *$\Sigma_m^\diamond = (\Sigma^\diamond, \circ_m)$ is a monoid if and only if m is associative.*

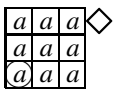
From now on let m be an arbitrary associative merging function. We will use the notation $\Sigma_m^\diamond$ for both the monoid and the set of directed figures itself. Abusing this notation, we will also write $X_m^\diamond$ to denote the set of all figures that can be composed by $\circ_m$ catenation from figures in $X \subseteq \Sigma_m^\diamond$.

Observation 2 *Monoid $\Sigma_m^\diamond$ is never free, since its basis must contain "unit figures" (figures of the type*

$$a_E = \boxed{a}\diamond, a_N = \begin{array}{c}\diamond\\ \boxed{a}\end{array}, a_W = \diamond\boxed{a}, a_S = \begin{array}{c}\boxed{a}\\ \diamond\end{array},$$

for all $a \in \Sigma$), contradicting the freeness.

Example 3 *Regardless of the merging function, the following figure can be decomposed as e.g. $a_N a_N a_E a_S a_S a_E a_N a_N a_E$ and $a_E a_E a_N a_W a_W a_N a_E a_E a_E$.*



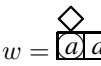
3 Necessary condition for codicity

Definition 3 (Code) *$X \subseteq \Sigma_m^\diamond$ is a* code *if for any $x_1, \ldots, x_k, y_1, \ldots, y_l \in X$ the equality*

$$x_1 \circ_m \ldots \circ_m x_k = y_1 \circ_m \ldots \circ_m y_l$$

implies $k = l$ and $x_i = y_i$ for each $i \in \{1, \ldots, k\}$.

Example 4 *It is clear that $X = \{$* *$\}$ $\subseteq \{a\}_{\pi_1}^\diamond$ is a code because the figures are directed in the plane.*

Example 5 *$X = \{w =$* *, $x =$* *, $y =$* *, $z =$*  *$\}$ $\subseteq \{a\}_{\pi_1}^\diamond$ is not a code since*

$$wx = yz = \text{ }.$$

Theorem 1 (Necessary condition) *Let $X = \{x_1, \ldots, x_n\} \subseteq \Sigma_m^\diamond$. If there exist $\alpha_{i_1}, \ldots, \alpha_{i_k} \in \mathbb{Z}_+$, where $\{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$, such that*

$$\sum_{j=1}^{k} \alpha_{i_j} \text{tran}(x_{i_j}) = (0, 0),$$

then X is not a code.

Proof: Let

$$x = \underbrace{x_{i_1} \cdots x_{i_1}}_{\alpha_{i_1}} \underbrace{x_{i_2} \cdots x_{i_2}}_{\alpha_{i_2}} \cdots \underbrace{x_{i_k} \cdots x_{i_k}}_{\alpha_{i_k}}.$$

Now consider the following set of directed figures:

$$Y = \{x^i\}_{i=1}^\infty.$$

Since $\text{tran}(x) = (0, 0)$, every element of Y has the same domain. There is only a finite number of possible labellings of this domain, which implies that regardless of the merging function and labelling of x, there exist $a, b \in \mathbb{N}$, $a \neq b$ such that $x^a = x^b$. Hence X is not a code; *cf.* Examples 9 and 10. □

Observation 3 *Let $v_1, \ldots, v_k \in \mathbb{Z}^2$. Following conditions are equivalent ($\cdot$ denotes the usual dot product):*

1. *$\sum_i \alpha_i v_i \neq 0$ for all $\alpha_1, \ldots, \alpha_k \in \mathbb{Z}_+$,*
2. *there exists $v \in \mathbb{Z}^2$ such that $v \cdot v_i > 0$ for all i, i.e., there exists a line passing through $(0, 0)$ such that all v_i's lie on one side of it.*

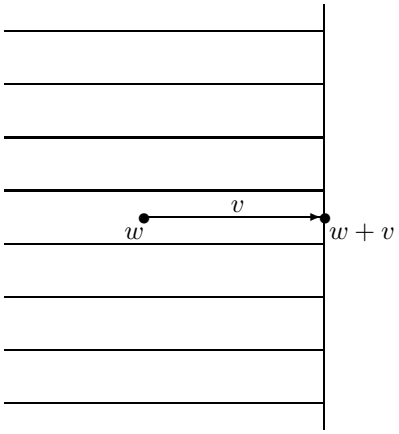


Fig. 1: A half-plane $HP(v, w)$.

