Lyndon + Christoffel = Digitally Convex ⋆

S. Brlek [a], J.-O. Lachaud [b], X. Provençal [a], C. Reutenauer [a],

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

[b] *Laboratoire de Mathématiques, UMR 5127 CNRS, Université de Savoie,*
73376 Le Bourget du Lac, France

Abstract

Discrete geometry redefines notions borrowed from Euclidean geometry creating a need for new algorithmical tools. The notion of convexity does not translate trivially, and detecting if a discrete region of the plane is convex requires a deeper analysis. To the many different approaches of digital convexity, we propose the combinatorics on words point of view, unnoticed until recently in the pattern recognition community. In this paper we provide first a fast optimal algorithm checking digital convexity of polyominoes coded by their contour word. The result is based on linear time algorithms for both computing the Lyndon factorization of the contour word, and the recognition of Christoffel factors that are approximations of digital lines. By avoiding arithmetical computations the algorithm is much simpler to implement and much faster in practice. We also consider the convex hull computation and relate previous work in combinatorics on words with the classical Melkman algorithm.

Key words: Digital Convexity, Lyndon words, Christoffel words, Convex hull

1 Introduction

In Euclidean geometry, a given region R is said to be *convex* if and only if for any pair of points p_1, p_2 in R the line segment joining p_1 to p_2 is completely included in R. In discrete geometry on square grids, the notion does

⋆ with the support of NSERC (Canada) and Canada Research Chair
Email addresses: brlek.srecko@uqam.ca (S. Brlek), jacques-olivier.lachaud@univ-savoie.fr (J.-O. Lachaud), xavierprovencal@gmail.com (X. Provençal), reutenauer.christophe@uqam.ca (C. Reutenauer).

not translate trivially, since the only convex (in the Euclidean sense) regions are isolated points when pixels are considered as points in the plane, or rectangles when pixels are seen as unit squares. Many attempts have been made to fill the gap, and a first definition of discrete convexity based on discretization of continuous object came from Sklansky [1] and Minsky and Papert [2]: it is not convenient since, in some cases, sets of arbitrary distant points are considered digitally convex (see Fig. 1).

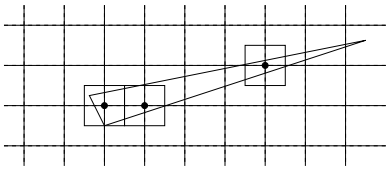


Fig. 1. The discretization of a continuous objet resulting in a non-connected set.

Later, Kim [3,4] then Kim and Rosenfeld [5] provided several equivalent characterizations of discrete convex sets, and finally Chaudhuri and Rosenfeld [6] proposed a new definition of digital convexity based this time on the notion of digital line segments (see [7] for a review of digital straightness). Given a finite subset S of $\mathbb{Z}^2$ its *convex hull* is defined as the intersection of all Euclidean convex sets containing S. Of course all the vertices of the convex hull are points from S. Therefore, throughout this work, a polyomino P (which is the interior of a closed non-intersecting grid path of $\mathbb{Z}^2$) is called *convex* if and only if its convex hull contains no integer point outside P. This definition of convexity is called **H-convexity** by Eckhardt in his review on digital convexity [8]. Since we only consider polyominoes which are 4-connected subsets of $\mathbb{Z}^2$, these objects are considered as convex under most common definitions of digital convexity (see for instance Eckardt for a precise statement of several equivalences). Debled-Rennesson et al. [9] already provided an algorithm deciding if a given polyomino is convex. Their method uses arithmetical tools to compute the sequence of maximal segments along the polyomino. Digital convexity is then achieved if and only if the induced sequence of slopes is monotonous. From this observation, a linear — and thus optimal — time algorithm is obtained using optimal time methods for computing the tangential cover of a digital contour, which relies on a moving digital straight line recognition algorithm [10,11]. It is worthy to note that digital convex sets and hulls have specific properties that are not shared by their Euclidean counterpart. For instance, Hübler et al [12] (or easiest to find [13]) have shown that the convex hull of any 4-connected set of discrete points can be computed in linear time. Even more surprising, the digital convex hull of an implicitly defined euclidean convex body can be computed in sublinear time $O(d^{2/3} \log d)$, where d is the diameter of the set [14].

In their survey on digital straightness, Klette and Rosenfeld [7] mention that

"*Related work even earlier on the theory of words, specifically, on mechanical*

or Sturmian words, remained unnoticed in the pattern recognition community."

Indeed, the study of words goes back to Bernouilli, Markov, Thue and Morse (see Lothaire's books [15–17] for an exhaustive bibliographic account) and suggest to build a bridge between discrete geometry and combinatorics on words which will benefit to both areas. This approach gave an elementary proof and a generalization [18] of a result of Daurat and Nivat [19] relating salient and reentrant points in discrete sets. It also provided an optimal algorithm for recognizing tiles that tile the plane by translation [20,21].

Here we study the problem of deciding whether a polyomino coded by its contour word, also called Freeman chain code, is convex or not. To achieve this we use well known tools of combinatorics on words. The first is the existence of a unique Lyndon factorization, and its optimal computation by a linear algorithm (Fredricksen and Maiorana [22], Duval [23]). The second concerns the Christoffel words, a class of finite factors of Sturmian words, that are discrete approximations of straight lines. After recalling the combinatorial background and basic properties, we propose another linear time algorithm deciding convexity of polyominoes. This new purely discrete algorithm is however much simpler to implement and some experiments reveal that it is 10 times faster than previous linear algorithms. Furthermore, one of its main interests lies in the explicit link between combinatorics on words and discrete geometry. Since our method does not rely on geometric and vector computations, it also shows that digital convexity is much more fundamental and abstract property than general convexity.

We also consider the convex hull computation in our framework: while the classical Melkman algorithm [24] applies to any simple polygonal curve, our algorithm applies to words coding the contour of a polyomino and appears as a discrete version of it.

This paper is an extended and enhanced version of a paper presented at the 14th International Conference on Discrete Geometry for Computer Imagery (DGCI 2008) held in Lyon (Fance) on April 14-16, 2008 [25].

