On the Tiling by Translation Problem ⋆

S. Brlek *, X. Provençal

LaCIM, Université du Québec à Montréal,
C. P. 8888 Succursale "Centre-Ville", Montréal (QC), CANADA H3C 3P8

Jean-Marc Fédou

Laboratoire I3S - CNRS - UMR 6070, Université de Nice
Les Algorithmes - bt. Euclide B, BP.121, 06903 Sophia Antipolis - Cedex

Abstract

On square or hexagonal lattices tiles or polyominoes are coded by words. The polyominoes that tile the plane by translation are characterized by the Beauquier-Nivat condition. By using the constant time algorithms for computing the longest common extensions in two words, we provide a linear time algorithm in the case of pseudo-square polyominoes, improving the previous quadratic algorithm of Gambini and Vuillon. For pseudo-hexagon polyominoes not containing arbitrarily large square factors we also have a linear algorithm. The results are extended to more general tiles.

Key words: Tiling polyominoes, plane tesselation, longest common extensions

1 Introduction

In discrete geometry many results are based on an arithmetic approach for characterizing and recognizing patterns having a certain shape. Here we take a combinatorics on words point of view that enable us with new tools for analyzing a shape in discrete planes. We consider the problem of determining if a given word, which encodes the boundary of a discrete figure, tiles the plane by translation.

⋆ with the support of NSERC (Canada)

* Corresponding author.
Email addresses: brlek.srecko@uqam.ca (S. Brlek), provenca@lacim.uqam.ca (X. Provençal), fedou@unice.fr (Jean-Marc Fédou).

The way of tiling planar surfaces takes its roots in the ancient times for decorative purposes. More recently, connections were established with computational theory, mathematical logic and discrete geometry, where tilings are often regarded as basic objects for proving undecidability results for planar problems. Tilings have been also used in physics, as powerful tools for studying quasicrystal structures: in particular these structures can be better understood by representing them as rigid tilings decorated by atoms in a uniform fashion. Their long-range order can consequently be investigated in a purely geometrical framework, after assigning to every tiling a structural energy. A classical result of Berger [1] states that given a set of tiles, it is not decidable whether there exists a tiling of the plane which involves all its elements. This result implies the existence of aperiodic sets of tiles, and Berger provided an exemple using 20426 tiles. Berger's result has been strengthened afterwards by Gurevich and Koriakov [2] to the periodic case. It was therefore natural to seek manageable problems, and polyominoes appeared as good candidates. Invented by Golomb [3] who coined the term *polyomino*, these objects, also called n-ominoes or lattice animals, gained some interest after being popularized by Gardner in mathematical games [4]. In statistical physics they appear as models for percolation theory and their combinatorial properties have been extensively studied. These nowadays well studied combinatorial objects are still related to many challenging problems, such as tiling problems [5,6], games [4], among many others (see Weisstein [7] for more pointers). Their enumeration is also an open problem despite the fact that restricted classes have been fully described.

There are different types of polyominoes and here we consider a *polyomino* as a finite union of unit lattice closed squares (pixels) in the discrete plane whose boundary consists of a simple closed polygonal path using 4-connectedness. In particular, polyominoes are simply connected (contain no holes), and have no multiple points (Figure 1(a)).

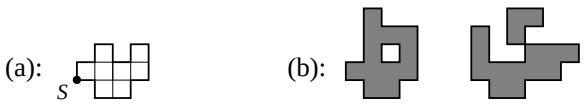


Fig. 1. (a) a polyomino; (b) not a polyomino.

The problem of deciding if a given polyomino tiles the plane by translation goes back to Wisjhoff and Van Leeuven [8] who coined the term *exact polyomino* for these, and also provided a polynomial algorithm for solving the problem. Polyominoes may be coded by words on a 4-letter alphabet $\Sigma = \{a, \overline{a}, b, \overline{b}\}$ identified with the unit steps $\{\rightarrow, \leftarrow, \uparrow, \downarrow\}$, also known as the Freeman chain codes [9,10], coding their boundaries (see [11] for further reading). For instance, the boundary $\mathbf{b}(P)$ of the polyomino in Figure 1 (a), in a counterclockwise manner and starting from the lowest of the leftmost points S, is coded by the word $w = a\,\overline{b}\,a\,a\,b\,a\,b\,b\,\overline{a}\,\overline{b}\,\overline{a}\,b\,\overline{a}\,\overline{b}\,\overline{a}\,\overline{b}$.

Observe that we may consider the words as circular which avoids a fixed origin. The *perimeter* of a polyomino P is the length of its boundary word $\mathbf{b}(P)$ and is of even length $2n$. Beauquier and Nivat [12] gave a characterization stating that the boundary of such a polyomino P satisfies the following (not necessarily in a unique way) factorization

$$\mathbf{b}(P) = A \cdot B \cdot C \cdot \widehat{A} \cdot \widehat{B} \cdot \widehat{C} \tag{1}$$

where at most one of the variables is possibly empty. Hereafter, this condition is refered as the BN-factorization. The operation $\widehat{(\ )}$ appearing in (1) is defined by $\widehat{U} = \widetilde{\overline{U}}$, where $\widetilde{(\ )}$ is the usual reversal operation and $\overline{(\ )}$ the complement on $\Sigma = \{a, \overline{a}, b, \overline{b}\}$.

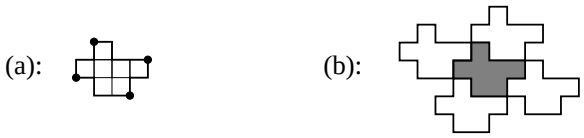


Fig. 2. (a) an exact polyomino; (b) the associated tiling.

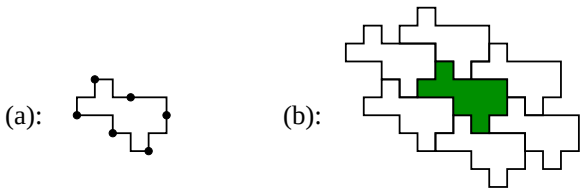
For instance, the exact polyomino in Figure 2 can be coded by the circular word

$$w = a\,\overline{b}\,a\,a\,b\,a\,b\,\overline{a}\,\overline{a}\,b\,\overline{a}\,\overline{b}\,\overline{a}\,\overline{b}$$

its semi-perimeter is 7, and its boundary may be factorized as

$$\mathbf{b}(P) = A \cdot B \cdot \widehat{A} \cdot \widehat{B} = a\,\overline{b}\,a\,a \cdot b\,a\,b \cdot \overline{a}\,\overline{a}\,b\,\overline{a} \cdot \overline{b}\,\overline{a}\,\overline{b},$$

and for the exact polyomino P' below



the boundary may be factorized as

$$\mathbf{b}(P') = a\,a\,\overline{b} \cdot a\,\overline{b}\,a \cdot b\,a\,b \cdot b\,\overline{a}\,\overline{a} \cdot \overline{a}\,b\,\overline{a} \cdot \overline{b}\,\overline{a}\,\overline{b}.$$

Therefore the problem of deciding wether a polyomino tiles the plane by translation or not, reduces to finding a factorization satisfying the Beauquier and Nivat condition. A naive $\mathcal{O}(n^4)$ algorithm solves the problem, but recently, Gambini and Vuillon [13] improved that bound by designing an $\mathcal{O}(n^2)$ algorithm that checks the BN-condition (1).

