UNIVERSITÉ DU QUÉBEC À MONTRÉAL

COMBINATOIRE DES MOTS, GÉOMÉTRIE DISCRÈTE ET PAVAGES

THÈSE

PRÉSENTÉE

COMME EXIGENCE PARTIELLE

DU DOCTORAT EN MATHÉMATIQUES

PAR

XAVIER PROVENÇAL

SEPTEMBRE 2008

Remerciements

En tout premier lieu, je tiens à remercier mon directeur de thèse : Srečko Brlek. Tu as cru en moi dès le début et je t'en serai toujours reconnaissant. Tout au long de ces quatre années, tu as su trouver les bons problèmes sur lesquels me faire travailler et, plus important encore, comment me pousser à travailler ! Tu as été plus qu'un directeur comme en témoignent nos anecdotes de voyage ainsi que les nombreux *soupers au sommet*. Pour tout, je te dis merci.

J'aimerais également remercier mes co-auteurs. Tout d'abord, Jean-Marc Fédou ton hospitalité sans borne fera à jamais partie de mes bons souvenirs. Jacques-Olivier Lachaud pour ta grande dévotion envers moi et, surtout, pour avoir accepté de me prendre comme postdoc ! J'espère que je serai à la hauteur de tes attentes. Je tiens également à remercier Michel Koskas pour ton hospitalité et ta collaboration qui a donné un merveilleux résultat. Finalement, Christophe Reutenauer pour tes nombreux conseils rigoureusement judicieux.

Sophie, mon amour, tu es toujours là pour m'aider, m'écouter et me conseiller quand j'en ai besoin et c'est fou comme j'en ai besoin ! Merci d'avoir choisi de m'accompagner, ta présence rend les moments difficiles plus agréables et les bons deviennent tout simplement merveilleux. Merci d'être à mes côtés.

Un gros merci à ma *coloc de bureau*, Annie. Grâce à toi, j'ai pu trouver la force d'aller à ce fameux bureau jour après jour. Également grâce à toi, je suis parvenu à faire avancer cette thèse quand je me rendais compte à quel point j'étais en retard par rapport à toi ! J'en serais encore au chapitre 1... Merci pour toutes ces heures de *travail*. Merci d'avoir à l'occasion servi d'aide mémoire, de précurseure bureaucratique et de fanfaronne !

Je n'oublie pas mes *amis d'université* Baptiste et Clément. Je n'ose imaginer ces quatres années sans vous, sans ces doses quotidiennes de délire, de "Faut que je vous montre ça !" et de "Je dis ça, je dis rien". C'est la fin d'une belle époque !

Je veux également remercier ma famille car sans leur support inconditionnel, je ne me serais sans doute jamais lancé dans une telle aventure ! Merci papa. Merci maman. Merci soeurette. Merci Manù. Et oui, merci Luca ! Tu es arrivé juste au bon moment, ton rire est un anti-stress qu'il faudra breveter !

Je tiens également à remercier tous mes amis qui m'ont encouragé dans ce projet farfelu qu'est le doctorat. Vous l'avez tous fait à votre manière et je vous en suis extrêmement reconnaissant.

Avant de conclure, je me dois de remercier deux femmes qui ont su me guider, m'épauler et parfois même me sauver de la noyade dans l'océan de la bureaucratie universitaire. Un GROS merci à Lise Tourigny et Manon Gauthier.

Finalement, je me dois de mentionner que rien de tout ceci n'aurait été possible sans le support financier du FQRNT et de l'ISM. Ces bourses sont une bénédiction pour les étudiants.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

RÉSUMÉ

L'objet de cette thèse est d'étudier les liens entre la géométrie discrète et la combinatoire des mots. Le fait que les figures discrètes soient codées par des mots sur l'alphabet à quatre lettres $\Sigma = \{0, 1, \overline{0}, \overline{1}\}$, codage introduit par Freeman en 1961, justifie l'utilisation de la combinatoire des mots dans leur étude. Les droites discrètes sont des objets bien connus des combinatoriciens, car étant identifiés par les mots Sturmiens, dont on trouve déjà une description assez complète dans les travaux de Christoffel à la fin du XIXe siècle à la suite de travaux précurseurs de Bernouilli et Markov. Alors que l'on comprend bien la structure des droites discrètes, on connait beaucoup moins bien les courbes en général.

Cet ouvrage porte sur l'étude de propriétés géométriques de courbes fermées, codées sur l'alphabet Σ. On s'intéresse tout d'abord à la représentation des chemins dans le plan discret $\mathbb{Z}^2$ et de ceux qui codent les polyominos.

Dans un premier temps, l'emploi d'une structure arborescente quaternaire permet d'élaborer un algorithme optimal afin de tester si un mot quelconque sur Σ code un polyomino ou non. Ce résultat est fondamental d'abord parce qu'il est nouveau, élégant et qu'il se généralise en dimension supérieure. En outre, la linéarité de ce test rend les algorithmes subséquents vraiment efficaces.

À la suite de résultats précurseurs de Lyndon, Spitzer et Viennot sur la factorisation des mots, il existe une interprétation combinatoire de la convexité discrète. En géométrie algorithmique, des algorithmes linéaires furent établis par McCallum et Avis en 1979, puis par Melkman en 1987, pour calculer l'enveloppe convexe d'un polygone. Debled-Rennesson et al. ont obtenu, en 2003, un algorithme linéaire pour décider de la convexité discrète d'un polyomino par des méthodes arithmétiques. Nous avons obtenu grâce aux propriétés spécifiques des mots de Lyndon et de Christoffel un algorithme linéaire pour tester si un polyomino est digitalement convexe. L'algorithme obtenu est extrêmement simple et s'avère dix fois plus rapide que celui de Debled-Rennesson et al.

Finalement, le calcul de la plus longue extension commune à deux mots en temps constant – obtenu par Gusfield à l'aide des arbres suffixes – et le théorème de Fine et Wilf permettent d'élaborer de nouveaux algorithmes qui, grâce à la caractérisation de Beauquier-Nivat, testent si un polyomino pave le plan par translation. En particulier, on obtient un algorithme optimal en $\mathcal{O}(n)$ pour détecter les pseudo-carrés. Dans le cas des pseudo-hexagones ayant des facteurs carrés pas trop longs on obtient également un algorithme linéaire optimal, tandis que pour les pseudo-hexagones quelconques nous avons obtenu un algorithme en $\mathcal{O}\left(n(\log n)^3\right)$ que nous croyons ne pas être optimal.

Mots clés : combinatoire des mots, géométrie discrète, droites digitales, pavages du plan, algorithmique.

INTRODUCTION

Le traitement numérique consécutif à l'avènement des ordinateurs a connu un essor remarquable en synthèse et en analyse d'images. Toutes les technologies d'affichage modernes sont basées sur la représentation d'images réelles sur des grilles de pixels. Les images qu'on représente figurent dans un plan où les principes de la géométrie euclidienne s'appliquent. Le passage du monde continu ou euclidien, au monde discret ne s'effectue pas sans difficultés. En effet, de nombreuses notions bien définies dans le monde euclidien ne peuvent s'appliquer directement aux espaces discrets. Par exemple, la notion bien connue de ligne droite doit être repensée lorsqu'on passe au discret : au lieu d'être représentée par deux points distincts, elle se trouve représentée par un chemin sur le réseau carré constitué des pixels. Ce chemin est un mot infini sur un alphabet de deux lettres connu sous le nom de mot de Sturm et certains segments finis de ce chemin sont des mots dits de Christoffel.

L'analyse et la synthèse d'images sont deux domaines qui se sont récemment imposés dans pratiquement tous les secteurs scientifiques et techniques. Les problèmes soulevés par le développement de leurs applications relèvent de la géométrie discrète.

Comme le mentionnent Klette et Rozenfeld dans la deuxième phrase de l'introduction de leur synthèse sur les droites digitales (Klette et Rosenfeld, 2004) :

> "*Related work even earlier on the theory of words, specifically, on mechanical or Sturmian words, remained unnoticed in the pattern recognition community.*"

faire le pont entre les méthodes issues de la combinatoire des mots et celles provenant de l'analyse et le traitement d'images correspond à un besoin. Jusqu'à présent, en géométrie discrète les objets sont caractérisés à l'aide de propriétés arithmétiques, de sorte que les outils employés pour détecter ces propriétés reposent sur l'arithmétique. Une approche combinatoire offre un point de vue différent et propose de nouveaux outils algorithmiques pour étudier ces objets.

Le but de cette thèse est de renforcer les liens entre la géométrie discrète et la combinatoire des mots. Plus précisément, montrer comment la combinatoire des mots fournit des outils efficaces et d'implémentation simple afin d'étudier différents problèmes issus de la géométrie discrète.

Le premier chapitre est consacré à la définition des objets combinatoires et géométriques qui seront étudiés et employés par la suite. Ces objets et concepts sont en général bien connus et aucun nouveau résultat n'y est présenté. Les définitions et notations relatives à la combinatoire des mots sont en majeure partie tirées des livres de M. Lothaire (Lothaire, 1997; Lothaire, 2002; Lothaire, 2005) qui sont considérés comme la référence en ce domaine. Après avoir introduit quelques résultats classiques sur les mots, on présente le codage de Freeman (Freeman, 1961; Freeman, 1970) qui permet de coder par un mot sur l'alphabet $\{0, 1, \overline{0}, \overline{1}\}$ tout chemin constitué de déplacements unitaires dans le plan discret $\mathbb{Z}^2$. Ce codage est ensuite employé afin de coder les polyominos qui sont le principal objet d'étude de cette thèse. Une fois un polyomino représenté par un mot, on étudie les propriétés géométriques du polyomino par l'intermédiaire de la structure combinatoire de ce mot.

Au chapitre 2, on s'intéresse au problème suivant :

> "*Étant donné un mot w sur l'alphabet $\{0, 1, \overline{0}, \overline{1}\}$, est ce que le chemin codé par w passe deux fois par le même point.*"

Ce problème, très simple à première vue, présente de nombreuses solutions algorithmiques dont la complexité temporelle est dans $\mathcal{O}(n \log n)$. L'utilisatoin d'un radix-tree quaternaire auquel on ajoute des *liens de voisinages* permet d'éviter ce facteur logarithmique proposant ainsi une solution optimale. L'algorithme présenté est tiré de l'article (Brlek, Koskas et Provençal, 2008). On conclut en montrant de quelle manière cet algorithme permet directement de tester si un mot code le bord d'un polyomino. Ceci s'avère de première importance car les deux chapitres qui suivent portent sur l'analyse de ces *mots de contour* afin d'en tirer des informations géométriques en temps linéaire. Il est donc nécessaire d'être d'abord en mesure de déterminer si ce mot code le bord d'un polyomino ou non, sans affecter la complexité temporelle du traitement.

Le chapitre 3 est consacré à l'étude de la convexité discrète des polyominos. Les résultats présentés sont issus des articles (Brlek, Lachaud et Provençal, 2008) (Brlek et al., 2008).

On y propose d'abord une condition nécessaire et suffisante pour tester la convexité discrète. En effet, un polyomino est digitalement convexe si et seulement si la factorisation en mots de Lyndon décroissants de son mot de contour est uniquement composée de mots de Christoffel. Cette condition permet d'élaborer un test de convexité optimal dont l'implémentation est d'une simplicité surprenante. On s'attarde ensuite au problème du calcul de l'enveloppe convexe et, encore une fois, on y propose une solution optimale.

Le quatrième et dernier chapitre de cette thèse étudie les polyominos dit *exacts*, c'est-à-dire ceux avec lequels il est possible de paver le plan par translation. On s'intéresse d'abord à la détection de ces polyominos particuliers. Wijshoff et van Leeuven (Wijshoff et van Leeuven, 1984) ont montré qu'il suffit de considérer les pavages *réguliers* démontrant par le fait même que, contrairement au problème plus général de pavage du plan par un ensemble de polyominos qui est indécidable (Berger, 1966), déterminer si un polyomino seul pave le plan est décidable en temps polynômial. Un critère sur les mots établi par Beauquier et Nivat (Beauquier et Nivat, 1991) sépare les polyominos exacts en deux catégories, soit les *pseudo-carrés* et les *pseudo-hexagones*. On propose trois algorithmes, basés sur ce critère qui permettent d'abaisser la borne quadratique établie par Gambini et Vuillon (Gambini et Vuillon, 2003) :

1. un algorithme optimal pour détecter les pseudo-carrés. (Brlek et Provençal, 2006c),
2. un algorithme optimal pour détecter les pseudo-hexagones dont les bords ne possèdent pas de répétitions trop longues. (Brlek et Provençal, 2006a; Brlek et Provençal, 2006b),
3. un algorithme dans $\mathcal{O}\left(n(\log n)^3\right)$ pour détecter tout pseudo-hexagone.

Ensuite, on étudie la structure particulière des polyominos de type pseudo-carré qui pavent le plan de deux manières différentes. On termine en proposant deux généralisations provenant de l'article (Brlek, Fédou et Provençal, 2008). Premièrement, on considère des chemins fermés qui ne codent pas nécessairement des polyominos, mais des régions plus générales. Deuxièmement, on s'intéresse au passage de la grille carrée à la grille hexagonale.

On conclut en proposant diverses voies d'exploration futures qui suivent naturellement de ces travaux.

Il est important de préciser que les résultats présentés dans cette thèse sont en majorité le fruit de

collaborations internationales. Tout d'abord, l'algorithme présenté au chapitre 2 est le résultat d'une collaboration avec Michel Koskas de l'AgroParisTech à Paris. D'autre part, des travaux réalisés conjointement avec Jacques-Olivier Lachaud de l'Université de Savoie à Chambéry ont quant à eux mené à l'élaboration du test de convexité présenté au chapitre 3. Finalement, les généralisations proposées à la fin du chapitre 4, c'est-à-dire la notion de *mot de contour généralisé* et le passage à la grille hexagonale, découlent d'une collaboration avec Jean-Marc Fédou de l'Université de Nice Sophia-Antipolis à Nice.

En terminant, j'aimerais souligner que tous les algorithmes présentés dans cette thèse ont été implémentés. Ces programmes, écrits principalement en Maple ou en C++, seront tous recodés et intégrés au projet *Sage-Words* développé au LaCIM.

Chapitre I

DÉFINITIONS ET NOTATIONS

La combinatoire des mots est considérée comme une discipline récente en mathématiques. Ses premiers travaux sont attribués à Bernouilli, au XVIIIe siècle. Ensuite, on remarque les travaux de Christoffel et Markov aux XIXe et Morse au XXe. À cette époque, on ne considère pas la combinatoire des mots comme une discipline en soit. Ces grands mathémaciens en utilisent simplement les idées de base afin de mener à bien leurs travaux respectifs dans divers domaines des mathématiques : principalement en algèbre mais également en traitement algébrique des courbes continues et en théorie des nombres.

On s'en doute, c'est l'avènement de l'ordinateur qui, dans la deuxième moitié du XXe siècle, créa un véritable intérêt envers cette discipline. En informatique, toute information est représentée sous la forme de chaînes de caractères. Le traitement de celles-ci soulève des questions qui relèvent de la combinatoire des mots.

Le développement des ordinateurs a provoqué une véritable explosion dans l'étude des mathématiques discrètes créant ce qu'on appelle aujourd'hui l'informatique théorique. Les pionniers de cette discipline, Schützenberger, Chomsky, Kleene, Eilenberg et Ginzburg pour ne nommer que ceux-là, ont jeté les bases de la théorie des langages formels de laquelle sont issus les compilateurs, interpréteur et autres logiciels d'usage courant.

La combinatoire des mots a aujourd'hui atteint un niveau de développement important dont on trouve une présentation exhaustive dans la série de livres par M. Lothaire (Lothaire, 1997; Lothaire, 2002; Lothaire, 2005).

1.1 Alphabet, mot et facteur

Étant donné un ensemble de symboles Σ appelé *alphabet*, on définit le mot w comme étant un n-uplet $(w_1, w_2, \ldots, w_n)$. Afin d'alléger l'écriture, on omet les parenthèses et les virgules de manière à écrire simplement $w = w_1 w_2 \cdots w_n$; on appelle w_k la k-ième lettre de w. On utilisera à l'occasion la notion $w[k]$ pour désigner cette lettre.

Définition 1. Soit $w \in \Sigma^n$, on appelle n la *longueur* de w, notée $|w| = n$.

Par exemple, soit $\Sigma = \{a, b\}$ le mot de longueur six $w = abbaba$ correspond au 6-tuple

$$(a, b, b, a, b, a) \in \Sigma^6.$$

Définition 2. Étant donné $u \in \Sigma^n$ et $v \in \Sigma^m$, la *concaténation* de u et v, notée uv, est le $(n+m)$-uplet

$$uv = (u_1, u_2, \ldots, u_n, v_1, v_2, \ldots, v_m) \in \Sigma^{n+m}.$$

La concaténation est souvent appelée le *produit* de deux mots. Conséquemment, pour tout entier $k \geq 0$, la notation w^k désigne la concaténation de k copies du mot w. On note ε le mot de longueur zéro appellé *mot vide*. Naturellement, on a que pour tout mot w,

$$\varepsilon w = w\varepsilon = w \quad \text{et} \quad w^0 = \varepsilon.$$

L'ensemble des mots finis possède donc une structure de monoïde dont ε est l'élément neutre.

Notation 1. L'ensemble des mots finis sur Σ est noté

$$\Sigma^* = \bigcup_{n \geq 0} \Sigma^n.$$

On appelle Σ^* le *monoïde libre*. L'ensemble des mots finis non-vides est quant à lui noté

$$\Sigma^+ = \Sigma^* \setminus \{\varepsilon\}.$$

Le mot obtenu en inversant l'ordre des lettres d'un mot $w = w_1 w_2 \cdots w_n$ est appellé son *miroir* et est défini par $\widetilde{w} = w_n w_{n-1} \cdots w_1$. Un invariant sous l'action de l'opérateur $\widetilde{\ }$ est appelé un *palindrome*.

Définition 3. L'ensemble des palindromes sur l'alphabet Σ est noté :

$$\mathrm{Pal}(\Sigma^*) = \{w \in \Sigma^* | w = \widetilde{w}\}.$$

Remarque 1. Un mot $w \in \Sigma^n$ est un palindrome si et seulement si pour $i \in \{1, 2, \ldots, n\}$ on a $w_i = w_{n-i+1}$.

Par exemple, les mots $w_1 = \varepsilon$, $w_2 = a$, et $w_3 = abbabba$ sont tous éléments de $\mathrm{Pal}(\{a, b\}^*)$.

Définition 4. Soit w un mot fini sur l'alphabet Σ et $x, y, z \in \Sigma^*$ trois mots tels que $w = xyz$, on dit alors que x est un *préfixe* de w, y est un *facteur* de w, et z est un *suffixe* de w. De plus, un préfixe, un facteur ou un suffixe de w est dit *propre* s'il n'est pas égal à w.

On note $\mathrm{Pref}(w)$ l'ensemble des préfixes de mot w, $\mathrm{Fact}(w)$ l'ensemble des facteurs de w et $\mathrm{Suff}(w)$ l'ensemble des suffixes de w. Par définition, pour tout $w \in \Sigma^*$ on a

$$\{\varepsilon, w\} \subset \mathrm{Pref}(w) \cap \mathrm{Fact}(w) \cap \mathrm{Suff}(w).$$

Par exemple, considérons encore une fois le mot $w = abbaba$,

$$\mathrm{Pref}(w) = \{\varepsilon, a, ab, abb, abba, abbab, abbaba\},$$
$$\mathrm{Fact}(w) = \{\varepsilon, a, b, ab, bb, ba, abb, bba, bab, aba, abba, bbab, baba, abbab, bbaba, abbaba\},$$
$$\mathrm{Suff}(w) = \{\varepsilon, a, ba, aba, baba, bbaba, abbaba\}.$$

Étant donné un mot $w \in \Sigma^*$ et deux entiers k et l, $1 \leq k \leq l \leq |w|$, on note $w[k..l]$ le facteur $w_k w_{k+1} \cdots w_l$ de w.

Bien qu'il existe de nombreux ordres sur les mots, dans le cadre de cet ouvrage le seul ordre considéré sera l'*ordre lexicographique*. Souvent appelé *ordre du dictionnaire*, l'ordre lexicographique étend un ordre total sur l'alphabet Σ à l'ensemble des mots finis Σ^*.

Définition 5. Soient $u, v \in \Sigma^*$ et $<$ un ordre total sur les lettres de Σ. Le mot u est lexicographiquement plus petit que v, noté également $u < v$, si une des deux conditions suivantes est respectée

1. u est un préfixe propre de v.

2. Il existe un mot $x \in \Sigma^*$ ainsi que deux lettres $a, b \in \Sigma$, $a < b$ tels que $xa \in \mathrm{Pref}(u)$, $xb \in \mathrm{Pref}(v)$.

Le fait que les deux ordres soient notés de la même manière n'entraîne pas de confusion car l'ordre lexicographique correspond à l'ordre de base sur Σ lorsqu'on considère des mots de longueur 1. Afin d'alléger la présentation, on introduit les notations suivantes :

Notation 2. Étant donné un mot non-vide $w \in \Sigma^+$, on note $f(w) = w_1$ la première lettre de w et $l(w) = w_{|w|}$ sa dernière lettre.

Notation 3. Étant donné $x, y \in \Sigma$ on note ${}_x\Sigma_y = \{w \in \Sigma^+ | f(w) = x \text{ et } l(w) = y\}$, l'ensemble des mots sur Σ qui commencent par la lettre x et terminent par la lettre y.

Notons que ${}_x\Sigma_y$ est un langage rationnel car

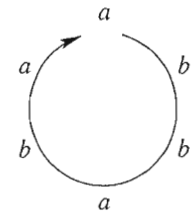
$$ {}_x\Sigma_y = \begin{cases} \{x\} \cdot \Sigma^* \cdot \{y\} & \text{si } x \neq y \\ \{x\} \cup (\{x\} \cdot \Sigma^* \cdot \{x\}) & \text{si } x = y. \end{cases} $$

Une autre notion fondamentale en combinatoire des mots est la conjugaison.

Définition 6. Deux mots finis u et v sur l'alphabet Σ sont dit *conjugués* s'il existe $x, y \in \Sigma^*$ tels que $u = xy$ et $v = yx$. On note la conjugaison $u \equiv v$.

On appelle la *classe de conjugaison* de u l'ensemble de tous les mots v tels que $u \equiv v$. Notons que la conjugaison est une relation d'équivalence sur les mots qui correspond à les voir comme des mots circulaires. Par exemple, la classe de conjugaison du mot $u = abbaba$ est :

$$\{abbaba, bbabaa, babaab, abaabb, baabba, aabbab\}.$$



Il devient alors évident que la classe de conjugaison d'un mot $u \in \Sigma^*$ est toujours entièrement incluse dans l'ensemble $\mathrm{Fact}(u^2)$.

1.2 Périodicité et théorème de Fine et Wilf

Quand on s'intéresse à la structure d'un mot, la périodicité joue un rôle fondamental. Habituellement, lorsqu'un mot possède une période petite relativement à sa taille, la périodicité est la première propriété structurelle qu'on remarque.

Définition 7. Un entier $p \geq 1$ est une *période* d'un mot $w \in \Sigma^n$ si pour tout $i \in \{1, 2, \ldots, n-p\}$ on a $w_i = w_{i+p}$.

On note $\mathrm{Per}(w)$ l'ensemble des périodes de w. Parmi toutes les périodes d'un mot, la plus significative est bien entendu la plus petite. On dira d'un entier p qu'il est la *période primitive* du mot w si $p = \min(\mathrm{Per}(w))$.

Notons qu'un mot $w \in \Sigma^n$ admet une période $p < n$ si et seulement si il possède un préfixe de longueur $n-p$ qui est également un suffixe. C'est-à-dire qu'il existe $u \in \Sigma^{n-p}$ et $v, v' \in \Sigma^p$ tels que $w = uv = v'u$.

Le résultat le plus important concernant la périodicité des mots est sans aucun doute le théorème de Fine et Wilf (Fine et Wilf, 1965). Il décrit de manière précise la structure des mots possédant plusieurs périodes.

Théorème 1. *Si un mot w admet deux périodes p et q et que $|w| \geq p + q - \mathrm{pgcd}(p,q)$ alors $\mathrm{pgcd}(p,q)$ est aussi une période de w.*

Voir (Lothaire, 1997) Proposition 1.3.5 pour une preuve de ce résultat et (Giancarlo et Mignosi, 1994) pour une généralisation bi-dimensionnelle. À titre d'exemple, soit w le mot de longueur 13 suivant :

$$w = 0100100100100,$$

ce mot admet entre autre les périodes 6 et 9. Comme $|w| \geq 6+9-\mathrm{pgcd}(6,9) = 12$, le théorème de Fine et Wilf assure que w admet aussi la période $3 = \mathrm{pgcd}(6,9)$.

Il est également important de noter que ce thérorème propose une borne exacte. C'est-à-dire qu'il existe une classe infinie de mots w admettant deux périodes p et q avec $\mathrm{pgcd}(p,q) = 1$ tels que $|w| = p + q - 2$ mais n'admettant pas la période 1 (voir Section 1.4, Théorème 3).

1.3 Mots de Lyndon

Introduits par Lyndon (Lyndon, 1954; Lyndon, 1955), les mots de Lyndon jouent un rôle fondamental en combinatoire des mots. Relativement à l'ordre lexicographique, ils constituent les représentants canoniques des classes de conjugaison des mots primitifs. Étant donné un mot w sur un alphabet ordonné Σ, on définit les mots de Lyndon de la manière suivante :

Définition 8. Un mot $w \in \Sigma^*$ est de *Lyndon* si pour tous $u, v \in \Sigma+$, $w = uv$ implique que $w < vu$.

Il s'agit donc des mots qui sont strictement plus petits (relativement à l'ordre lexicographique) que tous les autres mots de leur classe de conjugaison. On remarque qu'un mot de Lyndon est forcément primitif puisque sinon il y aurait égalité avec au moins un conjugué. Initialement, les mots de Lyndon furent étudiés pour leur rôle déterminant dans le calcul de bases d'algèbres de Lie libres. C'est dans ce contexte que Chen, Fox et Lyndon ont énoncé le résultat suivant (Chen, Fox et Lyndon, 1958) :

Théorème 2 (Lyndon). *Tout mot w sur un alphabet ordonné Σ admet une unique décomposition :*

$$w = l_1^{n_1} l_2^{n_2} \cdots l_k^{n_k}$$

avec $n_1, n_2, \ldots, n_k \geq 1$ et où les l_i forment une suite des mots de Lyndon strictement décroissants $l_1 > l_2 > \cdots > l_k$.

Voir (Lothaire, 1997) pour une preuve élégante. Notons que cette décomposition est unique mais dépend de la relation d'ordre sur les lettres de l'alphabet. Par exemple, le mot $w = aabbaaabbbaa$ se décompose en :

- $w = (aabb) \cdot (aaabbb) \cdot (a)^2$ si $a < b$.
- $w = (a)^2 \cdot (bbaaa) \cdot (bbbaa)$ si $b < a$.

Un algorithme linéaire, et donc optimal, pour calculer cette factorisation est du à Fredricksen et Maiorana (Fredricksen et Maiorana, 1978) qui l'ont élaboré dans un tout autre contexte. Cet algorithme porte aujourd'hui le nom d'*algorithme de Duval* car ce dernier fut le premier

à montrer que cette factorisation se calcule en temps linéaire (Duval, 1980; Duval, 1983). La version présentée ici provient de (Lothaire, 2005) et procède de la manière suivante : on commence par calculer le premier terme $l_1^{n_1}$ de la factorisation en mots de Lyndon décroissants de façon à obtenir la factorisation $w = l_1^{n_1} w'$ puis on procède récursivement sur le mot w' jusqu'à ce qu'il soit entièrement factorisé.

Algorithme 1 (PremierFacteurDeLyndon).

Entrée : $w \in \mathcal{A}^n$;

Sortie : (l_1, n_1);

1 : $i \leftarrow 1$; $j \leftarrow 2$;
2 : **Tant que** $j \leq n$ **et** $w[i] \leq w[j]$ **faire**
3 : **Si** $w[i] < w[j]$ **alors**
4 : $i \leftarrow 1$;
5 : **sinon**
6 : $i \leftarrow i + 1$;
7 : **fin si**
8 : $j \leftarrow j + 1$;
9 : **fin Tant que**
10 : **retourne** $(w[1..j-i], \lfloor (j-1)/(j-i) \rfloor)$;

1.4 Mots de Christoffel

Les mots de Christoffel constituent un excellent exemple d'objets mathématiques qui dressent des ponts entre différents domaines d'études. Introduits par Christoffel en 1875, ils lient étroitement la combinatoire des mots à l'arithmétique des développements en fractions continues, mais surtout, ce qui est l'intérêt du présent ouvrage, à la géométrie discrète. Voir (Berstel et al., 2008) pour une présentation complète de la théorie des mots de Christoffel. Il existe plusieurs définitions équivalentes des mots de Christoffel. La première provient de l'article original de Christoffel (Christoffel, 1875).

Définition 9. Étant donné $k, n \in \mathbb{N}$, $k < n$, deux nombres relativement premiers, le *mot de Christoffel primitif* $C_{n,k} = w_1 w_2 w_3 \dots w_n$ est défini par :

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

où r_i est le reste de la division de (ik) par n.

Par exemple, si on prend $n = 17$ et $k = 5$, on obtient les valeurs suivantes :

r_0	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9	r_{10}	r_{11}	r_{12}	r_{13}	r_{14}	r_{15}	r_{16}	r_{17}
0	5	10	15	3	8	13	1	6	11	16	4	9	14	2	7	12	0

	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9	w_{10}	w_{11}	w_{12}	w_{13}	w_{14}	w_{15}	w_{16}	w_{17}
	0	0	0	1	0	0	1	0	0	0	1	0	0	1	0	0	1

Ainsi, le mot de Christoffel primitif associé à $n = 17$ et $k = 7$ est $C_{17,5} = 00010010001001001$.

En général on dira qu'un mot est de *Christoffel* s'il s'agit d'une puissance d'un mot de Christoffel primitif. Dans les années 1990, Borel et Laubie ont relancé l'étude des mots de Christoffel en mettant en valeur leur interprétation géométrique ainsi que leurs liens avec les mots de Lyndon (Borel et Laubie, 1993). Plusieurs auteurs considèrent deux types de mots de Christoffel : les positifs et les négatifs. La Définition 9 ne définit en fait que les mots de Christoffel positifs et ils seront les seuls considérés dans cet ouvrage, l'ensemble des mots de Christoffel négatifs étant obtenus en prenant le miroir des positifs.

1.4.1 Interprétation graphique de mots de Christoffel

À chaque mot de Christoffel $(C_{n,k})^m$ est associé un chemin dans le réseau carré. Pour ce faire, on relie par le segment s les points $(0, 0)$ et $(m(n-k), mk)$ et on considère l'ensemble des chemins composés uniquement de pas unitaires vers la droite et de pas vers le haut qui partent de $(0, 0)$ et se terminent à $(m(n-k), mk)$ en restant toujours en dessous du segment s. Parmi ces chemins, celui qui reste le plus près de la droite d est le chemin correspondant au mot de Christoffel $(C_{n,k})^m$. À partir d'un tel chemin on retrouve le mot de Christoffel en associant à chaque pas horizontal la lettre 0 et à chaque pas vertical la lettre 1, tel que l'illustre la Figure 1.1.

Cette représentation graphique introduit la notion de *pente* d'un mot sur un alphabet à deux lettres. On peut ainsi redéfinir les mots de Christoffel en fonction des pentes de leurs préfixes.