4 Decidability of verification

In this section we prove that testing whether a given set of figures is a code is decidable. We begin with observations that allow us to construct a "bounding area" for figures. We then proceed with properties that imply finiteness of possible configuration sets and, consequently, decidability of the problem in question.

Let $X = \{x_1, \ldots, x_n\} \subseteq \Sigma_m^\diamond$. Either there exists a vector $\tau_E \in \mathbb{Z}^2$ such that for all $x \in X$

$$\tau_E \cdot \text{tran}(x) > 0$$

or, by Theorem 1, X is not a code. If τ_E exists, we can take it to be long enough so that for all $x \in X$

$$\text{dom}(x) \cup \{\text{end}(x)\} \subseteq HP(\tau_E, \text{begin}(x)),$$

where, for given $v, w \in \mathbb{R}^2$, $HP(v, w)$ denotes a half-plane $\{u \in \mathbb{Z}^2 \mid v \cdot (u - (w + v)) \leq 0\}$.

Observe that this implies

$$\text{dom}(x) \cup \{\text{end}(x)\} \subseteq HP(\tau_E, \text{end}(x))$$

for all $x \in X$. Without loss of generality we can assume

$$\angle(R_{-\frac{\pi}{2}}(\tau_E), \text{tran}(x_1)) \leq \angle(R_{-\frac{\pi}{2}}(\tau_E), \text{tran}(x_2)) \leq \ldots \leq \angle(R_{-\frac{\pi}{2}}(\tau_E), \text{tran}(x_n))$$

(where R_ϕ denotes a rotation by ϕ and $\angle$ is the angle spanned by two vectors) and $\text{begin}(x) = (0, 0)$ for all $x \in X$.

Now choose constants $r_S, r_N, r_W > 0$ such that the vectors

$$\begin{aligned} \tau_N &= r_N R_{\frac{\pi}{2}}(\text{tran}(x_n)), \\ \tau_W &= -r_W \tau_E, \\ \tau_S &= r_S R_{-\frac{\pi}{2}}(\text{tran}(x_1)) \end{aligned}$$

define a "bounding area" for figures in X, *i.e.*, for all $x \in X$

$$\text{dom}(x) \cup \{\text{end}(x)\} \subseteq \bigcap_{\tau \in \{\tau_N, \tau_W, \tau_S\}} \{HP(\tau, \text{begin}(x))\}.$$

Figure 2 shows half-planes $HP(\tau, \text{begin}(x))$ for $\tau \in \{\tau_E, \tau_N, \tau_W, \tau_S\}$.

For $x \in X_m^\diamond$ define

$$\begin{aligned} CE^+(x) &= HP(\tau_S, \text{end}(x)) \cap HP(\tau_N, \text{end}(x)) \cap HP(\tau_W, \text{end}(x)), \\ CE^-(x) &= \mathbb{Z}^2 \setminus CE^+(\text{end}(x)), \\ CW^+(x) &= \bigcup_v \{v + (CE^+(\text{end}(x)) \cap HP(\tau_E, \text{end}(x)))\}, \\ CW^-(x) &= \mathbb{Z}^2 \setminus CW^+(\text{end}(x)), \end{aligned}$$

where the union in the definition of $CW^+(x)$ is taken over $v \in \mathbb{Z}^2$ lying within an angle spanned by vectors $-\text{tran}(x_1)$ and $-\text{tran}(x_n)$.