2 Preliminaries

A word w is a finite sequence of letters $w_1 w_2 \cdots w_n$ on a finite alphabet Σ, that is a function $w : [1..n] \longrightarrow \Sigma$, and $|w| = n$ is its *length*. Consistently its number of a letters, for $a \in \Sigma$, is denoted $|w|_a$. The set of words of length n is denoted Σ^n and the set of all finite words is Σ^*, the free monoid on Σ. The empty word is denoted ε and by convention $\Sigma^+ = \Sigma^* \setminus \{\varepsilon\}$. The k-th power of

word w is defined by $w^k = w^{k-1} \cdot w$ with the convention that $w^0 = \varepsilon$. A word is said *primitive* when it is not the power of a nonempty word. A *period* of a word w is a number p such that $w_i = w_{i+p}$, for all $1 \leq i \leq |w| - p$. The *reversal* of a word $w_1 w_2 \cdots w_n$ is the word $\widetilde{w} = w_n w_{n-1} \cdots w_2 w_1$, and a *palindrome* is a word such that $w = \widetilde{w}$. The set of palindromes in Σ^* is denoted $\mathrm{PAL}(\Sigma)$.

Given a total order $<$ on Σ, the *lexicographic ordering* extends this order to words on Σ by using the following rule :

$$w < w' \text{ if either } \quad \text{(i) } w' \in w\Sigma^+, \qquad \text{(ii) } w = uav \text{ and } w' = ubv' \text{ with } a < b, a, b \in \Sigma, u \in \Sigma^*.$$

Two words w, w' on the alphabet Σ are said to be *conjugate*, written $w \equiv w'$, if there exist u, v such that $w = uv$ and $w' = vu$. The conjugacy class of a word is defined as the set of all its conjugates and is equivalent to the set of all circular permutations of its letters.

Let w be a finite word over the alphabet $\{0, 1\}$. We denote by $\overrightarrow{w}$ the vector $(|w|_0, |w|_1)$. For any word w, the partial function $\phi_w : \mathbb{N} \longrightarrow \mathbb{Z} \times \mathbb{Z}$ associates to any integer j, $1 \leq j \leq |w|$, the vector $\phi_w(j) = \overrightarrow{w_1 w_2 \cdots w_j}$. In other words, this map draws the word as a 4-connected path in the plane starting from the origin, going *right* for a letter 0 and *up* for a letter 1 (see Fig. 2). This

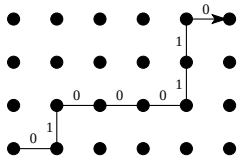


Fig. 2. Path encoded by the word $w = 01000110$.

extends naturally to more general paths by using the four letter alphabet $\Sigma = \{0, 1, \bar{0}, \bar{1}\}$, associating the letter $\bar{0}$ to a *left* step and $\bar{1}$ to a *down* step. This notation allows to code the border of any polyomino by a 4-letter word known as the Freeman chain code.

The lexicographic order $<$ on points of $\mathbb{R}^2$ or $\mathbb{Z}^2$ is such that $(x, y) < (x', y')$ when either $x < x'$ or $x = x'$ and $y < y'$. The *convex hull* of a finite set S of points in $\mathbb{R}^2$ is the intersection of all convex sets containing these points and is denoted by $\mathrm{Conv}(S)$. S being finite, it is clearly a polygon in the plane whose vertices are elements of S. The *upper convex hull* of S, denoted by $\mathrm{Conv}^+(S)$, is the clockwise oriented sequence of consecutive edges of $\mathrm{Conv}(S)$ starting from the lowest vertex and ending on the highest vertex. The *lower convex hull* of S, denoted by $\mathrm{Conv}^-(S)$, is the clockwise oriented sequence of consecutive edges of $\mathrm{Conv}(S)$ starting from the highest vertex and ending on the lowest vertex.

3 Combinatorics on words

Combinatorics on words has imposed itself as a powerful tool for the study of large number of discrete, linear, non-commutative objects. Such objects appears in almost any branches of mathematics and discrete geometry is not an exception. Traditionally, discrete geometry works on characterization and recognition of discrete objects using arithmetic approach or computational geometry. However combinatorics on words provide mathematical tools and efficient algorithms to address this problem as already mentioned. Lothaire's books [15–17] constitute the reference for presenting a unified view on combinatorics on words and many of its applications.

3.1 Lyndon words

Introduced as *standard lexicographic sequences* by Lyndon in 1954, Lyndon words have several characterizations (see [15]). We shall define them as words being strictly smaller than any of their circular permutations.

Definition 1 *A Lyndon word $l \in \Sigma^+$ is a word such that $l = uv$ with $u, v \in \Sigma^+$ implies that $l < vu$.*

Note that Lyndon words are always primitive. An important result about Lyndon words is the following unique factorization theorem (see Lothaire [15] Theorem 5.1.1).

Theorem 2 *Any word $w \in \Sigma^+$ admits a unique factorization as a sequence of decreasing Lyndon words :*

$$w = l_1^{n_1} l_2^{n_2} \cdots l_k^{n_k}, \quad l_1 > l_2 > \cdots > l_k \tag{1}$$

where $n_i \geq 1$ and l_i is a Lyndon word, for all i such that $1 \leq i \leq k$.

There exist several algorithms for factorizing a word $w = w_1 w_2 \cdots w_n$ into Lyndon words.

1. The naive one, which mimics the proof of existence of the factorization (see e.g.[15], Theorem 5.1.5 and its proof) is the following :

L1. start with the list $(l_1, l_2, \ldots, l_n) \leftarrow (w_1, w_2, \ldots, w_n)$;
L2. if $l_i < l_{i+1}$ for some i, then $(l_1, l_2, \ldots, l_n) \leftarrow (l_1, \ldots, l_{i-1}, l_i l_{i+1}, l_{i+2}, \ldots, l_n)$, and stop otherwise.

2. The second algorithm rests on the following result : let $w = l_1 l_2 \cdots l_k$ be the factorization of w into a nonincreasing product of Lyndon words. Then l_k

is the smallest suffix of w for the lexicographical order (see [15], Proposition 5.1.6).

3. The previous algorithms are not linear. A linear algorithm, which is moreover very subtle and elegant, was found by Duval [23]. It works by reading from left to right, with at most $2n$ comparisons of letters (see also [26], Section 7.4). We use this algorithm later and it is explained in full detail in Section 5.