The underlying idea of our approach is to search efficiently the pairs of homologous factors $X, \widehat{X}$. Our algorithms borrow from Lothaire [14] (for instance) that the *Longest-Common-Factor*, the *Longest-Common-Prefix* and the *Longest-Common-Suffix* in two words may be computed in linear time. The approach is also inspired by the linear algorithm of Gusfield and Stoye [15] for detecting *tandem repeats* in a word, and by the linear algorithm used to detect repetitions with gaps, as shown in Lothaire [14]. More precisely, the computation of the *Longest-Common-Left-Extension* (LCLE(u, v)) and *Longest-Common-Right-Extension* (LCRE(u, v)) is achieved in constant time, provided a linear pre-processing is performed on u and v, by a clever utilization of suffix-trees (see Gusfield [16]). Taking advantage of these algorithms we provide a linear algorithm, with respect to the length of words, for pseudo-square polyominoes (Theorem 9). We establish a first step in order to provide a linear algorithm for pseudo-hexagons as well. Indeed, for boundary words not having two large square repetitions there is a linear algorithm to decide whether a polyomino tiles the plane by translation or not (Theorem 12).

2 Preliminaries

Let Σ be a finite alphabet whose elements are called *letters*. Finite words are sequences of letters, that is, functions $w : [0..n-1] \longrightarrow \Sigma$. For practical reasons we use the functions $f(w) = w[0]$, and $\ell(w) = w[n-1]$. The set of words of length n is Σ^n, and the free monoid $\Sigma^* = \cup_{n=0}^{\infty}\Sigma^n$ is the set of all finite words and the empty word is denoted ε.

A *morphism* is a function $\sigma : \Sigma^* \longrightarrow \Sigma^*$ such that $\sigma(uv) = \sigma(u)\sigma(v)$. Clearly a morphism is defined by the image of the letters. On the other hand, the operator $\widehat{(\ )}$ is an *antimorphism* since $\widehat{uv} = \widehat{v}\widehat{u}$. Moreover this operator is involutive since $\widehat{\widehat{u}} = u$. A *factor* u of w is a word $u \in \Sigma^*$ such that $w = xuy$, for some words $x, y \in \Sigma^*$. If $x = \varepsilon$ (resp. $y = \varepsilon$) then u is called *prefix* (resp. *suffix*). The length of a word w is $|w|$, and the number of occurrences of a factor $u \in \Sigma^*$ is $|w|_u$. A word is said to be *primitive* if it is not a power of another word. A useful operator is the circular permutation ρ defined by $\rho(w) = w[1..(n-1)] \cdot w[0]$. Two words u and v are *conjugate* when there are words x, y such that $u = xy$ and $v = yx$. Equivalently, u and v are conjugate if and only if there exists an index k such that $u = \rho^k(v)$. Conjugation is an equivalence relation written $u \equiv v$. Paths on the square lattice $\mathbb{Z} \times \mathbb{Z}$ are encoded by pairs (S, w), where S is a point in $\mathbb{Z} \times \mathbb{Z}$ and w a word on $\Sigma = \{a, \overline{a}, b, \overline{b}\}$ identified with the unit steps $\{\rightarrow, \leftarrow, \uparrow, \downarrow\}$. Moreover, the relative positions of the starting and ending point of any path is completely determined by the sum of the unit vectors corresponding to each letter, that

is, for a word $w : [0..n-1] \to \Sigma$, there is a naturally defined vector

$$\overrightarrow{w} = \sum_{k=0}^{n-1} \overrightarrow{w_k}.$$

Note that $\overrightarrow{w} = 0$ if and only if w is a closed path, and that $\overrightarrow{w} = -\overrightarrow{\widehat{w}}$.

The classical notion of *equipollent* vectors also extends to paths, so that equipollent paths always define a translation. Two vectors $\overrightarrow{v_1}$ and $\overrightarrow{v_2}$ are called *homologous* if either $\overrightarrow{v_1} = \overrightarrow{v_2}$ or $\overrightarrow{v_1} = -\overrightarrow{v_2}$, and the notion extends to paths as well. When two paths (S, u) and (S', v) are homologous, we also say that the words u and v are *homologous*. More precisely, two words u and v are said to be *homologous* when either (i) $u = v$, or (ii) $u = \widehat{v}$. An exact polyomino P whose boundary is $\mathbf{b}(P) \equiv A \cdot B \cdot C \cdot \widehat{A} \cdot \widehat{B} \cdot \widehat{C}$ is called a *pseudo-hexagon* if none of the variables is empty and a *pseudo-square* otherwise. Such factorization defines a pair of vectors, $u = \overrightarrow{A} + \overrightarrow{B}$ and $v = \overrightarrow{B} + \overrightarrow{C}$, whose translations induce a regular tiling of the plane. For instance, lets consider the word $w \equiv \overline{b}\,a\,\overline{b}\,a\,a\,\overline{b}\,a\,b\,a\,a\,b\,\overline{a}\,b\,\overline{a}\,b\,\overline{a}\,\overline{a}\,\overline{a}\,\overline{a}\,\overline{b}$ with factorization

$$w \equiv A \cdot B \cdot C \cdot \widehat{A} \cdot \widehat{B} \cdot \widehat{C} = \overline{b}\,a\,\overline{b} \cdot a\,a\,\overline{b}\,a \cdot b\,a\,a \cdot b\,\overline{a}\,b \cdot \overline{a}\,b\,\overline{a}\,\overline{a} \cdot \overline{a}\,\overline{a}\,\overline{b}.$$

The associated vectors are $u = \overrightarrow{A} + \overrightarrow{B} = (4, -3)$ and $v = \overrightarrow{B} + \overrightarrow{C} = (5, 0)$ as shown in Figure 3.

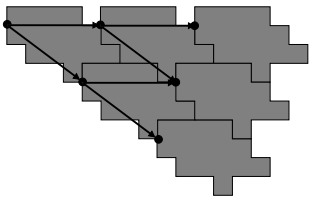


Fig. 3. Translations defined by a BN-factorization.

The next lemma gives two useful criteria for identifying closed paths.

Lemma 1 *For a path $w \in \Sigma^*$, the following conditions are equivalent*
(i) *w is a closed path;*
(ii) *$|w|_\alpha = |w|_{\overline{\alpha}}$, for all $\alpha \in \Sigma$;*
(iii) *$\overrightarrow{w} = 0$.*

Denote the set of closed paths in Σ^* by $\mathcal{C}$.

3 Searching the homologous factors

Since polyominoes are coded by circular words w, in order to find the homologous factors it is convenient to work with $w \cdot w$ since a pair of homologous

factors might be split, depending on the starting point. Therefore, finding the homologous factors suggests to look for the *longest common factor* of ww and $\widehat{ww}$ denoted $\mathrm{LCF}(ww, \widehat{ww})$. For instance the longest common factors of the polyomino-tile P in Figure 2 are

$$\mathrm{LCF}(ww, \widehat{ww}) = \{a\,\overline{b}\,a\,a, \overline{a}\,\overline{a}\,b\,\overline{a}\}$$

and they are homologous sides by definition. Indeed, since we know the positions i and j of $a\,b\,a\,a$ and $\overline{a}\,\overline{a}\,b\overline{a}$ in w, this is easy to check in linear time. Clearly the boundary of P may be written as

$$\mathbf{b}(P) = w = a\,\overline{b}\,a\,a \cdot u \cdot \overline{a}\,\overline{a}\,b\,\overline{a} \cdot v$$

and then we have $v = \widehat{u}$. Unfortunately the situation is not always that good. Indeed, let $w = a\,a\,b\,b\,b\,a\,a\,b\,\overline{a}\,\overline{a}\,b\,\overline{a}\,\overline{b}\,\overline{b}\,\overline{b}\,\overline{a}\,\overline{a}\,\overline{b}\,a\,\overline{b}$. Then the longest homologous factors of w are $\mathrm{LCF}(ww, \widehat{ww}) = \{a\,a\,b\,b\,b\,a, \overline{a}\,\overline{b}\,\overline{b}\,\overline{b}\,\overline{a}\,\overline{a}\}$(see Figure 4).