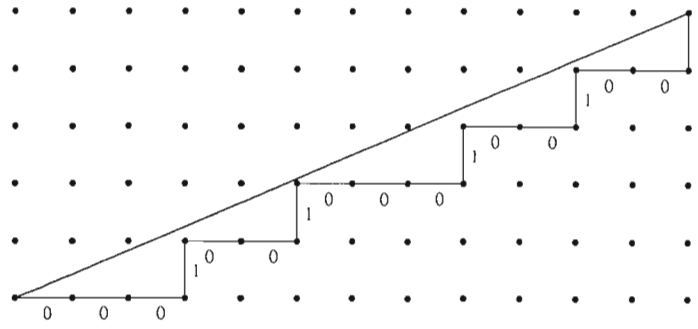


Figure 1.1 Chemin associé au mot de Christoffel $C_{17,5} = 00010010001001001001$.

Les notations qui suivent proviennent de (Berstel et de Luca, 1997). Soit ρ la fonction :

$$\rho : \{0,1\}^* \longrightarrow \mathbb{Q} \cup \{\infty\}$$

$$\rho(w) = \begin{cases} 1 \text{ si } w = \varepsilon, \\ |w|_1/|w|_0 \text{ si } w \neq \varepsilon \end{cases}$$

On supposera ici que $1/0 = \infty$ et on dira de que $\rho(w)$ est la *pente* du mot w. Ainsi la pente du mot de Christoffel primitif $C_{n,k}$ est $\rho(C_{n,k}) = k/(n-k)$.

Pour toute paire $w \in \Sigma^+$ et $k \in \{1, 2, \ldots, |w|\}$ on définit l'ensemble

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

et la mesure

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

Formalisant leur interprétation géométrique, cette deuxième définition des mots de Christoffel provient de (Borel et Laubie, 1993).

Définition 10. Un mot w est de Christoffel si et seulement si pour tout $k \in \{1, 2, \ldots, |w|\}$ on a

$$\rho(w_1 w_2 \cdots w_k) = \mu_k(w).$$

Les mots de Christoffel possèdent de nombreuses propriétés remarquables. En particuler, les quatre suivantes s'avèreront particulièrement utiles. Les trois premières proviennent de Borel et Laubie (Borel et Laubie, 1993) et la dernière provient de (Berstel et de Luca, 1997).

Propriété 1. *Soient c_1, c_2 deux mots de Christoffel et c un mot de Christoffel primitif.*

(i) *c est un mot de Lyndon.*

(ii) *$c_1 < c_2$ si et seulement si $\rho(c_1) < \rho(c2)$.*

(iii) *S'il existe deux entiers $k, l \geq 1$ tels que $c_1^k = c_2^l$ alors*

$$\rho(c_1) = \rho(c_2).$$

(iv) *Si $|c| \geq 2$ alors*

$$c = 0w1 \text{ pour un certain } w \in \mathrm{Pal}(\Sigma^*).$$

Notons qu'un mot de Christoffel primitif débute toujours par la lettre 0 et se termine par 1, à l'exception du mot 0 qui est associé à la droite horizontale et le mot 1 qui est associé à la droite verticale. Le mot obtenu en retirant la première et la dernière lettre d'un mot de Christoffel est appelé un *mot central*. Ces mots possèdent des propriétés combinatoires surprenantes. Le théorème suivant, du à Berstel et de Luca (Berstel et de Luca, 1997), caractérise entièrement les mots de Christoffel en fonction des périodes de leurs mots centraux.

Théorème 3. *Un mot primitif $w \in \Sigma^n$, $n \geq 2$ est de Christoffel si et seulement si $w = 0w'1$ pour un mot w' possédant deux périodes p et q relativement premières entre elles tels que $|w'| = p + q - 2$.*

Cette caractérisation est particulièrement intéressante car elle situe les mots de Christoffel primitifs comme les cas limites de l'application du théorème de Fine et Wilf.

En géométrie discrète, on définit habituellement les droites par une paire d'inégalités diophantiennes (voir (Reveillès, 1991)).

Définition 11. Trois entiers a, b, c tels que a et b sont relativement premiers définissent la *droite 4-connexe* $D_{a,b,c}$ de pente a/b comme étant l'ensemble :

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

Parmi les points de la droite $D_{a,b,c}$, ceux qui satisfont l'équation $ax - by = c$ sont appelés *points d'appui supérieurs*. Un mot de Christoffel de pente a/b code justement l'unique chemin inclus dans la droite 4-connexe $D_{a,b,c}$ qui relie deux points d'appui supérieurs.

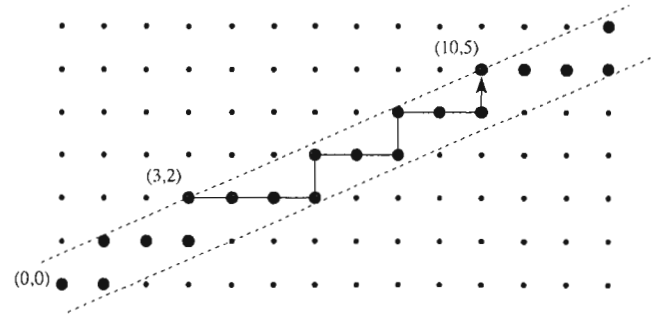


Figure 1.2 La droite 4-connexe $D_{3,7,-5} = \{(x, y) \in \mathbb{Z}^2 | -5 \leq 3x - 7y < 5\}$ et le chemin associé au mot de Christoffel de pente $3/7$ reliant deux points d'appui supérieurs.

1.4.2 Un algorithme optimal pour détecter les mots de Christoffel

Tester si un mot primitif est de Christoffel se fait en un temps qui est linéaire en fonction de la longueur de ce mot. La Définition 9 décrit exactement le travail à effectuer. La Propriété 1, (iv) permet même d'accélérer le travail puisqu'elle assure que le mot central de tout mot de Christoffel primitif est un palindrome. Donc, un mot de Christoffel primitif c de longueur $n \geq 2$, est toujours tel que pour tout $i \in \{2, 3, \ldots, \lceil n/2 \rceil\}$ on a $c_i = c_{n-i+1}$.

Algorithme 2 (estChristoffelPrimitif).

Entrée : $w \in \Sigma^n$

1 . $k \leftarrow |w|_1$; $i \leftarrow 1$; $r \leftarrow k$
2 : $rejete :=$ **non**$(w_1 = 0$ **et** $w_n = 1)$
3 : **Tant que non**$(rejete)$ **et** $i < \lceil n/2 \rceil$ **do**
4 : $i \leftarrow i + 1$; $r' \leftarrow r + k \mod n$
5 : **Si** $r < r'$ **alors**
6 : $rejete \leftarrow$ **non**$(w_i = w_{n-i+1} = 0)$
7 : **else**
8 : $rejete \leftarrow$ **non**$(w_i = w_{n-i+1} = 1)$
9 : **fin si**
10 : $r \leftarrow r'$
11 : **fin tant que**
12 : **retourne non**$(rejete)$

Notons qu'il est possible de modifier légèrement cet algorithme afin d'en obtenir un qui génère un mot de Christoffel primitif à partir des entiers n et k. Pour ce faire, il suffit de se débarrasser de la variable $rejete$ et de remplacer les tests des lignes 2, 6 et 8 par des affectations.

1.5 Mots et géométrie discrète

1.5.1 Chemins et codage de Freeman

L'interprétation géométrique des mots de Christoffel présentée à la Section 1.4.1 propose un lien entre les mots et les chemins dans le plan. Le codage de Freeman (Freeman, 1961; Freeman, 1970) permet de représenter tout chemin composé de pas élémentaires dans le plan discret $\mathbb{Z}^2$ comme un mot sur un alphabet à quatre lettres. Ce lien entre la géométrie discrète et la combinatoire des mots est à l'origine de tous les résultats présentés dans cet ouvrage.

Définition 12. Soit $p = (p_x, p_y) \in \mathbb{Z}^2$ un point du plan discret. Le 4-*voisinage* de p est l'ensemble

$$\left\{(x, y) \in \mathbb{Z}^2 \middle| |p_x - x| + |p_y - y| = 1\right\}.$$

Une suite de points voisins deux à deux forme un chemin.

Définition 13. Un 4-*chemin* est un sous-ensemble ordonné de points du plan discret $C = [(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)]$ tel que le point (x_{i+1}, y_{i+1}) fait partie du 4-voisinage du point (x_i, y_i) pour $i = 1, 2, \ldots, (n-1)$.

Soit C' un 4-chemin débutant au point $p \in \mathbb{Z}^2$, on construit le mot $w_{C'}$ sur l'alphabet $\{0, 1, \overline{0}, \overline{1}\}$ lettre par lettre en parcourant le 4-chemin C et en notant dans quelle direction est effectué chacun des déplacements unitaires.

- Un pas vers la droite ($\rightarrow$) est noté 0.
- Un pas vers la gauche ($\leftarrow$) est noté $\overline{0}$.
- Un pas vers le haut ($\uparrow$) est noté 1.
- Un pas vers le bas ($\downarrow$) est noté $\overline{1}$.

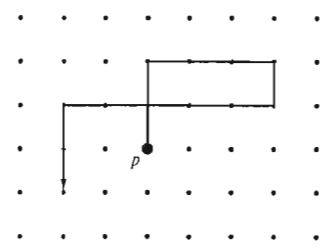


Figure 1.3 Un chemin codé par le mot $w = 1\,1\,0\,0\,0\,\overline{1}\,\overline{0}\,\overline{0}\,\overline{0}\,\overline{0}\,\overline{0}\,\overline{1}\,\overline{1}$.

Ainsi, tout mot w sur l'alphabet $\{0, 1, \overline{0}, \overline{1}\}$ correspond à un 4-chemin de sorte que la position relative du point final de ce chemin par rapport au point initial correspond à la somme des pas unitaires de ce chemin.

Notation 4. Soit $w \in \{0, 1, \overline{0}, \overline{1}\}^*$, on note $\overrightarrow{w}$ le vecteur :

$$\overrightarrow{w} = \sum_{1 \leq i \leq |w|} \overrightarrow{w_i},$$

où $\overrightarrow{0} = (1, 0)$, $\overrightarrow{1} = (0, 1)$, $\overrightarrow{\overline{0}} = (-1, 0)$, $\overrightarrow{\overline{1}} = (0, -1)$.

On dit d'un mot w qu'il code un chemin *fermé* si $\overrightarrow{w} = \overrightarrow{0}$.

1.5.2 Polyominos

Les polyominos, introduits par Gardner dans ses célèbres *Mathematical Games* (Gardner, 1958), peuvent être considérés comme les éléments de base de la géométrie discrète. Ils sont, d'une certaine manière, l'équivalent discret des ensembles simplement connexes d'aire finie en géométrie euclidienne. C'est Golomb (Golomb, 1996) qui utilisa en premier le mot *polyomino* pour désigner ces objets. La notion de chemin dans le plan discret permet d'abord d'étendre la notion familière de connexité en géométrie euclidienne au plan discret.

Définition 14. Un sous-ensemble $S \in \mathbb{Z}^2$ est *4-connexe* si pour toute paire de points $p, q \in S$, il existe un 4-chemin $\mathcal{C} = [p_1, p_2, \ldots, p_n]$ entièrement inclus dans S tel que $p = p_1$ et $q = p_n$.

On peut maintenant définir un polyomino :

Définition 15. Un *polyomino* P est un sous-ensemble fini, 4-connexe du plan discret $\mathbb{Z}^2$ qui ne possède pas de trous.

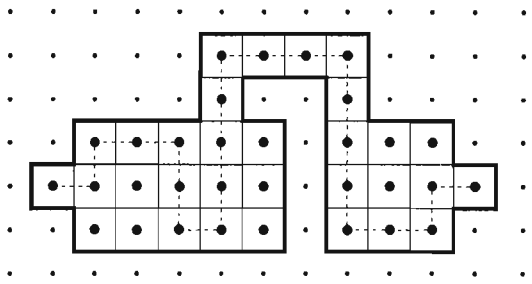


Figure 1.4 Un polyomino avec en gras son bord et en pointillé un chemin 4-connexe reliant son point le plus à gauche à son point le plus à droite.

On dit qu'un sous-ensemble fini de $\mathbb{Z}^2$ contient un trou si son complément n'est pas 4-connexe. Il est agréable de visualiser les ensembles de points discrets comme des pixels, c'est-à-dire des carrés unitaires centrés sur un point à coordonnée entière. Pour cette raison, on représente un polyomino par un ensemble de carrés avec un point en leur centre (voir Figure 1.4). On représente souvent un polyomino en ne traçant que le *bord*, c'est-à-dire seulement les côtés extérieurs des carrés unitaires formant le contour du polyomino. Il est important de garder en tête que seuls les points centraux, élément de $\mathbb{Z}^2$, font réellement partie du polyomino.

Une version moins restrictive de la notion de connexité discrète est la 8-connexité. Dans ce cas-ci, on définit le 8-voisinage d'un point de la manière suivante :

Définition 16. Soit $p = (p_x, p_y) \in \mathbb{Z}^2$ un point du plan discret. Le 8-*voisinage* de p est l'ensemble

$$\left\{(x, y) \in \mathbb{Z}^2 \middle| max\left\{|p_x - x|, |p_y - y|\right\} = 1\right\}.$$

Il en découle alors les définitions d'un 8-*chemin* et de la 8-*connexité* de la même manière que dans le cas précédent. Notons que tout ensemble 4-connexe est également 8-connexe.

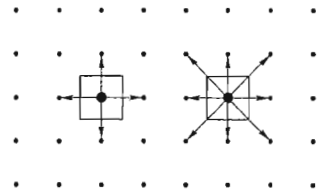


Figure 1.5 Illustration de la 4-connexité (à gauche) et de la 8-connexité (à droite).

1.5.3 Mots de contour

En plus de représenter les 4-chemins par des mots, le codage de Freeman permet également de coder les polyominos par des mots sur l'alphabet $\{0, 1, \overline{0}, \overline{1}\}$. Ceci ouvre la porte à l'étude des propriétés géométriques des polyominos en observant la structure combinatoire des mots qui les codent.

Définition 17. Soit P un polyomino et $p \in \mathbb{Z}^2 - (1/2, 1/2)$ un point sur le bord de P. Le mot w code le *contour* du polyomino P à partir du point p si, lorsqu'on translate le polyomino P par le vecteur $\overrightarrow{v} = (1/2, 1/2)$, le chemin codé par w à partir du point $p + \overrightarrow{v}$ effectue un parcours du bord de P en sens horaire.

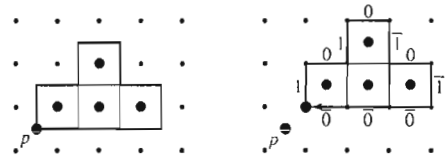


Figure 1.6 Un polyomino dont le mot de contour à partir du point p est $w = 1\,0\,1\,0\,\overline{1}\,0\,\overline{1}\,\overline{0}\,0\,\overline{0}$.

On dira d'un mot w qu'il s'agit d'un *mot de contour* s'il code le contour d'un polyomino. Comme cette façon d'encoder les polyominos dépend du point de départ choisi, plusieurs mots de contour codent le même polyomino.

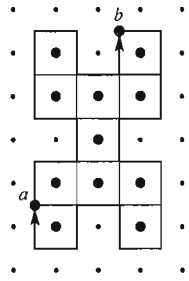
Notation 5. Soit P un polyomino, on note $b(\mathrm{P})$ l'ensemble des mots de contour de P.

Évidemment, si deux mots de contour u et v codent le même polyomino, on a alors que $u \equiv v$. Ainsi, l'ensemble $b(\mathrm{P})$ correspond à la classe de conjugaison d'un des mots de contour de P.

Étant donné un polyomino $\mathbf{P}$, on peut s'intéresser au mot codant une partie de son bord.

Notation 6. Soit $\mathbf{P}$ un polyomino et $a, b \in \mathbb{Z}^2 - (1/2, 1/2)$ deux points distincts situés sur le bord de $\mathbf{P}$. On note $\mathbf{P}[\mathbf{a..b}]$ le mot codant la partie du bord de $\mathbf{P}$ allant de a à b en sens horaire.

Bien entendu, pour toute paire de points a, b sur le bord d'un polyomino $\mathbf{P}$, $\mathbf{P}[a..b] \cdot \mathbf{P}[b..a] \in b(\mathbf{P})$, comme l'illustre la figure ci-dessous.



Un polyomino $\mathbf{P}$ dont le bord est codé par
$w = 101\overline{0}110\overline{1}010\overline{1}\overline{1}0\overline{1}0\overline{1}\overline{1}0\overline{1}0\overline{1}01 \in b(\mathbf{P})$,
$P[a..b] = 101\overline{0}110\overline{1}01$,
$P[b..a] = 0\overline{1}\overline{1}0\overline{1}0\overline{1}\overline{1}0\overline{1}0\overline{1}01$.

Un bel exemple de propriété géométrique se traduisant par une propriété combinatoire sur les mots de contour est maintenant présentée. Daurat et Nivat (Daurat et Nivat, 2003) ont étudié le nombre de coins *saillants* et *rentrants* d'un polyomino et ont établi que le nombre de coins saillants est toujours égal au nombre de coins rentrants plus quatre. Brlek, Labelle et Lacasse ont montré (Brlek, Labelle et Lacasse, 2005; Brlek, Labelle et Lacasse, 2006) que ce résultat peut être obtenu combinatoirement comme suit. Soient

$$\mathcal{G} = \{01, 1\overline{0}, \overline{0}\overline{1}, \overline{1}0\},$$
$$\mathcal{D} = \{0\overline{1}, 10, \overline{0}1, \overline{1}\overline{0}\},$$

respectivement l'ensemble des virages à gauche et des virages à droite. Par abus de notation, on notera $|w|_{\mathcal{G}}$ (respectivement $|w|_{\mathcal{D}}$) le nombre de virages à gauche (resp. à droite) effectués lors du parcours du chemin codé par w. Pour des raisons pratiques, on définit Δ la fonction qui compte la différence entre le nombre de virages à gauche et le nombre de virages à droite par

$$\Delta(w) = |w|_{\mathcal{G}} - |w|_{\mathcal{D}}.$$

La fonction Δ est *additive* au sens où

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

Lorsque le mot w code un chemin fermé, il y a possiblement un virage supplémentaire à la fin du chemin. C'est pourquoi on définit la spécialisation suivante de la fonction Δ aux chemins fermés

$$\Delta_C(w) = \Delta(w \cdot f(w)).$$

On remarque que la fonction Δ_C est invariante par conjugaison du mot.

Propriété 2 (Brlek, Labelle et Lacasse). *Soit $w \in \Sigma^*$ un mot codant un chemin fermé qui ne se croise pas sur la grille carrée, alors $\Delta_C(w) = 4$ pour un parcours effectué en sens anti-horaire et $\Delta_C(w) = -4$ pour un parcours en sens horaire.*

Notons que cette propriété s'applique à une classe de mots plus générale que ceux qui codent le bord des polyominos.

On peut maintenant donner une condition nécessaire et suffisante pour qu'un mot soit un mot de contour.

Propriété 3. *Un mot $w \in \{0, 1, \overline{0}, \overline{1}\}^*$ est un mot de contour si et seulement si les conditions suivantes sont vérifiées :*

(i) $\Delta_C(w) = -4$;

(ii) *Pour tout* $u \in \mathrm{Fact}(w)$, *on a* $\overrightarrow{u} = \overrightarrow{0} \iff u \in \{\varepsilon, w\}$.

Preuve. Soit w un mot de contour. La Proposition 2 assure que la condition (i) est respectée. Puisque w code le bord d'un polyomino, il ne passe jamais deux fois au même point et donc il ne contient aucun facteur propre $u \neq \varepsilon$ tel que $\overrightarrow{u} = \overrightarrow{0}$ d'où la condition (ii).

Inversement, la condition (ii) assure que le chemin est fermé et ne se croise pas, tandis que la condition (i) assure que le parcours est fait en sens horaire. ∎

En particulier, un facteur de la forme $a\overline{a}$ où $a \in \{0, 1, \overline{0}, \overline{1}\}$ ne peut jamais faire partie d'un mot de contour.

1.5.4 Opérations sur les mots de contour

Le codage de Freeman relie directement la structure d'un mot aux propriétés géométriques de la figure qu'il code. Il est donc naturel de se poser la question suivante :

"*Soit* P *un polyomino et* $w \in b(\mathrm{P})$, *étant donné une fonction*

$$\phi \,.\, \{0,1,\overline{0},\overline{1}\}^* \longrightarrow \{0,1,\overline{0},\overline{1}\}^*,$$

que peut-on dire du chemin codé par $\phi(w)$ *?*"

Considérons tout d'abord le morphisme σ défini par

$$\begin{aligned} &\sigma : \{0,1,\overline{0},\overline{1}\}^* \longrightarrow \{0,1,\overline{0},\overline{1}\}^* \\ &\sigma(0) = 1 \qquad \sigma(\overline{0}) = \overline{1} \\ &\sigma(1) = \overline{0} \qquad \sigma(\overline{1}) = 0 \end{aligned}$$

Proposition 1. *Soit* P *un polyomino et* $w \in b(\mathrm{P})$, *le mot* $\sigma(w) \in b(\mathrm{Q})$ *où* Q *est l'image de* P *par une rotation de* $\pi/2$.

Ce morphisme ainsi que cette interprétation géométrique sont présentés dans (Brlek, Labelle et Lacasse, 2005; Brlek, Labelle et Lacasse, 2006). Considérons maintenant l'opérateur $\overline{\ }$ qui remplace chaque lettre par son barré et vice-versa. Cet opérateur correspond à une rotation de π puisque pour toute lettre $a \in \{0,1,\overline{0},\overline{1}\}$, l'image de a obtenue en appliquant deux fois le morphisme σ est $\sigma^2(a) = \overline{a}$.

Par exemple, considérons le mot de contour suivant ainsi que son image par applications successives du morphise σ :

$$\begin{aligned} w &= 1\,0\,1\,0\,\overline{1}\,0\,0\,\overline{1}\,\overline{0}\,\overline{0}\,\overline{0}\,\overline{0}, \\ \sigma(w) &= \overline{0}\,1\,\overline{0}\,1\,0\,1\,1\,0\,\overline{1}\,\overline{1}\,\overline{1}\,\overline{1}, \\ \sigma^2(w) &= \overline{1}\,\overline{0}\,\overline{1}\,\overline{0}\,1\,\overline{0}\,\overline{0}\,1\,0\,0\,0\,0, \\ \sigma^3(w) &= 0\,\overline{1}\,0\,\overline{1}\,\overline{0}\,\overline{1}\,\overline{1}\,\overline{0}\,1\,1\,1\,1. \end{aligned}$$

Les polyominos codés par $w, \sigma(w), \sigma^2(w)$ et $\sigma^3(w)$ sont représentés à la Figure 1.7.

On s'intéresse maintenant à l'interprétation géométrique de l'opérateur $\widetilde{\ }$ qui à un mot lui associe son miroir.

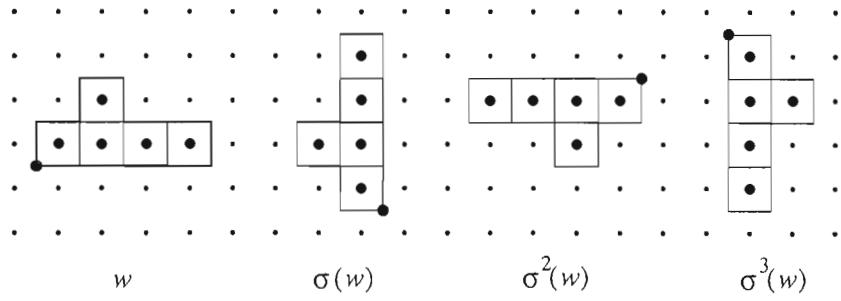


Figure 1.7 Un polyomino dont le mot de contour à partir du point p est $w = 1010\bar{1}00\bar{1}\bar{0}\bar{0}\bar{0}\bar{0}$ et son image par l'application du morphisme σ.

Proposition 2. *Soit* P *un polyomino et* Q *son image par une rotation de* π. *Si* $w \in b(\mathrm{P})$, *alors le chemin codé par* $\widetilde{w}$ *code le bord de* Q *mais parcouru en sens anti-horaire.*

Par exemple, considérons encore une fois le mot $w = 1010\bar{1}00\bar{1}\bar{0}\bar{0}\bar{0}\bar{0}$, la Figure 1.8 illustre le chemin codé par $\widetilde{w} = \bar{0}\bar{0}\bar{0}\bar{0}\bar{1}00\bar{1}0101$.

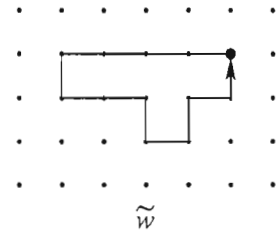


Figure 1.8 Chemin codé par $\widetilde{w} = \bar{0}\bar{0}\bar{0}\bar{0}\bar{1}00\bar{1}0101$.

On introduit finalement pour usage ultérieur l'opérateur $\widehat{\ } = {}^{-} \circ {}^{\sim}$. Cet anti-morphisme joue un rôle fondamental dans la détection des polyominos exacts, problème étudié en détail au Chapitre 4.

Proposition 3. *Soit* P *un polyomino et* $w \in b(\mathrm{P})$, *le chemin codé par* $\widehat{w}$ *code le bord de* P *mais parcouru en sens anti-horaire.*

En reprenant le même exemple, le mot $\widehat{w} = 0000\bar{1}\bar{0}\bar{0}1\bar{0}\bar{1}\bar{0}\bar{1}$ code le chemin suivant :

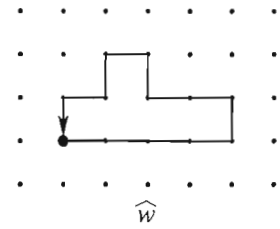


Figure 1.9 Chemin codé par $\widehat{w} = 0\,0\,0\,0\,1\,\overline{0}\,\overline{0}\,1\,\overline{0}\,\overline{1}\,\overline{0}\,\overline{1}$.

1.6 Arbres suffixes

En informatique, il est classique de représenter un langage fini par un arbre dont les arêtes sont étiquetées par des mots. Dans cet arbre, la concaténation des étiquettes des arêtes formant un chemin de la racine à une feuille donne un des mots du langage. L'*arbre des suffixes* d'un mot w est un cas particulier où l'ensemble de mots considéré est formé des suffixes du mot w. La présente section a pour but d'exposer la structure des arbres suffixes, voir (Gusfield, 1997) pour une explication détaillée de leur construction ainsi que leurs principales applications. L'arbre des suffixes d'un mot en expose toute sa structure interne et constitue un outil algorithmique majeur pour le traitement des chaînes de caractères, voir (Apostolico, 1985).

Définition 18. L'*arbre des suffixes* $\mathcal{A}$ d'un mot $w \in \Sigma^n$ est un arbre possédant les caractéristiques suivantes :

- Il possède exactement n feuilles étiquetées de 1 à n.
- Chaque noeud interne, à l'exception possible de la racine, possède au moins deux fils.
- Chaque arête est étiquetée par un facteur non-vide de w.
- Il ne peut y avoir deux arêtes partant d'un même noeud dont les étiquettes ont un même préfixe non-vide.
- La concaténation des étiquettes des arêtes reliant la racine à la feuille i forme le suffixe $w[i..n]$.

Le premier résultat théorique important relatif aux arbres des suffixes provient de Weiner (Weiner, 1973) qui appelait ces structures des *arbres de positions*. Dans son article de 1973, il a montré comment les construire en un temps linéaire en fonction de la longueur du mot

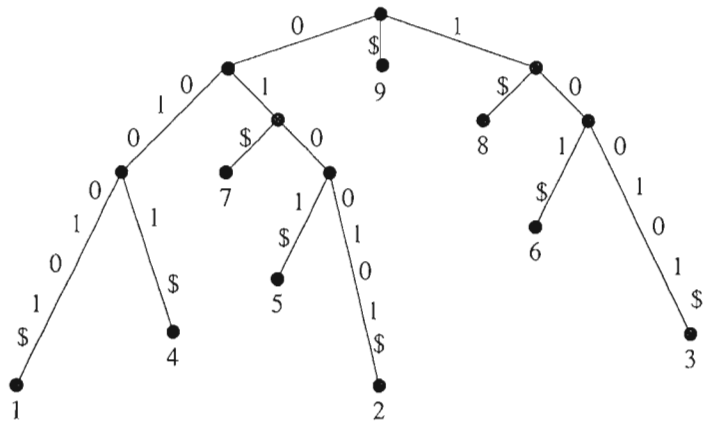


Figure 1.10 L'arbres des suffixes du mot 00100101$.

considéré. Quelques années plus tard, McCreight (McCreight, 1976) proposa une méthode plus simple et plus efficace pour construire l'arbre des suffixes d'un mot, mais le tout demeure relativement complexe n'attirant que peu d'intérêt sur les arbres suffixes. Il faudra attendre près de 20 ans pour qu'Ukkonen (Ukkonen, 1995) élabore un algorithme permettant la construction *en-ligne* de l'arbre des suffixes. Cette dernière méthode est significativement plus simple que les précédentes et tout aussi efficace quant à la complexité temporelle. Voir (Giegerich et Kurtz, 1997) pour une comparaison détaillée de ces trois algorithmes.

Afin de satisfaire la définition de l'arbre des suffixes, on ajoute un caractère spécial, en général $, à la fin des mots considérés. Ce caractère ne se retrouvant nulle part ailleurs dans le mot assure que tous les suffixes terminent à une feuille de l'arbre. Considérons l'exemple de la Figure 1.10, sans le caractère $ à la fin du mot, les suffixes $0, 01$ et 010 termineraient sur des noeuds internes de l'arbre.

1.6.1 Recherche du plus bas ancêtre commun en temps constant

La recherche du plus bas ancêtre commun est un problème motivé par l'affirmation naïve suivante :

> *"Tout facteur d'un mot est le préfixe d'un suffixe."*

Considérons les suffixes $s_i = w[i..n]$ et $s_j = w[j..n]$ avec $i \neq j$ du mot $w \in \Sigma^+$. Soit u le plus long préfixe commun à s_i et s_j, il existe alors deux lettres distinctes $a, b \in \Sigma$ et $v, v' \in \Sigma^*$ tels que $s_i = uav$ et $s_j = ubv'$.

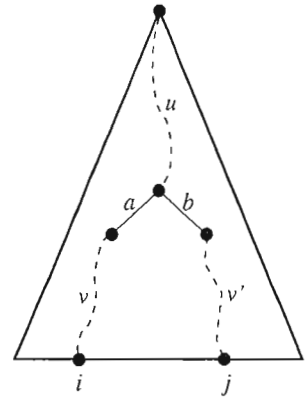


Figure 1.11 Les chemins reliant la racine d'un arbre des suffixes aux feuilles i et j.

Commme l'illustre la Figure 1.11, le chemin allant de la racine au plus bas ancêtre commun de deux feuilles code exactement le plus long préfixe commun à ces deux suffixes. On remarque encore une fois que la présence d'un caractère spécial à la fin du mot assure qu'aucun suffixe n'est préfixe d'un autre suffixe. Par exemple, dans l'arbre illustré à la Figure 1.10, le plus bas ancêtre commun aux feuilles 2 et 5 code le facteur 010 ce qui est bien le plus long préfixe commun aux suffixes $w[2..n] = 0100101\$$ et $w[5..n] = 0101\$$.

Définition 19. Étant donné un mot $w \in \Sigma^n \cdot \{\$\}$ et $i, j \in \{1, 2, \ldots, n\}$, la *plus longue extension* de i et j est la longueur du plus long préfixe commun aux suffixes $w[i..n]$ et $w[j..n]$.