4. Another algorithm, which turns out to be linear, rests on the concept of *suffix standardization* of the word w, that is the permutation σ on $[1, 2, \ldots, n]$ such that the j-th suffix of w for the lexicographical order is $w_i w_{i+1} \cdots w_n$, where $\sigma(i) = j$. For example, let

$$w = 1\,0\,1\,1\,0\,1\,0\,0\,1.$$

Then its suffix-standard permutation is

$$\sigma = 8\,4\,9\,7\,3\,6\,1\,2\,5,$$

since the suffixes of w in increasing order are 001, 01, 01001, 01101001, 1, 1001, 101001, 101101001, 1101001. Now, consider the sequence of *left-to-right minima* of σ : a left-to-right minimum of σ is an index k such that $\sigma(i) > \sigma(k)$ for any $i < k$. In the example, the left-to-right minima are $k = 1, 2, 5, 7$ corresponding to the values $\sigma(k) = 8, 4, 3, 1$. Then the Lyndon factorization of w is obtained by cutting w just before each left-to-right minimum of its suffix-standard permutation σ :

$$\begin{aligned} \sigma &= \underline{8}\,\underline{4}\,9\,7\,\underline{3}\,6\,\underline{1}\,2\,5, \\ w &= (1)\,(0\,1\,1)\,(0\,1)\,(0\,0\,1). \end{aligned}$$

This algorithm is linear : indeed, suffix standard permutation is equivalent to the *suffix array* of w, which may be computed in linear time [27]. Moreover, the sequence of left-to-right minima of a permutation is clearly computable in linear time. This algorithm was known to Duval and Lefebvre ([28], p. 250), who attribute it to Crochemore [27]. Note that suffix standardization may be used also to build the tree (or nonassociative word) associated to a Lyndon word, hence the associated Lie polynomial, which gives the Lyndon basis of the free Lie algebra (see [29]). Finally, note that the suffix standardization, when applied to a Christoffel word w of length n, gives the Cayley graph of $\mathbb{Z}/n\mathbb{Z}$ which generated it. For example for $w = 0010101$, we obtain $\sigma = 1473625$, and w is recovered by reading cyclically σ and writing 0 for an ascent and 1 for a descent (see [30]).

3.2 Christoffel words

Introduced by Christoffel [31] in 1875 and reinvestigated recently by Borel and Laubie [32] who pointed out some of their geometrical properties, Christoffel words reveal an important link between combinatorics on words and discrete geometry.

This first definition of Christoffel word, borrowed from Berstel and de Luca [33], highlights their geometrical properties and helps to understand the main result of this work stated in Proposition 7. Let $\Sigma = \{0, 1\}$. The *slope* of a word is a map $\rho : \Sigma^* \to \mathbb{Q} \cup \{\infty\}$ defined by

$$\rho(\epsilon) = 1, \quad \rho(w) = |w|_1/|w|_0, \quad \text{for } w \neq \epsilon.$$

It is assumed that $1/0 = \infty$. It corresponds to the slope of the straight line joining the first and the last point of the path coded by w. For each k, $1 \leq k \leq |w|$, we define the set

$$\delta_k(w) = \{u \in \Sigma^k | \rho(u) \leq \rho(w)\},$$

of words of length k whose slope is not greater than the slope of w. The quantity

$$\mu_k(w) = \max\{\rho(u) | u \in \delta_k(w)\}$$

is used to define Christoffel words (see Fig. 3).

Definition 3 *A word w is a Christoffel word if for any prefix v of w one has $\rho(v) = \mu_{|v|}(w)$.*

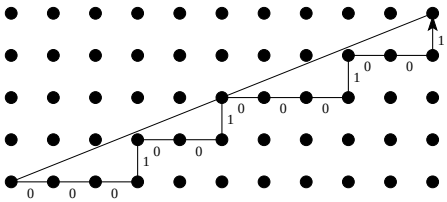


Fig. 3. The path coded by the Christoffel word $w = 00010010001001$ staying right under the straight line of slope $r = 2/5$.

A direct consequence of this definition is that given a Christoffel word $u^r = v^s$ for some $r, s \geq 1$, both words u and v are also Christoffel words. From an arithmetical point of view, discrete lines (see Reveillès [34]) are defined by a pair of Diophantine inequalities.

Definition 4 *Three integers a, b, c such that a and b are relatively prime define the* 4-connected straight line *$D_{a,b,c}$ of slope a/b as the set :*

$$D_{a,b,c} = \left\{(x, y) \in \mathbb{Z}^2 | c \leq ax - by < c + |a| + |b|\right\}.$$

Among the points of the set $D_{a,b,c}$ those which verify the equation $ax - by = c$ are called the *upper leaning points*, and a Christoffel word is a connected subset of a 4-connected straight line joining upper leaning points (see Fig. 4).

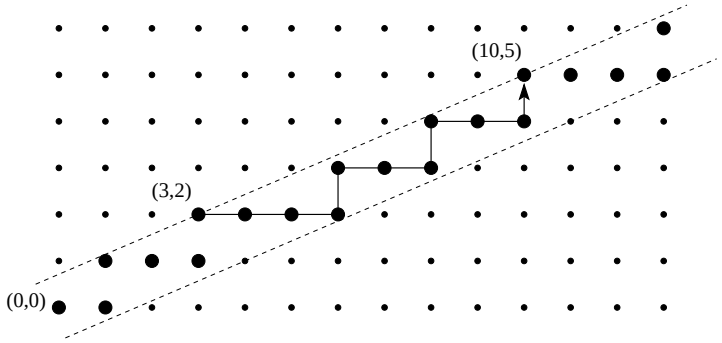


Fig. 4. The 4-connected straight line $D_{3,7,-5} = \{(x, y) \in \mathbb{Z}^2 | -5 \leq 3x - 7y < 5\}$ and the path associated to the primitive Christoffel word of slope 3/7 joining two upper leaning points.

The next properties are taken from Borel and Laubie [32].

Property 1 [Borel and Laubie[32]] *Christoffel words satisfy the following properties.*

(i) *All primitive Christoffel words are Lyndon words.*
(ii) *Given c_1 and c_2 two Christoffel words, $c_1 < c_2$ iff $\rho(c_1) < \rho(c_2)$.*
(iii) *Given $r \in \mathbb{Q}^+ \cup \{\infty\}$, let F_r be the set of words w on the alphabet $\{0, 1\}$ such that $\rho(v) \leq r$ for all non-empty prefix v of w. F_r corresponds to the words coding paths starting from $(0, 0)$ and staying below the Euclidean straight line of slope r. Among these paths, those being the closest ones to the line and having their last point located on it are Christoffel words.*

Originally Christoffel [31] defined these words as follows. Given $k < n$ two relatively prime numbers, a (primitive) Christoffel word $w = w_1 w_2 \dots w_n$ is defined by :

$$w_i = \begin{cases} 0 & \text{if } r_{i-1} < r_i, \\ 1 & \text{if } r_{i-1} > r_i, \end{cases}$$

where r_i is the remainder of $(i\,k) \mod n$. The slope of this word is then $k/(n-k)$. For example if $k = 5$ and $n = 8$, we have the following values

i	0	1	2	3	4	5	6	7	8
$(ik) \bmod n$	0	5	2	7	4	1	6	3	0
w_i		0	1	0	1	1	0	1	1

and the Christoffel word is $w = 01011011$.