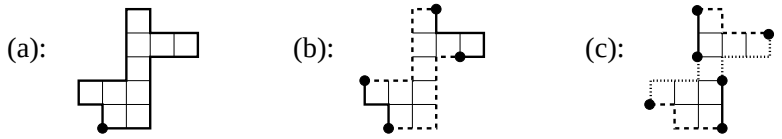


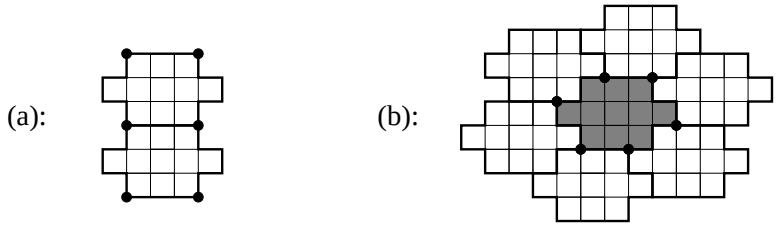
Fig. 4. (b) longest homologous factors; (c) a good factorization.

However, $w = a\,a\,b\,b\,b\,a \cdot a\,b\,\overline{a}\,\overline{a}\,b \cdot \overline{a}\,\overline{b}\,\overline{b}\,\overline{b}\,\overline{a}\,\overline{a} \cdot \overline{b}\,a\,\overline{b}$ does not satisfy the BN-factorization. A good factorization is $w \equiv b\,b \cdot b\,a\,a\,b \cdot \overline{a}\,\overline{a}\,b\,\overline{a} \cdot \overline{b}\,\overline{b} \cdot \overline{b}\,\overline{a}\,\overline{a}\,\overline{b} \cdot a\,\overline{b}\,a\,a$. This means that not all the homologous factors provide a factorization, and good candidates are those separated by factors of same length.

Definition 2 *Let $w \equiv \mathbf{b}(P)$ be the boundary of a polyomino P. A factor A of w is* admissible *if*

(i) $w \equiv Ax\widehat{A}y$, *for some x, y such that $|x| = |y|$;*
(ii) A *is* maximal, *that is, $f(x) \neq \overline{\ell(x)}$ and $f(y) \neq \overline{\ell(y)}$.*

Not all admissible factors lead to a BN-factorization. For instance, in the polyomino $w \equiv a\,a\,a\,b\,a\,b\,\overline{a}\,b\,\overline{a}\,\overline{a}\,\overline{a}\,\overline{b}\,\overline{a}\,\overline{b}\,a\,\overline{b}$ shown below the factor aaa is admissible



but does not provide a correct factorization of w. Indeed, $A = aa$ is the admissible factor ($w = Ax\widehat{A}y$ with $x = abab\overline{a}b$, $y = \overline{a}\overline{b}\overline{a}\overline{b}a\overline{b}$) yielding a correct factorization with $B = aba$ and $C = b\overline{a}b$:

$$w \equiv a\,a \cdot a\,b\,a \cdot b\,\overline{a}\,b \cdot \overline{a}\,\overline{a} \cdot \overline{a}\,\overline{b}\,\overline{a} \cdot \overline{b}\,a\,\overline{b}.$$

The following technical property is established in [17,18], generalizing a result of Daurat and Nivat ([5], IWCIA'03) on the number of salient and reentrant points of discrete sets. More precisely, let $\mathcal{L} = \{ab, b\overline{a}, \overline{a}\overline{b}, \overline{b}a\}$, that is the set of "left turns" moves. Similarly $\mathcal{R} = \{a\overline{b}, \overline{b}\overline{a}, \overline{a}b, ba\}$ is the set of "right turns". Then $|w|_{\mathcal{R}}$ (respectively $|w|_{\mathcal{L}}$) is the number of right (resp. left) turns in a path w on a square lattice. For practical reason, we define for every word $w \in \Sigma^*$ an auxiliary function recording the difference between left and right turns by

$$\Delta(w) = |w|_{\mathcal{L}} - |w|_{\mathcal{R}}. \tag{2}$$

The function Δ is *additive* in the following sense

$$\Delta(uv) = \Delta(u) + \Delta(\ell(u) \cdot f(v)) + \Delta(v).$$

Observe that $\Delta(w) = -\Delta(\widehat{w})$, for all paths $w \in \Sigma^*$. When the path w is closed there is possibly an extra left or right turn at the end of the path, so that we define the following specialization of Δ for closed paths

$$\Delta_{\mathcal{C}}(w) = \Delta(ww_0).$$

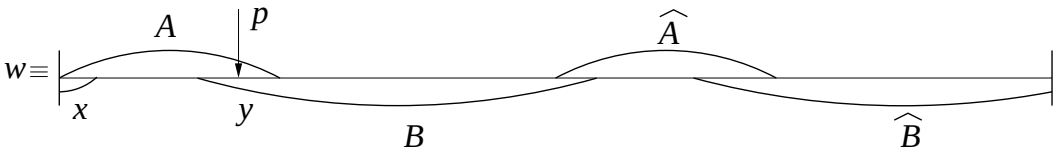
Note that $\Delta_{\mathcal{C}}$ is invariant by conjugation of words. A path in a grid is said to be *non-crossing* , if it does not traverse itself in a grid point visited twice.

Property 3 [B.L.L. [17], DLT2005] *Let $w \in \mathcal{C}$ be a non-crossing closed path in the square lattice. Then $|\Delta_{\mathcal{C}}(w)| = 4$.*

Obviously, Property 3 above holds for words coding the boundary of polyominoes since such paths are non-crossing . The next proposition establishes a useful property.

Proposition 4 *Let p be any fixed position on the contour $w = \mathbf{b}(P) \in \Sigma^{2n}$ of a polyomino P. Let X be the set of all admissible factors that include position p and $\widehat{X}$ be the set of their respective homologous factors. Then, there exists at least one position in w that is not covered by any element of $X \cup \widehat{X}$.*

Proof. By contradiction, assume that there is no such point. Let $A \in X$ be the factor that starts at the leftmost position and $B \in X$ be the one that ends at the rightmost position as shown below. Let x be the overlap between A and



$\widehat{B}$, and y be the overlap between A and B. Without loss of generality we may assume that $|y| \geq |x|$. Note that since y is a prefix of B and a suffix of A, then $\widehat{y}$ is a suffix of $\widehat{B}$ and a prefix of $\widehat{A}$. This implies that if $|x| = |y|$, then $x = \widehat{y}$ and the factorization is $w \equiv x\,U\,\widehat{x}\,V\,x\,\widehat{U}\,\widehat{x}\,\widehat{V}$.

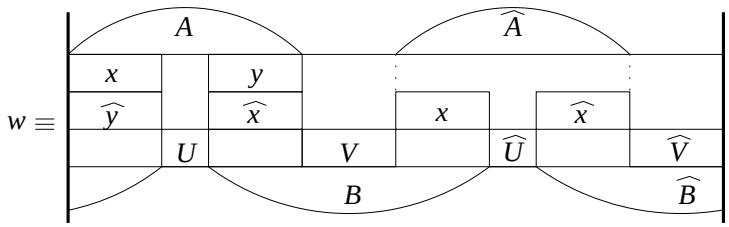


Fig. 5. Case 1 : $|x| = |y|$.

Since all turns in a factor are cancelled by those of its homologue, we only have to consider the turns between consecutive factors: $x\,U$ is cancelled by $\widehat{U}\,\widehat{x}$, $U\,\widehat{x}$ by $x\,\widehat{U}$, $\widehat{x}\,V$ by $\widehat{V}x$ (the word w is circular), and $V\,x$ by $\widehat{x}\,\widehat{V}$. Hence the difference between right and left turns is 0 and, by Property 3, w is not the boundary of a polyomino. Contradiction.