Une solution naïve au calcul de la plus longue extension consiste à comparer les lettres des deux suffixes une par une jusqu'à ce qu'une différence soit constatée. Cette méthode pouvant nécessiter un temps de traitement en $\mathcal{O}(n)$ est amplement satisfaisante si on veut calculer un nombre limité d'extensions. Il existe par contre des solutions algorithmiques qui permettent, après un prétraitement nécessitant un temps en $\mathcal{O}(n)$, de calculer de telles extensions en temps constant.

Développé initialement par Harel et Tarjan (Harel et Tarjan, 1984) puis simplifiée par Schieber et Vishkin (Schieber et Vishkin, 1988), la recherche du plus bas ancêtre commun en temps constant est un résultat algorithmique remarquable. Il s'agit d'un outil puissant permettant de traiter de nombreux problèmes relatifs à la détection de répétitions dans les mots. On trouve dans (Gusfield, 1997) une présentation détaillée de cet algorithme et nous en donnons une application qui nous sera utile par la suite : le calcul de la plus longue extension commune dans deux mots.

1.6.2 Plus longue extension commune

Le problème du calcul de la plus longue extention se transpose naturellement au calcul d'extensions commune de deux mots. En effet, étant donné deux mots $w_1, w_2 \in \Sigma^+$ et deux symboles extérieurs à l'alphabet $\$, \# \notin \Sigma$, l'utilisation de la recherche en temps constant du plus bas ancêtre commun dans l'arbre des suffixes du mot $w = w_1 \# w_2 \$$ calcule directement la plus longue extension commune aux deux mots.

Définition 20. Étant donné une position i dans le mot w_1 et une position j dans le mot w_2, la *plus longue extension commune à droite*, notée $\mathrm{PLECD}(w_1, w_2, i, j)$, est le plus grand entier k tel que $w_1[i..i+k-1] = w_2[j..j+k-1]$. On définit similairement la *plus longue extension commune à gauche* notée $\mathrm{PLECG}(w_1, w_2, i, j)$.

Par exemple, étant donné les mots :

$$\begin{aligned} w_1 &= 0\underline{0011}1\underline{00}100101010, \\ w_2 &= 101\underline{0011}1\underline{00}110101011. \end{aligned}$$

On a $\mathrm{PLECD}(w_1, w_2, 6, 8) = 4$ et $\mathrm{PLECG}(w_1, w_2, 6, 8) = 5$.

Théorème 4. *Après un prétraitement en temps linéaire, le calcul de la plus longue extension commune à deux mots peut être effectuée en temps constant.*

Ce résultat constitue le point de départ des algorithmes de détection des polyominos exacts présentés au Chapitre 4.

Chapitre II

CHEMINS AUTO-ÉVITANTS

2.1 Introduction

La notion de *chemin auto-évitant*, c'est-à-dire un chemin qui ne passe pas deux fois au même point, vient naturellement lorsqu'on considère le codage de Freeman. Algorithmiquement, "tracer" le chemin codé nécessite un espace mémoire beaucoup trop grand. En effet, une manière naïve de résoudre ce problème serait d'utiliser une matrice creuse initialisée avec des 0 partout et littéralement tracer le chemin codé en écrivant des 1 dans les cases visitées tout au long du parcours. Malheureusement, une telle approche nécessite l'initialisation d'une matrice dont la taille est quadratique en fonction de la longueur du mot analysé.

Une autre méthode consiste à utiliser une structure de données dans laquelle seront rangées les coordonnées des points visités. Pour chaque point du chemin codé par w, il suffit donc de tester si ce point est déjà présent dans la structure et de l'ajouter s'il n'y est pas. L'utilisation d'une structure arborescente auto-équilibrante, arborescence AVL par exemple, fournit un algorithme en $\mathcal{O}(n \log n)$. L'utilisation d'une table de hachage assure un temps linéaire en moyenne mais $\mathcal{O}(n \log n)$ au pire cas. Ceci est insatisfaisant d'autant plus que les deux prochains chapitres présentent des algorithmes linéaires en fonction de la longueur d'un mot de contour permettant de déterminer des propriétés géométriques d'un polyomino. De tels outils s'avèrent peu intéressants en l'absence d'un algorithme linéaire pour déterminer si un mot quelconque sur l'alphabet $\{0, 1, \overline{0}, \overline{1}\}$ code un chemin auto-évitant car tester si un mot code le bord d'un polyomino se ramène à vérifier s'il passe deux fois par le même point.

Le présent chapitre vient alors remplir ce vide algorithmique en proposant un algorithme linéaire, donc optimal, permettant de tester si un chemin donné par une suite de pas élémentaires passe deux fois au même point.

Avant de commencer, il est important de noter qu'en général lorsqu'on analyse la complexité temporelle d'un algorithme, on suppose que les *petits* nombres se manipulent en temps constant. Par *petits* on entend les nombres dont la taille est dans $\mathcal{O}(\log n)$. L'algorithme présenté ici ne nécessite pas cette hypothèse et reste linéaire même sur un modèle de machine plus restrictif. Pour y parvenir, on utilise une structure arborescente dont les noeuds représentent des points du plan mais on ne stocke jamais leurs coordonnées. La structure de l'arbre à elle seule encode les coordonnées de chacun des points et permet d'assurer la cohérence de la structure de sorte que si le chemin codé passe deux fois par le même point, un noeud sera alors visité deux fois.

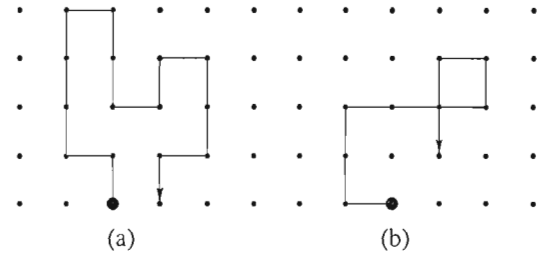


Figure 2.1 (a) Chemin auto-évitant codé par $1\overline{0}111\overline{01}\overline{10}10\overline{1}101$. (b) Chemin qui se croise codé par $\overline{0}1100010\overline{11}$.

2.2 Structure de données

Dans un premier temps, on supposera que le chemin considéré est tel que si son point de départ est situé à l'origine $(0,0)$, alors tous les points du chemin sont situés dans le premier quadrant. Définissons d'abord la notion de voisins dans le premier quadrant.

Définition 21. Pour chaque $\alpha \in \{0,1,\overline{0},\overline{1}\}$, le α-*voisin* d'un point $(x,y) \in \mathbb{N}^2$ est le point $(x',y') \in \mathbb{N}^2$ tel que $(x',y') = (x+\alpha_x, y+\alpha_y)$. Les *voisins* d'un point sont les éléments de l'ensemble des α-voisins pour tous les $\alpha \in \Sigma$.

Notons que chaque point possède exactement quatre voisins à l'exception du point $(0,0)$ qui

n'en possède que deux ((0,1) et (1,0)) et les points de la forme $(x, 0)$ et $(0, y)$ pour $x.y \geq 1$ qui n'admettent que trois voisins.

L'idée de cet algorithme est de bâtir un graphe de type arborescent dont les noeuds représentent des points du plan. Chacun de ces points peut avoir deux états possibles : *visité* ou *non-visité*. En lisant le mot $w = w_1 w_2 \cdots w_n$ de gauche à droite, on crée un nouveau noeud pour chaque lettre lue de manière à ce que le noeud créé après avoir lu la lettre w_i correspond au point du plan $\overrightarrow{w[1..i]}$. Chacun de ces points est alors marqué comme visité et, bien sûr, si à un certain moment un point est visité deux fois, c'est que le chemin n'est pas auto-évitant et l'algorithme s'arrête.

On définit le graphe $\mathcal{G} = (N, R, V)$ où N est un ensemble de noeuds associés aux points du plan, on appellera le noeud (x, y) le noeud représentant le point de coordonnées (x, y). Notons que ces étiquettes sont virtuelles puisqu'il est inutile d'inscrire cette information dans le noeud. R et V sont quant à eux deux ensembles distincts d'arêtes orientées. Les arêtes de l'ensemble R donnent une structure de *quadtree* alors que les arêtes de V relient les paires de noeuds représentant deux points voisins.

2.2.1 Radix-tree quaternaire

Le sous-graphe (N, R) forme un radix-tree dont la racine est le noeud $(0, 0)$ et chaque noeud (à l'exception de la racine) peut avoir jusqu'à quatre fils ; les arêtes menant aux enfants d'un noeud sont étiquetées par des paires dans l'ensemble $\{(0,0), (0,1), (1,0), (1.1)\}$. On définit la *hauteur* de $\mathcal{G}$ comme étant la hauteur de l'arbre (N, R).

Si l'arête reliant le noeud (x, y) à son fils possède l'étiquette $(\varepsilon_x, \varepsilon_y)$, alors le fils représente le point $(2x + \varepsilon_x, 2y + \varepsilon_y)$. Évidemment, la racine possède au maximum trois fils puisqu'une arête étiquetée par $(0, 0)$ partant de la racine devrait revenir sur elle-même.

On remarque que chaque couple (x, y) d'entiers positifs peut être représenté exactement une seule fois dans un tel arbre. De plus, lorsqu'on écrit les nombres x et y en base 2, en ajoutant des zéros devant l'écriture du plus court de manière à obtenir des mots de même longueur, la séquence de couples de bits obtenus en lisant simultanément ces deux mots (d'abord la première

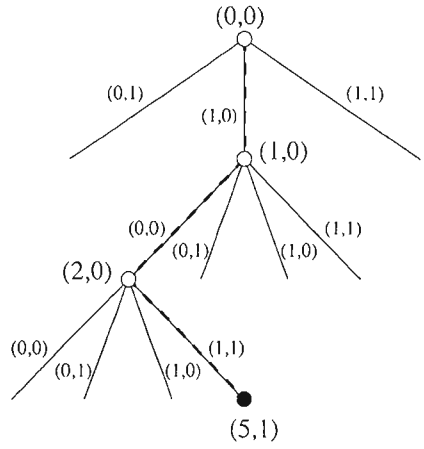


Figure 2.2 Un radix-tree et le chemin allant de la racine au point $(5, 1) = (101_2, 001_2)$.

lettre de chacun des deux mots, puis la deuxième, et ainsi de suite) correspond exactement aux étiquettes de l'unique chemin menant de la racine au noeud (x, y).

2.2.2 Liens de voisinage

A cette structure on ajoute ce que nous appelons des *liens de voisinage*. Chacune des arêtes de l'ensemble V est étiquetée par une des quatres translations élémentaires de l'ensemble $\{(0,1),(0,-1),(1,0),(-1,0)\}$ de manière à ce qu'une arête portant l'étiquette $(\varepsilon_x, \varepsilon_y)$ part du noeud (x, y) et termine au noeud $(x + \varepsilon_x, y + \varepsilon_y)$.

2.3 Le principe de base

Lorsqu'on additionne 1 à un nombre écrit en base 2, il y a deux possibilités : soit ce nombre se termine par un 0 et il suffit de le changer en 1, soit ce nombre se termine par une suite de 1 précédée d'un 0 et il faut changer ces 1 en 0 et le 0 en 1. Ce processus requiert donc un nombre d'étape proportionnel à la longueur de la séquence de 1.

Supposons maintenant qu'on souhaite additionner 1 à un nombre impair x mais que l'on connaisse déjà le résultat de $\lfloor \frac{x}{2} \rfloor + 1$. Il suffit alors d'ajouter un 0 à l'écriture en base 2 de $\lfloor \frac{x}{2} \rfloor + 1$ pour obtenir le nombre $x + 1$.

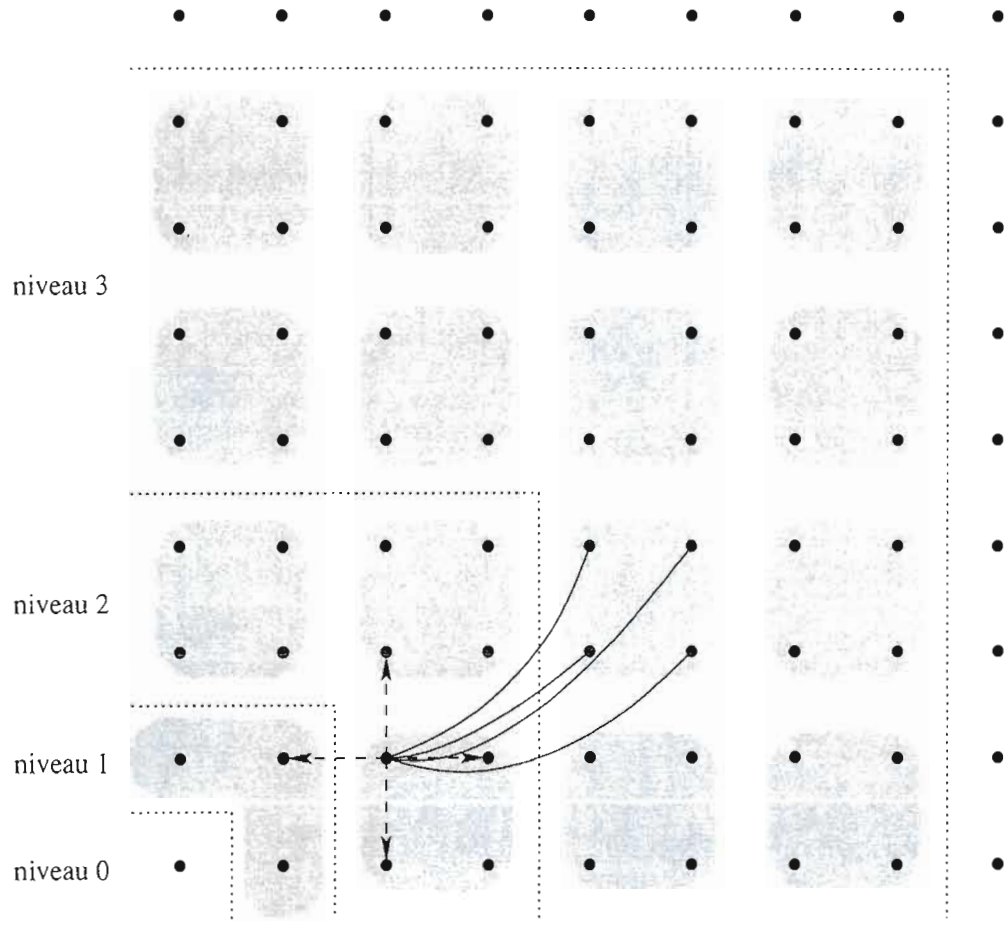


Figure 2.3 Le point $(2, 1)$ avec ses 4 fils (lignes pleines) et ses 4 voisins (flèches en tirets). Les zones grises regroupent les fils d'un même noeud et les pointillés séparent les différents niveaux de l'arbre.

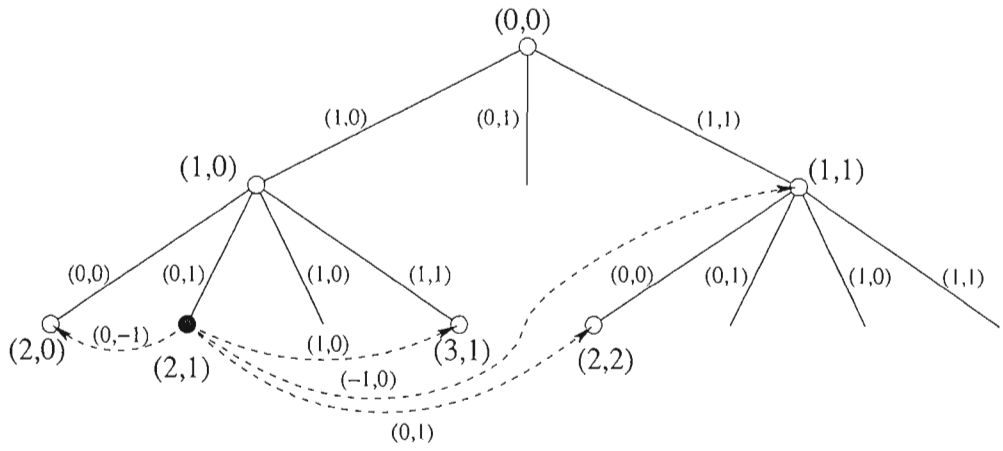


Figure 2.4 Le noeud $(2, 1)$ et ses quatres liens de voisinages.

Limitons nous pour l'instant au cas unidimensionnel et supposons qu'on ait un radix-tree contenant des nombres écrits en base 2 ainsi que des liens vers leurs voisins (ici un noeud peut avoir deux voisins, un voisin "+1" et un voisin "-1"). Soit n un noeud représentant le nombre impair x dont l'écriture binaire est $x = \alpha 0 \underbrace{11\ldots1}_{k \text{ fois}}$, on peut passer au noeud n' représentant le nombre $x + 1$ en suivant les trois étapes suivantes :

1. Remonter au père de n (ce noeud représente le nombre $\lfloor \frac{x}{2} \rfloor$).
2. Suivre le lien vers son voisin "+1".
3. Descendre à son fils en suivant l'arête d'étiquette 0.

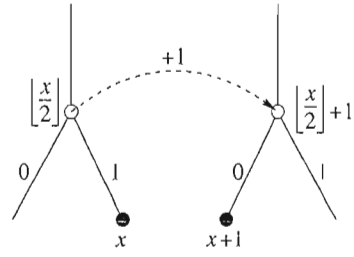


Figure 2.5 Le cas unidimensionnel avec un nombre impair x.

2.3.1 Application à notre structure

Dans le problème que nous considérons, chacune des quatre translations élémentaires $(0,1)$, $(1,0)$, $(0,-1)$ et $(-1,0)$ laisse une des deux coordonnées inchangée. On va donc effectuer le traitement décrit précédemment sur la coordonnée modifiée et mémoriser le dernier bit de la coordonnée inchangée. Ainsi, pour effectuer la translation $(1,0)$ à partir du noeud n représentant le point (x,y) dont l'écriture binaire est $(\alpha 0\underbrace{11\ldots1}_{k \text{ fois}},\beta\nu)$ on fera :

1. Remonter au père de n.
2. Suivre le lien de voisinage (1,0).
3. Descendre à son fils en suivant l'arête d'étiquette $(0,\nu)$.

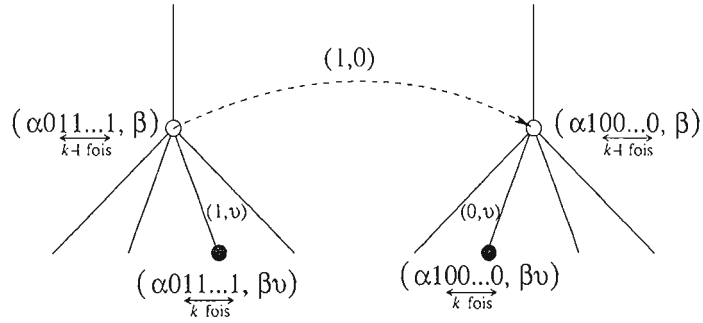


Figure 2.6 Le cas bidimensionnel avec un radix-tree.

Dans le cas où le lien reliant le père à son voisin n'a pas été construit, il suffit de répéter le processus récursivement.

2.3.2 Exemple

Au début de l'algorithme, le graphe ne contient qu'un seul sommet visité, soit la racine $(0,0)$. On ajoute immédiatement ces deux fils et voisins, les sommets $(0,1)$ et $(1,0)$. On appelle le graphe initial $\mathcal{G}_\varepsilon$.

En liant ainsi la racine à ses deux voisins, notons que la racine n'en possède que deux, on s'assure que lorsqu'on remonte l'arbre de manière récursive on finit toujours par arriver à un noeud relié à ses voisins. Supposons maintenant qu'on lise le mot $w = 0011$, on débute par

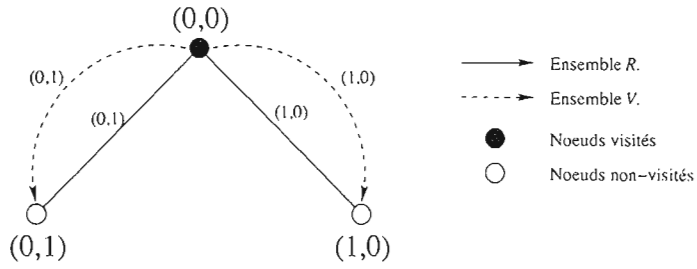


Figure 2.7 Le graphe initial $\mathcal{G}_\varepsilon$.

la lettre $w_1 = 0$ correspondant à la translation $(1,0)$. En partant de la racine, il suffit de suivre le lien de voisinage étiqueté par $(1,0)$ pour arriver au sommet $(1,0)$ et de marquer ce dernier comme visité. On appelle ce nouveau graphe $\mathcal{G}_0$.

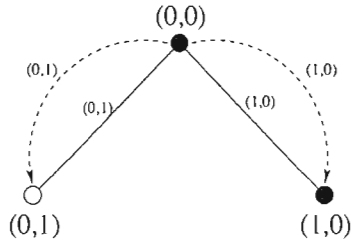


Figure 2.8 Le graphe $\mathcal{G}_0$ obtenu après la lecture de la lettre 0.

On passe ensuite à la deuxième lettre $w_2 = 0$ correspondant encore une fois à la translation $(1,0)$. Il est important de se rappeler que les étiquettes ne sont pas inscrites dans les noeuds et qu'aucun compteur ne calcule les coordonnées du point considéré, les coordonnées sont données implicitement par la structure de l'arbre. Ainsi, pour effectuer la translation $(1,0)$ à partir du sommet $(1,0)$, il faut :

1. Remonter au père de $(1,0)$ soit la racine $(0,0)$.
2. Suivre son lien de voisinage étiqueté par $(1,0)$. On retourne ainsi sur le noeud $(1,0)$.
3. Puisque le noeud $(1,0)$ ne possède pas de fils dont l'arête est étiquetée par $(0,0)$, on le crée. Il s'agit du noeud $(2,0)$.
4. On ajoute un lien de voisinage débutant à $(1,0)$, terminant à $(2,0)$ et d'étiquette $(1,0)$.

5. On marque le sommet $(2, 0)$ comme visité.

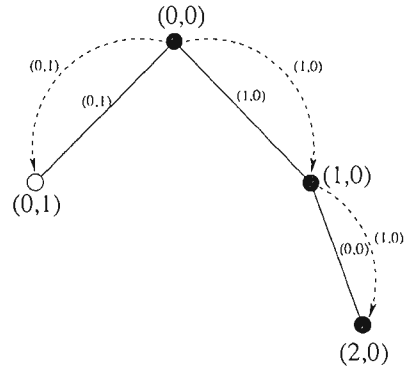


Figure 2.9 Le graphe $\mathcal{G}_{00}$.

On lit ensuite la lettre $w_3 = 1$ correspondant à la translation $(0, 1)$. On remonte au père du sommet $(2, 0)$. Puisque l'arête (de l'ensemble R) reliant le noeud $(2, 0)$ à son père $(1, 0)$ possède l'étiquette $(0, 0)$ on sait que la coordonnée en y du noeud $(2, 0)$ est un nombre pair et donc que son translaté de $(0, 1)$ possède le même père. Il suffit de créer ce nouveau noeud et d'ajouter le lien de voisinage entre $(2, 0)$ et $(2, 1)$.

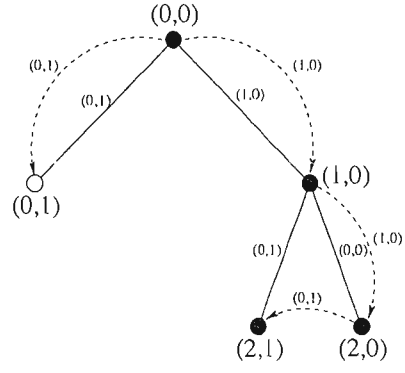


Figure 2.10 Le graphe $\mathcal{G}_{001}$.

Finalement, on lit la lettre $w_4 = 1$ correspondant à la translation $(0, 1)$. Cette fois-ci il faudra effectuer une récursion sur le père du sommet $(2, 1)$.

1. On remonte au père de $(2, 1)$: le noeud $(1, 0)$.

2. Comme il ne possède pas de lien de voisinage étiqueté par $(0,1)$ on monte à son père : la racine $(0,0)$.
3. L'arête reliant $(1,0)$ à $(0,0)$ possède l'étiquette $(1,0)$, le $(1,0)$-voisin du noeud $(1,0)$ possède donc le même père que lui.
4. On crée le noeud non-visité $(1,1)$ et on l'ajoute comme fils du noeud $(0,0)$ avec une arête d'étiquette $(1,1)$.
5. On ajoute le lien de voisinage débutant à $(1,0)$, terminant à $(1,1)$ et d'étiquette $(0,1)$.
6. On crée le noeud $(2,2)$ et on l'ajoute comme fils du noeud $(1,1)$ avec une arête étiquetée par $(0,0)$.
7. On ajoute le lien de voisinage débutant à $(2,1)$, terminant à $(2,2)$ et d'étiquette $(0,1)$.
8. On marque le noeud $(2,2)$ comme visité.

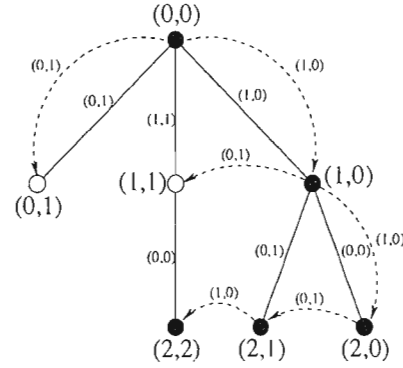


Figure 2.11 Le graphe $\mathcal{G}_{0011}$.

2.4 Algorithme

Étant donné un mot $w \in \{0,1,\overline{0},\overline{1}\}^*$ et une lettre $\alpha \in \{0,1,\overline{0},\overline{1}\}$, l'algorithme suivant construit le graphe $\mathcal{G}_w$ en lisant les lettres de w une à une.

Algorithme 3 (lireMot).

Entrée : $w \in \{0,1,\overline{0},\overline{1}\}^*$

0 : $\mathcal{G} \leftarrow \mathcal{G}_\varepsilon$; $\quad n \leftarrow$ racine de $\mathcal{G}$;

1 : **Pour** i **de** 1 **à** $|w|$ **faire**
2 : $\alpha \leftarrow w_i$;
3 : $n' \leftarrow$ trouverVoisin$(\mathcal{G}, n, \alpha)$;
4 : **Si** n' est visité **alors**
5 : Le mot w n'est pas auto-évitant.
6 : **fin si**
7 : **Marquer** n' *visité*;
8 : $n \leftarrow n'$;
9 : **fin pour**
10 : Le mot w est auto-évitant.

L'algorithme *trouverVoisin* se charge de trouver, et créer au besoin, le voisin d'un noeud.

Algorithme 4 (trouverVoisin).

Entrée : $\mathcal{G} = (N, R, V)$; $n \in N$; $\alpha \in \{0, 1, \overline{0}, \overline{1}\}$;

1 : **Si** n possède un lien de voisinage d'étiquette α **alors**
2 : $n' \leftarrow$ le α-voisin de n;
3 : **sinon**
4 : $p \leftarrow$ le père de n;
5 : **Si** le α-voisin de n est lui aussi fils de p **alors**
6 : $p' \leftarrow p$;
7 : **sinon**
8 : $p' \leftarrow$ trouverVoisin$(\mathcal{G}, p, \alpha)$;
9 : **fin si**
10 : $n' \leftarrow$ le fils de p' correspondant au α-voisin de n ; (Il faut possiblement le créer)
11 : Ajouter un lien de voisinage d'étiquette (α_x, α_y) de n à n'.
12 : **fin si**
13 : **retourner** n' ;

La récursion (ligne 8) détermine la complexité de cet algorithme puisque tous les autres tests et affectations s'effectuent en temps constant. On remarque également qu'après l'exécution de cet algorithme, le noeud n et son α-voisin n' sont forcément reliés par un lien de voisinage d'étiquette (α_x, α_y).

2.5 Analyse de la complexité

La clé de l'analyse de cet algorithme se trouve dans le fait qu'après un appel récursif (ligne 8 de l'Algorithme 4) sur un noeud p, il y aura forcément eu l'ajout d'un lien de voisinage entre un des fils de p et un autre noeud qui n'est pas un de ses fils. Comme un noeud possède au maximun quatre fils et que chacun d'eux possède au plus deux voisins qui n'ont pas le même père, le nombre d'appels récursifs effectués sur le noeud p est borné par 8. Il ne reste plus qu'à déterminer le nombre de noeuds construits lors de l'exécution de cet algorithme est linéaire en fonction de la longueur du mot w.

Premièrement, remarquons que pour chaque lettre lue, un noeud est marqué comme *visité*. Le nombre de noeuds visités est donc égal à la longeur du mot w. Afin d'apposer une borne sur le nombre de noeuds non-visités, définissons la fonction :

Définition 22. Étant donné un graphe $\mathcal{G}_w = (N, R, V)$ et un sous-ensemble $M \subset N$, on définit la fonction P :

$$P(M) = \{n \in N \mid n \text{ est le père d'au moins un des noeuds de } M\}.$$

Soit h la hauteur de $\mathcal{G}_w$, on a évidemment que $P^h(N)$ est l'ensemble contenant uniquement la racine de $\mathcal{G}_w$.

Lemme 1. *Soit $M \subset N$ une suite de cinq sommets voisins deux à deux alors $|P(M)| \leq 4$.*

Preuve. Comme l'illustre la Figure 2.3, l'ensemble des fils d'un noeud donné forme un carré de dimension 2×2. Si on considère cinq points voisins deux à deux, il y en a forcément au moins une paire qui ont le même père, d'où la borne $|P(M)| \leq 4$. ∎

On peut ainsi borner le nombre total de noeuds.

Lemme 2. *Étant donné un mot $w \in \{0, 1, \overline{0}, \overline{1}\}^n$ et son graphe $\mathcal{G}_w = (N, R, V)$, le nombre de noeuds dans N est linéaire en fonction de n.*

Preuve. Soit N_v l'ensemble des sommets visités de N et h la hauteur de $\mathcal{G}_w$. On a alors que

$$N = \bigcup_{0 \leq i \leq h} P^i(N_v),$$

et donc que

$$|N| \leq \sum_{0 \leq i \leq h} |P^i(N_v)|. \tag{2.1}$$

Par construction, l'ensemble N_v forme une chaîne de noeuds voisins deux à deux puisqu'ils correspondent aux points visités lors du parcours du chemin codé par w. Ainsi, on peut appliquer le lemme précédent en coupant ce chemin en blocs de longueur 5. On obtient alors :

$$|P(N_v)| \leq 4 \left\lceil \frac{|N_v|}{5} \right\rceil \leq \frac{4}{5}(|N_v| + 4). \tag{2.2}$$

Étant donné que deux noeuds voisins peuvent soit partager le même père soit avoir des pères qui sont eux-aussi voisins, l'argument précédent peut être appliqué aux ensembles

$$P(N_v), P^2(N_v), \ldots, P^h(N_v).$$

Ainsi, en appliquant l'équation 2.2 à l'équation 2.1, on peut borner la cardinalité de N :

$$\begin{aligned}
|N| &\leq \sum_{0 \leq i \leq h} \left|P^i(N_v)\right| \\
&\leq \sum_{0 \leq i \leq h} \left(\left(\frac{4}{5}\right)^i |N_v| + \sum_{0 \leq j \leq i} \left(\frac{4}{5}\right)^j 4 \right) \\
&\leq |N_v| \left(\frac{1}{1 - \frac{4}{5}} \right) + 4 \sum_{0 \leq i \leq h} \left(\frac{1}{1 - \frac{4}{5}} \right) \\
&\leq 5|N_v| + 20h
\end{aligned}$$

Puisque la hauteur de l'arbre correspond à la longueur de l'écriture binaire des coordonnées associées à un des noeuds, $h \in \mathcal{O}(\log n)$ et donc $|N| \in \mathcal{O}(n)$. ∎

Notons que pour simplifier la présentation, la constante de linéarité obtenue ici est largement supérieure à la réalité. Le but recherché n'étant pas d'effectuer une analyse fine mais uniquement de prouver la linéarité de l'algorithme.