In [33] Berstel and de Luca provided an alternative characterization of primitive Christoffel words. Let CP be the set of primitive Christoffel words, PAL the set of palindromes and PER the set of words w having two periods p and q such that $|w| = p + q - 2$. The following relations hold :

$$\mathrm{CP} = \Big(\{0,1\} \cup 0 \cdot \mathrm{PER} \cdot 1\Big) \subset \Big(\{0,1\} \cup 0 \cdot \mathrm{PAL} \cdot 1\Big).$$

These properties of Christoffel words are useful for deciding if a given word is Christoffel or not.

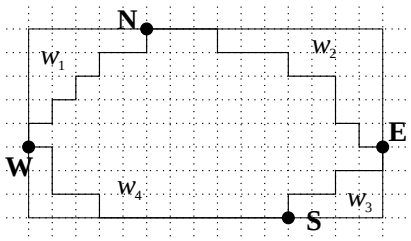
4 Digital convexity

There are several (more or less) equivalent definitions of digital convexity, depending on whether or not one asks the digital set to be connected. We say that a finite 4-connected subset S of $\mathbb{Z}^2$ is *digitally convex* if it is the Gauss digitization of a convex subset X of the plane, i.e. $S = \mathrm{Conv}(X) \cap \mathbb{Z}^2$.

The *border* $\mathrm{Bd}(S)$ of S is the 4-connected path that follows clockwise the pixels of S that are 8-adjacent to some pixels not in S. This path is a word in $\{0, 1, \bar{0}, \bar{1}\}^*$, starting by convention from the lowest point and in clockwise order.

Definition 5 *A word w is said to be* digitally convex *if it is conjugate to the word coding the border of some finite 4-connected digitally convex subset of $\mathbb{Z}^2$.*

Note that implicitly, a digitally convex word is necessarily closed. Now, every closed path coding the boundary of a region is contained in a smallest rectangle such that its contour word w may be factorized as follows. Four extremal points are defined by their coordinates:



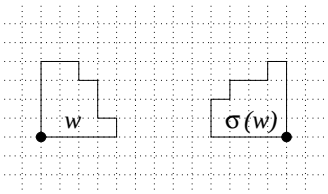
W is the lowest on the Left side;
N is the leftmost on the Top side;
E is the highest on the Right side;
S is the rightmost on the Bottom side;
So that $w \equiv w_1 w_2 w_3 w_4$.

This factorization is called the *standard decomposition*. We say that a word w_1 in $\{0,1\}^*$ is *NW-convex* iff there are no integer points between the upper convex hull of the points $\{\phi_{w_i}(j)\}_{j=1\ldots|w_i|}$ and the path w_i.

Define the counterclockwise $\pi/2$ circular rotation by

$$\sigma : (0, 1, \bar{0}, \bar{1}) \longmapsto (1, \bar{0}, \bar{1}, 0),$$

as illustrated in the following example.



$$w = 1111001\bar{0}1\bar{1}\bar{1}0\bar{1}\bar{0}\bar{0}\bar{0}\bar{0}\bar{0},$$
$$\sigma(w) = \bar{0}\bar{0}\bar{0}\bar{0}11010010\bar{1}\bar{1}\bar{1}\bar{1}.$$

Then w_2 in $\{0, \bar{1}\}^*$ is *NE-convex* iff $\sigma(w_2)$ is NW-convex, and more generally, for the standard decomposition $w \equiv w_1w_2w_3w_4$ we have

$$w_i \quad \text{is convex} \iff \sigma^{i-1}(w_i) \quad \text{is NW-convex.}$$

Clearly, the convexity of w requires the convexity of each w_i for $i = 1, 2, 3, 4$ and we have the following obvious property.

Proposition 6 *Let $w \equiv w_1w_2w_3w_4$ be the standard decomposition of a polyomino. Then w is digitally convex iff $\sigma^{i-1}(w_i)$ is NW-convex, for all i.*

Let $\mathrm{Alph}(w)$ be the set of letters of w. Observe that if for some i, w_i contains more than 2 letters, that is if $\mathrm{Alph}(\sigma^{i-1}(w_i)) \not\subseteq \{0, 1\}$, then w is not digitally convex.

We are now in position to state the main result which is used in Section 5 to design an efficient algorithm for deciding if a word is convex.

Proposition 7 *A word v is NW-convex iff its unique Lyndon factorization $l_1^{n_1}l_2^{n_2}\cdots l_k^{n_k}$ is such that all l_i are primitive Christoffel words.*

In order to prove Proposition 7, we first need the following lemmas.

Lemma 8 *Let $v \in \{0, 1\}^*$ be a word coding an NW-convex path and let e be one of the edges of its convex hull. The factor u of v corresponding to the segment of the path determined by e is a Christoffel word.*

This is a direct consequence of Property 1(iii) since both the starting and ending points of an edge of the convex hull of a discrete figure are necessarily part of its border. Note that we also have by the same Property 1(iii) that

Lemma 9 *Any Christoffel word is NW-convex.*

We may now proceed to the proof of Proposition 7.

Proof. Let v be a word coding a NW-convex path and let the ordered sequence of edges $(e_1, e_2, \ldots, e_k)$ be the border of its convex hull. For each i from 1 to k, let u_i be the factor of v determined by the edge e_i so that $v = u_1u_2\cdots u_k$. Let l_i be the unique primitive word such that $u_i = l_i^{n_i}$. By Lemma 8, u_i is

a Christoffel word, which implies that l_i is a primitive Christoffel word. By Property 1(i), l_i is also a Lyndon word. Now, since $(e_1, e_2, \ldots, e_k)$ is the convex hull of w, it follows that the slope s_i of the edge e_i is greater than the slope s_{i+1} of the edge e_{i+1} leading to the following inequality :

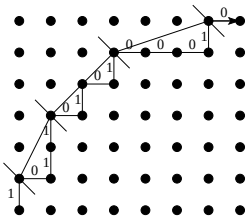
$$\rho(l_i) = \rho(u_i) = s_i > s_{i+1} = \rho(u_{i+1}) = \rho(l_{i+1}).$$

By Property 1(ii) we conclude that $l_i > l_{i+1}$. Thus $l_1^{n_1} l_2^{n_2} \cdots l_k^{n_k}$ is the unique factorization w as a decreasing sequence of Lyndon words.