If $|x| \neq |y|$ we have the following situation where $\widehat{y}$ does not overlap $\widehat{A}$

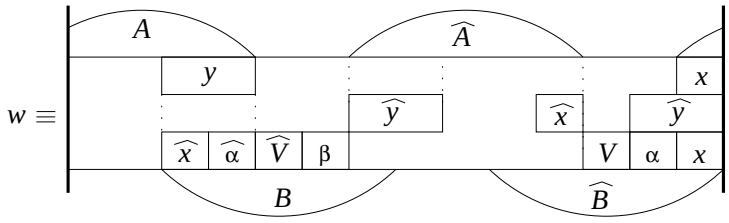


Fig. 6. Case 2 : $\widehat{y}$ does not overlap $\widehat{A}$ in $\widehat{B}$.

in $\widehat{B}$. Letting V be the factor between $\widehat{A}$ and $\widehat{y}$, we have the factorization $w \equiv A\,\widehat{V}\,\beta\,\widehat{A}\,V\,\alpha$, where α is the factor between V and A in $\widehat{B}$ and β is the factor between $\widehat{V}$ and $\widehat{A}$ in B. Passing to vectors, and using commutativity of addition, we have

$$\overrightarrow{w} = \overrightarrow{A} + \overrightarrow{\widehat{A}} + \overrightarrow{V} + \overrightarrow{\widehat{V}} + \overrightarrow{\beta} + \overrightarrow{\alpha} = \overrightarrow{\beta} + \overrightarrow{\alpha} = \overrightarrow{0}\,.$$

But $\widehat{y} = \alpha x$, so that β is followed by α in w. Therefore $\beta\alpha$ is a nonempty closed path on the boundary of P. Contradiction.

In the case where $\widehat{y}$ overlaps $\widehat{A}$ in $\widehat{B}$ we have the following situation where

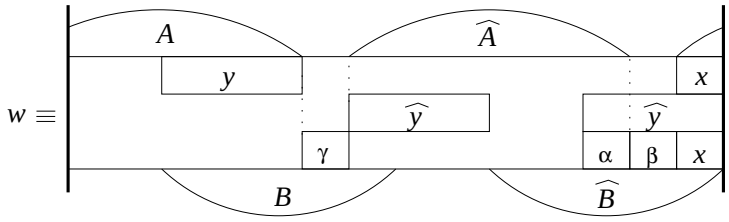


Fig. 7. Case 3 : $\widehat{y}$ does overlap $\widehat{A}$ in $\widehat{B}$.

$\overrightarrow{\gamma} + \overrightarrow{\beta} = 0$ (by closure property $\overrightarrow{w} = 0 = \overrightarrow{A} + \overrightarrow{\gamma} + \overrightarrow{\widehat{A}} + \overrightarrow{\beta}$). Moreover, $\widehat{y} = \alpha\beta x$, so that $y\,\gamma\,\widehat{y}$ contains the nonempty factor $\widehat{\alpha}\,\gamma\,\alpha\,\beta$ corresponding to a closed path. Contradiction. ∎

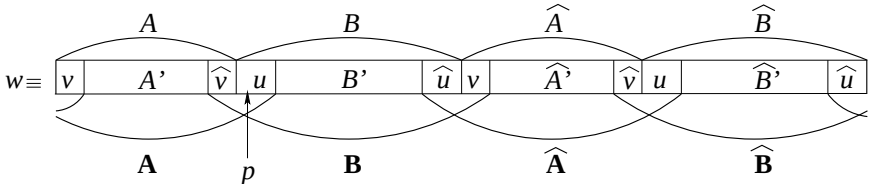
Consequently, admissibility is ensured for words having a BN-factorization, as we show now.

Corollary 5 *Let $w \equiv ABC\widehat{A}\widehat{B}\widehat{C}$ be a BN-factorization of the boundary $\mathbf{b}(P)$ of an exact polyomino P. Then A, B and C are admissible.*

Proof. Condition (i) of Definition 2 is a direct consequence of the fact that $|u| = |\widehat{u}|$ for all $u \in \Sigma^*$. For condition (ii) of Definition 2, we proceed by contradiction. Assume that A is not maximal, that is $f(BC) = \overline{\ell(BC)}$.

In the case of a pseudo-hexagon factorization, A, B and C are nonempty. Let $x \in \Sigma$ and $B', C' \in \Sigma^*$ be such that $B = xB'$ and $C = C'\overline{x}$, then the factor $\widehat{B}\widehat{C} = \widehat{xB'}\widehat{C'\overline{x}} = \widehat{B'}\overline{x}x\widehat{C'}$. But $\overline{x}x$ cannot be a factor of a word coding the boundary of a polyomino. Contradiction.

In the case of a pseudo-square factorization $w \equiv AB\widehat{A}\widehat{B}$, the assumption implies that $f(B) = \overline{\ell(B)}$. Let $u \in \Sigma^+$ be the longest prefix of B such that $B = uB'\widehat{u}$ for some word $B' \in \Sigma^*$. Let $\mathbf{A} = \widehat{u}Au$, the factor $\mathbf{A}$ is admissible because $f(B') \neq \overline{\ell(B')}$. Similarly, let $\mathbf{B} = \widehat{v}A'v$ where v is the longest prefix of A such that $A = vA'\widehat{v}$. Note that if B is admissible, then v is the empty word and $\mathbf{B} = B$.



Since u is nonempty, there exists a position p in w such that every other position in w is overlapped by an admissible factor overlapping p or its homologue. This is in contradiction with Proposition 4. ∎

A similar approach was used by Gambini and Vuillon ([13], section 3.1) by using the geometric result of Daurat and Nivat [5]. The (shorter) proof above differs substantially since it only depends on combinatorial arguments on words.

Proposition 4 specializes for pseudo-squares as follows. Assume that a pseudo-square P has two factorizations

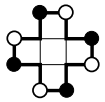
$$w = \mathbf{b}(P) \equiv AB\widehat{A}\widehat{B} \equiv XY\widehat{X}\widehat{Y}$$

where $A = sXt$. Then, by using the same argument as in Proposition 4, the boundary of P contains a loop yielding a contradiction.

Corollary 6 *If $w = \mathbf{b}(P) \equiv AB\widehat{A}\widehat{B} \equiv XY\widehat{X}\widehat{Y}$ are two non-trivially distinct factorizations of the boundary of a pseudo-square P, that is $A \notin \{X, Y, \widehat{X}, \widehat{Y}\}$, then there exist $\alpha \in \mathrm{Suff}(A)$ and $\beta \in \mathrm{Pref}(B)$ such that $\alpha\beta \in \{X, Y, \widehat{X}, \widehat{Y}\}$.*

As an example, for a cross polyomino, we have the following pseudo-square factorizations

$$\circ : w \equiv a\,b\,a \cdot b\,\overline{a}\,b \cdot \overline{a}\,\overline{b}\,\overline{a} \cdot \overline{b}\,a\,\overline{b},$$
$$\bullet : w \equiv b\,a\,b \cdot \overline{a}\,b\,\overline{a} \cdot \overline{b}\,\overline{a}\,\overline{b} \cdot a\,\overline{b}\,a.$$



The problem of enumerating all the factorizations of a given pseudo-square will be addressed in a forthcoming paper.

3.1 A linear time algorithm for detecting pseudo-squares

The main idea used to achieve linear time factorization, is to choose a position p in w and then list all the *admissible factors* A that overlap this fixed position. The following auxiliary functions are useful. The *Longest-Common-Right-Extension* (LCRE) and *Longest-Common-Left-Extension* (LCLE) of two words u and v at positions respectively m and n are partial functions

$$\mathrm{LCRE}, \mathrm{LCLE} : \Sigma^* \times \Sigma^* \times \mathbb{N} \times \mathbb{N} \longrightarrow \mathbb{N}$$

defined as follows. For $u, v \in \Sigma^*$, let i and j be such that $0 \leq i \leq |u|$ and $0 \leq j \leq |v|$, then

$$\mathrm{LCRE}(u, v, i, j) = \mathrm{LCP}(\rho^i(u), \rho^j(v)),$$
$$\mathrm{LCLE}(u, v, i, j) = \mathrm{LCS}(\rho^{|u|-i}(u), \rho^{|v|-j}(v)),$$

where LCP and LCS are respectively the lengths of the longest common prefix and suffix.