Corollaire 1. *Étant donné un mot $w \in \{0, 1, \overline{0}, \overline{1}\}^n$, déterminer si le chemin codé par w passe deux fois par le même point est décidable en $\Theta(n)$. De plus, cette borne est optimale.*

Preuve. Notons tout d'abord que $\Omega(n)$ constitue une borne inférieure pour ce problème puisque chacune des lettres du mot w doit être lue au moins une fois.

Il ne reste plus qu'à montrer comment adapter la solution présentée précédemment afin d'être en mesure d'enlever la condition stipulant que si le chemin considéré débute en $(0,0)$, alors il demeure dans le premier quadrant. Pour ce faire, il suffit d'utiliser simultanément quatre graphes

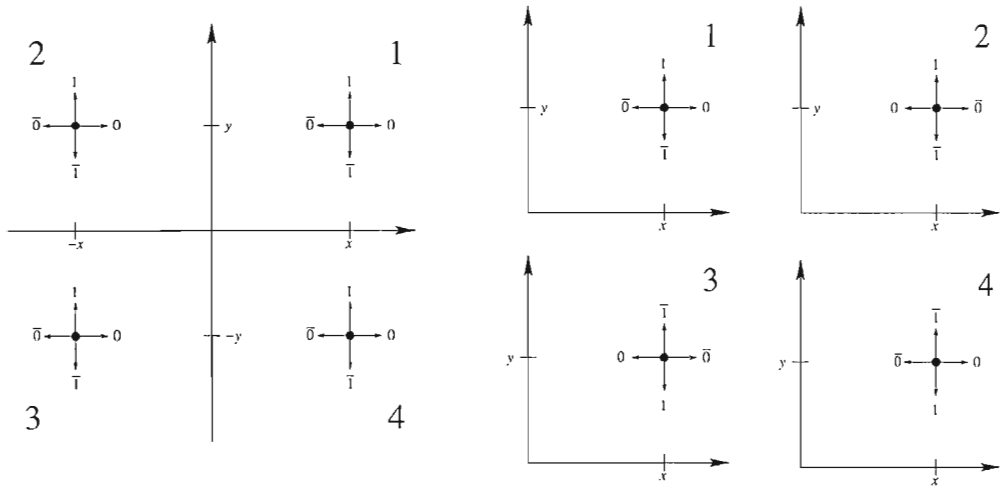


Figure 2.12 Permutations des translations élémentaires associées à chacune des lettres en fonction du quadrant considéré.

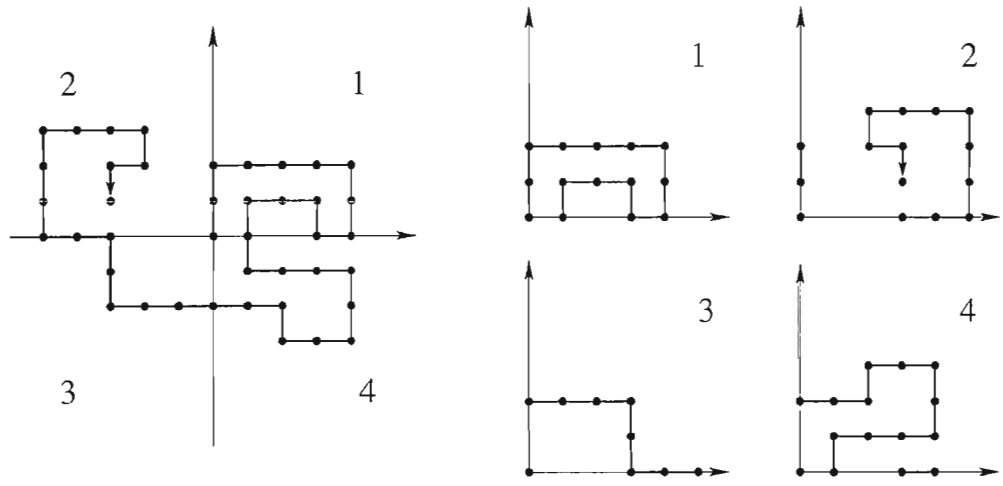


Figure 2.13 Un chemin dans le plan et sa représentation en quatre quadrants.

$\mathcal{G}_1, \mathcal{G}_2, \mathcal{G}_3, \mathcal{G}_4$, un pour chacun des quadrants du plan et on lie virtuellement les noeuds (*hard-code*) correspondant à des points présents dans plus d'un quadrant. Ainsi, l'origine $(0,0)$ sera

représentée dans les quatres graphes, les points de la forme $(x, 0)$ avec $x > 0$ seront représentés dans le graphe $\mathcal{G}_1$ et $\mathcal{G}_4$. De plus, comme le père d'un tel noeud est lui aussi de la forme $(x, 0)$, il est également présent dans les deux arbres. La cohérence de la structure finale est donc assurée.

■

2.6 Chemins auto-évitants et mots de contour

On s'intéresse maintenant à déterminer si un mot $w \in \{0, 1, \overline{0}, \overline{1}\}^*$ est un mot de contour. La Propriété 3 indique qu'il suffit de tester premièrement que $\Delta_{\mathcal{C}}(w) = -4$ et deuxièmement que le mot $w[1..n-1]$ code un chemin auto-évitant alors que le mot w code un chemin qui se termine au point où il a commencé.

Corollaire 2. *Étant donné un mot $w \in \{0, 1, \overline{0}, \overline{1}\}^n$, déterminer si w est un mot de contour est décidable en $\Theta(n)$.*

Preuve. L'algorithme présenté dans le présent chapitre permet de tester en temps linéaire si un chemin est auto-évitant. Il suffit donc de tester le mot $w[1..n-1]$ et de vérifier que le pas associé à la lettre w_n fait terminer le chemin là où il a commencé c'est-à-dire à l'origine $(0, 0)$. Tout au long de l'algorithme, le mot w est lu lettre par lettre, on peut donc en profiter pour compter le nombre de virages à droite et le nombre de virages à gauche afin de déterminer si $\Delta_{\mathcal{C}}(w) = -4$. ■

Si un mot w est tel que $w[1..n-1]$ code un chemin auto-évitant, $\overrightarrow{w} = \overrightarrow{0}$ mais que $\Delta_{\mathcal{C}}(w) = 4$, alors w code bien le bord d'un polyomino mais le parcours est effectué dans le sens anti-horaire. Le mot w n'est donc pas un mot de contour mais $\widehat{w}$ est en un. La fonction $\Delta_{\mathcal{C}}$ joue ainsi le rôle d'indicateur du sens de parcours.

Chapitre III

CONVEXITÉ DISCRÈTE

3.1 Introduction

La convexité est une notion géométrique fondamentale. En traitement d'image, on décompose souvent une image en un ensemble de formes géométriques convexes.

Les travaux présentés dans le cadre de ce chapitre découlent d'une observation de Christophe Reutenauer qui avait considéré les chemins qui approximent inférieurement une fonction concave. Il avait alors remarqué que si un mot w code un tel chemin, alors la factorisation de w en mots de Lyndon décroissants est composée uniquement de mots de Christoffel. Par exemple, la Figure 3.1 illustre l'approximation discrète inférieure de la fonction concave $f(x) = 2\sqrt{x}$ pour x allant de 0 à 10.

Le chemin ainsi formé est codé par le mot

$$w = 0110010100010010.$$

La factorisation de w en mots de Lyndon décroissants est :

$$w = (011) \cdot (00101) \cdot (0001001) \cdot (0),$$

et on a bien que 001, 00101, 0001001 et 0 sont des mots de Christoffel.

À partir de cette observation, on élabore une condition nécessaire et suffisante qui caractérise la convexité discrète d'un polyomino. Un algorithme linéaire, donc optimal, pour tester la convexité discrète en est ensuite déduit. Habituellement, de telles propriétés géométriques sont

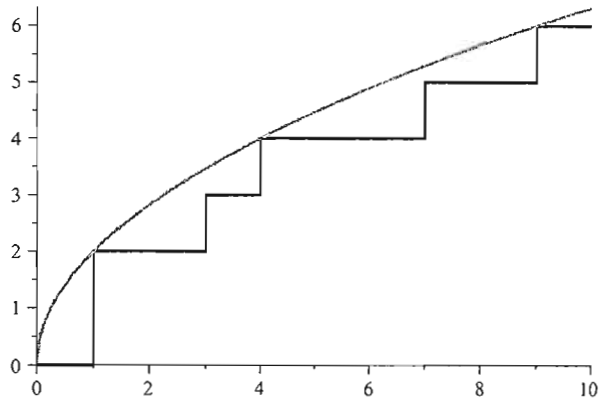


Figure 3.1 approximation discrète inférieure de la fonction concave $f(x) = 2\sqrt{x}$.

testées par des méthodes arithmétiques. L'algorithme de Debled-Rennesson et Al. (Debled-Rennesson, Rémy et Rouyer-Degli, 2003), basé sur la reconnaissance des segments maximaux de droites discrètes en est un bon exemple. Celui présenté ici se démarque en privilégiant l'utilisation d'arguments combinatoires.

Finalement, on s'intéresse au calcul de l'enveloppe convexe d'un polyomino. Le calcul de l'enveloppe convexe d'un ensemble de points discrets est étudié depuis longtemps et de nombreux algorithmes linéaires sont aujourd'hui connus. Le premier est celui de McCallum et Avis (McCallum et Avis, 1979). À la fin de ce chapitre, on présente un algorithme, lui aussi linéaire, pour calculer l'enveloppe convexe d'un polyomino dit hv-convexe. Cet algorithme tire son intérêt du fait qu'il est basé sur des résultats issus de combinatoire des mots datant des années 1950.

Avant d'aller plus loin, il faut préciser ce qu'on entend par *convexité discrète*. En géométrie euclidienne, la notion intuitive de convexité s'exprime de la manière suivante :

Définition 23. Une région R du plan euclidien est *convexe* si pour toute paire de points p_1, p_2 de cette région, le segment de droite qui relie p_1 à p_2 appartient à R.

Puisque l'intersection de deux ensembles convexes est convexe, il s'ensuit une définition toute aussi intuitive de l'enveloppe convexe :

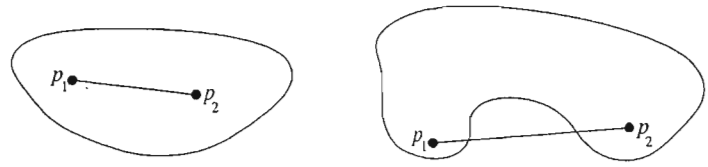


Figure 3.2 Deux régions de $\mathbb{R}^2$. Celle de gauche est convexe alors que celle de droite ne l'est pas.

Définition 24. L'*enveloppe convexe* d'une région R du plan euclidien est l'intersection de toutes les régions convexes qui contiennent R.

Ces définitions ne peuvent être appliquées directement à la géométrie discrète. Une première approche consiste à considérer les polyominos dans le plan euclidien $\mathbb{R}^2$ comme une union de carrés unitaires, appelés pixels. Cette façon de faire n'est guère satisfaisante car les seuls polyominos qui satisfont une telle définition de la convexité sont les rectangles orientés dans le même sens que les axes (voir Figure 3.3).

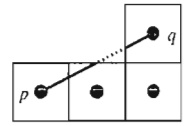


Figure 3.3 Mauvaise définition de la convexité discrète.

Il y a donc un travail non-trivial nécessaire afin de traduire la notion de convexité au monde discret. Une première définition de la convexité basée sur la discrétisation d'objets continus est due à Minsky et Papert (Minsky et Papert, 1969) et Sklansky (Sklansky, 1970).

Définition 25. Un sous-ensemble S du plan discret $\mathbb{Z}^2$ est *digitalement convexe* s'il correspond à la discrétisation de Gauss d'un sous-ensemble convexe R de $\mathbb{R}^2$. C'est-à-dire, $S = Conv(R) \cap \mathbb{Z}^2$.

Contrairement à la version euclidienne, cette définition de convexité discrète n'implique pas la connexité de l'ensemble considéré (voir Figure 3.4) ce qui est contre-intuitif. De plus, cette

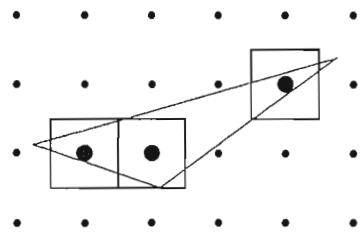


Figure 3.4 La discrétisation d'une figure convexe qui n'est pas 8-connexe.

définition de la convexité discrète ne permet pas l'élaboration directe d'un test de convexité efficace. Kim a proposé plusieurs caractérisations des ensembles discrets convexes et à montré (Kim, 1981; Kim, 1982) que sous l'hypothèse préalable de la 8-connexité elles s'avèrent toutes équivalentes à la Définition 25. Ce prérequis rend la notion de convexité discrète fidèle à l'intuition. Voici cinq des caractérisations proposées par Kim.

- *Propriété de lignes.* Ils n'existe pas trois points colinéaires $p_1, p_2, p_3 \in \mathbb{Z}^2$ tels que p_1 et p_3 appartiennent à S, p_2 est situé entre p_1 et p_3 mais $p_2 \notin S$.
- *Propriété de triangles.* Pour tout triplet de points $p_1, p_2, p_3 \in S$ la discrétisation du triangle euclidien formé par p_1, p_2 et p_3 est incluse dans S.

Bien que ces deux propriétés ne soient pas équivalentes en général (voir Figure 3.5), elles le sont dans le cas d'ensembles 8-connexes. Voir (Ronse, 1985) pour une preuve directe.

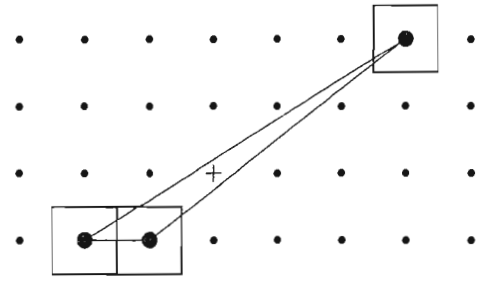


Figure 3.5 La propriété de lignes est satisfaite par cet ensemble mais pas celle de triangles.

Contrairement aux deux précédentes, les deux suivantes impliquent d'elles-mêmes la 8-connexité de l'ensemble considéré.

– *Propriété de cordes.* Pour toute paire de points $p_1, p_2 \in S$ et pour tout point $(x, y) \in \mathbb{R}^2$ tel que (x, y) est situé sur le segment reliant p_1 à p_2, il existe un point $(h, k) \in S$ tel que $\max\{|x-h|, |y-k|\} < 1$.

Une autre caractérisation, proposée par Kim et Rosenfeld (Kim et Rosenfeld, 1982), se démarque des autres par le fait qu'elle n'utilise que des objets discrets évitant ainsi toute référence au monde continu $\mathbb{R}^2$.

– *Propriété de droite digitale.* Pour toute paire de points $p_1, p_2 \in S$ il existe un segment de droite digitale reliant p_1 à p_2 qui est entièrement inclus dans S.

Finalement, cette dernière propriété servira de point de départ pour l'élaboration de notre test de convexité présenté en Section 3.3.

– *Propriété d'enveloppe.* L'enveloppe convexe euclidienne de S ne contient aucun point à coordonnées entières à l'extérieur de S.

La Figure 3.6 montre un exemple d'ensemble 8-connexe correspondant exactement à la discrétisation de son enveloppe convexe euclidienne.

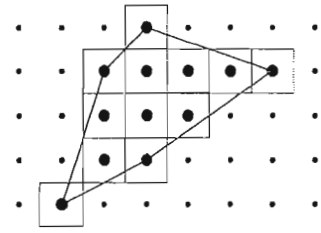


Figure 3.6 Un ensemble 8-connexe convexe et son enveloppe convexe euclidienne.

Étant donné que nous nous intéressons ici aux polyominos, qui sont par définition 4-connexes, toutes ces définitions de convexité discrète s'avèrent équivalentes. Voir (Eckhardt, 2001) pour une revue complète des liens et équivalences entre les différentes définitions de la convexité discrète.

Le bord d'un polyomino se décompose naturellement en quatre parties déterminées par ses points extrémaux. On identifie, parmi les points situés sur le bord d'un polyomino, les points N, S, E, O de la manière suivante : (comme l'illustre la Figure 3.7)

- N le point le plus à gauche de la ligne la plus haute.

- S le point le plus à droite de la ligne la plus basse.
- E le point le plus haut de la colone la plus à droite.
- O le point le plus bas de la colone la plus à gauche.

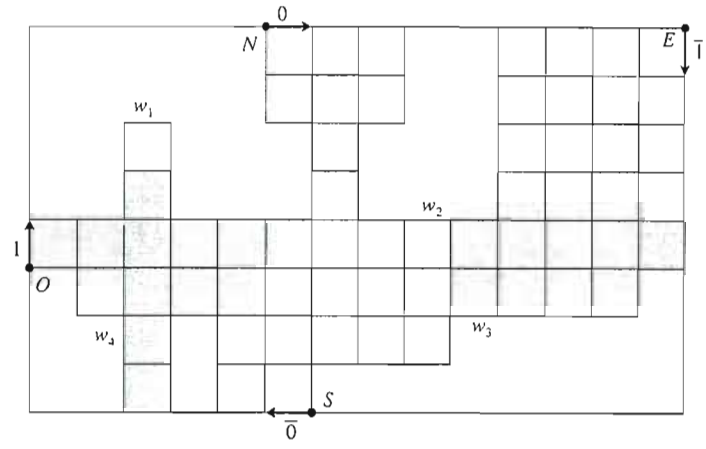


Figure 3.7 Un polyomino et sa décomposition standard $w \equiv w_1 \cdot w_2 \cdot w_3 \cdot w_4$.

Ces points définissent une factorisation du mot codant le bord du polyomino.

Définition 26. Soit $w \in \{0, 1, \overline{0}, \overline{1}\}^*$ un mot codant le bord d'un polyomino $\mathbf{P}$. On appelle *décomposition standard* l'unique factorisation $w \equiv w_1 \cdot w_2 \cdot w_3 \cdot w_4$ telle que w_1 code le bord de $\mathbf{P}$ du point O au point N, w_2 du point N au point E, w_3 du point E au point S et w_4 du point S au point O.

On remarque que les quatre mots w_1, w_2, w_3, w_4 sont forcément non-vides et que w_1 commence et termine par la lettre 1, w_2 par 0, w_3 par $\overline{1}$ et w_4 par $\overline{0}$. De plus, cette décomposition se calcule en temps linéaire en effectuant une seule passe sur le mot w.

3.2 hv-convexité

Une notion préalable à la convexité, classique dans l'étude de la tomographie discrète et en combinatoire énumérative, est la *hv-convexité*. Contrairement à la convexité classique, cette notion passe trivialement du monde continu au monde discret et vice-versa.

Définition 27. Un sous-ensemble $R \subset \mathbb{R}^2$ est *horizontalement convexe*, en abrégé *h-convexe*, si pour toute paire de points $(x_1, y), (x_2, y) \in R$, on a :

$$x_1 < x < x_2 \implies (x, y) \in R.$$

On définit similairement la *v-convextité* :

Définition 28. Un sous-ensemble $R \subset \mathbb{R}^2$ est verticalement convexe, en abrégé *v-convexe*, si pour toute paire de points $(x, y_1), (x, y_2) \in R$, on a :

$$y_1 < y < y_2 \implies (x, y) \in R.$$

Finalement, une région est dite *hv-convexe* si elle est à la fois h-convexe et v-convexe. Dans le cas discret, on dira donc qu'une figure est h-convexe (resp. v-convexe) si tous les points appartenant à une même ligne (resp. colonne) sont consécutifs dans cette ligne (resp. colonne).

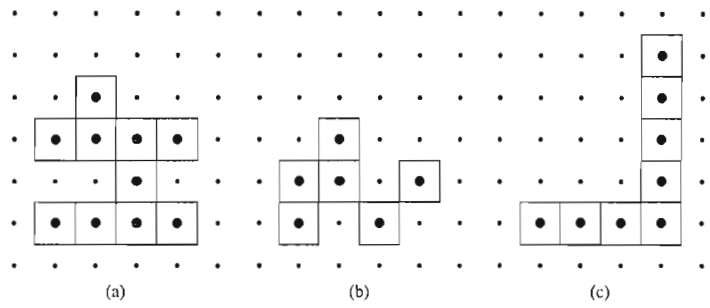


Figure 3.8 (a) Une figure h-convexe. (b) Une figure v-convexe. (c) Une figure hv-convexe.

Évidemment, la hv-convexité est une condition nécessaire à la convexité discrète mais pas suffisante (voir Figure 3.8 (c)). Cette propriété impose une structure particulière au mot de contour d'une figure hv-convexe. Un mot $w \in \{0, 1, \overline{0}, \overline{1}\}^*$ est dit *hv-convexe* s'il code le bord d'une figure hv-convexe.

Proposition 4. *Soit $w \in \{0, 1, \overline{0}, \overline{1}\}^*$ un mot de contour. w est hv-convexe si et seulement si le*

mot w admet une factorisation $w \equiv w_1w_2w_3w_4$ telle que

$$\begin{aligned} w_1 &\in {}_1\{0,1\}_1 \subset \{0,1\}^*, \\ w_2 &\in {}_0\{0,\overline{1}\}_0 \subset \{0,\overline{1}\}^*, \\ w_3 &\in {}_{\overline{1}}\{\overline{0},\overline{1}\}_{\overline{1}} \subset \{\overline{0},\overline{1}\}^*, \\ w_4 &\in {}_{\overline{0}}\{\overline{0},1\}_{\overline{0}} \subset \{\overline{0},1\}^*. \end{aligned}$$

De plus, dans un tel cas $w_1w_2w_3w_4$ forme la décomposition standard de w.

Preuve. ($\Leftarrow$) Soit $w_1 \in {}_1\{0,1\}_1$, $w_2 \in {}_0\{0,\overline{1}\}_0$, $w_3 \in {}_{\overline{1}}\{\overline{0},\overline{1}\}_{\overline{1}}$ et $w_4 \in {}_{\overline{0}}\{\overline{0},1\}_{\overline{0}}$ tels que leur concaténation $w = w_1w_2w_3w_4$ code le bord du polyomino $\mathbf{P}$. Soient O, N, S, E les points sur le bord de $\mathbf{P}$ tels que $w_1 = \mathbf{P}[O,N]$, $w_2 = \mathbf{P}[N,E]$, $w_3 = \mathbf{P}[E,S]$, $w_4 = \mathbf{P}[S,O]$ (voir Figure 3.9). Puisque toutes les occurrences de la lettre 0 sont situées dans le préfixe w_1w_2, et que toutes les occurrences de $\overline{0}$ sont dans le suffixe w_3w_4, le point O est situé dans la colonne la plus à gauche de $\mathbf{P}$ et E dans la plus à droite. Si $\mathbf{P}$ n'est pas h-convexe, c'est que soit

1. Le mot w_1w_2 contient un facteur de la forme $\overline{1}0^k1$, avec $k \geq 1$.
2. Le mot w_3w_4 contient un facteur de la forme $1\overline{0}^{k'}\overline{1}$, avec $k' \geq 1$.

Étant donné les alphabets respectifs des mots w_1, w_2, w_3 et w_4, ces deux cas sont impossibles, donc $\mathbf{P}$ est h-convexe. On montre de manière semblable que $\mathbf{P}$ est v-convexe, ce qui permet de conclure que w est hv-convexe. On remarque également que le point O correspond forcément au point le plus bas de la colonne la plus à gauche, que N correspond au point le plus à gauche de la ligne la plus haute, que E correspond au point le plus haut de la colonne la plus à droite et que S correspond au point le plus à droite de la ligne la plus basse. On conclut donc que $w_1w_2w_3w_4$ est la décomposition standard de w.

($\Rightarrow$) Soit $\mathbf{P}$ un polyomino hv-convexe. Soient O, N, E, S quatre points sur le bord de $\mathbf{P}$ tels que O est le plus bas de la colonne la plus à gauche, N est le plus à gauche de la ligne la plus haute, E est le point le plus haut de la colonne la plus à droite et S est le plus à droite de la ligne la plus basse (voir Figure 3.9). On pose $w_1 = \mathbf{P}[O,N]$, $w_2 = \mathbf{P}[N,E]$, $w_3 = \mathbf{P}[E,S]$ et $w_4 = \mathbf{P}[S,O]$. Par cette construction, w_1 débute et termine par la lettre 1, w_2 par la lettre 0, w_3 par la lettre $\overline{1}$ et w_4 par la lettre $\overline{0}$.

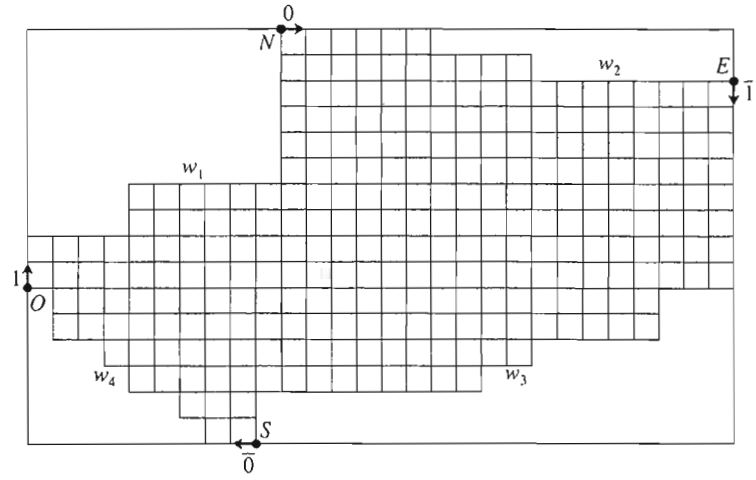


Figure 3.9 Une figure hv-convexe et la factorisation de son mot de contour $w = w_1w_2w_3w_4$.

Le facteur w_1w_2 de w ne contient aucune occurrence de la lettre $\overline{0}$. Supposons que ce n'est pas le cas et considérons une occurrence de la lettre $\overline{0}$ dans w_1w_2. La colonne traversée, lors du pas vers la gauche correspondant à la lettre $\overline{0}$, avait déjà été traversée lors d'un pas vers la droite puisque le chemin codé par w_1w_2 débute au point le plus à gauche de $\mathbf{P}$. Cette colonne devra également être retraversée lors d'un autre pas vers la droite car w_1w_2 termine au point le plus à droite de $\mathbf{P}$. Cependant, lors du parcours du bord d'une figure hv-convexe, chaque colonne est traversée exactement deux fois. Il y a donc une contradiction. On montre de la même manière que le facteur w_2w_3 de w ne contient aucune occurrence de la lettre 1, que w_3w_4 ne contient aucune occurrence de la lettre 0 et finalement que w_4w_1 ne contient aucune occurrence de la lettre $\overline{1}$.

On conclut que le mot $w_1 \in {}_1\{0,1\}_1$, $w_2 \in {}_0\{0,\overline{1}\}_0$, $w_3 \in {}_{\overline{1}}\{\overline{0},\overline{1}\}_{\overline{1}}$ et $w_4 \in {}_{\overline{0}}\{\overline{0},1\}_{\overline{0}}$. ■

Ainsi, déterminer si un mot w est hv-convexe revient dans un premier temps à calculer sa décomposition standard, puis à vérifier que chacun des mots w_1, w_2, w_3, w_4 est bien sur l'alphabet à deux lettres approprié.

3.3 Détection de la convexité discrète

Étant donné un mot $w \in \{0, 1, \overline{0}, \overline{1}\}^*$, on dira qu'il est convexe s'il code le bord d'une figure digitalement convexe. Soit $w_1 w_2 w_3 w_4$ la décomposition standard d'un mot convexe, on dira de w_1 qu'il est *NO-convexe* car il code la partie *nord-ouest* d'une figure convexe. On dira similairement de w_2 qu'il est *NE-convexe*, de w_3 qu'il est *SE-convexe* et de w_4 qu'il est *SO-convexe*.

Supposons maintenant que le mot w est hv-convexe. Afin de déterminer si le mot w est convexe, on commence par tester si le mot w_1 est NO-convexe, c'est-à-dire, s'il code le côté nord-ouest d'une figure digitalement convexe. Nous verrons ensuite comment réutiliser ce test afin de décider de la convexité du mot w. Pour décider de la NO-convexité d'un mot sur l'alphabet $\{0, 1\}$, il faut détecter s'il existe des points à coordonnées entières entre le chemin codé par ce mot et la partie supérieure de l'enveloppe convexe des points de ce chemin. Le théorème suivant provient de (Brlek et al., 2008).

Théorème 5. *Un mot $v \in \{1\} \cdot \{0, 1\}^*$ est NO-convexe si et seulement si son unique factorisation en mots de Lyndon décroissants $w = l_1^{n_1} l_2^{n_2} \cdots l_k^{n_k}$ est composée uniquement de mots de Christoffel primitifs.*

Afin de démontrer ce résultat, considérons le lemme suivant :

Lemme 3. *Soit $v \in \{0, 1\}^*$ un mot codant un chemin NO-convexe et soit e une des arêtes de son enveloppe convexe. Soit u le facteur de v qui correspond à la partie du chemin délimitée par e ; alors u est un mot de Christoffel.*

Ceci est une conséquence directe de la Définition 10 qui assure que le chemin associé à un mot de Christoffel reste toujours le plus près possible de la droite reliant son point de départ à son point d'arrivée sans jamais la traverser. Nous pouvons maintenant passer à la preuve du Théorème 5.

Preuve. ($\Rightarrow$) Soit v un mot codant un chemin NO-convexe et soit $(e_1, e_2, \ldots, e_k)$ la suite d'arêtes qui forment le bord de son enveloppe convexe. Pour chaque i de 1 à k, soit u_i le

facteur de v déterminé par l'arête e_i et soit l_i l'unique mot primitif tel que $u_i = l^{n_i}$. On a alors que

$$v = u_1 u_2 \cdots u_k = l_1^{n_1} l_2^{n_2} \cdots l_k^{n_k}.$$

Par la définition de la NO-convexité et le Lemme 3 on a que les u_i tout comme les l_i sont des mots de Christoffel. Par la Propriété 1 (i) (Section 1.4.1), les l_i sont également des mots de Lyndon. Puisque $(e_1, e_2, \ldots, e_k)$ forme l'enveloppe convexe supérieure de v, il s'ensuit que la pente p_i de l'arête e_i est plus grande que la pente p_{i+1} de l'arête e_{i+1}. Ceci implique l'inégalité suivante :

$$\rho(l_i) = \rho(u_i) = p_i > p_{i+1} = \rho(u_{i+1}) = \rho(l_{i+1}). \tag{3.1}$$

Par la Propriété 1 (ii) (Section 1.4.1) on conclut que $l_i > l_{i+1}$. Ainsi $l_1^{n_1} l_2^{n_2} \cdots l_k^{n_k}$ est l'unique factorisation en mots de Lyndon décroissants de v et chacun de ces facteurs est un mot de Christoffel.