Conversely, let $v \in \{0,1\}^+$ be such that its Lyndon factorization $l_1^{n_1} l_2^{n_2} \cdots l_k^{n_k}$ consists of primitive Christoffel words. For each i from 1 to k, let e_i be the segment joining the starting point of the path coded by $l_i^{n_i}$ to its ending point. We shall show that $(e_1, e_2, \ldots, e_k)$ is the upper convex hull of ϕ_v. Since $l_i^{n_i}$ is a Christoffel word, Lemma 9 ensures that the path always stays below the segment. By hypothesis, $l_i > l_{i+1}$. Using the same argument as before we have that the slope of e_i is strictly greater than the slope of e_{i+1}. We have just built a sequence of edges whose vertices are points of ϕ_v, which stay above it, and with decreasing slopes. $(e_1, e_2, \ldots, e_k)$ is thus exactly the upper convex hull of ϕ_v.

Now, for all $i = 1 \ldots k$, since $l_i^{n_i}$ is Christoffel (Lemma above), there are no integer points lying between e_i and $\phi_{l_i^{n_i}}$. It follows that there are no integer points between ϕ_v and its upper convex hull, which means v is NW-convex. □

For example, consider the following NW-convex path $v = 1011010100010$.



The Lyndon factorization of v is

$$v = (1)^1 \cdot (011)^1 \cdot (01)^2 \cdot (0001)^1 \cdot (0)^1,$$

where 0, 011, 01, 0001 and 0 are all Christoffel words.

It is interesting to note that many results in the theory of Sturmian word have been obtained by using geometrical properties. This close relation between the two domains raises a number of combinatorial problems such as enumeration of convex words of given length. Another interesting property is the factorial closure of digitally convex words: while a geometrical proof is rather easy to obtain, it begs for a purely combinatorial proof. A formal language characterization would also be of interest. These more theoretical issues goes beyond the scope of this paper and will be addressed in a future work.

5 Algorithm to check word convexity

We give now a linear time algorithm checking digital NW-convexity for paths encoded on $\{0,1\}$. This is achieved in two steps: first we compute the prefix $l_1^{n_1}$ of the word w using the Fredricksen and Maiorana algorithm [22] (rediscovered by Duval [23]), and then Algorithm 2 below checks that the Lyndon factor $l_1 \in \mathrm{CP}$. Iterating this process on all Lyndon factors of w provides the answer whether all l_i are primitive Christoffel words.

Given a word $w \in \Sigma^*$ whose Lyndon factorization is $w = l_1^{n_1} l_2^{n_2} \dots l_k^{n_k}$, the following algorithm, taken from Lothaire's book [17], computes the pair (l_1, n_1).

Algorithm 1 (FirstLyndonFactor)
Input $w \in \Sigma^n$; **Output** (l_1, n_1);
1 : $(i,j) \leftarrow (1,2)$;
2 : **while** $j \leq n$ **and** $w_i \leq w_j$ **do**
3 : **If** $w_i < w_j$ **then**
4 : $i \leftarrow 1$;
5 : **else**
6 : $i \leftarrow i+1$;
7 : **end if**
8 : $j \leftarrow j+1$;
9 : **end while**
10 : **return** $(w_1 w_2 \cdots w_{j-i}, \lfloor (j-1)/(j-i) \rfloor)$;

The underlying principle of the algorithm is to find the longest prefix u of w such that u is smaller than all its suffixes. Assume then, that at some point of execution we obtained the factorization $w = l^n p v$ where l is a Lyndon factor, $n \geq 1$ and p is a non-empty prefix of l. Translating this relation into the values of i and j gives $j - i = |l|$, $n = \lfloor (j-1)/(j-i) \rfloor$ and $j = |l^n p|$. Then $l = p\alpha q$ for some suffix q and some letter α. Three cases may occur:

1. if $w_{i+1} < w_{j+1}$ then $l' = l^n p w_{i+1}$ is a Lyndon factor and $(l', 1)$ becomes the current value of the pair (l_1, n_1);
2. if $w_{i+1} = w_{j+1}$ then p is replaced by $p w_{j+1}$;
3. if $w_{i+1} > w_{j+1}$ then algorithm stops and returns (l, n).

Clearly this algorithm is linear in $n_1|l_1|$, and hence the Lyndon factorization of w is computed in linear time with respect to $|w|$. On the other hand, given a primitive word $w \in \{0,1\}^*$, checking whether it is a Christoffel word is also achieved in linear time using the original definition from [31] (see example at the end of Section 3):

C1. compute $k = |w|_1$ and $n = |w|$;
C2. compute successively $r_1, r_2, \dots, r_{\lceil n/2 \rceil}$ where $r_i = (i\,k) \mod n$ and check

that w_i satisfies the definition.

Note that since

$$\mathrm{CP} \setminus \{0,1\} \subset 0 \cdot \mathrm{PAL} \cdot 1,$$

the second half of w is checked at the same time by verifying that $w_i = w_{n-i+1}$ for $2 \leq i \leq \lceil n/2 \rceil$. This yields the following algorithm.

Algorithm 2 (IsChristoffelPrimitive)
Input $w \in \Sigma^n$
1 : $k \leftarrow |w|_1$; $i \leftarrow 1$; $r \leftarrow k$;
2 : $rejected :=$ **not**$(w_1 = 0$ **and** $w_n = 1)$;
3 : **while not**$(rejected)$ **and** $i < \lceil n/2 \rceil$ **do**
4 : $i \leftarrow i + 1$; $r' \leftarrow r + k \mod n$;
5 : **If** $r < r'$ **then**
6 : $rejected \leftarrow$ **not**$(0 = w_i$ **and** $0 = w_{n-i+1})$;
7 : **else**
8 : $rejected \leftarrow$ **not**$(1 = w_i$ **and** $1 = w_{n-i+1})$;
9 : **end if**
10 : $r \leftarrow r'$;
11 : **end while**
12 : **return not**$(rejected)$;

Combining these two algorithms provides this following algorithm that checks NW-convexity of a given word $w \in \Sigma^*$.

Algorithm 3 (IsNW-Convex)
Input $w \in \Sigma^n$;
1 : $index \leftarrow 1$; $rejected \leftarrow false$;
2 : **while not**$(rejected)$ **and** $index \leq n$ **do**
3 : $(l_1, n_1) \leftarrow$ **FirstLyndonFactor**$(w_{index} w_{index+1} \cdots w_n)$;
4 : $rejected \leftarrow$ **not**(**IsChristoffelPrimitive**(l_1));
5 : $index \leftarrow index + n_1 |l_1|$;
6 : **end while**
7 : **return not**$(rejected)$;

Equation (1) ensures that $\sum_i |l_i| \leq |w|$ so that this algorithm is linear in the length of the word w.