Remark 7 It is clear from the definition above that LCRE and LCLE may be computed in linear time. Their computation may also be performed directly by the following formulas. Since we use circular words w, denote $\underline{i} = i \mod |w|$. If $u[\underline{i}] = v[\underline{j}]$ then

(i) $\mathrm{LCRE}(u, v, \underline{i}, \underline{j}) = \max\{k \in \mathbb{N} \mid u[\underline{i}..(\underline{i+k})] = v[\underline{j}..(\underline{j+k})]\} + 1,$
(ii) $\mathrm{LCLE}(u, v, \underline{i}, \underline{j}) = \max\{k \in \mathbb{N} \mid u[(\underline{i-k})..\underline{i}] = v[(\underline{j-k})..\underline{j}]\} + 1,$

and, otherwise, $\mathrm{LCRE}(u, v, \underline{i}, \underline{j}) = \mathrm{LCLE}(u, v, \underline{i}, \underline{j}) = 0$.

For example, if $u = a\,a\,b\,b\,b\,a\,a\,b\,a\,a\,b\,a\,b\,a\,b\,a\,b\,a$, $v = b\,a\,b\,a\,a\,b\,b\,b\,a\,a\,b\,b\,a\,b\,a\,b\,a\,b\,b$, $i = 4$ and $j = 7$ then (note that words all start at position 0) we have

$$u = \underline{a\,a\,b\,b}\,\mathbf{b}\,\underline{a\,a\,b}\,a\,a\,b\,a\,b\,a\,b\,a\,b\,a,$$
$$v = b\,a\,b\,\underline{a\,a\,b\,b}\,\mathbf{b}\,\underline{a\,a\,b}\,b\,a\,b\,a\,b\,a\,b\,b,$$

and $\mathrm{LCRE}(u, v, 4, 7) = 4, \mathrm{LCLE}(u, v, 4, 7) = 5$. On the other hand $\mathrm{LCRE}(u, v, 4, 1) = \mathrm{LCLE}(u, v, 4, 1) = 0$.

Later we need to perform these computations $\mathcal{O}(n)$ times. Fortunately, the computation of LCLE and LCRE is achieved in constant time, provided a linear pre-processing is performed on u and v, by a clever utilization of suffix-trees (see Gusfield [16], section 9.1, or Gusfield and Stoye [15], page 531).

Lemma 8 *Let $w = \mathbf{b}(P) \in \Sigma^{2n}$ be the boundary of P. For any position p in w, listing all the admissible factors that include p is computable in linear time.*

Proof. Since the longest common right and left extension problem can be solved in constant time after linear time pre-processing, the following algorithm lists all admissible factors containing the $(p+1)$-th letter of w in linear time.

Algorithm 1
Input : $w = \mathbf{b}(P) \in \Sigma^{2n}$
0 : *Pre-process w and $\widehat{w}$ so that* LCLE *and* LCRE *take constant time*
1 : **For** $i := 0$ **to** $2n-1$ **do**
2 : $\quad l := \mathrm{LCLE}(w, \widehat{w}, p, i) - 1$
3 : $\quad r := \mathrm{LCRE}(w, \widehat{w}, p, i) - 1$
4 : $\quad A := w[p-l, \ldots, p+r]$
5 : $\quad$ **If** $w \equiv Ax\widehat{A}y$ **with** $|x| = |y|$ **then**
6 : $\quad\quad$ *Add A to the list of admissible factors*
7 : $\quad$ **end if**
8 : **end for**

Using the modulo in managing the positions is superfluous. Indeed, since w is a circular word, we may take a convenient conjugate such that the modulo is not used. Note that, by definition of LCRE and LCLE, the factor A in this algorithm is necessarily maximal. Given an occurrence of A in $\widehat{w}$, one computes in constant time the corresponding position of $\widehat{A}$ in w. Whether or not $\widehat{A}$ overlaps A in w is decidable in constant time. If $\widehat{A}$ and A do not overlap then, $u \equiv Ax\widehat{A}y$ and A is an admissible factor if $|x| = |y|$, a condition obtained in $\mathcal{O}(1)$. ∎

This lemma implies that the number of admissible factors in a word is linear. To determine a precise upper bound remains an open problem which is similar to the problem of determining a tight upper bound for the number of distinct squares in a word (see for instance Lothaire [14] or Ilie [19]).

Theorem 9 *Let $w = \mathbf{b}(P) \in \Sigma^{2n}$ be the boundary of P. Determining if w codes a* pseudo-square *is decidable in linear time.*

Proof. If w encodes an exact polyomino, any position belongs to some admissible factor of the BN-factorization. Therefore, it suffices to apply Lemma 8 to an arbitrary position p. Then, Algorithm 1 provides the list of all admissi-

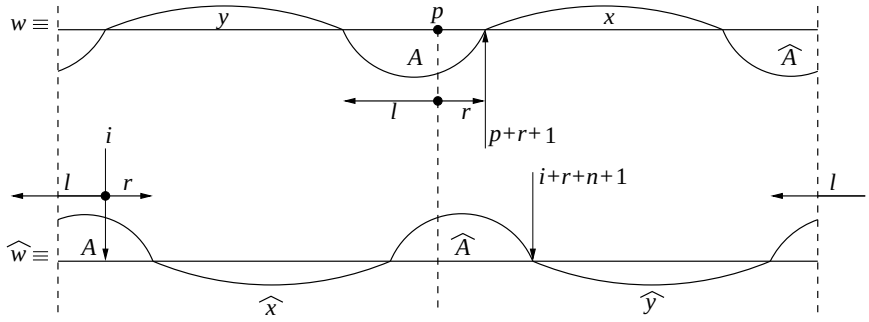


Fig. 8. An admissible factor A in w and $\widehat{w}$.

ble factors overlapping the position p, and it only remains to check, for each admissible factor, if $x = \widehat{y}$. Lemma 5 ensures that if $w \equiv AB\widehat{A}\widehat{B}$ then B is maximal. As shown in Figure 8, it suffices now to replace step 6 in Algorithm 1 by:

$6a:$ **If** $\mathrm{LCRE}(w, \widehat{w}, p+r+1, \underline{i+r+n+1}) = |x|$ **then**
$6b:$ P is a *pseudo-square* : $w \equiv Ax\widehat{A}\widehat{x}$
$6c:$ **end if**

Since LCRE is computed in constant time, the overall algorithm is linear. ■

To illustrate this algorithm take the word $w \equiv a\,a\,\overline{b}\,a\,a\,b\,a\,a\,b\,\overline{a}\,\overline{a}\,b\,\overline{a}\,\overline{a}\,\overline{b}\,\overline{a}\,\overline{a}\,\overline{b}$ as shown in Figure 9. It satisfies the condition of Theorem 9. Since p is an

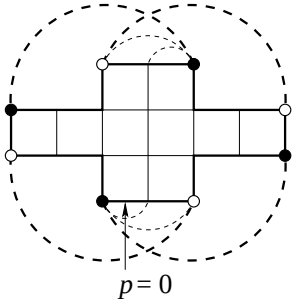


Fig. 9. A pseudo-square with its admissible factors

arbitrary chosen position we may take $p = 0$. All the admissible factors are displayed with dashed lines, and the homologous factors are displayed with thick lines. In this case we obtain two different pseudo-square factorizations distinguished by $\circ$ and $\bullet$:

$\circ : w \equiv a\,a\,\overline{b}\,a\,a \cdot b\,a\,a\,b \cdot \overline{a}\,\overline{a}\,b\,\overline{a}\,\overline{a} \cdot \overline{b}\,\overline{a}\,\overline{a}\,\overline{b}$;
$\bullet : w \equiv a\,a\,b\,a\,a \cdot b\,\overline{a}\,\overline{a}\,b \cdot \overline{a}\,\overline{a}\,\overline{b}\,\overline{a}\,\overline{a} \cdot \overline{b}\,a\,a\,\overline{b}$.