The following properties are now immediately proven:

Proposition 1 *For all $x, y \in X_m^\diamond$ labels of $CE^-(x)$ cannot be overwritten in xy, i.e.,*

$$\begin{aligned} u \in CE^-(x) \cap \text{dom}(x) &\Rightarrow \text{label}(x)(u) = \text{label}(xy)(u), \\ u \in CE^-(x) \setminus \text{dom}(x) &\Rightarrow u \notin \text{dom}(xy). \end{aligned}$$

Proposition 2 *For all $x \in X_m^\diamond$ labels of $CW^-(x)$ are not defined in x, i.e.,*

$$u \in CW^-(x) \Rightarrow u \notin \text{dom}(x).$$

Proposition 3 *For all $x, y \in X_m^\diamond$*

$$CE^+(xy) \subseteq CE^+(x).$$

Proposition 4 *For all $x, y \in X_m^\diamond$*

$$CW^+(x) \subseteq CW^+(xy).$$

We define a *configuration* as a pair (x, y), with $x, y \in X_m^\diamond$. We say that $(x', y') \in (X_m^\diamond)^2$ is a *successor* of (x, y) and write $(x, y) \prec (x', y')$ if

$$\begin{aligned} &x' = xx'' \text{ for some } x'' \in X \text{ and } y = y', \text{ or} \\ &y' = yy'' \text{ for some } y'' \in X \text{ and } x = x'. \end{aligned}$$

By $\prec^*$ we denote the transitive closure of $\prec$. Obviously X is not a code if and only if there exist $x, y \in X$ and $z \in X_m^\diamond$ such that $x \neq y$ and $(x, y) \prec^* (z, z)$.

Our goal is either to find a configuration (x, y) such that $x \neq y$ and

$$(x, y) \prec \ldots \prec (z, z)$$

(then X is not a code), or to prove that such a configuration does not exist (then X is a code). A configuration satisfying the above condition will be called an *eventually terminating configuration*.

Unfortunately, there are potentially infinitely many configurations to check. The following propositions will let us reduce the number of configurations under consideration.