According to Proposition 6, we have to check convexity for each term in the standard decomposition $w \equiv w_1 w_2 w_3 w_4$. Instead of applying the morphism σ to each w_i, which requires a linear pre-processing, it suffices to implement a more general version of Algorithm 1 and Algorithm 2, with the alphabet and its order relation as a parameter. For that purpose, ordered alphabets are denoted as lists $Alphabet = [\alpha, \beta]$ with $\alpha < \beta$.

The resulting algorithm is the following where we assume that w is the contour of a non-empty polyomino.

Algorithm 4 (IsConvex)
Input $w \in \Sigma^n$
0 : *Compute the standard decomposition* $w \equiv w_1w_2w_3w_4$;
1 : $rejected \leftarrow false; i \leftarrow 1; Alphabet \leftarrow [0,1]$;
2 : **while not**$(rejected)$ **and** $i \leq 4$ **do**
3 : $u \leftarrow w_i; k \leftarrow |u|$;
4 : **if** Alph$(u) \subseteq Alphabet$ **then**
5 : $index \leftarrow 1$;
6 : **while not**$(rejected)$ **and** $index \leq k$ **do**
7 : $(l_1, n_1) \leftarrow$ **FirstLyndonFactor**$([u_{index}u_{index+1}\cdots u_k], Alphabet)$;
8 : $rejected \leftarrow$ **not**(**IsChristoffelPrimitive**$(l_1), Alphabet)$;
9 : $index \leftarrow index + n_1|l_1|$;
10 : **end while**
11 : **else**
12 : $rejected \leftarrow true$;
13 : **end if**
14 : $i \leftarrow i + 1; Alphabet \leftarrow [\sigma^{i-1}(0), \sigma^{i-1}(1)]$;
15 : **end while**
16 : **return not**$(rejected)$;

Remark. For more efficiency, testing that the letters of w_i belong to the set $\sigma^{i-1}(\{0,1\}^*)$ (Line 4) can be embedded within the algorithm **FirstLyndonFactor** or in the computation of the standard decomposition (Line 0) and returning an exception.

6 Computing the convex hull

Given a word w on the alphabet $\{0,1\}$, the NW-convex hull of the path coded by w is given by what is called a Spitzer factorization of w (see [15], page 95). Given a morphism $\Phi : \{0,1\}^* \to \mathbb{R}$ where $\mathbb{R}$ is seen as an additive monoid, then for all $r \in \mathbb{R}$ let

$$C_r = \left\{ v \in \{0,1\}^+ \middle| \Phi(v) = r|v| \right\}, \text{ and } B_r = C_r \setminus \left(\bigcup_{s \geq r} C_s \cdot \{0,1\}^+ \right).$$

We provide an example of this definition in Fig. 5. The following theorem is attributed by Lothaire to Spitzer [35], where it appears in a different context.

Theorem 10 (Spitzer, 1956) *Every word* $w \in \{0,1\}^*$ *admits a unique fac-*

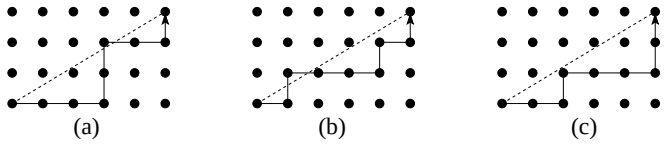


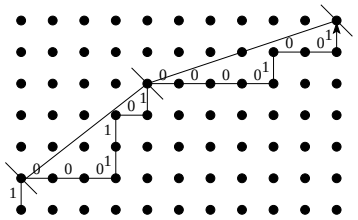
Fig. 5. Given $\Phi(0) = -1$ and $\Phi(1) = +1$, three paths coded by words from the set $C_{-1/4}$. The word in (c) is also in $B_{-1/4}$ while in (a) the prefix $00011 \in C_{-1/5}$ and in (b) the prefix $01 \in C_0$.

torization as

$$w = b_1 b_2 \cdots b_k,$$

$b_i \in B_{r_i}$ *for* $i = 1, 2, \ldots, k$ *and if* $i \neq k$ *then* $r_i \geq r_{i+1}$.

By taking the morphism Φ defined by $\Phi(0) = -1$ and $\Phi(1) = +1$, the separation $b_i \cdot b_{i+1}$ between each pair of consecutive factors such that $r_i > r_{i+1}$ corresponds exactly to the vertices of the convex hull of the path coded by w. As an example consider the word w as shown below :



$w = 1000110100001001,$
$w = 1 \cdot 0001101 \cdot 00001001,$
$1 \in B_1,$
$0001101 \in B_{-1/7},$
$00001001 \in B_{-1/2}.$

More precisely the algorithm showing the fusion $b_i b_{i+1}$ is given by

$$w = (1) \cdot (0(0(((01)1)))(01)) \cdot ((0(0(0(01))))(0(01))).$$

Note that the slope of a word $v \in B_r$ is $\rho(v) = (1 + r)/(1 - r)$, so that if $v \in B_r$ and $v' \in B_{r'}$ then

$$r < r' \iff \rho(v) < \rho(v').$$

As in the case of Viennot factorizations [36], the family of sets $(B_r)_{r \geq 0}$, with $\Phi(0) = -1$ and $\Phi(1) = +1$, has the following obvious property :

Property 2 *Let* $u \in B_r$ *and* $v \in B_t$, *if* $r < t$ *then the word* uv *belongs to a* B_s *for some* $r < s < t$.

As a consequence, the Spitzer factorization of a word $w \in \{0, 1\}^n$ can be computed as follows

S1. start with the list $(b_1, b_2, \ldots, b_n) \leftarrow (w_1, w_2, \ldots, w_n)$;
S2. if $\rho(b_i) < \rho(b_{i+1})$ for some i then
$(b_1, b_2, \ldots, b_k) \leftarrow (b_1, \ldots, b_{i-1}, b_i b_{i+1}, b_{i+2}, \ldots, b_k)$, and stop otherwise.