Note that the algorithm above lists all the pseudo-square factorizations but can be easily modified to stop after the first one if there is such.

3.2 A linear algorithm for k-square-free pseudo-hexagons

A factor f of a word w is called a *square* if $f = xx$ for some $x \in \Sigma^+$. The set of squares of a word w is Squares(w).

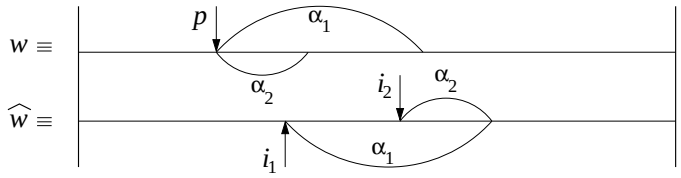
Definition 10 *A word w is* k-square-free *if and only if it satisfies the inequality* $\max\{|f| : f \in \text{Squares}(w)\} < k$.

The following technical lemma is useful.

Lemma 11 *Let $w = \mathbf{b}(P) \in \Sigma^{2n}$ a k-square-free word, and let p be any position in w. Then, the number of admissible factors that overlap the position p in w is bounded by $4k + 2\log(n)$.*

Proof. Let $A_1, A_2, \ldots, A_r$ be all the admissible factors that include p in w. Since $w \equiv A_i x_i \widehat{A_i} y_i$ with $|x_i| = |y_i|$ for all $1 \leq i \leq r$, all their homologous occurrences $\widehat{A_i}$ include the position $p' = p+n$. Thus, there is a position q such that all A_i include q in $\widehat{w}$. In Algorithm 1, all the admissible factors including p in w are listed through a loop such that each iteration can detect at most one of them. Let i_1, i_2 be such that $0 \leq i_1 < i_2 < q$ and assume that admissible factors are detected when $i = i_1$ and $i = i_2$.

Let $\alpha_1 = \widehat{w}[i_1, \ldots, q]$ and $\alpha_2 = \widehat{w}[i_2, \ldots, q]$. By definition of *common extension* we have $\alpha_1 = w[p, \ldots, p + |\alpha_1| - 1]$ and $\alpha_2 = w[p, \ldots, p + |\alpha_2| - 1]$, as shown below.



This implies that α_2 is prefix and suffix of α_1, so that Lothaire's Proposition 1.3.4 [20] applies. It follows that there are two words $u, v \in \Sigma^*$ such that $\alpha_1 = (uv)^m u$, for some integer m with

$$m(|\alpha_1| - |\alpha_2|) \geq |\alpha_2| \geq (m-1)(|\alpha_1| - |\alpha_2|). \tag{3}$$

If $|\alpha_1| < 2|\alpha_2|$, equation (3) requires m to be greater than 1 and thus α_1 contains a square of length at least $\frac{1}{2}|\alpha_1|$. This implies that if $|\alpha_2| \geq 2k$ then $|\alpha_1| > 2|\alpha_2|$ because otherwise it would contain a square longer than k.

Since this holds for any pair of admissible factors detected while i is strictly smaller than $q-2k$, the number of such admissible factors is bounded by $\log_2(q-2k) < \log(n)$. So, in Algorithm 1, as i runs from 0 to $2n-1$, the number of admissible factors is bounded by :

- $\log n$ for i from 0 to $q-2k$.
- $4k$ for i from $q-2k+1$ to $q+2k-1$.
- $\log n$ for i from $q+2k$ to $2n-1$.

Summing up all these provides the bound. ∎

Theorem 12 *Let $w = \mathbf{b}(P) \in \Sigma^{2n}$ be a k-square-free word encoding the boundary of a polyomino P, with $k \in \mathcal{O}(\sqrt{n})$. Determining if w codes a* pseudo-hexagon *is decidable in linear time.*

Proof. The idea is to construct convenient, and not too long, lists of admissible factors and then to use the constant time LCRE function.

Algorithm 2
Input : $w = \mathbf{b}(P) \in \Sigma^{2n}$
0 : **Build** L_1 : *list of all admissible factors containing position* p;
1 : $m :=$ *position of the rightmost letter of* w *included in a factor of* L_1;
2 : **Build** L_2 : *list of all admissible factors containing position* $(m+1)$;
3 : **For** *all* $A \in L_1$ **do**
4 : **For** *all* $B \in L_2$ **do**
5 : **If** $w \equiv ABx\widehat{A}\widehat{B}y$ **or** $w \equiv AxB\widehat{A}y\widehat{B}$ **then**
6 : **Compute** i : *position of* x *in* w;
7 : **Compute** j : *position of* $\widehat{y}$ *in* $\widehat{w}$;
8 : **If** $\mathrm{LCRE}(w, \widehat{w}, i, j) = |x|$ **then** P *is a* pseudo-hexagon;
9 : **end if**
10 : **end if**
11 : **end for**
12 : **end for**

Since w is k-square-free, Lemma 11 ensures that L_1 and L_2 each contain less than $4k + 2\log n$ elements. The nested loops perform at most $(4k+2\log n)^2$ iterations, and thus, the overall complexity is $\mathcal{O}(n + (4k+2\log n)^2) = \mathcal{O}(n)$.
∎

The k-square-freeness condition in the hypothesis of Theorem 12 is sufficient to ensure the linear complexity, but not necessary as shown in the following examples. Indeed, let $w \equiv a^{t+1}bab\overline{a}^t\overline{b}\overline{a}^2\overline{b}$, shown in Fig. 10 (a) with $t = 7$: the longest square is a^8 and $8 > \sqrt{22}$ which does not satisfy the condition. In that case the result is provided still in linear time, and there is a unique solution.

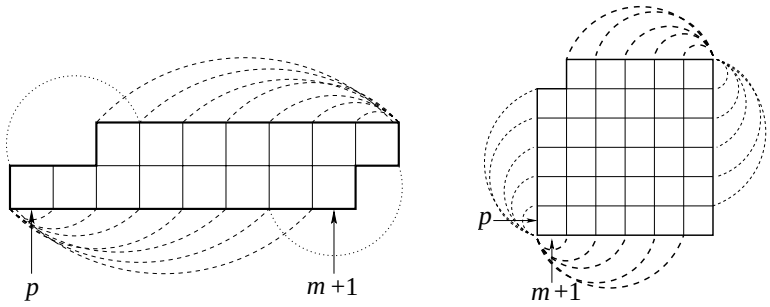


Fig. 10. (a) a pseudo-hexagon; (b) a bad pseudo-hexagon.

On the other hand, the polyomino in Fig. 10 (b) is coded by $w \equiv a^t b^t \overline{a}^{t-1} \overline{b}\,\overline{a}\,\overline{b}^{t-1}$ with $t = 6$. In that case the longest square is a^6 while the length of $|w| = 24$, so that $6 > \sqrt{24}$ which violates the condition: the solution is still unique but the loops in Algorithm 2 run through two lists of size $\mathcal{O}(n)$, providing the result in $\mathcal{O}(n^2)$. This example also shows that checking all the possible pairs is sometimes unavoidable.