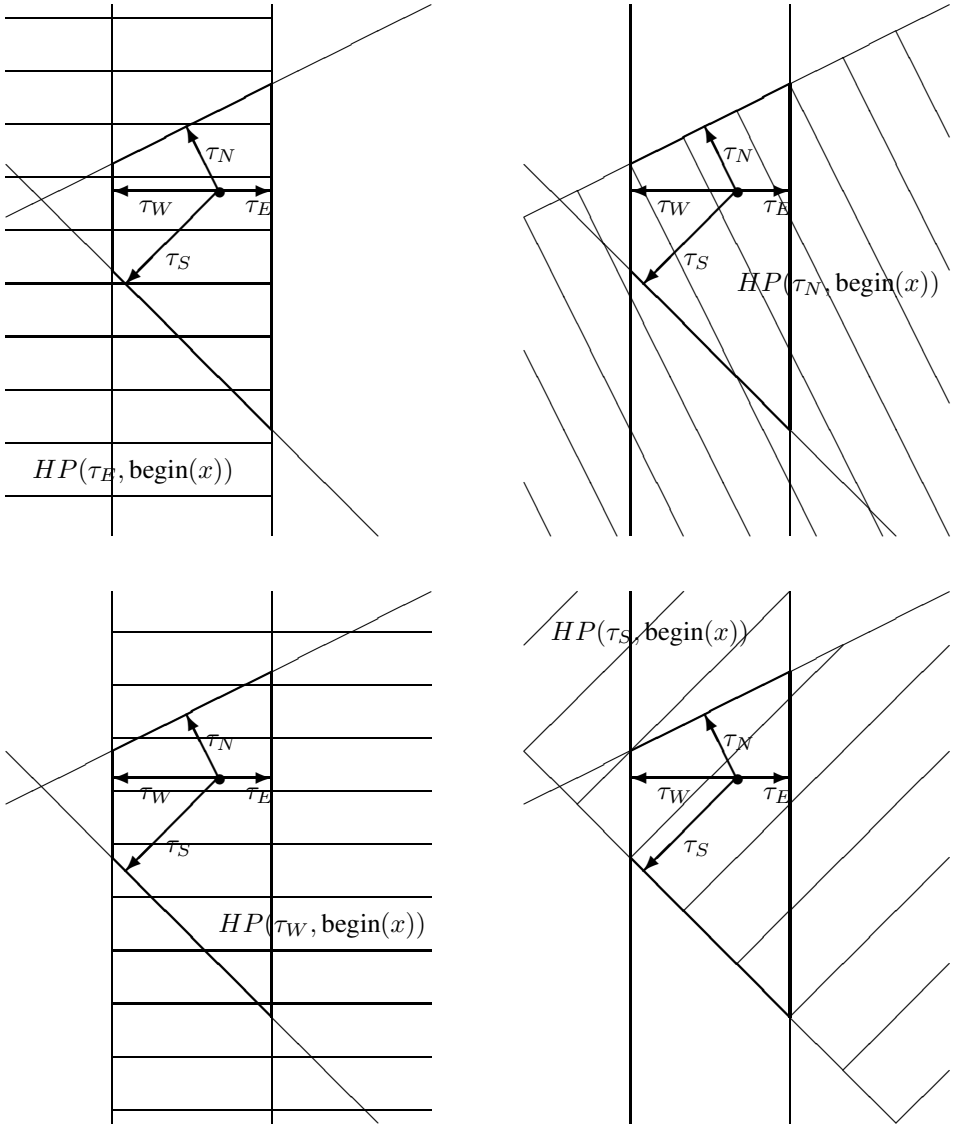


Fig. 2: Half-planes $HP(\tau, \text{begin}(x))$ for $\tau \in \{\tau_E, \tau_N, \tau_W, \tau_S\}$; the black dot denotes the start point of x.

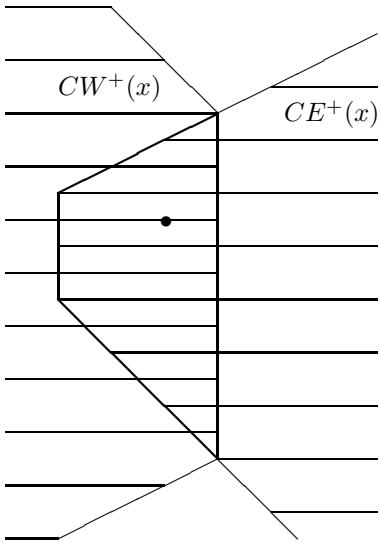


Fig. 3: $CW^+(x)$ and $CE^+(x)$ regions; the black dot denotes the end point of x.

Proposition 5 *If (x, y) is eventually terminating and $(x', y') \prec (x, y)$, then (x', y') is eventually terminating.*

Proof: Obvious. □

Proposition 6 *If (x, y) is eventually terminating, then*

$$\text{end}(x) \in CW^+(y) \cup CE^+(y),$$
$$\text{end}(y) \in CW^+(x) \cup CE^+(x).$$

Proof: See definitions of CW^+ and CE^+. □

Proposition 7 *If (x, y) is eventually terminating, then*

$$\text{label}(x) \mid_{CE^-(x) \cap CE^-(y)} \equiv \text{label}(y) \mid_{CE^-(x) \cap CE^-(y)} .$$

Proof: See definition of CE^- and Proposition 1. □

Notice that we do not need all of the information contained in configurations, just those labellings that can be changed by future catenations. By Proposition 7, instead of (x, y) we can consider a *reduced configuration* defined as a pair $(\pi_{RC}(x, y), \pi_{RC}(y, x))$ where

$$\pi_{RC}(z, z') = (\text{end}(z), \text{label}(z) \mid_{\text{dom}(z) \setminus (CE^-(z) \cap CE^-(z'))}).$$

Now Proposition 5 implies that we need only consider configurations where the span along τ_E is bounded by $|\tau_E|$, *i.e.*, $|\tau_E \cdot (\text{end}(x) - \text{end}(y))| \leq |\tau_E|^2$, since no single figure advances $\text{end}(x)$ or $\text{end}(y)$ by more than $|\tau_E|$. Moreover, Proposition 6 restricts the perpendicular span (in the direction of $R_{-\frac{\pi}{2}}(\tau_E)$). Hence the number of reduced configurations, up to translation, is finite.