($\Leftarrow$) Soit $v \in \{0,1\}^*$ un mot tel que sa factorisation en mots de Lyndon décroissants $l_1^{n_1} l_2^{n_2} \cdots l_k^{n_k}$ est uniquement composée de mots de Christoffel primitifs. Pour chaque i de 1 à k, soit e_i le segment de droite reliant le point de départ du chemin codé par $l_i^{n_i}$ à son point final. Il reste à voir que $(e_1, e_2, \ldots, e_k)$ forme l'enveloppe convexe supérieure de v. Puisque $l_i^{n_i}$ est un mot de Christoffel, la Définition 10 (Section 1.4.1) assure qu'il n'existe aucun point à coordonnées entières entre le chemin codé par $l_i^{n_i}$ et le segment de droite e_i et, en plus, que jamais le chemin ne traverse le segment. Par le théorème de factorisation unique en mots de Lyndon décroissants (Théorème 2), on a que $l_i > l_{i+1}$. En utilisant encore une fois l'Équation 3.1, on conclut que la pente de e_i est strictement plus grande que celle de e_{i+1} .

Ainsi, la suite de segments $(e_1, e_2, \ldots, e_k)$ forme l'enveloppe convexe supérieure du chemin codé par v . Comme il n'y a aucun point entre le chemin et son enveloppe convexe, on conclut que v est NO-convexe. ∎

Ceci permet l'élaboration directe d'un algorithme vérifiant si un mot donné code le bord d'une région digitalement convexe.

3.3.1 Algorithme optimal

Traditionnellement, les algorithmes de géométrie discrète s'appuient sur des outils arithmétiques. L'algorithme de Debled-Renneson et al. (Debled-Rennesson, Rémy et Rouyer-Degli, 2003) en est un bon exemple. Cet algorithme construit, à l'aide d'outils arithmétiques, une série de segments maximaux de droites digitales (Reveillès, 1991) et parvient ainsi à déterminer si une figure discrète est convexe ou non. Notons que cette méthode est linéaire en fonction du périmètre de la figure analysée. L'algorithme qui suit utilise des résultats issus de la combinatoire des mots (voir Chapitre 1) afin de résoudre ce problème en temps linéaire également mais avec une constante plus petite.

Problème 1. *Étant donné un mot $w \in \{0,1,\overline{0},\overline{1}\}^*$, est-ce que le chemin codé par w forme le bord d'une figure discrète convexe ?*

La première étape consiste à nous assurer que w code bien le bord d'un polyomino hv-convexe. Ceci s'effectue en trois étapes :

1. Vérifier que w code bien le bord d'un polyomino à l'aide de l'algorithme présenté en Section 2.
2. Calculer w_1, w_2, w_3, w_4 la décomposition standard de w.
3. Vérifier que le mot $w_1 \cdot w_2 \cdot w_3 \cdot w_4$ code une figure $hv - convexe$.

Chacune de ces étapes s'effectue en temps linéaire (voir les sections respectives) n'affectant pas la complexité totale de l'algorithme. L'algorithme suivant teste la NO-convexité d'un mot v correspondant au facteur w_1 de la décomposition standard du mot de contour. Il s'agit d'une application directe du Théorème 5. Chaque mot de Lyndon de la factorisation unique de v en mots de Lyndon décroissants est testée afin de vérifier qu'il s'agit bien d'un mot de Christoffel primitif.

Algorithme 5 (estNOConvexe).

Entrée : $v \in \{0,1\}^n$

1 : $convexe \leftarrow vrai$;
2 : $index \leftarrow 1$;

3 : **tant que** $convexe$ **et** $index \leq n$ **faire**
4 : $(l_1, n_1) \leftarrow$ **PremierFacteurDeLyndon**$(u_{index} u_{index+1} \cdots u_n)$;
5 : $convexe \leftarrow$ **estChristoffelPrimitif**(l_1);
6 : $index \leftarrow index + n_1|l_1|$;
7 : **fin tant que**
8 : **retourne**$(convexe)$:

La factorisation en mots de Lyndon décroissants d'un mot v étant $v = l_1^{n_1} l_2^{n_2} \cdots l_k^{n_k}$, on a forcément que $|v| > \sum_i |l_i|$. L'algorithme précédent est donc linéaire en fonction de la longueur du mot v.

Ensuite, pour tester que le facteur w_2 est NE-convexe il suffit d'utiliser le morphisme σ introduit à la Section 1.5.4. En effectuant une rotation de $\frac{\pi}{2}$, la partie nord-est devient alors la partie nord-ouest et on peut alors utiliser l'algorithme **estNOConvexe** pour en tester la convexité. La proposition suivante généralise ceci.

Proposition 5. *Un mot hv-convexe w dont la décomposition standard est $w_1 w_2 w_3 w_4$ est convexe ssi $\sigma^{i-1}(w_i)$ est NO-convexe, pour tout i.*

Ceci constitue donc un test de convexité linéaire en fonction de la longueur du mot codant le bord d'une figure discrète. Ce test de convexité discrète constitue une solution algorithmique particulièrement efficace car elle est entièrement basée sur la manipulation d'objets discrets. C'est là la force de l'approche combinatoire de la géométrie discrète.

3.3.2 Raffinement de l'algorithme

L'algorithme présenté précédemment, bien qu'optimal, requiert plusieurs passes sur le mot, en particulier lors du prétraitement. On peut éviter ce prétraitement en insérant ces étapes au coeur de l'algorithme.

La premère modification consiste à modifier l'algorithme **PremierFacteurDeLyndon** de façon à passer en paramètre l'alphabet ordonné sur lequel factoriser le mot. On notera $[a_1, a_2, \ldots, a_n]$ l'alphabet $\{a_1, a_2, \ldots, a_n\}$ muni de l'ordre total : $a_i < a_j \iff i < j$. Le Tableau 3.1

Facteur	Alphabet	Ordre	Alphabet Ordonné
w_1	$\{0,1\}$	$0<1$	$[0,1]$
w_2	$\{0,\overline{1}\}$	$\overline{1}<0$	$[\overline{1},0]$
w_3	$\{\overline{0},\overline{1}\}$	$\overline{0}<\overline{1}$	$[\overline{0},\overline{1}]$
w_4	$\{\overline{0},1\}$	$1<\overline{0}$	$[1,\overline{0}]$

Tableau 3.1 Alphabets ordonnés en fonction du facteur considéré.

montre les alphabets ordonnés sur lesquels doivent être factorisés chacun des facteurs de la décomposition standard du mot w.

Une deuxième modification portée à cet algorithme consiste à compter le nombre d'occurrences de chacune des lettres et à retourner cette information avec la paire (l_1, n_1) calculée. Appelons ce nouvel algorithme **PremierFacteurDeLyndon$^+$**.

On modifie également l'algorithme **estChristoffelPrimitif** de façon à passer en paramètre l'alphabet utilisé ainsi que le nombre d'occurrences de chacune des lettres. Appelons ce nouvel algorithme **estChristoffelPrimitif$^+$**.

Afin de minimiser le prétraitement, au lieu de calculer la décomposition standard du mot w puis de tester la hv-conexité, on va plutôt supposer que w est hv-convexe et donc que les changements d'alphabets à deux lettres correspondent exactement à la décomposition standard w_1, w_2, w_3, w_4. C'est-à-dire que $w_1 \in \{0,1\}^*$, $w_2 \in \{0,\overline{1}\}^*$, $w_3 \in \{\overline{0},\overline{1}\}^*$, $w_4 \in \{\overline{0},1\}^*$.

L'algorithme optimisé suivra donc les étapes suivantes :

1. Trouver le début d'un des facteurs w_i.
   1.1 Appelons w_{i-1} le facteur parmi w_1, w_2, w_3 et w_4 dans lequel est situé la première lettre de w.
   1.2 Puisque le mot w contient au moins une occurrence de chacune des lettres de son alphabet, il existe $a, b \in \{0,1,\overline{0},\overline{1}\}, a \neq b$, telles que $w = uv$ avec $u \in \{a,b\}^* \cdot \{ab^k\}$, $k \geq 1$ et Première$(v) = c \notin \{a,b\}$. On pose $w' = vu$.
   1.3 Si w est hv-convexe alors le facteur w_i est un préfixe de $b^k \cdot w'$ sur l'alphabet ordonné [c,b].

2. Calculer le premier facteur de la factorisation en mots de Lyndon décroissants de w' sur l'alphabet ordonné $[c, b]$.
   2.1 Le préfixe b^k de w_i n'a pas besoin d'être considéré puisque b est le plus grand (lexicographiquement) mot de Lyndon sur l'alphabet ordonné [c,b] et que b^k est bien un mot de Christoffel.
   2.2 On utilise la convention que pour toute lettre $a \notin \{b, c\}$ on a $a < b$ et $a < c$ de manière à ce que l'algorithme **PremierFacteurDeLyndon**$^+$ s'arrête dès qu'une telle lettre est rencontrée. On sait alors qu'on doit traiter le facteur w_{i+1} sur l'alphabet ordonné $[a, c]$.
   2.3 Vérifier que la succession des alphabets ordonnés à deux lettres est bien conforme au Tableau 3.3.2.
3. Vérifier que le mot de Lyndon obtenu est bien un mot de Christoffel.
4. Répéter les étapes 2 et 3 jusqu'à ce qu'on ait traité les quatre facteurs w_1, w_2, w_3, w_4.
5. Puisqu'on a parcouru les mots w_1, w_2, w_3 et w_4, on a pu compter le nombre d'occurrences de chacune des lettres. On vérifie finalement que :
   5.1 Si $|w_1| + |w_2| + |w_3| + |w_4| < |w|$ alors w n'est pas hv-convexe.
   5.2 Si $|w|_0 \neq |w|_{\overline{0}}$ ou $|w|_1 \neq |w|_{\overline{1}}$ alors w ne code pas le bord d'un polyomino.

Cette méthode a pour avantage qu'elle ne nécessite aucun prétraitement sur le mot testé. De plus, presque tout le traitement s'effectue en une seule passe sur le mot. En effet, si on considère un facteur $l_i^{n_i}$ obtenu en calculant la factorisation en mots de Lyndon décroissants d'un des w_i, une seule des n_i occurrences de l_i sera utilisée pour tester qu'il s'agit bien d'un mot de Christoffel et sera alors parcourue une deuxième fois. Tout le reste du facteur n'est traité qu'une seule fois.

Ainsi, lors de l'analyse du bord d'un polyomino, il n'est nécessaire de stocker en mémoire que le mot de Lyndon considéré lors de la factorisation du mot w_i. Les facteurs précédents n'ont pas besoin d'être conservés en mémoire.

3.3.3 Enveloppe convexe

Étant basée uniquement sur la notion d'ordre lexicographique, la décomposition en mots de Lyndon décroissants d'un mot ne semble pas, à première vue, donner lieu à une interprétation géométrique. On voit bien cependant qu'elle s'avère extrêmement significative dans le cas des figures convexes puisqu'elle en détermine les points appartenant à l'enveloppe convexe. La preuve du Théorème 5 implique le corollaire suivant.

Corollaire 3. *L'enveloppe convexe d'un polyomino est un polygone dont les sommets correspondent aux points de factorisation en mots de Lyndon.*

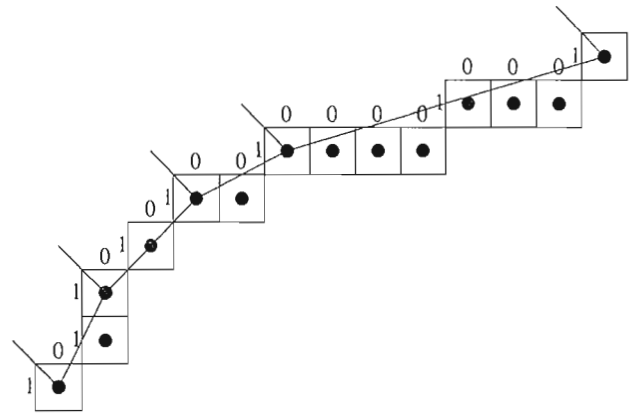


Figure 3.10 Un mot NO-convexe et sa factorisation en mots de Lyndon décroissants.

À titre d'exemple, considérons le chemin codé par $v = 10110101001000010001$. La décomposition en mots de Lyndon décroissants est :

$$v = (1)^1 \cdot (011)^1 \cdot (01)^2 \cdot (001)^1 \cdot (000010001)^1.$$

On remarque que les facteurs $1, 011, 01, 001, 000010001$ sont tous des mots de Christoffel primitifs. La Figure 3.10 illustre clairement le lien entre cette factorisation et les points correspondant aux sommets du polygone formant l'enveloppe convexe.

Il est important de noter que ce lien entre la factorisation en mots de Lyndon décroissants d'un mot et l'enveloppe convexe de son chemin associé n'est valide que dans le cas des figures convexes. Lorsqu'une figure n'est pas convexe, il n'y a pas de correspondance entre les deux.

Par exemple, considérons le mot $v = 10001110100100100011$. La décomposition en mots de Lyndon décroissants est :

$$v = (1)^1 \cdot (00011101001001)^1 \cdot (00011)^1.$$

Cela ne correspond pas à l'enveloppe convexe supérieure de ce chemin tel que l'illustre la Figure 3.11.

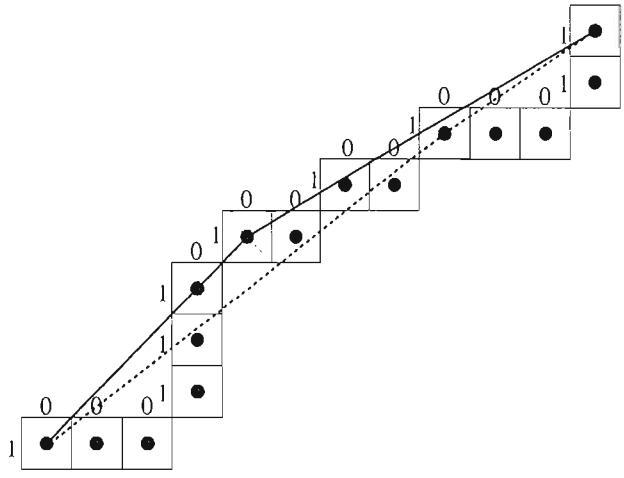


Figure 3.11 La décomposition en mots de Lyndon décroissants et l'enveloppe convexe.

L'enveloppe convexe euclidienne d'un ensemble discret $S \in \mathbb{Z}^2$ est forcément un polygone dont les sommets sont des points de S. Dans le cas où S est digitalement convexe, il est entièrement déterminé par ces points.

Étant donné un mot NO-convexe $v \in \{0,1\}^*$ dont la décomposition en mots de Christoffel décroissants est $v = l_1^{n_1} l_2^{n_2} \cdots l_k^{n_k}$, on pose (x_0, y_0) le point de départ du chemin codé par v. Pour chaque entier $i \in \{1, 2, \ldots, k\}$ on pose (x_i, y_i) comme étant le point où se termine le chemin codé par le facteur $l_i^{n_i}$ et les quantités :

$$\begin{aligned} \Delta x_i &= x_i - x_{i-1}, \\ \Delta y_i &= y_i - y_{i-1}. \end{aligned}$$

On définit ensuite les mots

$$u_i = \left(C_{\frac{\Delta x_i + \Delta y_i}{\mu_i}, \frac{\Delta y_i}{\mu_i}} \right)^{\mu_i},$$

où $\mu_i = \text{pgcd}(\Delta x_i, \Delta y_i)$ et $C_{n,k}$ est le mot de Christoffel primitif de longueur n avec k occurrences de la lettre 1 (voir Définition 9 de la Section 1.4).

À partir de la preuve du Théorème 5 et du Corollaire 3 on déduit directement que

$$v = u_1 u_2 \dots u_k.$$

Ceci fournit donc un algorithme optimal pour reconstruire une figure discrète digitalement convexe à partir des points correspondant aux sommets de son enveloppe convexe euclidienne. Il suffit de calculer les mots de Christoffel correspondants en utilisant une version légèrement modifiée de l'Algorithme 2 présenté à la Section 1.4.2.

3.4 Calcul de l'enveloppe convexe

Étant donné un mot w sur l'alphabet $\{0,1\}$, l'enveloppe convexe supérieure du chemin codé par w est donné par ce qu'on appelle sa *factorisation de Spitzer* (voir (Lothaire, 1997), page 95). On définit d'abord deux familles d'ensembles. Étant donné un morphisme $\Phi : \{0,1\} \to \mathbb{R}$, où $\mathbb{R}$ est considéré comme un monoïde additif, pour tout $r \in \mathbb{R}$ on pose

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

Le résultat suivant est attribué à Spitzer (Spitzer, 1956) qui l'a établi de manière plus générale dans un tout autre contexte.

Théorème 6 (Spitzer, 1956). *Tout mot $w \in \{0,1\}^*$ admet une unique factorisation de la forme*

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

où $b_i \in B_{r_i}$ pour $i = 1, 2, \dots, k$ et si $i \neq k$ alors $r_i \geq r_{i+1}$.

En prenant le morphisme défini par $\Phi(0) = -1$ et $\Phi(1) = +1$, les séparations entre les facteurs $b_i \cdot b_{i+1}$ tels que $r_i > r_{i+1}$ correspondent exactement aux points de l'enveloppe convexe du chemin codé par w tel que l'illustre la figure suivante :

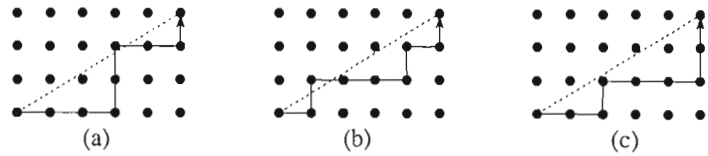
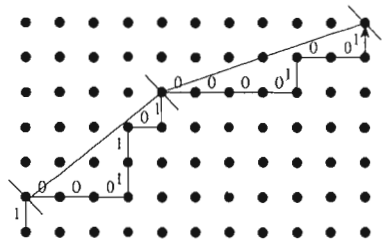


Figure 3.12 Étant donné le morphisme $\Phi(0) = -1$ et $\Phi(1) = +1$, trois chemins codés par des mots appartenant à l'ensemble $C_{-1/4}$. Le mot en (c) appartient également à l'ensemble $B_{-1/4}$ alors qu'en (a) le préfixe $00011 \in C_{-1/5}$ et en (b) le préfixe $01 \in C_0$.



$w = 1000011100010001,$

$w = 1 \cdot 0000111 \cdot 00010001,$

$1 \in B_1,$

$0000111 \in B_{-1/7},$

$00010001 \in B_{-1/2}.$

On remarque que la pente d'un mot $w \in C_r$ est $\rho(w) = (1+r)/(1-r)$, de sorte que si on a $u \in B_r$ et $v \in B_s$ alors

$$r < s \iff \rho(u) < \rho(v).$$

Tout comme dans le cas des factorisations de Viennot, la famille d'ensembles $(B_r)_{r \geq 0}$ admet la propriété suivante :

Propriété 4. *Soit $u \in B_r$ et $v \in B_s$ alors $r < s$ implique $uv \in B_t$ pour un certain $r < t < s$.*

En fait, on peut redéfinir la factorisation de Spitzer de manière à obtenir une factorisation de Viennot (Viennot, 1978). Cette seule propriété suffit par contre à élaborer un algorithme linéaire en fonction de la longueur d'un mot afin d'en calculer la factorisation de Spitzer et donc l'enveloppe convexe supérieure du chemin qu'il code. Ehrenfeucht, Haemer et Haussel on utilisé cette approche dans leur article (Ehrenfeucht, Haemer et Haussler, 1987) afin d'élaborer un algorithme optimal pour calculer l'enveloppe convexe d'une liste de points $T = (x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$, de coordonnées x distinctes et triée en fonction de cette coordonnée.

Étant donné un mot w, puisqu'on s'intéresse au nombre d'occurrences de chacune des lettres dans ses facteurs plutôt qu'aux facteurs eux-mêmes, on commence par calculer la liste $L = \big((a_1, b_1), (a_2, b_2), \ldots, (a_n, b_n)\big)$ définie par

$$a_i = \begin{cases} (1,0) \text{ si } w_i = 0, \\ (0,1) \text{ si } w_i = 1. \end{cases}$$

Ensuite, tant qu'il existe une valeur i telle que $a_i b_{i+1} \leq a_{i+1} b_i$, on remplace les couples (a_i, b_i) et (a_{i+1}, b_{i+1}) par $(a_i + a_{i+1}, b_i + b_{i+1})$ dans la liste L.

Il existe de nombreux algorithmes linéaires pour calculer l'enveloppe convexe d'un polygone. Le premier est dû à McCallum et Avis (McCallum et Avis, 1979) mais on lui préfère habituellement celui de Melkman (Melkman, 1987) (voir (Aloupis, web) pour une chronologie de ces algorithmes). Notons que si on implémente l'algorithme présenté ici à l'aide d'une pile, on obtient alors une version discrète et simplifiée de l'algorithme de Melkman. Bien entendu, l'algorithme ainsi obtenu ne s'applique qu'aux mots codant le bord d'un polyomino hv-convexe alors que celui de Melkman permet de calculer l'enveloppe convexe de n'importe quel polygone. L'algorithme présenté ici tire son intérêt du fait qu'il utilise des résultats issus de la combinatoire des mots antérieurs à l'algorithme de McCallum et Avis. D'un autre côté, la forte similarité avec l'algorithme de Melkman tend à montrer que l'utilisation des mots ne fait ici que masquer l'arithmétique du problème. La porte est donc toujours ouverte à l'élaboration d'une approche plus combinatoire au problème de calcul de l'enveloppe convexe d'un polyomino, comme les mots de Lyndon l'ont fait pour tester la convexité discrète.

Chapitre IV

PAVAGES ET DÉTECTION DES POLYOMINOS EXACTS

4.1 Introduction

L'idée de paver une surface à l'aide d'un motif répété tire ses racines des temps anciens autant pour des raisons pratiques qu'esthétiques. Plus récemment, les pavages ont été étudiés de manière théorique sous différents angles, en particulier en informatique théorique, en logique mathématique et géométrie discrète. Les pavages permettent de développer des outils efficaces pour démontrer l'indécidabilité d'un problème. En physique, on étudie les pavages afin de mieux comprendre la structure des quasi-cristaux.

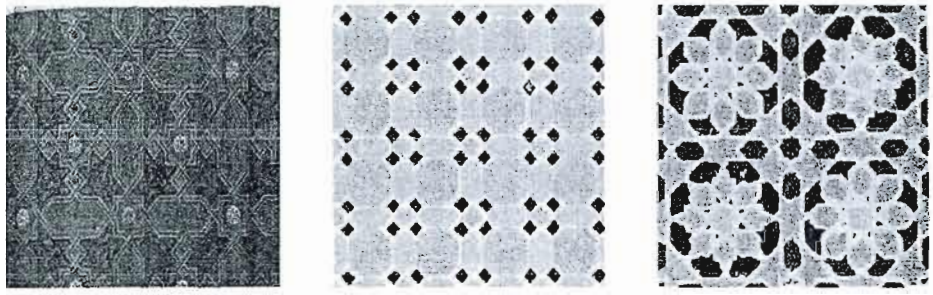


Figure 4.1 Trois des nombreux pavages qui décorent le palais de l'Alhambra à Granada en Espagne.

On s'intéresse ici au problème de pavage du point de vue de la théorie de le complexité. On définit habituellement un pavage par un ensemble de copies de polyominos qui recouvrent exactement une région donnée.

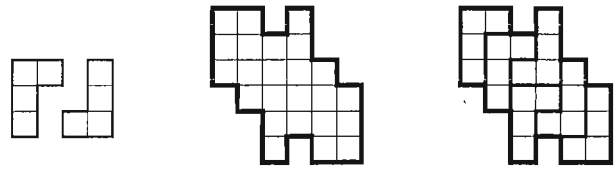
Définition 29. Un pavage $\mathcal{T}$ d'un sous-ensemble $S \subset \mathbb{Z}^2$ par un ensemble fini de polyominos $\mathcal{P}$ est un ensemble de couples $(p, \vec{u}) \in \mathcal{P} \times \mathbb{Z}^2$ tel que :

1. S est l'union des polyominos $p + \vec{u}$ pour tout les $(p, \vec{u}) \in \mathcal{T}$.
2. Pour toute paire distincte $(p, \vec{u}), (p', \vec{v}) \in \mathcal{T}$, les polyominos $p+\vec{u}$ et $p'+\vec{v}$ sont d'intersection vide.

Cette définition de pavage ne permet que les copies par translation des polyominos. Cette restriction n'entraîne aucune perte de généralité puisqu'on peut toujours inclure les images des polyominos considérés selon différentes transformations (rotation, symétrie, etc.) dans l'ensemble $\mathcal{P}$.

Définition 30 (Problème du pavage)**.** Étant donné un ensemble de polyominos $\mathcal{P}$ et un sous-ensemble $S \subset \mathbb{Z}^2$, est-ce que S admet un pavage par $\mathcal{P}$.

On supposera toujours que l'ensemble de S est 4-connexe puisque dans le cas contraire il suffit de traiter chacune des composantes 4-connexes indépendamment des autres.



(1) (2) (3)

Figure 4.2 (1) Un ensemble de polyominos $\mathcal{P}$, (2) un sous-ensemble $S \subset \mathbb{Z}^2$, (3) un pavage de S par $\mathcal{P}$.

Pour un ensemble de polyominos $\mathcal{P}$ fixé, dans le cas où le sous-ensemble du plan S est fini, le problème du pavage est clairement dans NP puisqu'étant donné un pavage de S par P on vérifie en un temps polynomial, en fonction de la taille de S, que le pavage est conforme à la Définition 29 (voir la Figure 4.2).

La complexité du problème du pavage varie de manière impressionnante selon les contraintes imposées. En particulier le pavage d'un ensemble fini S par des barres horizontales et verticales

illustre bien cette variation. Pour $k \geq 2$, on note h_k (resp. v_k) le polyomino formé d'une barre horizontale (resp. verticale) de k cellules consécutives et la complexité est donnée en fonction du nombre de cellules dans l'ensemble S.

- Le problème du pavage avec S sans trou et $\mathcal{P} = \{h_k, v_l\}$ se résout en $\mathcal{O}(n)$ (Kenyon et Kenyon, 1992).
- Le problème du pavage avec S possédant k trous et $\mathcal{P} = \{h_2, v_2\}$ se résout en $\mathcal{O}(nk + n\log(n))$ (Thiant, 2003).
- Le problème du pavage avec S possédant un nombre arbitraire de trous et $\mathcal{P} = \{h_k, v_l\} \neq \{h_2, v_2\}$ est NP-complet. [1]

Dans le cadre de cet ouvrage, on s'intéresse aux pavages de l'ensemble du plan, c'est-à-dire $S = \mathbb{Z}^2$. Puisque cette fois-ci l'ensemble à paver est infini, la notion de périodicité joue un rôle déterminant dans le traitement algorithmique de ces problèmes.

Définition 31. Un pavage $\mathcal{T}$ est dit périodique s'il existe une paire de vecteurs linéairement indépendants $\vec{u}$ et $\vec{v}$ tels que la translation par ces vecteurs ne modifie pas l'ensemble $\mathcal{T}$.

Définition 32. Un pavage $\mathcal{T}$ est dit semi-périodique s'il existe un vecteur $\vec{u}$ tel que la translation par ce vecteur ne modifie pas l'ensemble $\mathcal{T}$.

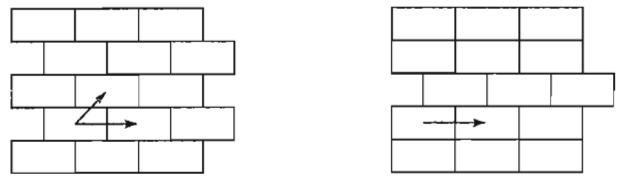


Figure 4.3 (1) Un pavage du plan périodique. (2) Un pavage du plan semi-périodique.

Remarque 2. Si un ensemble de polyominos $\mathcal{P}$ pave le plan de manière semi-périodique, alors il existe au moins un pavage du plan périodique par cet ensemble.

[1] Une démonstration de ce résultat est présentée dans (Beauquier et al., 1995) où elle est attribuée à un document non-publié de Garey, Johnson et Papadimitrou.

Le résultat le plus célèbre quand au pavages du plan provient de Berger qui à brisé la fameuse conjecture des dominos de Wang. En effet, dans (Wang, 1961) Wang avait considéré le problème de pavage du plan suivant :

Problème 2. *Étant donné un ensemble fini de carrés unitaires aux côtés colorés, appelés dominos, existe-t-il un pavage du plan tel que deux côtés adjacents soient toujours de la même couleur.*

Il en avait déduit la conjecture que voici.

Conjecture 1. *Si un ensemble de dominos pave le plan, alors il existe au moins un pavage du plan périodique par ces polyominos.*

Dans sa thèse de doctorat, Berger à montré indirectement que cette conjecture est fausse en prouvant l'indécidabilité du problème des dominos de Wang (Berger, 1966). Il a ensuite été en mesure de décrire explicitement un ensemble de dominos qui pavent le plan uniquement de manière apériodique. Évidemment le problème des dominos de Wang peut être réduit au problème de pavage du plan par un ensemble de polyominos, ce qui nous amène au résultat suivant :

Théorème 7 (Berger). *Le problème de pavage du plan par un ensemble fini de polyominos est indécidable.*

Pour un raffinement de ce résultat, voir également (Gurevich et Koriakov, 1972). Il est donc naturel d'étudier des versions du problème de pavage pour lesquelles des solutions algorithmiques peuvent être développées. Dans le cadre de ce chapitre, on considère le cas du pavage du plan par un seul polyomino.

4.2 Pavage du plan par un polyomino

Comme il est mentionné à la Section 1.5.3, un mot de contour code le bord d'un polyomino à partir d'un point arbitraire. C'est pour cette raison qu'on s'intéresse aux mots de contour à conjugaison près. Il est pour cela agréable de considérer les mots de contour comme des mots

circulaires mais, pour des raisons pratiques évidentes, on choisit de les représenter par des mots linéaires. On se permet donc de ne pas considérer les problèmes de bord en supposant qu'étant donné un mot de contour w de longueur n, pour tout entier k on a $w[k] = w[k+n]$.

Définition 33. Un polyomino est *exact* s'il existe un pavage du plan par ce polyomino.

Définition 34. Un pavage du plan $\mathcal{T}$ par un polyomino $\mathbf{P}$ est dit *régulier* s'il existe deux vecteurs $\overrightarrow{u}$ et $\overrightarrow{v}$ tels que $\mathcal{T} = \left\{(\mathbf{P}, a\overrightarrow{u} + b\overrightarrow{v}) \,|a, b \in \mathbb{Z}^2\right\}$.

Bien entendu, tout pavage régulier est périodique. Un premier résultat quant à la complexité du problème de pavage du plan par un seul polyomino provient de Wijshoff et van Leuveen (Wijshoff et van Leeuven, 1984) qui ont montré que, contrairement au cas général, si un polyomino pave le plan par translation alors il peut également le faire de manière régulière.

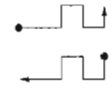
Théorème 8 (Wijshoff et van Leuveen). *Si* $\mathbf{P}$ *est un polyomino exact alors il existe un pavage du plan régulier par* $\mathbf{P}$.

Ceci fournit donc un premier test algorithmique puisqu'il suffit de tester s'il existe une paire de vecteurs $\overrightarrow{u}$ et $\overrightarrow{v}$ qui engendrent un pavage régulier du plan par le polyomino p. Puisqu'on peut borner la longueur de ces vecteurs en fonction de la taille du polyomino considéré, ceci permet d'établir la borne suivante quant à la complexité de ce problème.