4 Polyominoes on the hexagonal lattice

On a honeycomb lattice, a polyomino is encoded by a word on a 6-letter alphabet $\Sigma = \{a, \overline{a}, b, \overline{b}, c, \overline{c}\}$ identified with the set of unit steps $\{\rightarrow, \leftarrow, \nearrow, \swarrow, \nwarrow, \searrow\}$ representing the sequence of steps on its boundary. For in-

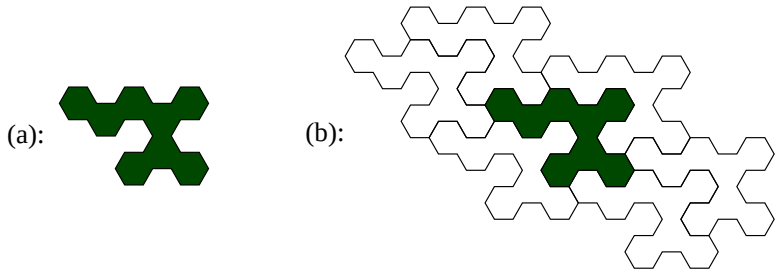


Fig. 11. (a) an exact polyomino; (b) the associated tiling.

stance, the boundary $\mathbf{b}(P)$ of the polyomino in Figure 11 (a), in a counter-clockwise manner starting from the bottom left point, is coded by the word $w = a\,b\,a\,\overline{c}\,a\,b\,c\,\overline{a}\,c\,b\,a\,b\,c\,\overline{a}\,\overline{b}\,\overline{a}\,c\,\overline{a}\,\overline{b}\,\overline{a}\,c\,\overline{a}\,\overline{b}\,\overline{c}\,a\,\overline{c}\,a\,b\,a\,\overline{c}\,\overline{b}\,\overline{a}\,\overline{b}\,\overline{c}$, and its boundary may be factorized as

$$\mathbf{b}(P) = b\,a\,\overline{c}\,a\,b \cdot c \cdot \overline{a}\,c\,b\,a\,b\,c\,\overline{a}\,\overline{b}\,\overline{a}\,c\,\overline{a} \cdot \overline{b}\,\overline{a}\,c\,\overline{a}\,\overline{b} \cdot \overline{c} \cdot a\,\overline{c}\,a\,b\,a\,\overline{c}\,\overline{b}\,\overline{a}\,\overline{b}\,\overline{c}\,a,$$

satisfying the BN-condition (1).

Not all pairs of units steps are allowed. Indeed the forbidden pairs of steps are

$$\Phi = \{\alpha\alpha \,|\, \alpha \in \Sigma\} \cup \{\alpha\overline{\alpha} \,|\, \alpha \in \Sigma\} \cup \{ac, a\overline{b}, b\overline{a}, b\overline{c}, ca, c\overline{b}, \overline{a}b, \overline{ac}, \overline{b}a, \overline{b}\overline{c}, \overline{ca}, \overline{c}b\}$$

so that these paths are words $w \in \mathcal{P}$ where $\mathcal{P}$ is given by

$$\mathcal{P} = \Sigma^* \setminus (\Sigma^* \cdot \Phi \cdot \Sigma^*) \,. \tag{4}$$

A noticeable difference between square and hexagonal lattices is established now. Indeed, pseudo-squares do not exist on an hexagonal lattice. In order to prove this we need a technical property established in [17,18], extending to hexagonal lattices the Daurat-Nivat result ([5], IWCIA'03) on the number of salient and reentrant points of discrete sets. In this case, we take $\mathcal{L} = \{ab, bc, c\overline{a}, \overline{a}\overline{b}, \overline{b}\overline{c}, \overline{c}a\}$, as the set of "left turns" moves, and $\mathcal{R} = \{a\overline{c}, ba, cb, \overline{a}c, \overline{b}a, \overline{c}b\}$ as the set of "right turns".

Property 13 [B.L.L. [18]] *Let $w \in \mathcal{C}$ be a non-crossing closed path in the hexagonal lattice. Then $|\Delta_{\mathcal{C}}(w)| = 6$.*

Proposition 14 *On an hexagonal lattice, a polyomino P tiles the plane by translation if and only if it is a pseudo-hexagon.*

Proof. It suffices to show that there is no pseudo-square BN-factorization of the boundary of $w = \mathbf{b}(P)$. By contradiction, assume that $w \equiv A \cdot B \cdot \widehat{A} \cdot \widehat{B}$. Applying the $\Delta_{\mathcal{C}}$ function, we have

$$\begin{aligned} \Delta_{\mathcal{C}}(w) = & \Delta(A) + \Delta(\ell(A) f(B)) + \Delta(B) + \Delta(\ell(B) f(\widehat{A})) \\ & + \Delta(\widehat{A}) + \Delta(\ell(\widehat{A}) f(\widehat{B})) + \Delta(\widehat{B}) + \Delta(\ell(\widehat{B}) f(A)) \\ & \leq 4 \,, \end{aligned}$$

contradicting Property 13. ∎

4.1 Polyominoes with holes

The results above generalize to more general tilings. Indeed, since the BN-factorization involves path properties, there is no need for a tile to be a polyomino. For instance, the tile T in Figure 12

is a pseudo-hexagon and its BN-factorization is (starting from $\mathbf{S}$)

$$\mathbf{b}(T) \equiv baba \cdot bc \cdot \overline{a}c\overline{a}\overline{b}c\overline{b}\overline{a}c\overline{a}cba\overline{c}abc\overline{a}c \cdot \overline{a}\overline{b}\overline{a}\overline{b} \cdot \overline{c}\overline{b} \cdot \overline{c}a\overline{c}\overline{b}\overline{a}c\overline{a}\overline{b}\overline{c}a\overline{c}abcba\overline{c}a \,.$$

The contour path is non-crossing , that is, it may not cross itself in a (i, j) coordinate point visited twice or more times. In a polyomino each point on the boundary is visited once, while in a non-crossing closed path a point may

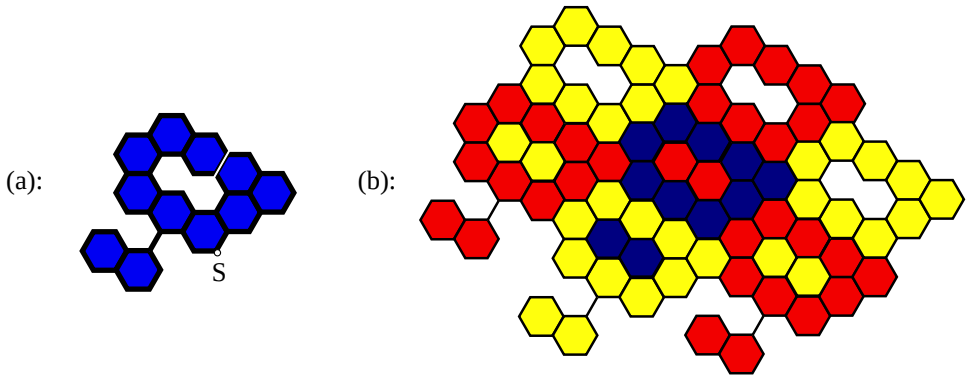


Fig. 12. (a) An hexagonal tile with (b) the associated tiling.

appear several times. A sequence of elementary steps may therefore visit a contiguous sequence of multiple points, defining a maximal null area region between two factors u and $\widehat{u}$ at some fixed positions.

Definition 15 *Let $w \in \mathcal{C}$ be a closed path and $u \in \mathcal{P}$ be such that*

(i) *$w \equiv xuy\widehat{u}$ for some non empty words $x, y \in \mathcal{C}$, and*
(ii) $\Delta\left(\ell(x)f(u)\right) = \Delta\left(\ell(u)f(y)\right) = \Delta\left(\ell(y)f(\widehat{u})\right) = \Delta\left(\ell(\widehat{u})f(x)\right)$.

The pair $u, \widehat{u}$ is called a channel.