This leads us to the main theorem of the paper:

Theorem 2 *It is decidable whether a given finite set $X \subseteq \Sigma_m^\diamond$ is a code.*

5 Algorithm

From the proof of Theorem 2 we can obtain an algorithm that verifies whether a given set of directed figures is a code. The following procedure returns **true** if the given input set is a code; otherwise it returns **false**. Additionally, if the given set is not a code, the algorithm finds $x_1, \ldots, x_k, x'_1, \ldots, x'_l$ such that $x_1 \cdots x_k = x'_1 \cdots x'_l$. Note that this is the reason for processing configurations (and not just reduced configurations) within the main loop of the algorithm.

1. **input** $X = \{x_1, \ldots, x_n\} \subseteq \Sigma_m^\diamond$;
2. **if** there exist $\alpha_{i_1}, \ldots, \alpha_{i_k} \in \mathbb{Z}_+$ ($\{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$) such that $\sum_{j=1}^{n} \alpha_{i_j} \text{tran}(x_{i_j}) = (0, 0)$ **then return false**;
3. **compute** $\tau_E, \tau_W, \tau_S, \tau_N$;
4. **set** $C = \emptyset$ (set of checked configurations),
   $RC = \emptyset$ (set of checked reduced configurations);
5. **for each** $x, y \in \{\tau_{-\text{begin}(x)}(x) \mid x \in X\}, x \neq y$ **do**
   (a) **if** (x, y) may be eventually terminating (*i.e.*, conditions from Propositions 6 and 7 are satisfied)
   and $(\pi_{RC}(x, y), \pi_{RC}(y, x)) \notin RC$
   then set
   $C = C \cup \{(x, y)\}$,
   $RC = RC \cup \{(\pi_{RC}(x, y), \pi_{RC}(y, x))\}$;
6. **set** $RC_{TMP} = \emptyset$;
7. **while** $RC \neq RC_{TMP}$ **do**
   (a) **set** $C_{TMP} = C$, $RC_{TMP} = RC$, $C = \emptyset$;
   (b) **for each** $(x, y) \in C_{TMP}$ **and** $z \in X$ **do**
      i. **if** $xz = y$ or $x = yz$
      then return false;
      ii. **if** (xz, y) may be eventually terminating
      and $|\tau_E \cdot (\text{end}(xz) - \text{end}(y))| \leq |\tau_E|^2$
      and $(\pi_{RC}(xz, y), \pi_{RC}(y, xz)) \notin RC$
      then set
      $C = C \cup \{(xz, y)\}$,
      $RC = RC \cup \{(\pi_{RC}(xz, y), \pi_{RC}(y, xz))\}$;

iii. **if** (x, yz) may be eventually terminating
and $|\tau_E \cdot (\text{end}(x) - \text{end}(yz))| \leq |\tau_E|^2$
and $(\pi_{RC}(x, yz), \pi_{RC}(yz, x)) \notin RC$
then set
$C = C \cup \{(x, yz)\}$,
$RC = RC \cup \{(\pi_{RC}(x, yz), \pi_{RC}(yz, x))\}$;

8. **return true**;

Note that configurations are considered up to a translation; in particular, this applies to "$\ldots \notin RC$" tests.

Observe that the actual implementation may omit the RC_{TMP} set and use $C \neq \emptyset$ as the loop condition (a new element is added to C if and only if a new element is added to RC). This would improve the efficiency but would obliterate the similarity of the algorithm to the algorithm of Sardinas and Patterson, *c.f.* Example 8. The implementation can also make use of the symmetrical nature of configurations.

The following examples show the algorithm behaviour for a code and a non-code. For the sake of simplicity, we depict elements of the set C only and we omit steps containing obvious assignments. We also omit pairs that can be obtained from another pair by exchanging the elements.