Corollaire 4. *Le problème de pavage du plan par un polyomino se résout en temps polynomial.*

Quelque années plus tard, dans (Beauquier et Nivat, 1991) Beauquier et Nivat ont proposé une caractérisation des mots de contour des polyominos exacts. Cette caractérisation crée un autre lien fort entre la géométrie discrète et la combinatoire des mots puisqu'une condition nécessaire et suffisante afin qu'un polyomino soit exact est entièrement exprimée en fonction de la structure combinatoire de son mot de contour. Pour ce faire, on utilise l'opérateur $\widehat{\ }$ présenté au Chapitre 1. L'exemple qui suit rappelle qu'étant donné un chemin codé par le mot w, le mot $\widehat{w}$ code exactement le même chemin mais parcouru en sens inverse.

$$w = 0010\overline{1}01, \qquad \widehat{w} = \overline{1}\,\overline{0}1\overline{0}\overline{1}\overline{0}\overline{0}.$$



Théorème 9 (Beauquier-Nivat). *Un polyomino* $\mathbf{P}$ *pave le plan par translation si et seulement s'il existe* $X, Y, Z \in \Sigma^*$ *tels que* $XYZ\widehat{X}\widehat{Y}\widehat{Z} \in b(\mathbf{P})$ *où au plus un des mots* X, Y, Z *est vide.*

Nous appellons une telle factorisation d'un mot de contour $w \equiv XYZ\widehat{X}\widehat{Y}\widehat{Z}$ une BN-factorisation. Notons que cette caractérisation s'applique à la classe de conjugaison du mot w puisque le point de départ du mot lui-même sur le bord de la figure qu'il encode est arbitraire. Ainsi, on dira de deux BN-factorisations qu'elles sont *équivalentes* si elles correspondent à une permutation circulaire des facteurs $X, Y, Z, \widehat{X}, \widehat{Y}, \widehat{Z}$. En général, seulement quelques conjugués de w admettent de telles factorisations ; il est par contre possible qu'un même conjugué en admette plusieurs.

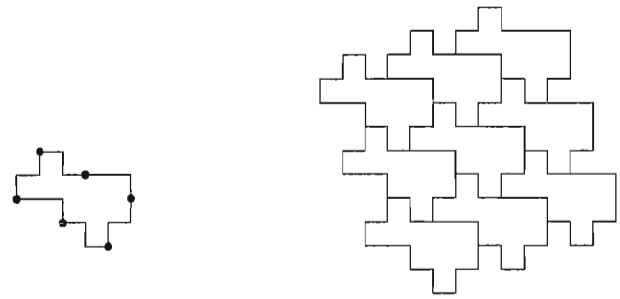
Considérons le polyomino exact, illustré à la Figure 4.4, dont le mot de contour est

$$w \equiv 1\,0\,1\,0\,\overline{1}\,0\,0\,0\,\overline{1}\,\overline{1}\,\overline{0}\,\overline{1}\,\overline{0}\,1\,\overline{0}\,1\,\overline{0}\,\overline{1}\,\overline{0}\,\overline{0}.$$

Ce mot admet la BN-factorisation suivante :

$$w \equiv 1\,0\,1 \cdot 0\,\overline{1}\,0 \cdot 0\,0\,\overline{1} \cdot \overline{1}\,\overline{0}\,\overline{1} \cdot \overline{0}\,1\,\overline{0} \cdot \overline{1}\,\overline{0}\,\overline{0}.$$

Une telle factorisation décrit explicitement de quelle manière on peut obtenir un pavage régulier



(1) (2)

Figure 4.4 (1) Un polyomino exact $\mathbf{P}$. (2) Un pavage du plan régulier par $\mathbf{P}$.

du plan. En effet, la définition de l'opérateur $\widehat{\ }$ assure que les facteurs X et $\widehat{X}$ codent exactement le même chemin et donc que les parties du bord du polyomino codées par ces facteurs vont s'emboiter parfaitement.

Remarque 3. Soit p polyomino exact dont le mot de contour w admet une décomposition $w \equiv XYZ\widehat{X}\widehat{Y}\widehat{Z}$ et soient $\overrightarrow{u}$ et $\overrightarrow{v}$ les vecteurs définis par $\overrightarrow{u} = \overrightarrow{X} + \overrightarrow{Y}$ et $\overrightarrow{v} = \overrightarrow{Y} + \overrightarrow{Z}$ alors $\mathcal{T} = \{(p, i\overrightarrow{u} + j\overrightarrow{v}) | i, j \in \mathbb{Z}^2\}$ forme un pavage régulier du plan.

Par exemple, considérons le mot $w \equiv 0010001010\bar{1}\bar{1}0\bar{1}0\bar{1}\bar{1}0\bar{1}0\bar{0}0\bar{1}0\bar{0}11\bar{0}1011$ avec la factorisation

$$\begin{aligned} w &\equiv X \cdot Y \cdot Z \cdot \widehat{X} \cdot \widehat{Y} \cdot \widehat{Z}, \\ &\equiv 0010 \cdot 0010 \cdot \bar{1}\bar{1}0\bar{1}0\bar{1}\bar{1} \cdot \bar{0}\bar{1}0\bar{0} \cdot \bar{0}\bar{1}0\bar{0} \cdot 11\bar{0}1011. \end{aligned}$$

Les vecteurs associés sont $\overrightarrow{u} = \overrightarrow{X} + \overrightarrow{Y} = (6, 2)$ et $v = \overrightarrow{Y} + \overrightarrow{Z} = (3, -4)$ tel qu'illustré à la Figure 4.5.

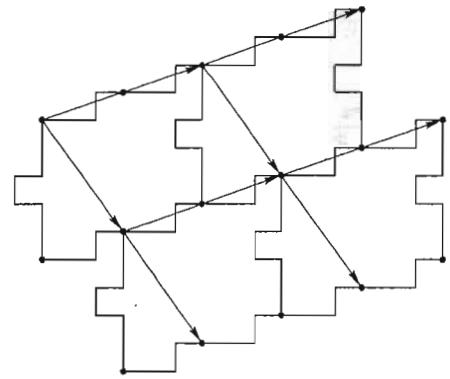


Figure 4.5 Les translations définies par une BN-factorisation.

La BN-factorisation définit deux types de polyminos exacts. Les pseudo-carrés et les pseudo-hexagones.

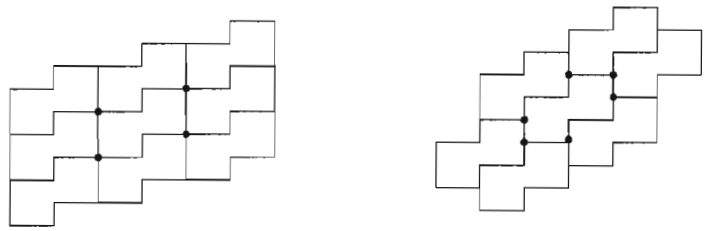
Définition 35. Un polyomino dont la BN-factorisation $w \equiv XYZ\widehat{X}\widehat{Y}\widehat{Z}$ est telle qu'aucun des facteurs X, Y et Z n'est le mot vide est appelé un *pseudo-hexagone*.

Définition 36. Un polyomino dont la BN-factorisation $w \equiv XYZ\widehat{X}\widehat{Y}\widehat{Z}$ est telle qu'un des facteurs X, Y ou Z est le mot vide est appelé un *pseudo-carré*.

Dans le cas d'un pseudo-carré, on supposera toujours que le facteur vide est Z. La factorisation de son bord sera donc $w \equiv XY\widehat{X}\widehat{Y}$. Notons qu'il est impossible que deux des facteurs X, Y

et Z soient vides car on aurait alors que $w \equiv u\widehat{u}$. Notons également qu'un polyomino peut admettre plusieurs BN-factorisations et peut être à la fois pseudo-hexagone et pseudo-carré.

Par exemple, le polyomino dont le mot de contour est $w \equiv 1\,1\,0\,0\,1\,0\,0\,\overline{1}\,\overline{1}\,\overline{0}\,\overline{0}\,\overline{1}\,\overline{0}\,\overline{0}$ admet deux BN-factorisations et est à la fois pseudo-carré et pseudo-hexagone.



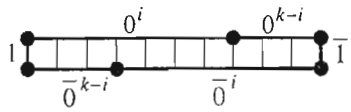
(1) (2)

Figure 4.6 (1) Un pavage de type pseudo-carré. (2) Un pavage de type pseudo-hexagone.

$$(1)\ w \equiv 1\,1 \cdot 0\,0\,1\,0\,0 \cdot \overline{1}\,\overline{1} \cdot \overline{0}\,\overline{0}\,\overline{1}\,\overline{0}\,\overline{0},$$

$$(2)\ w \equiv 1 \cdot 1\,0\,0\,1 \cdot 0\,0 \cdot \overline{1} \cdot \overline{1}\,\overline{0}\,\overline{0}\,\overline{1} \cdot \overline{0}\,\overline{0}.$$

En général le nombre de BN-factorisations que peut admettre un polyomino est linéaire en fonction de son périmètre. Le pire cas étant celui de la longue brique $w \equiv 1\,0^k\,\overline{1}\,\overline{0}^k$, pour tout $i \in \{0, 1, \ldots, k-1\}$ la factorisation $w \equiv 1 \cdot 0^i \cdot 0^{k-i} \cdot \overline{1} \cdot \overline{0}^i \cdot \overline{0}^{k-i}$ est une BN-factorisation valide, tel qu'illustré ci dessous.



Dans cet exemple notons que même si le nombre de factorisations peut être arbitrairement grand, une seule est du type pseudo-carré ($i = 0$ et $i = k$ étant deux cas équivalents) alors que toutes les autres sont de type pseudo-hexagone. La Section 4.4 étudie en détail les polyominos admettant plus d'une factorisation de type pseudo-carré.

Du point de vue algorithmique, la caractérisation de Beauquier-Nivat d'un mot se calcul naïvement par une méthode essais-erreurs en $\mathcal{O}(n^4)$. Gambini et Vuillon ont abaissé significative-

ment cette borne en proposant un algorithme en $\mathcal{O}(n^2)$ qui calcule toutes les factorisations de Beauquier-Nivat d'un mot. Dans la section qui suit, de nouveaux algorithmes, inspirés par celui de Gambini et Vuillon, permettent d'abaisser cette borne à $\mathcal{O}(n)$ pour la détection des pseudo-carrés et d'une certaine classe de pseudo-hexagones.

4.3 Algorithmes de détection des polyominos exacts

Le principe de base des algorithmes qui suivent est le suivant : si un mot admet une factorisation $w \equiv XYZ\widehat{X}\widehat{Y}\widehat{Z}$, alors toute lettre de ce mot appartient à un des trois facteurs X, Y ou Z. Ainsi on va débuter par rechercher une certaine classe de facteurs A tels que $w \equiv Ax\widehat{A}y$.

Définition 37. Soit w le mot de contour d'un polyomino. Un facteur A débutant à la position i de w est dit *admissible* s'il existe deux mots de même longueur x et y tels que

(i) $w \equiv Ax\widehat{A}y$.

(ii) A est *maximal* au sens que $f(x) \neq \overline{l(x)}$ et $f(y) \neq \overline{l(y)}$.

Remarque 4. Un facteur A peut avec plusieurs occurrences dans w. Ainsi, un facteur admissible est en fait désigné par la paire (A, i) puisqu'il s'agit de l'occurrence de A débutant à la position i de w. Par abus de notation, on écrit seulement A lorsque cela n'engendre pas de confusion.

Par exemple, considérons le mot de contour $w = 0000\overline{1}\overline{0}\overline{0}\overline{0}\overline{0}1$. Ce mot contient trois occurrences du facteur $A = 00$ débutant respectivement aux positions $1, 2$ et 3. Seulement celles débutant aux positions 1 et 3 sont admissibles car celle débutant à la position 2 ne satisfait pas la condition (ii).

$$(A,1): \quad w \equiv \quad Ax\widehat{A}y \equiv 00 \cdot 00\overline{1} \cdot \overline{0}\,\overline{0} \cdot \overline{0}\,\overline{0}1. \quad \text{(Admissible)}$$

$$(A,2): \quad w \equiv \quad Ax'\widehat{A}y' \equiv 00 \cdot 0\overline{1}\overline{0} \cdot \overline{0}\,\overline{0} \cdot \overline{0}1\overline{0}. \quad \text{(Non-admissible)}$$

$$(A,3): \quad w \equiv \quad Ax''\widehat{A}y'' \equiv 00 \cdot \overline{1}\overline{0}\overline{0} \cdot \overline{0}\,\overline{0} \cdot 1\overline{0}\overline{0}. \quad \text{(Admissible)}$$

Définition 38. Soit w un mot de contour et (A, i) un de ses facteurs admissibles. On appelle $\left(\widehat{A}, i + \frac{|w|}{2}\right)$ le facteur *homologue* de (A, i).

Étant donné une occurrence d'un facteur admissible, on s'intéresse à l'ensemble des lettres qui le constituent.

Définition 39. Soit (A, i) un facteur admissible du mot de contour w. Soit $n = |w|$ et $k = |A|$, on appelle l'ensemble $\{i, i+1, \ldots, i+k-1\}$ les positions de w *recouvertes* par le facteur admissible (A, i).

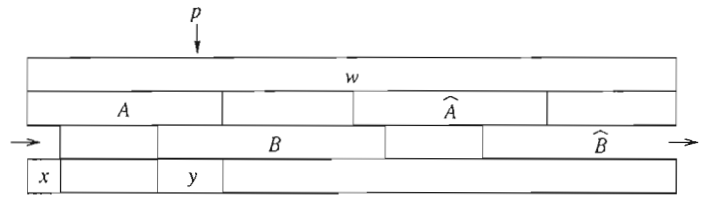
Par exemple, le facteur admissible $(00, 3)$ du mot de contour $w = 0\,0\,0\,0\,\overline{1}\,\overline{0}\,\overline{0}\,\overline{0}\,\overline{0}\,1$ recouvre les positions $\{3, 4\}$ et son homologue les positions $\{8, 9\}$.

On peut maintenant énoncer la proposition suivante qui provient de (Brlek et Provençal, 2006b).

Proposition 6. *Soit $w \in \Sigma^n$ le mot de contour d'un polyomino et $p \in \{1, 2, \ldots, n\}$. Soit $\mathcal{A}$ l'ensemble des facteurs admissibles qui recouvrent la position p et $\widehat{\mathcal{A}}$ l'ensemble de leurs homologues respectifs. Il existe au moins une position $q \in \{1, 2 \ldots, n\}$ telle que q n'est recouverte par aucun des éléments de $A \cup \widehat{\mathcal{A}}$.*

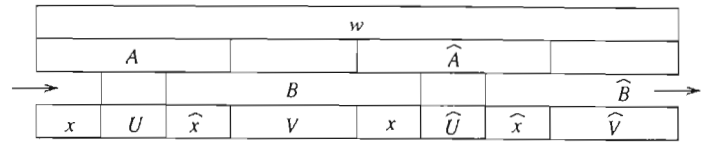
Preuve. Tout d'abord, remarquons que si A est un facteur admissible de w, on a alors que $|A| < |w|/2$. Puisqu'il existe $x, y \in \Sigma^*$ tels que $w \equiv Ax\widehat{A}y$, on a que $|A| \leq |w|/2$. Le seul cas à considérer est si $x = y = \varepsilon$. Ceci est impossible car on aurait alors que $w \equiv A\widehat{A}$ et donc $l(A)\overline{l(A)} \in \mathrm{Fact}(w)$. Contradiction, un mot de contour ne peut pas contenir un facteur de la forme $a\overline{a}$ où $a \in \Sigma$.

Maintenant, on procède par contradiction et on suppose que toutes les lettres de w sont recouvertes par des facteurs de $\mathcal{A}$ ou $\widehat{\mathcal{A}}$. Soit $A \in \mathcal{A}$ le facteur débutant à la position la plus à gauche par rapport à la position p et $B \in \mathcal{A}$ le facteur qui se termine en la position la plus à droite, tel qu'illustré ci-dessous.



Soit x le chevauchement entre les facteurs A et $\widehat{B}$ et y celui entre A et B. Sans perte de généralité, on peut supposer que $|x| \leq |y|$. Il y a alors deux cas à considérer.

1. Si $|x| = |y|$ alors en utilisant le fait que pour toute paire de mots u, v on a



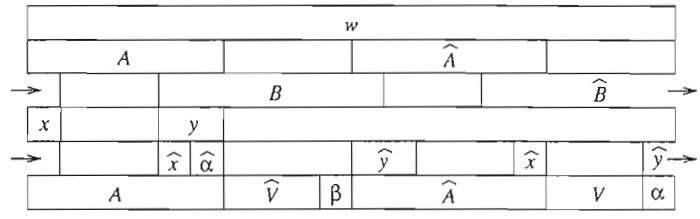
$$u \in \mathrm{Pref}(v) \iff \widehat{u} \in \mathrm{Suff}(\widehat{v}). \tag{4.1}$$

On obtient que $x = \widehat{y}$ et que w se factorise de la manière suivante

$$w \equiv x\,U\,\widehat{x}\,V\,x\,\widehat{U}\,\widehat{x}\,\widehat{V}.$$

On s'intéresse maintenant à la différence entre le nombre de virages à gauche et le nombre de virages à droite lorsqu'on parcourt le chemin codé par le mot w. On remarque que chaque virage à gauche dans un facteur est annulé par un virage à droite dans son facteur homologue et vice-versa. Ainsi il ne reste qu'à considérer les virages qui ont lieu entre ces facteurs. On remarque alors que si un virage a lieu entre les facteurs x et U alors ce virage sera annulé par son inverse entre les facteurs $\widehat{U}$ et $\widehat{x}$. Il en va de même pour tous les autres : $U\widehat{x}$ est annulé par $x\widehat{U}$, $\widehat{x}V$ par $\widehat{V}x$ (on s'intéresse au chemin fermé), et Vx par $\widehat{x}\widehat{V}$. On conclut alors que le nombre de virages à gauche est égal au nombre de virage à droite et que, par la Propriété 2 (Section 1.5.3), le chemin codé par w se croise. Contradiction.

2. Si $|x| < |y|$ on s'intéresse alors à la propagation du facteur y dans le mot w tel que décrite par l'équation (4.1).



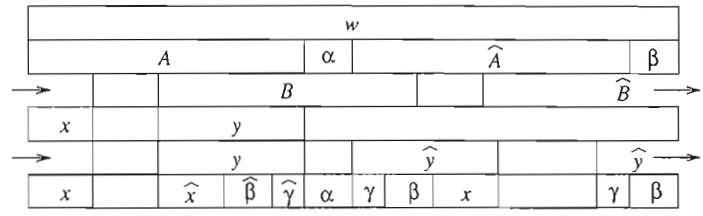
Dans le cas où le facteur $\widehat{y}$ ne chevauche pas $\widehat{A}$ dans $\widehat{B}$ (tel qu'illustré ci-dessus), on pose V comme étant le facteur entre $\widehat{A}$ et $\widehat{y}$. On obtient alors la factorisation $w \equiv A\widehat{V}\beta\widehat{A}V\alpha$

où α est défini comme le préfixe de $\widehat{y}$ tel que $\widehat{y} = \alpha x$ et β est le facteur entre $\widehat{V}$ et $\widehat{A}$. En passant aux vecteurs on obtient

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

Cependant $\widehat{y} = \alpha x$ de sorte que le facteur β est suivit par α dans w. Ainsi le facteur non-vide $\beta\alpha$ code une boucle fermée sur la frontière d'un polyomino. Contradiction.

Finalement, dans le cas où $\widehat{y}$ chevauche le facteur $\widehat{A}$ dans $\widehat{B}$, on a la situation suivante



où $w \equiv A\alpha\widehat{A}\beta$. On a alors que

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

On pose γ comme étant le chevauchement entre $\widehat{A}$ et $\widehat{y}$ dans $\widehat{B}$. Comme $\widehat{y} = \gamma\beta x$ le facteur $y\alpha\widehat{y}$ du mot w contient le facteur non-vide $\widehat{\gamma}\alpha\gamma\beta$ correspondant à une boucle fermée. Contradiction. ∎

Une conséquence directe de ce résultat est que l'admissibilité est prérequise pour les facteurs d'une BN-factorisation.

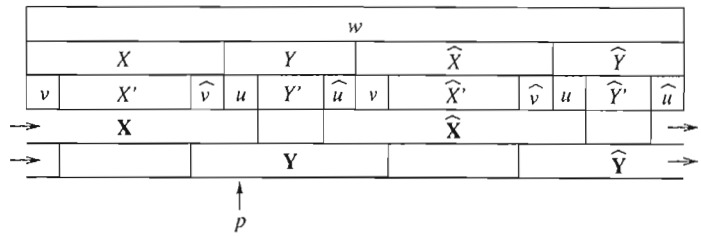
Corollaire 5. *Soit $w \equiv XYZ\widehat{X}\widehat{Y}\widehat{Z}$ la BN-factorisation d'un mot codant le bord d'un polyomino exact. Alors X, Y et Z sont des facteurs admissibles de w.*

Preuve. La condition (i) de la Définition 37 est une conséquence directe du fait que $|u| = |\widehat{u}|$ pour tout mot $u \in \Sigma^*$. Pour la condition (ii), procédons par contradiction. Supposons que le facteur X n'est pas *maximal*, c'est-à-dire que $f(YZ) = \overline{l(YZ)}$.

Dans le cas d'une factorisation de type pseudo-hexagone, les mots Y et Z sont non-vides. Il existe donc une lettre $a \in \Sigma$ et deux mots $Y', Z' \in \Sigma^*$ tels que $Y = aY'$ et $Z = Z'\overline{a}$. On a

alors que $\widehat{Y}\widehat{Z} = \widehat{Y'}\overline{a}a\widehat{Z'}$ ce qui est impossible puisque $\overline{a}a$ ne peut être facteur d'un mot codant le bord d'un polyomino. Contradiction.

Dans le cas d'une factorisation de type pseudo-carré $w \equiv XY\widehat{X}\widehat{Y}$, l'hypothèse de départ implique que $f(Y) = \overline{l(Y)}$. Soit $u \in \Sigma^+$ le plus long préfixe de Y tel que $Y = uY'\widehat{u}$ pour un certain $Y' \in \Sigma^*$. On pose $\mathbf{X} = \widehat{u}Xu$, le facteur $\mathbf{X}$ est admissible puisque par construction $f(Y') \neq \overline{l(Y')}$. Similairement, soit $\mathbf{Y} = \widehat{v}Y'v$ où v est le plus long préfixe de X tel que $X = vX'\widehat{v}$. Notons que contrairement à u, le mot v est possiblement vide.



Comme u est non-vide, il existe une position p dans w telle que la totalité du mot w est recouverte par les facteurs admissibles recouvrant cette position ou leurs homologues. Ceci contredit la Proposition 6. ∎

Une approche similaire a été utilisée par Gambini et Vuillon afin d'élaborer leur algorithme ((Gambini et Vuillon, 2003), section 3.1). Par contre, le point de vue diffère car nous privilégions ici des arguments combinatoires et non géométriques.

4.3.1 Détection des pseudo-carrés

Étant donné un mot de contour w, déterminer si w admet une factorisation de type pseudo-carré est un problème qui se résout en temps linéaire. L'idée de base afin d'atteindre cette borne est de choisir une position quelconque dans w et de lister tous les facteurs admissibles qui la recouvrent. Si ce mot admet une BN-factorisation, alors forcément un des facteurs admissibles listés en fera partie.

Lemme 4. *Soit $w \in \Sigma^*$ un mot de contour. Pour chaque position p de w lister l'ensemble des facteurs admissibles qui incluent la position p se calcule en temps linéaire.*

Preuve. Cette borne linéaire, est atteinte à l'aide du calcul des plus longues extensions en temps constant (voir Théorème 4, Section 1.6.2). L'idée est la suivante : au lieu de chercher un facteur A et son homologue $\widehat{A}$ dans le mot w, on recherche un facteur A commun aux mots w et $\widehat{w}$, puis à partir de sa position dans $\widehat{w}$ on calcule la position de $\widehat{A}$ dans w. L'algorithme suivant exploite cette idée afin de lister tous les facteurs admissibles qui recouvrent la position p du mot w.

Algorithme 6 (listeFacteursAdmissibles).

Entrée : $w \in \Sigma^n$ un mot de contour et $p \in \{1, 2, \ldots, n\}$.

1 : **Pour** i de 1 à n **faire**
2 : **Si** $w[p] = \widehat{w}[i]$ **alors**
3 : $g \leftarrow \text{PLECG}(w, \widehat{w}, p, i) - 1$;
4 : $d \leftarrow \text{PLECD}(w, \widehat{w}, p, i) - 1$;
5 : $A \leftarrow w[p-g, \ldots, p+d]$;
6 : **Si** $w \equiv Ax\widehat{A}y$ **et** $|x| = |y|$ **alors**
7 : Ajouter A à la liste de facteurs admissibles
8 : **fin si**
9 : **fin si**
10 : **fin pour**

Le mot w est considéré comme un mot circulaire mais une implémentation efficace de cet algorithme peut n'utiliser que des mots linéaires en les conjuguant de manière appropriée afin d'éviter les problèmes de bords. De plus, aux lignes 5, 6 et 7, pour que cet algorithme soit linéaire, il est important que l'implémentation ne manipule que la position et la longueur de chacun des facteurs considérés et non les facteurs eux-mêmes. Notons également que par la définition de PLECG et PLECD les facteurs A calculés à la ligne 5 sont forcément maximaux au sens de la Définition 37 (ii). À la ligne 6, la position de $\widehat{A}$ dans w est calculée à partir du fait que $A = \widehat{w}[i-g, \ldots, i+d]$. On peut ainsi déterminer si A et $\widehat{A}$ se chevauchent dans w et, dans le cas contraire, si $|x| = |y|$, le tout en $\mathcal{O}(1)$. ∎

Ce lemme implique que le nombre de facteurs admissibles dans un mot est au plus linéaire en sa taille. Déterminer une borne exacte au nombre de facteurs admissibles distincts d'un mot reste un problème ouvert qui s'apparente au problème de déterminer une borne supérieure exacte au nombre de carrés distincts d'un mot (voir entre autre (Lothaire, 1997) et (Ilie, 2005)). Le

résultat suivant provient de (Brlek et Provençal, 2006c).

Théorème 10. *Soit $w \in \Sigma^*$ un mot de contour. Déterminer si w admet une factorisation de type pseudo-carré se calcule en temps linéaire.*

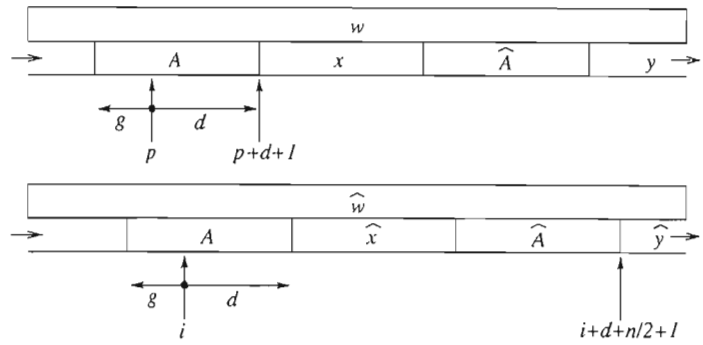


Figure 4.7 Un facteur admissible A dans les mots w et $\widehat{w}$.

Preuve. La première étape consiste à appliquer le Lemme 4 sur une position p quelconque du mot w. L'Algorithme 6 fournit alors la liste de tous les facteurs admissibles qui chevauchent cette position. Il ne reste alors plus qu'à vérifier, pour chaque facteur admissible, si $x = \widehat{y}$. Le Corollaire 5 assure que tous les facteurs d'une BN-factorisation sont maximaux. Ainsi, pour vérifier que $x = \widehat{y}$, il suffit de calculer la plus longue extension commune avec comme points de départ les premières lettres respectives de ces deux facteurs. Ceci peut être fait en remplaçant la ligne 7 de l'Algorithme 6 par :

7.1 : **Si** PLECD$(w, \widehat{w}, p + d + 1, i + d + \frac{n}{2} + 1) = |x|$ **alors**
7.2 : $w \equiv Ax\widehat{A}\widehat{x}$ est une factorisation pseudo-carrée.
7.3 : **Fin si**

La Figure 4.7 illustre cette situation. Cette modification ne change pas la complexité de l'algorithme puisque la fonction PLECD se calcule en temps constant. On obtient donc un algorithme linéaire pour détecter les pseudo-carrés. ∎

L'algorithme obtenu permet non seulement de détecter si un polyomino admet une factorisation de type pseudo-carré mais également de toutes les énumérer. Pour ce faire, lorsqu'on atteint la ligne 7.2 de l'algorithme, il suffit de stocker la factorisation obtenue et laisser rouler l'algorithme. Le tout demeure linéaire en fonction de la taille de w. Afin d'illustrer cet algorithme,

considérons le mot $w = 00\bar{1}00\bar{1}\bar{0}\bar{0}1\bar{0}\bar{0}1\bar{0}\bar{0}1001$. La Figure 4.8 montre les facteurs admissibles détectés par l'Algorithme 6 en prenant $p = 1$ comme point de départ.

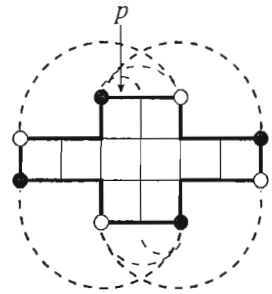


Figure 4.8 Un pseudo-carré avec ses facteurs admissibles qui recouvrent la position p.

Les deux facteurs admissibles représentés par les lignes plus épaisses sont ceux qui mènent à des factorisations de type pseudo-carré alors que les deux autres ne le permettent pas. Les deux factorisations obtenues sont identifiées sur la figure par ○ et •.

$$\circ \ : \ w \equiv 00100 \cdot \bar{1}00\bar{1} \cdot \bar{0}\bar{0}1\bar{0}\bar{0} \cdot 1\bar{0}\bar{0}1;$$

$$\bullet \ : \ w \equiv 00\bar{1}00 \cdot \bar{1}\bar{0}\bar{0}1 \cdot \bar{0}\bar{0}1\bar{0}\bar{0} \cdot 1001.$$

Il s'agit d'un cas rare de mot qui admet deux factorisations différentes de type pseudo-carré. La Section 4.4 étudie en détail la structure de ces mots particuliers.

4.3.2 Détection des pseudo-hexagones

La stratégie adoptée afin de détecter les pseudo-hexagones ressemble à celle employée pour détecter les pseudo-carrés. On commence encore une fois par lister l'ensemble des facteurs admissibles qui recouvrent une position quelconque du mot w. On obtient alors une factorisation de la forme $w \equiv Xx\widehat{X}y$ et il ne reste plus qu'à tester s'il existe Y et Z tels que $x = YZ$ et $y = \widehat{Y}\widehat{Z}$. Tester directement l'existence d'une telle paire de mots requiert forcément $\mathcal{O}(n)$ opérations ce qui porte la complexité totale de l'algorithme à $\mathcal{O}(n^2)$. On privilégie plutôt l'approche suivante qui consiste à bâtir deux listes de facteurs admissibles pour trouver une paire qui coïncide.

Algorithme 7 (DétectePseudoHexagone).

Entrée : $w \in \Sigma^n$ un mot de contour et $p \in \{1, 2, \ldots, n\}$.