A point in a closed path may have an $\mathcal{O}(n)$ multiplicity. In order to avoid pathological cases, we consider only the paths having a point multiplicity bounded by 2. Checking that a closed path is non-crossing requires some analysis, so we proceed with a criterion for checking that property. A channel

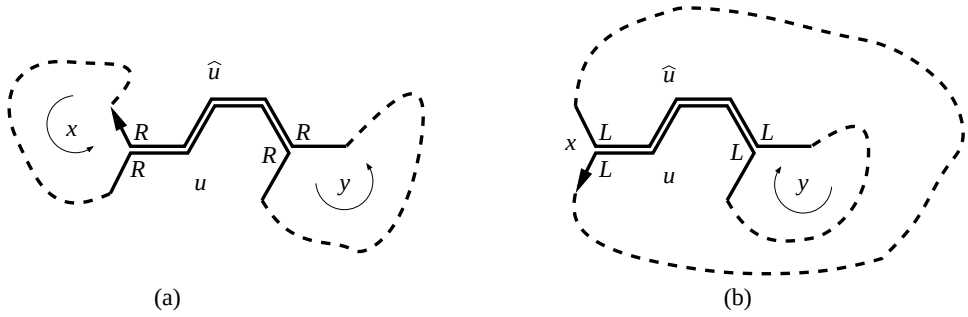
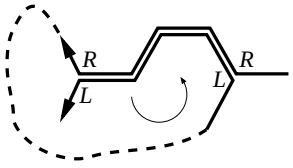


Fig. 13. Two types of channels.

(Figure 13) in any closed path $w \in \mathcal{P}$ is a maximal sequence of points having multiplicity 2 and bordered by two nonempty closed path. Note that the following case is excluded since it leads necessarily to a crossing path: indeed, it



contains a factor $\alpha\beta\alpha$ where $\alpha \neq \varepsilon$, with $\alpha\beta \in \mathcal{C}$. Note that a closed path may have several instances of a channel. It follows that a closed path is non-crossing if and only if every multiple point occurs in a channel.

At this point we need to generalize Proposition 4 to non-crossing paths.

Proposition 16 *Let $w \in \mathcal{C}$ be a non-crossing closed path, and let p be any fixed position in w. Let X be the set of all admissible factors overlapping the position p and $\widehat{X}$ be the set of their respective homologous factors. Then, there exists at least one position in w that is not covered by any element of $X \cup \widehat{X}$.*

Proof. The proof is an adaptation-continuation of that of Proposition 4, with the same notation. We have to go one step further in the two considered cases. In case 1, we use Property 13 instead of Property 3.

In case 2, since x is a prefix of A, $\widehat{x}$ is a suffix of $\widehat{A}$. This implies that $\widehat{x}V\alpha x$ is a suffix of $\widehat{B}$ and $\widehat{x}\widehat{\alpha}\widehat{V}x$ is a prefix of B. But $\widehat{x}\widehat{\alpha}\widehat{V}\beta$ is also a prefix of B so that x and β share a common nonempty prefix. Let u be such a common nonempty prefix and u' be such that $\beta\alpha = uu'$. The factor $\beta\widehat{y}$ of w is equal to $\beta\alpha x$ containing the factor $uu'u$ where uu' is a closed path. It is not a channel since it leads necessarily to either a multiplicity greater than 2 or a crossing. Contradiction.

In case 3, since $\widehat{\alpha}\gamma\alpha\beta$ is a closed loop and $\widehat{\alpha}\gamma\alpha\beta x$ is a factor of w, it suffices to show that $\widehat{\alpha}$ and x share a nonempty prefix. Let a be the first letter of x. By construction, a is also the first letter of A, and $\overline{a}$ is the last letter of $\widehat{A}$. Since α is a suffix of $\widehat{A}$, $\widehat{\alpha}$ also starts by the letter a. Contradiction. ∎

5 Concluding remarks

Corollary 5, the subsequent lemmas, and Theorem 12 apply straightforwardly to non-crossing paths (and therefore polyominoes) on the hexagonal lattice. On the other hand, the notion of *channel* applies to non-crossing paths over the square lattice but non-trivially since the square lattice allows pairs of unit steps that doesn't form a turn. This requires a more technical analysis and will appear in full detail in the second author's thesis [21].

Acknowledgements. The authors are grateful to the anonymous referees for their careful reading and suggestions that improved paper's presentation.

References

[1] R. Berger, The undecidability of the domino problem, Mem. Amer. Math. Soc. 66 (1966) 1–72.

[2] Y. Gurevich, I. Koriakov, A remark on Berger's paper on the domino problem, Siberian Journal of Mathematics 13 (1972) 459–463, (in Russian).

[3] S. W. Golomb, Checker boards and polyominoes, Amer. Math. Monthly 61 (1954) 675–682.

[4] M. Gardner, Mathematical games, Scientific American (1958) Sept. 182–192, Nov. 136–142.

[5] A. Daurat, M. Nivat, Salient and reentrant points of discrete sets, in: A. del Lungo, V. di Gesu, A. Kuba (Eds.), Proc. IWCIA'03, International Workshop on Combinatorial Image Analysis, Electronic Notes in Discrete Mathematics, Elsevier Science, Palermo, Italia, 2003.

[6] S. W. Golomb, Polyominoes: Puzzles, Patterns, Problems, and Packings, Princeton Academic Press, Princeton, 1996.

[7] E. Weisstein, Polyomino, from Wolfram Mathworld, Available electronically at http://mathworld.wolfram.com/Polyomino.html.

[8] H. A. G. Wijshoff, J. van Leeuven, Arbitrary versus periodic storage schemes and tesselations of the plane using one type of polyomino, Inform. Control 62 (1984) 1–25.

[9] H. Freeman, On the encoding of arbitrary geometric configurations, IRE Trans. Electronic Computer 10 (1961) 260–268.

[10] H. Freeman, Boundary encoding and processing, in: B. Lipkin, A. Rosenfeld (Eds.), Picture Processing and Psychopictorics, Academic Press, New York, 1970, pp. 241–266.

[11] J.-P. Braquelaire, A. Vialard, Euclidean paths: A new representation of boundary of discrete regions, Graphical Models and Image Processing 61 (1999) 16–43.

[12] D. Beauquier, M. Nivat, On translating one polyomino to tile the plane, Discrete Comput. Geom. 6 (1991) 575–592.

[13] I. Gambini, L. Vuillon, An algorithm for deciding if a polyomino tiles the plane by translations, Theoret. Informatics Appl. 41 (2007) 147–155.

[14] M. Lothaire, Applied Combinatorics on Words, Cambridge University Press, Cambridge, 2005.

[15] D. Gusfield, J. Stoye, Linear time algorithms for finding and representing all the tandem repeats in a string, Journal of Computer and System Sciences 69 (2004) 525–546.

[16] D. Gusfield, Algorithms on Strings, Trees and Sequences, Cambridge University Press, Cambridge (UK), 1997.

[17] S. Brlek, G. Labelle, A. Lacasse, A note on a result of Daurat and Nivat, in: C. de Felice, A. Restivo (Eds.), Proc. DLT 2005, 9-th International Conference on Developments in Language Theory, No. 3572 in LNCS, Springer-Verlag, Palermo, Italia, 2005, pp. 189–198.

[18] S. Brlek, G. Labelle, A. Lacasse, Properties of the contour path of discrete sets, Int. J. Found. Comput. Sci. 17 (3) (2006) 543–556.

[19] L. Ilie, A note on the number of distinct squares in a word, in: S. Brlek, C. Reutenauer (Eds.), Proc. Words2005, 5-th International Conference on Words, Vol. 36, Publications du LaCIM, Montreal, Canada, 2005, pp. 289–294.

[20] M. Lothaire, Combinatorics on Words, Cambridge University Press, Cambridge, 1997.

[21] X. Provençal, Ph.D. thesis, UQAM, Montréal, 2008, to appear.