Example 6 *Consider* $X = \{x =$ $, y =$ $\} \subseteq \{a\}^{\diamond}_{\pi_1}$.

Step 2: The condition clearly fails.

Step 3: Set $\tau_E = (1, 1), \tau_W = (-\epsilon, -\epsilon), \tau_S = (0, -\epsilon), \tau_N = (-\epsilon, 0)$*, where* ϵ *is a small positive constant less than* $\sqrt{2}/2$*. A larger value of* ϵ *would force the algorithm to overestimate the bounding area and thus process too many configurations.*

Step 5: $C = \{(x, y)\}$.

Step 7: C *is already empty after first iteration, since all candidate configurations, i.e.,* (xx, y)*,* (x, yx)*,* (xy, y) *and* (x, yy)*, fail the condition of Proposition 7.*

Step 8: Thus X *is a code.*

Example 7 *Consider* $X = \{w =$ $, x =$ $, y =$ $, z =$ $\} \subseteq \{a\}^{\diamond}_{\pi_1}$.

Step 2: The condition fails.

Step 3: Set $\tau_E = (1, 1), \tau_W = (-\frac{1}{2}, -\frac{1}{2}), \tau_S = (0, -1), \tau_N = (-\frac{1}{2}, 0)$.

Step 5: $C = \{(w, x), (w, y)\}$*; other pairs fail the condition of Proposition 7.*

Step 7, iteration 1: $C = \{(ww, x), (ww, y), (wx, y), (wz, y)\}$.

Step 7, iteration 2: One of configurations gives $wx = yz$*, hence* X *is not a code.*

The final example illustrates the similarity of the algorithm presented here to the well-known algorithm of Sardinas and Patterson (ASP) when figures resemble words.

Example 8 *Consider* $X = \{x = \boxed{a}\boxed{a}\diamond, y = \boxed{b}\diamond, z = \boxed{b}\boxed{a}\diamond\} \subseteq \{a, b\}^{\diamond}_{\pi_1}$.

Step 2: The condition fails.

Step 3: Set $\tau_E = (2, 0), \tau_W = (-\frac{1}{2}, 0), \tau_S = (0, -\frac{1}{2}), \tau_N = (0, \frac{1}{2})$.

Step 5: $C = \{(y, z)\}$*; other pairs fail the condition of Proposition 7. The respective reduced configuration, formally* $(((1, 0), \{\}), ((2, 0), \{(1, 0) \mapsto a\}))$*, corresponds to the -*$a$ *suffix computed by ASP.*

Step 7, iteration 1: $C = \{(y, zy)\}$*; other pairs fail various conditions:* (yx, z) *fails the "*$\ldots \notin RC$*" test,* (yy, z) *and* (yz, z) *fail the condition of Proposition 7;* (y, zx) *and* (y, zz) *fail the* τ_E*-span condition. Note a small dissimilarity with ASP, which is a consequence of extending configurations from both sides. This could be optimized by always extending only the "leftmost" component with respect to the* τ_E *direction.*

Step 7, iteration 2: $C = \emptyset$*, hence* X *is a code.*

Observe that the RC *set of the algorithm corresponds to the union of sets constructed by ASP in each step.*

6 Defect effect

In this section we present several counterexamples to disprove the defect theorem for directed figures. They show that the effect fails even for very simple sets. On the other hand, restricting figures to word-like shapes, with appropriately chosen start and end points, obviously guarantees the defect property. It would be interesting to characterize what restrictions on figures allow the defect effect to hold.

Theorem 3 (Classical defect effect) *Let* $X \subseteq \Sigma^+$ *be a non-code. There exists* $Y \subseteq \Sigma^+$ *such that* $|Y| < |X|$ *and* $X^+ \subseteq Y^+$.

Let us now consider the following examples:

Example 9 *Let* $\Sigma = \{a\}$, $m = \{(a, a) \mapsto a\}$ *and*

$$X = \{x = \boxed{a}\diamond, y = \diamond\boxed{a}\}.$$

For each $n \geq 1$

$$(xy)^n = \boxed{\diamond}\boxed{a}$$

which implies that X *is not a code. However, there exists no* Y *such that* $|Y| < 2$ *and* $X^+ \subseteq Y^+$.