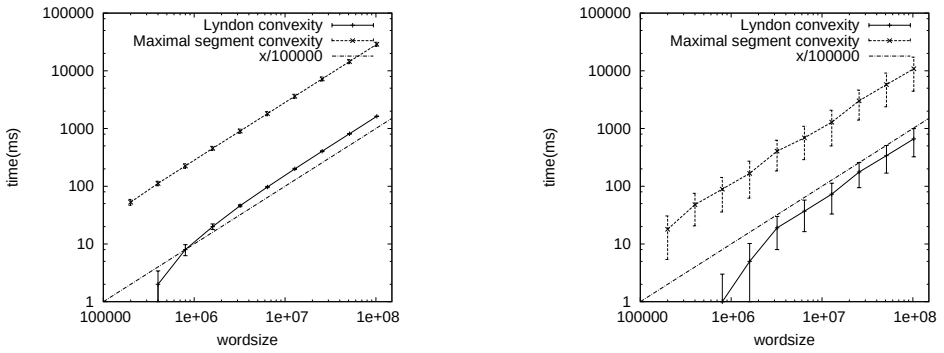
It is clear that if the number of occurrences of $0s$ and $1s$ of each b_i is stored then the factorization is computed in linear time. Note that if this computation is done using a stack, then the algorithm obtained appears as a discrete version of the classical Melkman's algorithm [24] for the construction of the convex hull of a polygon in linear time (see [37] for an historical review on the subject). Of course, the algorithm presented above only applies to words coding the north-west part of a hv-convex polyomino while Melkman's algorithm applies to any simple polyline. Note also that Ehrenfeucht, Haemer and Haussel used this approach in [38] in order to elaborate an optimal algorithm to compute the convex hull of an ordered list of points $T = (x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$, with distinct x coordinates and sorted in increasing order of this coordinate.

For sake of completeness we provide a general algorithm that computes the North-West part of the convex hull of any polyomino.

Algorithm 5 (NWConvexHull)
Input $w \in \{0, 1, \bar{0}, \bar{1}\}^n$ /* the non self-intersecting path from W to N */
Type *Subword* = **struct** { *i: integer* /* position */
a: integer /* vertical displacement */
b: integer /* horizontal displacement */ }
Var *P: Stack of Subword*
u, v: Subword
1 : **for** *i from* 1 *to n do*
2 : $u.i \leftarrow i$;
3 : **case** w_i **in**
4 : $0 : u.a \leftarrow 0; \quad u.b \leftarrow 1;$ **break**
5 : $1 : u.a \leftarrow 1; \quad u.b \leftarrow 0;$ **break**
6 : $\bar{0} : u.a \leftarrow 0; \quad u.b \leftarrow -1;$ **break**
7 : $\bar{1} : u.a \leftarrow -1; \; u.b \leftarrow 0;$ **break**
8 : **end case**
9 : $l \leftarrow true$;
10 : **while** *l and (P is not empty)* **do**
11 : $v \leftarrow top(P)$;
12 : **if** $v.a * u.b \leq v.b * u.a$ **then** /* if vector u is to the left of vector v */
13 : $u.i \leftarrow v.i$;
14 : $u.a \leftarrow u.a + v.a$; /* concatenate u and v */
15 : $u.b \leftarrow u.b + v.b$;
16 : $pop(P)$;
17 : **else** $l \leftarrow false$;
18 : **end if**
19 : **end while**
18 : $push(P, u)$;
20 : **end for**
21 : **return** P /* the stack contains the convex hull edges read from N to W */

7 Concluding remarks

The implementation of our algorithm to check digital convexity was compared to the method of Debled-Rennesson et al. [9], implemented in linear time with the optimization of [10,11]. The results (see Fig. 6) showed that our technique was 10 times faster than the technique of maximal segments.



(a) (b)

Fig. 6. Timing comparison of the digital convexity tests (maximal segments and Lyndon factorization). Test shapes are ellipses of increasing size (abscissa represents the contour word length). For each size, a hundred different axis ratios are tested and the timings are averaged. Figure (a) summarizes the timings on these ellipses, where each time the convexity test is positive (time in ms and standard deviation). Figure (b) also summarizes the timings but one 0 has been swaped with a 1, both letters at a randomly chosen place in the contour word : convexity test is then negative. Experiments have been runned on a 2.53 GHz Pentium 4 with cache 512 Kb, code written in C++ and compiled with GCC v4.1.2. Computation time decreases when the shape is no more convex, for both techniques. Speedup ratio for Lyndon technique is identical.

This speedup is partially due to the fact that computing maximal segments provides more geometrical informations while testing convexity is simpler. Nevertheless, our algorithm is much simpler conceptually and suggests that the notion of digital convexity might be a more fundamental concept than what is usually perceived. The fact that the combinatorial approach delivers such an elegant algorithm begs for a systematic study of the link between combinatorics on words and discrete geometry. In particular, the palindromic structure of Christoffel words allows to avoid part of the arithmetic calculation. Among the many problems that can be addressed with this new approach, we hope to develop a factorization providing the convex hull using mostly combinatoric properties. The one presented in Section 6 relies completely on arithmetic computations which explains the similarities with previously known convex hull computation techniques.

Acknowledgements Srečko Brlek is supported by a grant from NSERC

(Canada). Jacques-Olivier Lachaud was partially funded by ANR project GeoDIB, n° ANR-06-BLAN-0225-03. Xavier Provençal was supported by a scholarship from FQRNT (Québec). Christophe Reutenauer benefited from support of the Canada Research Chair program (Canada).

References

[1] J. Sklansky, Recognition of convex blobs, Pattern Recognition 2 (1) (1970) 3–10.

[2] M. Minsky, S. Papert, Perceptrons, 2nd Edition, MIT Press, 1988.

[3] C. Kim, On the cellular convexity of complexes, Pattern Analysis and Machine Intelligence 3 (6) (1981) 617–625.

[4] C. Kim, Digital convexity, straightness, and convex polygons, Pattern Analysis and Machine Intelligence 4 (6) (1982) 618–626.

[5] C. Kim, A. Rosenfeld, Digital straight lines and convexity of digital regions, Pattern Analysis and Machine Intelligence 4 (2) (1982) 149–153.

[6] B. Chaudhuri, A. Rosenfeld, On the computation of the digital convex hull and circular hull of a digital region, Pattern Recognition 31 (12) (1998) 2007–2016.

[7] R. Klette, A. Rosenfeld, Digital straightness—a review, Discrete Appl. Math. 139 (1-3) (2004) 197–230.

[8] U. Eckhardt, Digital lines and digital convexity, Digital and image geometry: advanced lectures (2001) 209–228.

[9] I. Debled-Rennesson, J.-L. Rémy, J. Rouyer-Degli, Detection of the discrete convexity of polyominoes, Discrete Appl. Math. 125 (1) (2003) 115–133.

[10] F. Feschet, L. Tougne, Optimal time computation of the tangent of a discrete curve: Application to the curvature, in: G. Bertrand, M. Couprie, L. Perroton (Eds.), Proc DGCI 1999, 8-th Int. Conf. on Discrete Geometry for Computer Imagery, Marne-la-Vallée, France, March 17-19 , No. 1568 in LNCS, Springer-Verlag, 1999, pp. 31–40.

[11] J.-O. Lachaud, A. Vialard, F. de Vieilleville, Fast, accurate and convergent tangent estimation on digital contours, Image and Vision Computing 25 (2007) 1572–1587.