1 : **Construire** L_1 : la liste de tous les facteurs admissibles qui recouvrent la position p.
2 : $m \leftarrow$ la position de la lettre la plus à droite dans un des facteurs de L_1.
3 : **Construire** L_2 : la liste de tous les facteurs admissibles qui recouvrent la position $m+1$.
4 : **Pour chaque** $A \in L_1$ **faire**
5 : **Pour chaque** $B \in L_2$ **faire**
6 : **Si** $w \equiv ABx\widehat{A}\widehat{B}y$ **ou** $w \equiv AxB\widehat{A}y\widehat{B}$ **alors**
7 : $i \leftarrow$ la position de la première lettre de x dans w.
8 : $j \leftarrow$ la position de la première lettre de $\widehat{y}$ dans $\widehat{w}$.
9 : **Si** $\mathrm{PLECD}(w, \widehat{w}, i, j) = |x|$ **alors**
10 : w admet une BN-factorisation.
11 : **fin si**
12 : **fin si**
13 : **fin pour**
14 : **fin pour**

On remarque que cet algorithme peut être modifié (très légèrement) afin d'énumérer toutes les factorisations de Beauquier-Nivat d'un mot de contour puisque les factorisations de type pseudo-carré correspondent au cas où, à la ligne 6, les variables x et y sont vides. Étant donné que le nombre de facteurs admissibles qui recouvrent une position donnée peut être linéaire en fonction de la longueur du mot, cet algorithme est quadratique au pire cas. Cependant, dans certains cas, il est possible de borner le nombre de facteurs admissibles ce qui diminue la complexité totale. En effet, la présence d'un grand nombre de facteurs admissibles recouvrant une position donnée implique la présence de répétitions arbitrairement longues dans le mot.

Définition 40. Un mot w est dit *sans-k-carrés*, pour $k \geq 2$ si pour tout facteur u de w on a que $u = xx \implies |u| < k$.

Par exemple, le $w = 0010011110000$ est un mot sans-k-carrés pour tout $k \geq 7$ puisque son plus long facteur carré est 001001.

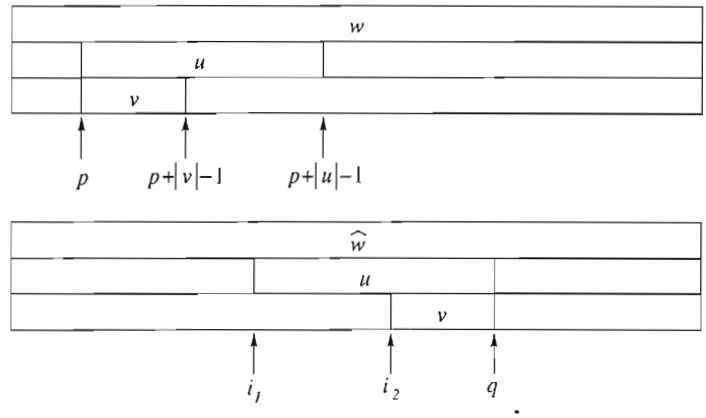
On peut borner la complexité de l'algorithme **DétectePseudoHexagone** dans le cas des mots qui ne possèdent pas de facteurs carrés trop longs.

Théorème 11. *Soit $w \in \Sigma^n$ un mot de contour sans-k-carrés. Déterminer si w admet une BN-factorisation se teste en temps $\mathcal{O}(n + k^2)$.*

Ce résultat découle directement du lemme suivant qui borne le nombre de facteurs admissibles et par la même occasion, la complexité des boucles imbriquées de l'Algorithme 7.

Lemme 5. *Soit $w \in \Sigma^n$ un mot de contour sans-k-carrés et p une position dans w. Le nombre de facteurs admissibles qui recouvrent la position p de w est borné par $2k + 2\log(n)$.*

Preuve. Soient $A_1, A_2, \ldots A_r$ les facteurs admissibles qui recouvrent la position p dans w. Par définition $w \equiv A_i x_i \widehat{A_i} y_i$ avec $|x_i| = |y_i|$ pour chaque $1 \leq i \leq r$ de sorte que tous les facteurs $\widehat{A_i}$ recouvrent la position $p' = p + \frac{n}{2}$. Il existe donc une position q dans $\widehat{w}$ telle que tous les facteurs A_i détectés dans $\widehat{w}$ recouvrent cette position. Dans l'Algorithme 6 (**listeFacteursAdmissibles**) les facteurs admissibles sont listés à travers une boucle telle que chaque itération peut détecter au maximum un d'entre eux. Soient i_1, i_2 tels que $1 \leq i_1 < i_2 \leq q$ et supposons que des facteurs admissibles ont été détectés lors des itérations où i a pris les valeurs i_1 et i_2.



Soit $u = \widehat{w}[i_1, \ldots, q]$ et $v = \widehat{w}[i_2, \ldots, q]$. Par définition de la plus longue extension commune, on a que $u = w[p, \ldots, p + |u| - 1]$ et $v = w[p, \ldots, p + |v| - 1]$ tel qu'illustré ci-dessus. On a donc que v est à la fois préfixe et suffixe de u ce qui implique que u possède la période $|u| - |v|$.

Considérons le cas où i_1 et i_2 sont inférieurs à $q - k$. Il est impossible que $|u| < 2|v|$. Par contradiction, supposons que c'est le cas. On a alors que le facteur u possède une période

inférieure à la moitié de sa longueur. Il existe donc α un facteur carré de u tel que $|\alpha| \geq \frac{|u|}{2}$. Par la définition de sans-k-carré, $|\alpha| < k$. On en déduit les inégalités suivantes

$$k > |\alpha| \geq \frac{|u|}{2} \geq |v|.$$

Or $|v| > k$ car on a supposé que $i_2 < q - k$. Contradiction.

La même situation se produit lorsque i_1 et i_2 sont supérieurs à $q + k$.

Ainsi, dans l'Algorithme 6, lorsque l'indice i va de 1 à n, le nombre de facteurs admissibles détectés est borné par :

- $\log n$ pour i de 1 à $q - k$,
- $2k$ pour i de $q - k$ à $q + k$,
- $\log n$ pour i de $q + k$ à n.

En additionant le tout, on obtient la borne énoncée. ∎

On en conclut le résultat suivant, tiré de (Brlek et Provençal, 2006a).

Corollaire 6. *Soit $w \in \Sigma^n$ un mot de contour sans-k-carrés avec $k \in \mathcal{O}(\sqrt{n})$. Déterminer si w admet une BN-factorisation se teste en temps linéaire.*

Cette contrainte sur la longueur des carrés est une condition suffisante pour assurer la linéarité mais pas nécessaire. En fait, il est possible qu'un mot possède des carrés arbitrairement longs, que le nombre de facteurs admissibles en une position donnée soit linéaire en fonction de la taille du mot w mais que l'Algorithme 7 reste linéaire. Considérons par exemple le mot de contour :

$$w = 0^k\, 1\, 0\, \bar{1}\, 0\, 0\, \bar{1}\, \bar{1}\, 0\, \bar{0}^k\, 1\, \bar{0}\, \bar{1}\, \bar{0}\, \bar{0}\, 1\, 1.$$

La Figure 4.9 illustre le cas $k = 8$. On remarque que même si la position $p = 1$ est recouverte par $k - 1$ facteurs admissibles, la position $m + 1$ n'est recouverte que par un seul. Ainsi, les deux boucles imbriquées donneront lieu à seulement $k - 1 \in \mathcal{O}(n)$ itérations.

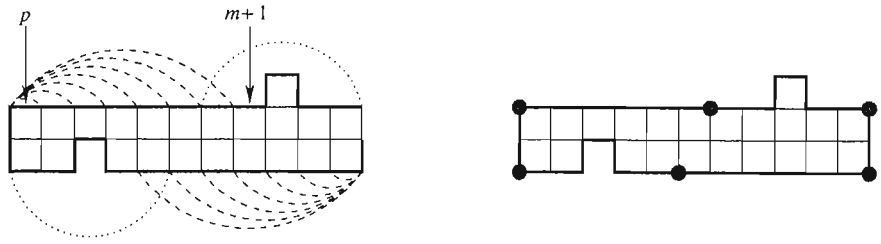


Figure 4.9 Les facteurs admissibles d'un polyomino exact et sa BN-factorisation.

4.3.3 Optimisations de l'algorithme

Une première optimisation de l'Algorithme 7 (**DétectPseudoHexagone**) consiste à stocker les facteurs admissibles de manière à pouvoir accéder directement à ceux débutant ou terminant en une position donnée. Par exemple, en utilisant un tableau de n listes d'entiers de manière à ce que la k-ième liste contienne les entiers l tels que $w[k..l]$ est un facteur admissible. On peut procéder de la même façon pour gérer les positions où se terminent les facteurs admissibles et ainsi éviter de boucler sur des paires de facteurs qui ne correspondent pas.

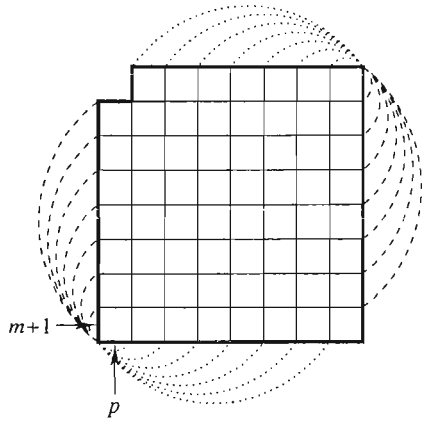


Figure 4.10 Un polyomino exact et ses deux listes de facteurs admissibles.

Cette optimisation n'est pas suffisante pour abaisser la complexité de l'algorithme car certains cas, comme celui présenté à la Figure 4.10, possèdent un nombre linéaire de facteurs admissibles se terminant à une position donnée et autant débutant à la position suivante. Ce genre de

situation peut mener à un nombre quadratique de tests.

De telles situations impliquent une périodicité dans le mot de contour. La détection de ces régularités permet d'éviter des tests inutiles et ainsi d'améliorer la performance de l'algorithme. Le lemme suivant fait le lien entre la périodicité et la présence de facteurs admissibles.

Lemme 6. *Soit $w \in \Sigma^n$ un mot de contour avec deux facteurs admissibles $A_1 = w[i, \ldots, p]$ (resp. $A_1 = w[p, \ldots, i]$) et $A_2 = w[j, \ldots, p]$ (resp. $A_2 = w[p, \ldots, j]$) avec $|A_1| > |A_2|$. Alors*

(i) *A_1 et A_2 ont la période $|j - i|$.*

(ii) *Soit m une période de A_1. Si m divise $|j - i|$ alors pour tout $0 \leq k \leq \left\lfloor \frac{p-i}{m} \right\rfloor$, le facteur $w[i + km, \ldots, p]$ (resp. $w[p, \ldots, i - km]$) est admissible.*

Preuve. (i) Par définition de facteur admissible, $w[i + \frac{n}{2}, \ldots, p + \frac{n}{2}] = \widehat{A_1}$ et $w[j + \frac{n}{2}, \ldots, l + \frac{n}{2}] = \widehat{A_2}$. Ainsi, $\widehat{A_2}$ est un suffixe de $\widehat{A_1}$. On a donc que A_2 est à la fois préfixe et suffixe de A_1, d'où A_1 a la période $|A_1| - |A_2| = j - i$.

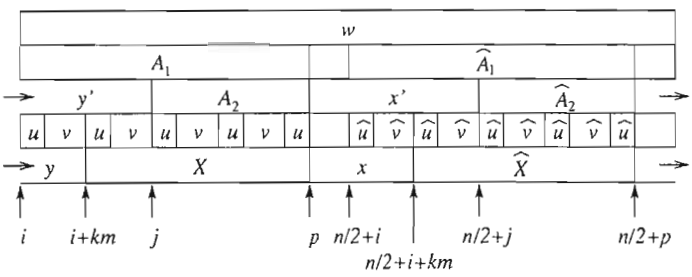
(ii) Pour $k = 0$ c'est trivial. On pose $X = w[i + km, \ldots, p]$ pour un $1 \leq k \leq \left\lfloor \frac{p-i}{m} \right\rfloor$. Comme A_1 a la période m, il existe deux mots u, v avec $|uv| = m$ tels que $A_1 = (uv)^\mu u$ et $X = (uv)^{\mu-k}u$. De plus, comme m divise $j - i$, $A_2 = (uv)^\nu u$ où $\nu = \left\lfloor \frac{j-p}{m} \right\rfloor$. D'un autre côté, on a

$$w[i + \frac{n}{2}, \ldots, p + \frac{n}{2}] = \widehat{A_1} = (\widehat{u}\widehat{v})^\mu \widehat{u},$$

ce qui implique que

$$w[i + km + \frac{n}{2}, \ldots, p + \frac{n}{2}] = (\widehat{u}\widehat{v})^{\mu-k}\widehat{u} = \widehat{X}.$$

Il existe donc x et y tels que $w \equiv Xx\widehat{X}y$ avec $|x| = |y|$.



Il reste à voir que $f(x) \neq \overline{l(x)}$ et $f(y) \neq \overline{l(y)}$. Comme A_2 est un facteur admissible, $w \equiv A_2 x' \widehat{A_2} y'$ avec $|x'| = |y'|$, $f(x') \neq \overline{l(x')}$ et $f(y') \neq \overline{l(y')}$. Comme A_1 et $\widehat{A_1}$ ont la période m, on a que $l(x) = l(x')$ et $l(y) = l(y')$. On obtient alors les inégalités voulues :

$$\begin{aligned} f(x) = f(x') &\neq \overline{l(x')} = \overline{l(x)}. \\ f(y) = f(y') &\neq \overline{l(y')} = \overline{l(y)}. \quad \blacksquare \end{aligned}$$

Ainsi, lorsque deux facteurs admissibles terminent à la même position, on a forcément une périodicité qui implique la présence d'autres facteurs admissibles. Par symétrie, le résultat précédent s'applique également aux paires de facteurs admissibles débutant à la même position. On a vu précédemment que les seules situations qui font monter la complexité de l'Algorithme 7 à $\mathcal{O}(n^2)$ étaient justement les cas où un grand nombre de facteurs admissibles se terminent tous à une position p donnée et un grand nombre débutent à position $p+1$.

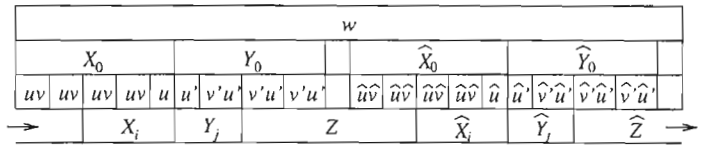
Définition 41. Soit w un mot de contour et $A = w[l, \ldots, p]$ (resp. $A = w[p, \ldots, l]$) un facteur admissible ayant la période $m \leq \frac{|A|}{2}$ tel que tous les $A_i = w[l + im, \ldots, p]$ (resp. les $A_i = w[p, \ldots, l + im]$) pour $0 \leq i \leq \left\lfloor \frac{p-l}{m} \right\rfloor$ sont admissibles. On appelle la suite $(A_i)_{0 \leq i \leq \lfloor \frac{p-l}{m} \rfloor}$ une *suite de facteurs admissibles de période m terminant (resp. débutant) à la position p de w.*

Par exemple, le polyomino illustré à la Figure 4.10 est codé par le mot $w = 0^8\, 1^7\, 0\, 1\, \bar{0}^7\, \bar{1}^8$ en débutant au point inférieur droit. Les facteurs admissibles représentés par des lignes pointillées forment une suite de facteurs admissibles de période 1 terminant à la position $p = 8$, alors que ceux représentés par des lignes en tirets forment une suite de facteurs admissibles de période 1 débutant à la position 9.

Le lemme suivant montre que dans une telle situation, il est inutile de tester toutes les paires de facteurs admissibles.

Lemme 7. *Soit w un mot de contour avec $(X_i)_{0 \leq i \leq k}$ une suite de facteurs admissibles de période m terminant à la position p et $(Y_i)_{0 \leq i \leq k'}$ une suite de facteurs admissibles de période m' débutant à la position $p+1$. Alors pour toute paire i, j telle que $w \equiv X_i Y_j Z \widehat{X_i} \widehat{Y_j} \widehat{Z}$, on a forcément* $\min(im, jm') < m + m' - \operatorname{pgcd}(m, m')$.

Preuve. Soient i, j tels que $w \equiv X_i Y_j Z \widehat{X_i} \widehat{Y_j} \widehat{Z}$ pour un certain mot Z. Par le Lemme 6 il existe $u, v \in \Sigma^*$ avec $|uv| = m$ tels que $X_0 = (uv)^k u$ et $X_i = (uv)^{k-i} u$. De même, il existe $u', v' \in \Sigma^*$ avec $|u'v'| = m'$ tels que $Y_0 = u'(v'u')^{k'}$ et $Y_j = u'(v'u')^{k'-j}$. Comme Y_j débute à la même position que Y_0 dans w, le mot Z admet $(v'u')^j$ comme préfixe. De la même manière, X_0 et X_i terminent à la même position dans w ce qui implique que $\widehat{Z}$ admet $(uv)^i$ comme suffixe. La figure ci-dessous illustre le cas où $k = 4$, $i = 2$, $k' = 3$, $j = 2$.



On a donc que Z admet $\widehat{(uv)^i}$ et $(v'u')^j$ comme préfixes. Soit x le préfixe de Z tel que $|x| = \min(im, jm')$. Le mot x possède les deux périodes m et m'. Le théorème de Fine et Wilf (Théorème 1, Section 1.2) stipule que si $|\alpha| \geq m + m' - \text{pgcd}(m, m')$ alors α possède la période $\text{pgcd}(m, m')$. Par contradiction, supposons que c'est le cas. Il existe alors un mot α de taille $\text{pgcd}(m, m')$ tel que $\widehat{uv}, v'u' \in \{\alpha\}^+$. Ainsi, $uv \in \{\widehat{\alpha}\}^+$ et il existe donc β, un conjugué de $\widehat{\alpha}$, tel que $vu \in \{\beta\}^+$. On a alors que $\beta\alpha$ est facteur de w car X_0 admet β comme suffixe alors que Y_0 admet α comme préfixe. En considérant les vecteurs associés, on obtient

$$\overrightarrow{\beta\alpha} = \overrightarrow{\beta} + \overrightarrow{\alpha} = \overrightarrow{\widehat{\alpha}} + \overrightarrow{\alpha} = -\overrightarrow{\alpha} + \overrightarrow{\alpha} = \overrightarrow{0}.$$

Le chemin de contour w contient donc une boucle fermée. Contradiction. ∎

Ceci permet encore une fois d'éviter du travail inutile et d'améliorer la performance de l'algorithme. Le lemme suivant établit une borne supérieure au travail nécessaire dans un tel cas.

Lemme 8. *Soit w un mot de contour de longueur n avec $(X_i)_{0 \leq i \leq k}$ une suite de facteurs admissibles de période m terminant à la position p et $(Y_j)_{0 \leq j \leq k'}$ une suite de facteurs admissibles de période m' débutant à la position $p + 1$. Vérifier s'il existe un couple i, j tel que $w \equiv X_i Y_j Z \widehat{X_i} \widehat{Y_j} \widehat{Z}$ requiert au plus $\mathcal{O}(n)$ opérations.*

Preuve. Le calcul de la plus longue extension en temps constant permet de tester chaque paire X_i, Y_j en temps constant. Le Lemme 7 montre qu'il ne faut considérer que les couples (i, j)

appartenant à l'ensemble suivant :

$$\mathcal{I} = \{(i,j) \mid 0 \leq i \leq k, 0 \leq j \leq k' \text{ et } \min(im, jm') < m + m' - \text{pgcd}(m,m')\}.$$

Clairement, plus les valeurs de m et m' sont rapprochés, plus la cardinalité de $\mathcal{I}$ est petite. Au pire cas $m = 1$ (ou, de manière équivalente, $m' = 1$) et alors

$$\begin{aligned} \min(im, jm') < m + m' - \text{pgcd}(m,m') &\implies \min(i, jm') < m', \\ &\implies \begin{cases} j = 0 \text{ et } 0 \leq i \leq k \\ \text{ou} \\ 1 \leq j \leq k' \text{ et } 0 \leq i < m'. \end{cases} \end{aligned}$$

Ainsi le nombre de tests nécessaires est bornée par $(k+1) + k'm'$. Comme $|X_0| \geq km$ et $|Y_0| \geq k'm'$ on a que $km, k'm' \in \mathcal{O}(n)$. On obtient ainsi la borne $\mathcal{O}((k + k'm') = \mathcal{O}(n)$. ∎

Maintenant, on peut borner le nombre de suite de facteurs admissibles périodiques terminant (ou débutant) en une position donnée, ceci permet d'établir une borne concrète au travail nécessaire afin de déterminer si un polyomino pave le plan.

Définition 42. Soit w un mot de contour avec $(A_i)_{0 \leq i \leq k}$, une suite de facteurs admissibles de période m terminant (resp. débutant) à la position p. $(A_i)_{0 \leq i \leq k}$ est dite *maximale* si

(i) Soit j la position de w telle que $A_0 = w[l, \ldots, p]$ (resp. $A_0 = w[p, \ldots, l]$) alors $w[l - m, \ldots, p]$ (resp. $w[p, \ldots, l + m]$) n'est pas un facteur admissible.

(ii) Pour toute période m' de A_0, si $m' \neq m$ alors m' ne divise pas m.

Le lemme suivant assure qu'il n'est nécessaire de considérer que les suites maximales.

Lemme 9. *Soit w un mot de contour, toute suite de facteurs admissibles périodiques qui termine (resp. commence) en une position donnée de w est incluse dans une suite maximale qui termine (resp. commence) à la même position.*

Preuve. Soit $(A_i)_{0 \leq i \leq k}$ une suites de facteurs admissibles de période m terminant à la position p de w. Soit μ la plus petite période de A_0 telle que μ divise m, et soit l tel que $A_0 = w[l, \ldots, p]$. On pose

$$\nu = \max\{\nu \in \mathbb{N} \mid \forall\, 0 \leq i \leq \nu \text{ le facteur } w[l - i\mu, \ldots, p] \text{ est admissible}\}.$$

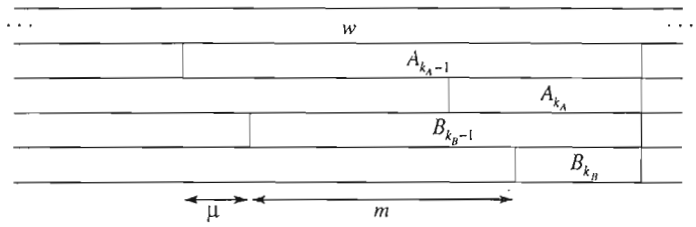
On pose $B_i = w[l + (i - \nu)\mu, \ldots, p]$ pour chaque $0 \leq i \leq \left\lfloor \frac{p-l}{\mu} \right\rfloor + \nu$. Par construction et le Lemme 6 (ii), la suite $(B_i)_{0 \leq i \leq \left\lfloor \frac{p-l}{\mu} \right\rfloor + \nu}$ forme une suite maximale de facteurs admissibles périodiques qui comprend tous les facteurs A_i pour $0 \leq i \leq k$. ∎

L'intérêt de ne considérer que les suites maximales est qu'étant donné une position dans un mot de contour, on peut fixer une borne logarithmique au nombre de suites de facteurs admissibles maximales débutant ou terminant en cette position.

Lemme 10. *Soit w un mot de contour de longueur n. Le nombre de suites maximales de facteur admissibles périodiques terminant (resp. débutant) en une position p est dans $\mathcal{O}(\log(n))$.*

Preuve. Soit $(A_i)_{0 \leq i \leq k_A}$ une suite de facteurs admissibles de période m_A et $(B_i)_{0 \leq i \leq k_B}$ une suite de facteurs admissibles de période $m_B \leq m_A$ toutes deux maximales et se terminant à la position p de w. Comme le mot w est de longueur n, il suffit de voir que $m_A \geq 2m_B$ pour obtenir la borne $\log_2(n)$.

Par contradiction, considérons premièrement le cas $m_A = m_B$. Par définition d'une suite de facteurs admissibles périodiques on a $|A_{k_A}| \leq m_A$ et $|B_{k_B}| \leq m_B$. Sans perte de généralité, on suppose que $|A_{k_A}| > |B_{k_B}|$ (il n'est pas nécessaire de considérer le cas où $|A_{k_A}| = |B_{k_B}|$ puisqu'on aurait alors $A_i = B_i$ pour tout i). On pose $\mu = |A_{k_A}| - |B_{k_B}| = |A_{k_A-1}| - |B_{k_B-1}|$.

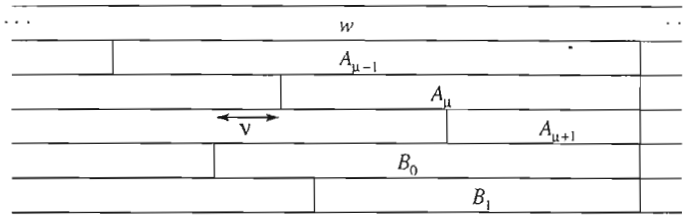


Par le Lemme 6, le mot A_{k_A-1} possède la période μ et $|A_{k_A-1}| = m + |A_{k_A}| \geq m + \mu$. Par le théorème de Fine et Wilf (Théorème 1, Section 1.2) A_{k-1} possède la période $\nu = \text{pgcd}(m, \mu) < m$. Encore une fois par le théorème de Fine et Wilf, on conclut que A_0 possède la période $\nu \neq \mu$ qui divise μ, ce qui contredit l'hypothèse de maximalité de la suite $(A_i)_{0 \leq i \leq k_A}$.

Supposons maintenant que $m' < m < 2m$. Il y a deux cas à considérer.

Si $|A_0| \leq |B_0|$ alors A_0 possède les périodes m et m' puisqu'il est un suffixe de B_0. Par définition, $|A_0| \geq 2m > m + m'$ et par le théorème de Fine et Wilf, A_0 possède la période $\text{pgcd}(m, m')$. Comme $m' < m$, le $\text{pgcd}(m, m') < m$ ce qui contredit la maximalité de la suite $(A_i)_{0 \leq i \leq k}$.

Si $|A_0| > |B_0|$ alors on pose μ comme étant l'entier qui minimise $\nu = ||A_\mu| - |B_0||$. On a alors que $\nu \leq \frac{1}{2}m < m'$.



Par le Lemme 6, le mot B_0 possède les périodes m' et ν alors que $|B_0| \geq 2m' > m' + \nu$. Par le théorème de Fine et Wilf, A'_0 possède la période $\text{pgcd}(m', \nu) < m'$. Contradiction. ∎

Lorsqu'on liste l'ensemble des facteurs admissibles qui recouvrent une position donnée, il est possible que des suites de facteurs admissibles périodiques terminent en différentes positions de w. Le lemme suivant borne le nombre de positions où se terminent de telles suites.

Lemme 11. *Soit w un mot de contour de longueur n et p une position quelconque de w. Le nombre de positions où peuvent se terminer (resp. commencer) des suites maximales de facteurs admissibles périodiques dont au moins 5 facteurs recouvrent la position p de w est borné par* $\log_2(n)$.

Preuve. Soit $(A_i)_{0 \leq i \leq k_A}$ une suite de facteurs admissibles de période m_A terminant en la position $q_A \geq p$ et $(B_i)_{0 \leq i \leq k_B}$ une deuxième suite de facteurs admissibles de période m_B terminant en la position $q_B > q_A$ toutes deux maximales et contenant au moins 5 facteurs recouvrant la position p de w.

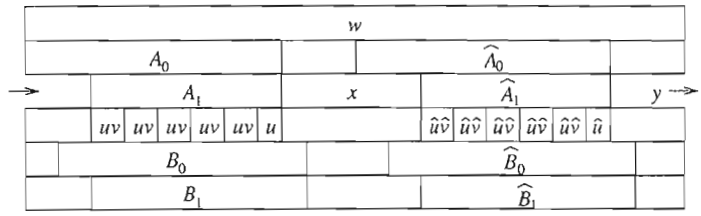
Soit j_A la position de w où débute le facteur A_0 et j_B la position où débute le facteur B_0. On a alors que

$$A_i = w[j_A + im_A, \ldots, q_A], \ \forall i \text{ tel que } 0 \leq i \leq k_A,$$

$$B_i = w[j_B + im_B, \ldots, q_B], \ \forall i \text{ tel que } 0 \leq i \leq k_B.$$

Afin d'obtenir la borne logarithmique, il suffit de voir que la période d'une des deux suites est au moins le double de celle de l'autre. Il y a deux cas à considérer.

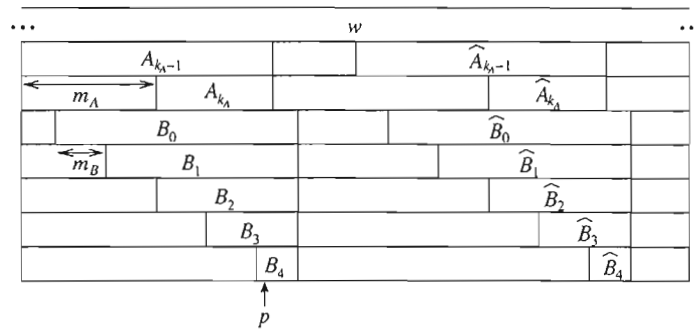
Si $j_B < j_A + m_A$, c'est-à-dire que le facteur B_0 commence avant le facteur A_1, on pose $\alpha = w[j_A + m_A, \ldots, p]$. Par construction, le facteur α est un facteur propre de A_1 et B_0, α possède donc les périodes m_A et m_B. Par hypothèse, au moins 5 des facteurs de la suite $(A_0, A_1, \ldots)$ recouvrent la position p de w et comme A_{i+1} est facteur propre de A_i, les facteurs A_0, A_1, A_2, A_3 et A_4 doivent absolument débuter avant la position p ce qui fait que $|\alpha| \geq 4m_A$. Supposons maintenant que $|\alpha| \geq m_A + m_B$. Par le théorème de Fine et Wilf (Théorème 1, Section 1.2) les mots α, A_0 et B_0 possèdent la période $\mu = \text{pgcd}(m_A, m_B)$. Il existe deux mots u, v tels que $|uv| = \mu$ et $A_i \in (uv)^* u$ tel que l'illustre la figure ci-dessous.



Soient x et y les facteurs de w tels que $w \equiv A_1 x \widehat{A_1} y$ avec $|x| = |y|$. Comme B_0 possède la période μ et par hypothèse se termine après le facteur A_1, on a $f(x) = f(v)$. De plus, comme le facteur $\widehat{B_0}$ commence avant le facteur $\widehat{A_1}$ on a également que $l(x) = l(\widehat{v})$. Ainsi, $f(x) = f(v) = \overline{l(\widehat{v})} = \overline{l(x)}$. Le facteur A_1 n'est donc pas admissible. Contradiction. On conclut que si $j_B < j_A + m_A$ alors $4m_A \leq |\alpha| < m_A + m_B$ et donc $3m_A < m_B$.

Inversement, si $j_B \geq j_A + m_A$, c'est-à-dire que le facteur B_0 commence après le facteur A_1 dans w, on pose $\alpha = w[j_B, \ldots, p]$. Comme les facteurs B_0, B_1, B_2, B_3 et B_4 recouvrent tous la position p, on a que $|\alpha| \geq 4m_B$. Si $\alpha \geq m_A + m_B$ alors par le théorème de Fine et Wilf, α, B_0 et A_0 possèdent la période $\mu = \text{pgcd}(m_A, m_B)$. Maintenant, si le facteur B_0 commence avant un facteur A_i tel que $|A_i| \geq \mu$, alors par le même argument que dans le cas précédent, on conclut que A_i n'est pas un facteur admissible car $w \equiv A_i x \widehat{A_i} y$ avec $|x| = |y|$ et $f(x) = \overline{l(x)}$. Ainsi on doit avoir $|\alpha| < m_A + m_B$ ce qui implique $4m_B < m_A + m_B$ et donc $m_A > 3m_B$.

Si, à l'opposé, il n'existe aucun facteur A_i de longueur plus grande ou égale à $\mu < m_A$ tel que B_0 commence avant A_i dans w, tel que l'illustre la figure ci-dessous,



c'est que les facteurs B_0, B_1, B_2, B_3 et B_4 commencent tous entre le début du facteur A_{k_A-1} et la position p. On conclut dans ce cas que $4m_B < 2m_A$. ∎

Finalement, certains facteurs admissibles ne font partie d'aucune suite de facteurs périodiques, on appelle ces facteurs *isolés*.