Thus, the defect effect does not hold even for two squares. The following example shows that in fact this is even worse: there are singleton non-codes.

Example 10 *Let* $\Sigma = \{a\}$, $m = \{(a, a) \mapsto a\}$ *and*

$$X = \{$$ $$\}$$

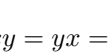
For each $n \geq 1$

$$x^n = x$$

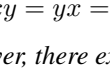
which implies that X *is not a code. However, there exists no* Y *such that* $|Y| < 1$ *and* $X^+ \subseteq Y^+$.

The defect theorem does not hold even for non-codes that do not allow a word x with $\text{tran}(x) = (0, 0)$ to be composed:

Example 11 *Let* $\Sigma = \{a, b, c\}$, $m = \{(a, \cdot) \mapsto a, (\cdot, a) \mapsto a, \ldots\}$ *(the remaining values of* m *can be set arbitrarily) and*

$$X = \{x =$$ $$, y =$$  $$\}.$$

Then

$$xy = yx =$$ 

which implies that X *is not a code. However, there exists no* Y *such that* $|Y| < 2$ *and* $X^+ \subseteq Y^+$.

7 Final remarks

It is interesting to note that the complexity of the codicity verification algorithm depends on the angle spanned by the translation vectors of figures. Bigger angles result in higher complexity, since there are more configurations to check. The obvious upper bound is exponential in the size of "bounding areas" defined in Section 4; this in turn grows like $\tan(\alpha/2)$, where α is the angle in question. However, at $\alpha = \pi$ the complexity drops radically because of Theorem 1. At the other end of spectrum, when the angle tends to zero, the figures resemble words and the algorithm becomes similar to the Sardinas and Patterson's algorithm, *cf.* Example 8. Estimating the complexity is an obvious subject for further research.

An interesting extension of our results would be to consider figures with no merging function, where catenation is a partial operation defined for those figures that do not overlap when catenated. Additional restrictions on figures would have to be imposed (*e.g.*, starting point on the inner boundary and end point on the outer boundary of a figure).

Another direction for further study is the behaviour of infinite catenations, like the usual infinite word asymptotics (*cf.* Foryś (2004)), and the density of codes (*cf.* Moczurad and Moczurad (2007)).

References

P. Aigrain and D. Beauquier. Polyomino tilings, cellular automata and codicity. *Theoretical Computer Science*, 147:165–180, 1995.

D. Beauquier and M. Nivat. A codicity undecidable problem in the plane. *Theoretical Computer Science*, 303:417–430, 2003.

J. Berstel and D. Perrin. *Theory of Codes*. Academic Press, 1985.

G. Costagliola, V. Deufemia, F. Ferrucci, and C. Gravino. On regular drawn symbolic picture languages. *Information and Computation*, 187:209–245, 2003.

G. Costagliola, F. Ferrucci, and C. Gravino. Adding symbolic information to picture models: definitions and properties. *Theoretical Computer Science*, 337:51–104, 2005.

W. Foryś. Asymptotic behaviour of bi-infinite words. *Theoretical Informatics and Applications RAIRO*, 38:27–48, 2004.

T. Harju and J. Karhumäki. Many aspects of defect theorems. *Theoretical Computer Science*, 324:35–54, 2004.

J. Karhumäki and S. Mantaci. Defect theorems for trees. *Fundamenta Informaticae*, 38:119–133, 1999.

S. Mantaci and A. Restivo. Codes and equations on trees. *Theoretical Computer Science*, 255:483–509, 2001.

M. Moczurad and W. Moczurad. Asymptotic density of brick and word codes. *Ars Combinatoria*, 83: 169–177, 2007.

W. Moczurad. Brick codes: families, properties, relations. *International Journal of Computer Mathematics*, 74:133–150, 2000.

W. Moczurad. Defect theorem in the plane. *Theoretical Informatics and Applications RAIRO*, 41:403–409, 2007.