[12] A. Hübler, R. Klette, K. Voss, Determination of the convex hull of a finite set of planar points within linear time, Elektronische Informationsverarbeitung und Kybernetik 17 (2/3) (1981) 121–139.

[13] K. Voss, Discrete Images, Objects, and Functions in Z^n, Springer-Verlag, 1993.

[14] S. Har-Peled, An output sensitive algorithm for discrete convex hulls, Comput. Geom. 10 (2) (1998) 125–138.

[15] M. Lothaire, Combinatorics on words, Cambridge Mathematical Library, Cambridge University Press, Cambridge, 1997.

[16] M. Lothaire, Algebraic combinatorics on words, Vol. 90 of Encyclopedia of Mathematics and its Applications, Cambridge University Press, Cambridge, 2002.

[17] M. Lothaire, Applied combinatorics on words, Vol. 105 of Encyclopedia of Mathematics and its Applications, Cambridge University Press, Cambridge, 2005.

[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] A. Daurat, M. Nivat, Salient and reentrant points of discrete sets, in: A. del Lungo, V. di Gesu, A. Kuba (Eds.), Proc. IWCIA'03, Int. Workshop on Combinatorial Image Analysis, Palermo, Italia, 14–16 May, Electronic Notes in Discrete Mathematics, Elsevier Science, 2003.

[20] S. Brlek, X. Provençal, An optimal algorithm for detecting pseudo-squares., in: A. Kuba, L. G. Nyúl, K. Palágyi (Eds.), Proc. DGCI 2006, 13-th Int. Conf. on Discrete Geometry for Computer Imagery, Szeged, Hungary, 25–27 Oct., No. 4245 in LNCS, Springer-Verlag, 2006, pp. 403–412.

[21] S. Brlek, J.-M. Fédou, X. Provençal, On the tiling by translation problem, Discrete Applied Math. To appear.

[22] H. Fredricksen, J. Maiorana, Necklaces of beads in k colors and k-ary de Bruijn sequences, Discrete Math. 23 (3) (1978) 207–210.

[23] J.-P. Duval, Factorizing words over an ordered alphabet, J. Algorithms 4 (4) (1983) 363–381.

[24] A. A. Melkman, On-line construction of the convex hull of a simple polyline, Inf. Process. Lett. 25 (1) (1987) 11–12.

[25] S. Brlek, J.-O. Lachaud, X. Provençal, Combinatorial view of convexity., in: D. Coeurjolly, I. Sivignon, L. Tougne, F. Dupont (Eds.), Discrete Geometry for Computer Imagery, 14th International Conference, DGCI 2008, Lyon, France, April 16-18, 2008. Proceedings, Vol. 4992 of Lecture Notes in Computer Science, Springer, 2008, pp. 57–68.

[26] C. Reutenauer, Free Lie algebras, Vol. 7 of London Mathematical Society Monographs. New Series, The Clarendon Press Oxford University Press, New York, 1993, oxford Science Publications.

[27] M. Crochemore, C. Hancart, T. Lecroq, Algorithms on strings, Cambridge University Press, Cambridge, 2007, translated from the 2001 French original.

[28] J.-P. Duval, A. Lefebvre, Words over an ordered alphabet and suffix permutations, Theor. Inform. Appl. 36 (3) (2002) 249–259.

[29] C. Hohlweg, C. Reutenauer, Lyndon words, permutations and trees, Theoret. Comput. Sci. 307 (1) (2003) 173–178, words.

[30] J. Berstel, A. Lauve, C. Reutenauer, F. Saliola, Combinatorics on words : Chritoffel words and repetition in words, CRM-AMS, Montreal, 2008, to appear.

[31] E. B. Christoffel, Observatio arithmetica, Annali di Mathematica 6 (1875) 145–152.

[32] J.-P. Borel, F. Laubie, Quelques mots sur la droite projective réelle, J. Théor. Nombres Bordeaux 5 (1) (1993) 23–51.

[33] J. Berstel, A. de Luca, Sturmian words, Lyndon words and trees, Theoret. Comput. Sci. 178 (1-2) (1997) 171–203.

[34] J.-P. Reveillès, Géométrie discrète, calcul en nombres entiers et algorithmique, Ph.D. thesis, Université Louis Pasteur, Strasbourg (December 1991).

[35] F. Spitzer, A combinatorial lemma and its application to probability theory, Trans. Amer. Math. Soc. 82 (1956) 323–339.

[36] G. Viennot, Algèbres de Lie libres et monoïdes libres, Vol. 691 of Lecture Notes in Mathematics, Springer, Berlin, 1978, bases des algèbres de Lie libres et factorisations des monoïdes libres.

[37] G. Aloupis, A history of linear-time convex hull algorithms for simple polygons., available electronically at http://cgm.cs.mcgill.ca/∼athens/cs601/.

[38] A. Ehrenfeucht, J. Haemer, D. Haussler, Quasi-monotonic sequences: theory, algorithms and applications, SIAM J. Algebraic Discrete Methods 8 (3) (1987) 410–429.

About the Author—SREČKO BRLEK obtained his Ph.D. from Université Bordeaux I in 1988. He is professor of computer science at Université du Québec à Montréal, and a regular member of the Laboratoire de Combinatoire et d'informatique Mathématique (LaCIM). In his research combinatorics on words play a key role, and cover automata theory, formal language theory, algorithms, discrete mathematics and also their applications in discrete geometry and protocol design.

About the Author—JACQUES-OLIVIER LACHAUD graduated from ENSIMAG engineering school in Computer Science in 1994 and received a Ph.D. degree in computer science from Joseph Fourier University (Grenoble, France) in 1998. He is currently a Professor in Computer Science at the university of Savoie (Chambry, France) and works in the mathematics laboratory (LAMA). His research interests are in image segmentation and analysis, more specifically deformable models, energy-minimizing techniques, digital geometry, topological models and invariants. He has written about thirty papers in international journals or conferences on these topics.

About the Author—XAVIER PROVENÇAL received B.Sc. and M.Sc. from Université du Québec à Montréal, and has completed his Ph.D (2008) in mathematics and computer science. Currently he is a postdoctoral fellow from FQRNT (Québec) in the laboratory of mathematics LAMA of the university of Savoie in Chambéry. His research interests are combinatorics on words, discrete geometry and theoretical computer science.

About the Author—CHRISTOPHE REUTENAUER is professor of mathematics at Université du Québec à Montréal. His research interests include Agebra, Combinatorics and Theoretical Computer Science. He is also member of the Laboratoire de Combinatoire et d'informatique Mathématique (LaCIM) and holds the Canada Research Chair in "Algebra, Combinatorics and Computer Science".