Définition 43. Soit w un mot de contour. Un facteur admissible $A = w[j, \ldots, p]$ (resp. $w[p, \ldots, j]$) est dit *isolé par rapport à la position* p s'il n'appartient à aucune suite de facteurs admissibles périodiques terminant (ou débutant) à la position p.

Le nombre de facteurs admissibles isolés est lui aussi borné par un facteur logarithmique.

Lemme 12. *Soit w un mot de contour, le nombre de facteurs admissibles isolés terminant (resp. commençant) à la position p de w est borné par* $\log_2(n)$.

Preuve. Soit $A = w[j_A, \ldots, p]$ et $B = w[j_B, \ldots, p]$ deux facteurs admissibles isolés de w. Sans perte de généralité, on suppose que $j_A < j_B$. Il suffit de voir que $|A| \geq 2|B|$ pour obtenir la borne $\log_2(n)$.

Par contradiction, supposons que $|A| \leq 2|B|$. Par le Lemme 6, A possède une période m qui divise $j_B - j_A$ telle que la suite $(A_i = w[j_A + im, \ldots, p])_{0 \leq i \leq \left\lfloor \frac{p - l_A}{m} \right\rfloor}$ est une suite de facteurs admissibles périodiques avec $A_0 = A$. Contradiction. ∎

Ceci permet enfin d'abaisser la borne quadratique de l'Algorithme 7.

Corollaire 7. *Soit w un mot de contour. Déterminer si w admet une BN-factorisation se teste en $\mathcal{O}\left(n(\log n)^3\right)$.*

Preuve. Conformément à l'Algorithme 7, on pose L_1 l'ensemble des facteurs admissibles qui recouvrent la position p de w, p étant choisi arbitrairement. Soit m la position la plus à droite où se termine un des facteurs de L_1, on pose L_2 l'ensemble des facteurs admissibles qui recouvrent la position $m+1$ de w. Pour chaque position q de w on s'intéresse aux facteurs admissibles X qui terminent en q et aux facteurs admissibles Y qui débutent en $q+1$. Ces paires X, Y sont les seules qui peuvent éventuellement mener à une factorisation $w \equiv XYZ\widehat{X}\widehat{Y}\widehat{Z}$.

Soit $\mathcal{T}_{\leq 4}$ l'ensemble des positions q telles que parmi les facteurs L_1, aucune suite maximale de facteurs admissibles périodiques de cardinalité supérieure à 4 y terminent .

Similairement, soit $\mathcal{C}_{\leq 4}$ l'ensemble des positions q telles que parmi les facteurs L_2, aucune suite maximale de facteurs admissibles périodiques de cardinalité supérieure à 4 y commencent.

Pour chaque $q \in \mathcal{T}_{\leq 4}$ (resp. $q \in \mathcal{C}_{\leq 4}$), il y a au maximun $5\log_2(n)$ facteurs de L_1 qui y terminent (resp. commencent). Effectivement, le Lemme 10, assure qu'il y a au maximum $\log_2(n)$ suites de facteurs admissibles périodiques maximales qui terminent (resp. débutent) en une position donnée et le Lemme 12 assure qu'il y a au maximum $\log_2(n)$ facteurs admissibles isolés qui terminent (resp. commencent) en une position donnée. Cela fait donc un total de $5\log_2(n)$, $4\log_2(n)$ pour les petites suites de facteurs admissibles périodiques et $\log_2(n)$ pour les facteurs admissibles isolés.

Soit $\mathcal{T}_{\geq 5}$ (resp. $\mathcal{C}_{\geq 5}$) l'ensemble des positions q telles que, parmi les facteurs L_1 (resp. L_2), au moins une suite de facteurs admissibles périodiques de cardinalité supérieure ou égale à 5 y termine (resp. commence).

Le Lemme 11 assure que la cardinalité des ensembles $\mathcal{T}_{\geq 5}$ et $\mathcal{C}_{\geq 5}$ est bornée par $\log_2(n)$.

Soit q une position de w, il y a trois cas possibles.

1. $q \in \mathcal{T}_{\leq 4}$ et $q+1 \in \mathcal{C}_{\leq 4}$. Le nombre de facteurs $X \in L_1$ qui terminent en q est borné par $5\log_2(n)$, tout comme le nombre de facteurs $Y \in L_2$ qui commencent en $q+1$. Pour chaque paire X, Y vérifier si $w \equiv XYZ\widehat{X}\widehat{Y}\widehat{Z}$ se teste en temps constant. Le temps de

traitement de chacune de ces positions est donc $\mathcal{O}\left((\log n)^2\right)$. Comme il y a $\mathcal{O}(n)$ telles positions q, le temps total pour toutes les traiter est $O\left(n(\log n)^2\right)$.

2. $q \in \mathcal{T}_{\leq 4}$ et $q+1 \in \mathcal{C}_{\geq 5}$. Comme dans le cas précédent, le nombre de facteurs de L_1 se terminant en q est borné par $5\log_2(n)$, alors que le nombre de facteurs de L_2 commençant en $q+1$ est dans $\mathcal{O}(n)$. En testant toutes les paires possibles, on obtient un temps de calcul en $\mathcal{O}(n\log n)$. Comme il y a au plus $\log(n)$ telles positions q, le temps total pour toutes les traiter est $\mathcal{O}\left(n(\log n)^2\right)$.

3. $q \in \mathcal{T}_{\geq 5}$ et $q+1 \in \mathcal{C}_{\leq 4}$. Ce cas est complètement symétrique au précédent. Le temps de traitement total est donc $\mathcal{O}\left(n(\log n)^2\right)$.

4. $q \in \mathcal{T}_{\geq 5}$ et $q+1 \in \mathcal{C}_{\geq 5}$. Soit $(X_i)_{0\leq i\leq k}$ une suite maximale de facteurs admissibles périodiques terminant en q et $(Y_i)_{0\leq i\leq k'}$ une suite maximale de facteurs admissibles périodiques commençant en $q+1$. Par le Lemme 10, le nombre de telles paires $(X_i),(Y_i)$ est borné par $(\log_2 n)^2$. Le Lemme 8 assure quant à lui que le temps de traitement pour chacune de ces paires est $\mathcal{O}(n)$. De plus, comme dans le cas précédent, il y a un maximum de $\log(n)$ facteurs admissibles supplémentaires qui terminent en q et autant qui commencent en $q+1$. Pour chacun de ces facteurs admissibles isolés il faut considérer un temps de traiment en $\mathcal{O}(n)$ puisque le nombre total de facteurs admissibles qui terminent, ou commencent, en une position est borné par $\mathcal{O}(n)$. Le temps total nécessaire pour traiter une telle position q est donc $\mathcal{O}\left(n(\log n)^2 + 2n\log n\right) = O\left(n(\log n)^2\right)$. Finalement, comme il y a au plus $\log(n)$ telles positions q, le temps total pour toutes les traiter est $\mathcal{O}\left(n(\log n)^3\right)$.

On conclut que tester s'il existe $X \in L_1$ et $Y \in L_2$ tels que $w \equiv XYZ\widehat{X}\widehat{Y}\widehat{Z}$ requiert un temps de traitement en $\mathcal{O}\left(n(\log n)^3\right)$. Finalement, il ne reste plus qu'à utiliser la même méthode pour chercher une paire $X \in L_1$ et $Y \in L_2$ tels que $w \equiv XZY\widehat{X}\widehat{Z}\widehat{Y}$. Il faut alors chercher un facteur $X \in L_1$ débutant en une position q et un facteur $Y \in L_2$ tel que son homologue $\widehat{Y}$ termine à la position $q-1$. La complexité totale d'un tel traitement est donc dans $\mathcal{O}\left(n(\log n)^3\right)$.

■

4.4 Caractérisation des doubles pseudo-carrés

On a vu précedemment qu'un polyomino exact peut admettre un nombre linéaire, en fonction de son périmètre, de factorisations de type pseudo-hexagone. Mais qu'en est-il des factorisations de type pseudo-carré ? Est-ce qu'un polyomino peut admettre plusieurs factorisations de type pseudo-carré ? Que peut-on dire de ces polyominos ? Cette section propose des problèmes ouverts portant sur ces objets géométriques très spéciaux.

On s'intéresse tout d'abord aux polyominos admettant deux factorisations de type pseudo-carré.

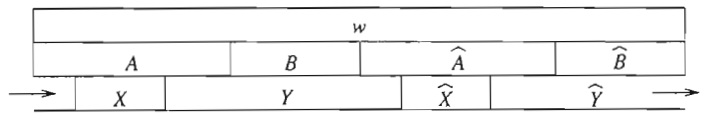
Définition 44. Soit w un mot de contour avec la BN-factorisation $w \equiv XYZ\widehat{X}\widehat{Y}\widehat{Z}$. Une deuxième BN-factorisation $w \equiv ABC\widehat{A}\widehat{B}\widehat{C}$ est dit *équivalente* si $[X, Y, Z, \widehat{X}, \widehat{Y}, \widehat{Z}]$ est une permutation circulaire de $[A, B, C, \widehat{A}, \widehat{B}, \widehat{C}]$.

Définition 45. Un *double pseudo-carré* est une polyomino dont le mot de contour admet au moins deux factorisations $XY\widehat{X}\widehat{Y}$ et $AB\widehat{A}\widehat{B}$ non-équivalentes.

De telles pièces existent, comme l'illustre la Figure 4.8. La Proposition 6 (Section 4.3) a une conséquence directe sur la structure des doubles pseudo-carrés.

Corollaire 8. *Soit w le mot de contour d'un double pseudo-carré et A, B, X, Y quatre mots tels que $w \equiv XY\widehat{X}\widehat{Y} \equiv AB\widehat{A}\widehat{B}$ forment deux factorisations non-équivalentes. Soit P_X l'ensemble des positions de w recouvertes par le facteur X et P_A celles recouvertes par le facteur A, alors $P_X \not\subset P_A$.*

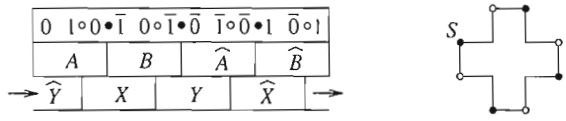
Preuve. Par contradiction, supposons que les positions recouvertes par le facteur X sont toutes recouvertes par A, on aurait alors la situation suivante :



Comme les deux factorisations ne sont pas équivalentes, il existe forcément une position p dans A qui n'est pas dans le facteur X. On a alors que les facteurs admissibles qui recouvrent

la position p et leurs homologues respectifs recouvrent toutes les positions du mot w ce qui contredit la Proposition 6. ∎

Ainsi les deux factorisations d'un double pseudo-carré doivent être décalées l'une par rapport à l'autre. Considérons par exemple le polyomino codé par le mot $w \equiv 0\,1\,0\,\bar{1}\,0\,\bar{1}\,\bar{0}\,\bar{1}\,\bar{0}\,1\,\bar{0}\,1$. Il s'agit du mot le plus court admettant deux factorisations pseudo-carrées.



Étant donné un polyomino sur la grille carrée, on peut toujours redessiner ce polyomino en utilisant comme grille de base un pavage du plan régulier par un pseudo-carré. On appelle cette opération le produit de polyominos.

Définition 46. Soit $\mathbf{P}$ un polyomino et $\mathbf{C}$ un pseudo-carré codé par le mot de contour $XY\widehat{X}\widehat{Y}$. Le *produit* de $\mathbf{P}$ par $\mathbf{C}$, noté $\mathbf{P} \circ \mathbf{C}$, est le polyomino dont le mot de contour est $\sigma(w)$ où w est le mot de contour de $\mathbf{P}$ et σ est le morphisme défini par

$$\sigma(0) = X, \qquad \sigma(1) = Y,$$
$$\sigma(\bar{0}) = \widehat{X}, \qquad \sigma(\bar{1}) = \widehat{Y}.$$

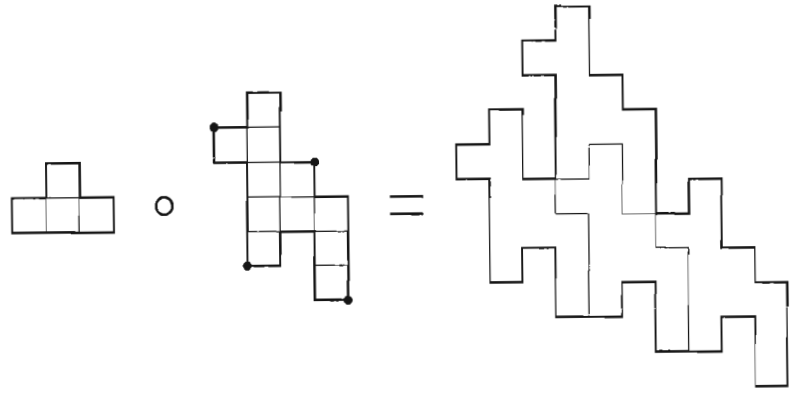


Figure 4.11 Produit du polyomino $\mathbf{P}$ par le pseudo-carré $\mathbf{C}$.

Par exemple, considérons le pseudo-carré C dont le mot de contour w_{C} est :

$$\begin{aligned} w_{\mathrm{C}} &= XY\widehat{X}\widehat{Y}, \\ &= 0\,1\,0\,\overline{1}\,\overline{1}\,0 \cdot \overline{1}\,0\,\overline{1}\,\overline{1}\,\overline{1} \cdot \overline{0}\,1\,1\,\overline{0}\,\overline{1}\,\overline{0} \cdot 1\,1\,1\,\overline{0}\,1, \end{aligned}$$

et **P** le polyomino dont le mot de contour est $w_P = 1\,0\,1\,0\,\overline{1}\,0\,\overline{1}\,\overline{0}\,0\,\overline{0}$, tels qu'illustrés à la Figure 4.11. Le produit de P par C est le polyomino défini par le mot de contour $\sigma(w_{\mathrm{P}})$ où σ est le morphisme défini par

$$\begin{aligned} \sigma(0) &= 0\,1\,0\,\overline{1}\,\overline{1}\,0, & \sigma(1) &= \overline{1}\,0\,\overline{1}\,\overline{1}\,\overline{1}, \\ \sigma(\overline{0}) &= \overline{0}\,1\,1\,\overline{0}\,\overline{1}\,\overline{0}, & \sigma(\overline{1}) &= 1\,1\,1\,\overline{0}\,1. \end{aligned}$$

Remarque 5. Le carré unitaire est un pseudo-carré dont la factorisation $X = 0, Y = 1, \widehat{X} = \overline{0}$ et $\widehat{Y} = \overline{1}$ est l'élément neutre de ce produit.

Définition 47. Un polyomino Q est dit *premier* si pour toute paire de polyominos P, C telle que $\mathrm{Q} = \mathrm{P} \circ \mathrm{C}$ on a alors que Q = P ou Q = C.

Remarque 6. Tout polyomino dont l'aire est un nombre premier est forcément premier.

Le Tableau 4.4 présente, à symétries diédrales près, la liste des premiers doubles pseudo-carrés.

La présence de deux factorisations de type pseudo-carrés est tellement contraignante qu'il est difficile d'en imaginer une troisième.

Conjecture 2. *Il n'existe aucun mot de contour admettant trois factorisations non-équivalentes de type pseudo-carré.*

De plus, Laurent Vuillon à remarqué que les côtés de tous les doubles pseudo-carrés primitifs observés jusqu'ici possèdent une *centro-symétrie* (Vuillon, 2008). Soit w un mot de contour d'un double pseudo-carré et X un des facteurs d'une de ses deux factorisations. Le segment de droite reliant le point de départ du chemin codé par X au point d'arrivée croise ce chemin exactement en son centre et les deux moitiés obtenues sont parfaitement symétriques par rapport à ce point.

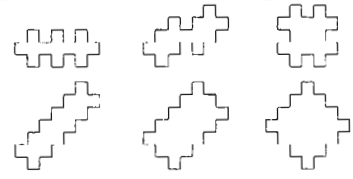
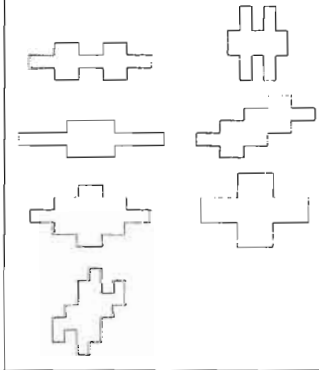
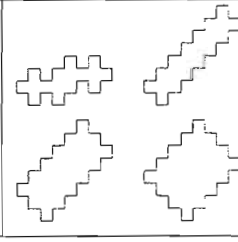
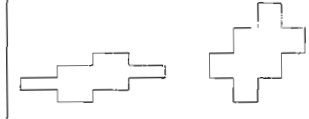
Périmètre	Premiers	Composés
12		
16		
18		
20		
24		
28		
30		
32		

Tableau 4.1 Les doubles pseudo-carrés de périmètre inférieur ou égal à 32.

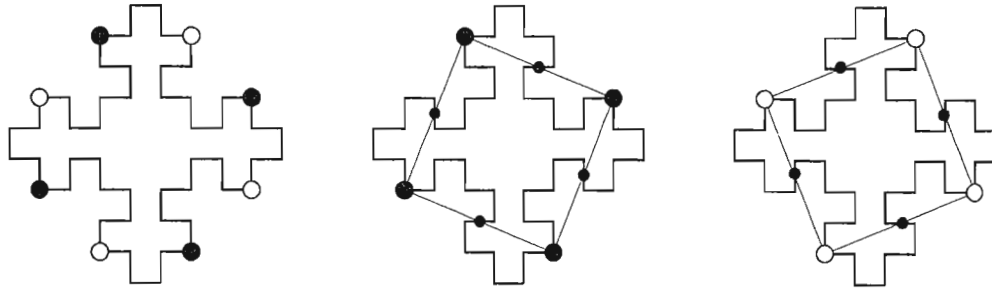


Figure 4.12 Un double pseudo-carré primitif et ses huit facteurs centro-symétriques.

La propriété géométrique de centro-symétrie se traduit sur les mots par la palindromicité, d'où la conjecture suivante.

Conjecture 3. *Soit w le mot de contour d'un double pseudo-carré premier. Si les mots X, Y sont tels que $w \equiv XY\widehat{X}\widehat{Y}$, alors $X, Y \in \mathrm{Pal}(\Sigma^*)$.*

Le fait que les facteurs X, Y sont des palindromes fait en sorte que si on pose $u = XY$ alors $w \equiv u\overline{u}$, ce qui impose également une structure très contraignante.

4.5 Polyominos avec des trous

Les résultats algorithmiques présentés précédemment peuvent s'appliquer à des tuiles plus générales que les polyominos. Étant donné que la BN-factorisation concerne uniquement le chemin codant le bord d'une pièce, on peut l'appliquer à des formes possédant des sections d'aire nulle. Considérons l'exemple présenté à la Figure 4.13. Débutant au point S, le bord de cette figure est codé par le mot

$$w = 1100100\bar{1}10\bar{1}\bar{1}\bar{1}001\bar{0}100\bar{1}\bar{1}\bar{1}\bar{0}\bar{0}\bar{1}\bar{1}\bar{1}0\bar{1}\bar{0}\bar{0}111\bar{0}\bar{1}\bar{1}0\bar{1}\bar{0}\bar{0}11\bar{0}\bar{0}.$$

Ce mot admet la BN-factorisation $w = XYZ\widehat{X}\widehat{Y}\widehat{Z}$ suivante :

$$w = 11001 \cdot 001\bar{0}110\overline{111}001\bar{0}1 \cdot 00\overline{11} \cdot \overline{10011} \cdot \overline{1}0\overline{100}111\overline{0}1\overline{1}0\overline{100} \cdot 11\overline{00}.$$

Cette factorisation correspond au pavage illustré à la Figure 4.13. Le bord d'une telle pièce correspond à un chemin qui passe deux fois par le même point mais qui ne se croise pas. Nous appellerons un *canal* une région d'aire nulle sans croisement.

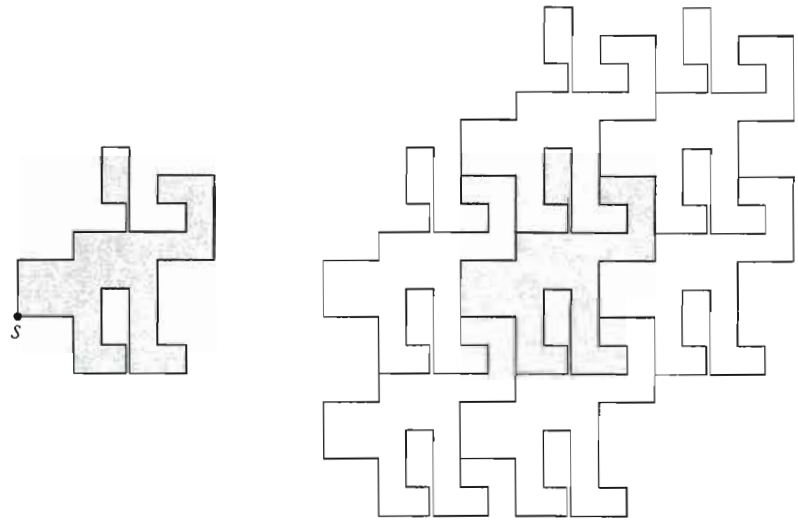


Figure 4.13 Un polyomino avec trou et un pavage du plan par cette pièce.

Définition 48. Soit $w \in \Sigma^*$ un mot codant un chemin. Un *canal* est un mot u tel que $w \equiv xuy\widehat{u}$ pour deux mots non-vides x et y satisfaisant $\overrightarrow{x} = \overrightarrow{y} = \overrightarrow{0}$ et

(i) u est maximal au sens que $f(x) \neq \overline{l(x)}$ et $f(y) \neq \overline{l(y)}$.

(ii) Soit $V = \{(l(x) \cdot f(uy)), (l(xu) \cdot f(y)), (l(y) \cdot f(\widehat{u}x)), (l(y\widehat{u}) \cdot f(x))\} \setminus \{aa | a \in \Sigma\}$ l'ensemble des virages effectués au début et à la fin de ce canal, il doit y avoir au moins un virage $V \neq \emptyset$, et ils doivent tous être du même côté $V \subset \mathcal{G}$ ou $V \subset \mathcal{D}$.

La Figure 4.14 illustre les deux types de canaux possibles. Le premier relie deux régions entre elles alors que le deuxième relie un trou à l'extérieur de la région considérée.

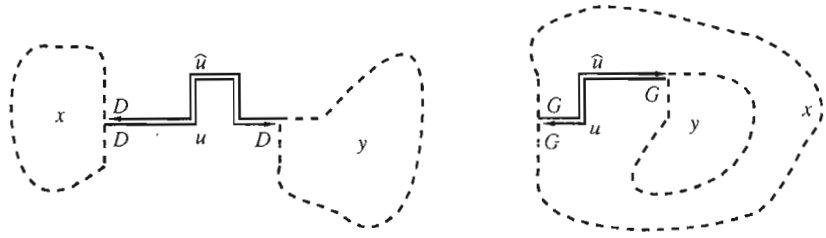


Figure 4.14 Deux types de canaux

Notons que le mot vide peut également correspondre à un canal. La Figure 4.15 en montre un exemple, en posant $u = \varepsilon$ on a bien $w \equiv xuy\widehat{u}$ avec $\overrightarrow{x} = \overrightarrow{y} = \overrightarrow{0}$ et les deux conditions de la définition sont respectées.

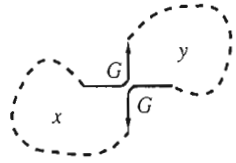


Figure 4.15 Canal défini par le mot vide.

La condition (ii) de la Définition 48 traduit le fait que deux chemins se touchent mais ne se croisent pas. La Figure 4.16 illustre trois exemples où cette contidion n'est pas respectée et à chaque fois le chemin contient forcément un croisement.

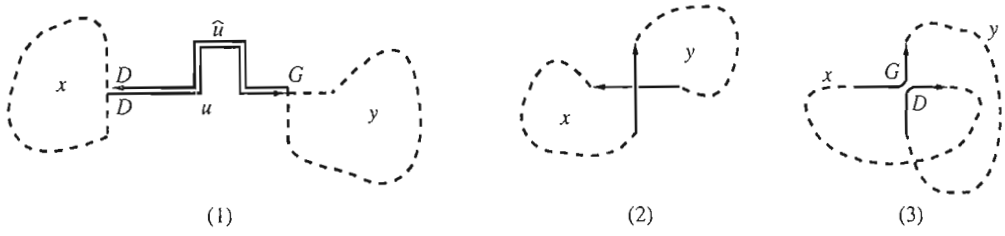


Figure 4.16 Chemins qui se croisent.

Afin d'éviter de devoir considérer des cas pathologiques, nous définissons la notion de multiplicité d'un point.

Définition 49. Soit w un mot codant un chemin débutant au point (x, y), la multiplicité d'un point (x', y') est le nombre d'entiers $1 \leq k \leq |w|$ tels que $(x, y) + \overrightarrow{w[1..k]} = (x', y')$.

La multiplicité d'un point correspond donc au nombre de fois où le chemin y passe. On peut caractériser les mots de contour de polyominos comme étant l'ensemble des mots w tels que $\overrightarrow{w} = \overrightarrow{0}$ et que tous les points visités lors du parcours du chemin codé par w sont de multiplicité un.

Nous pouvons maintenant généraliser cette notion de mot de contour à des chemins qui passent possiblement deux fois en certains points mais qui ne se croisent pas.

Définition 50. Un *mot de contour généralisé* est un mot w tel que

(i) $\overrightarrow{w} = \overrightarrow{0}$.

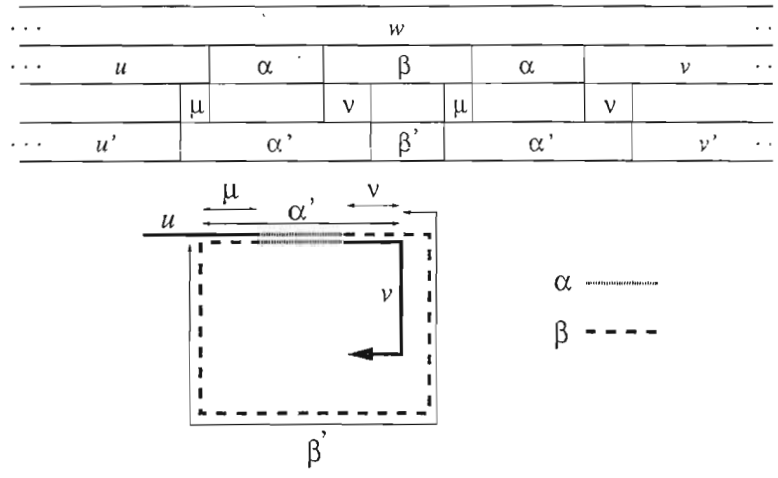
(ii) Tous les points du chemin codé par w sont de multiplicité 1 ou 2.

(iii) Tous les points de multiplicité 2 font partie de canaux.

Les mots de contour généralisés permettent à généraliser la notion de polyomino à des ensembles possédant des trous. La pièce utilisée pour paver le plan à la Figure 4.13 en est un exemple.

Lemme 13. *Un mot de contour généralisé w ne peut contenir un facteur de la forme $\alpha\beta\alpha$ avec $\overrightarrow{\alpha\beta} = \overrightarrow{0}$ et $\alpha \neq \varepsilon$.*

Preuve. Par contradiction, supposons que le mot de contour généralisé w possède un tel facteur. On peut donc écrire $w \equiv u\alpha\beta\alpha v$ pour certains mots u et v. Soit μ le plus long suffixe commun à u et β et ν le plus long préfixe commun à v et β. On pose $\alpha' = \mu\alpha\nu$ tel qu'illustré ci-dessous.



La partie du chemin codée par α' correspond à une région maximale d'aire nulle dont tous les points sont de multiplicité au moins 2. Le facteur α' doit donc former un canal, d'où $\alpha' = \widehat{\alpha'}$. Ceci est impossible car

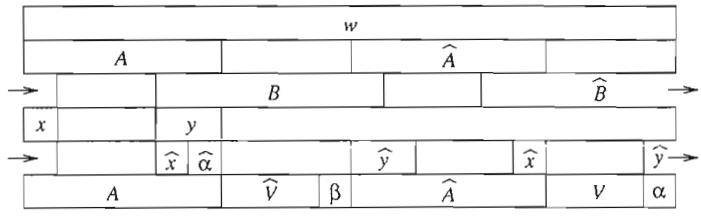
1. Si $|\alpha'|$ est pair, alors les deux lettres au centre de ce mot forment un facteur de la forme $a\overline{a}$. Contradiction.
2. Si $|\alpha'|$ est impair, alors la lettre centrale doit être égale à son propre barré. Contradiction.

■

On peut maintenant généraliser la Proposition 6 aux mots de contour généralisés.

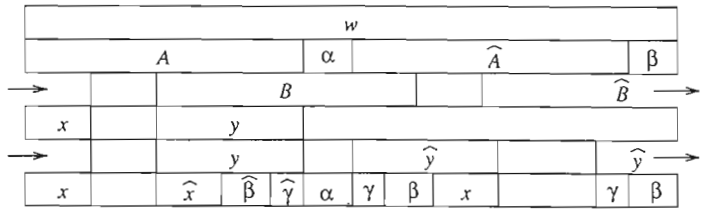
Proposition 7. *Soit $w \in \Sigma^n$ un mot de contour généralisé et $p \in \{1, 2, \ldots, n\}$. Soit $\mathcal{A}$ l'ensemble des facteurs admissibles qui recouvrent la position p et $\widehat{\mathcal{A}}$ l'ensemble de leurs homologues respectifs. Il existe au moins une position $q \in \{1, 2 \ldots, n\}$ telle que q n'est recouverte par aucun des éléments de $A \cup \widehat{A}$.*

Preuve. Il s'agit d'un prolongement de la preuve de Proposition 6 (Section 4.3) en utilisant les même notations. Le cas 1 est traité de manière identique puisque la Propriété 2 (Section 1.5.3) s'applique non seulement aux polyominos mais aussi aux chemins de contour généralisés car il s'agit d'un chemin qui ne se croise pas tel qu'établi dans l'article (Brlek, Labelle et Lacasse, 2006).



Pour le cas 2, considérons d'abord le cas où le facteur $\widehat{y}$ ne chevauche pas $\widehat{A}$ dans $\widehat{B}$, tel qu'illustré ci-dessus. Puisque par définition x est un préfixe de A, $\widehat{x}$ est un suffixe de $\widehat{A}$, ce qui implique que $\widehat{x}V\alpha x$ est un suffixe de $\widehat{B}$. Ainsi le facteur B a les deux préfixes suivants : $\widehat{x}\widehat{\alpha}\widehat{V}x$ et $\widehat{x}\widehat{\alpha}\widehat{V}\beta$. Les facteurs x et β possèdent donc un préfixe commun non-vide, appelons le u et soit v tel que $\beta\alpha = uv$. Le mot w contient donc le facteur $\beta\widehat{y} = \beta\alpha x$ qui admet uvu comme préfixe contredisant le Lemme 13 car

$$\overrightarrow{uv} = \overrightarrow{\alpha} + \overrightarrow{\beta} = \overrightarrow{0}.$$



Finalement, pour le cas où le facteur $\widehat{y}$ chevauche $\widehat{A}$ dans $\widehat{B}$ (illustré ci-dessus), on a que $\widehat{\gamma}\alpha\gamma\beta$ est une boucle fermée et que $\widehat{\gamma}\alpha\gamma\beta x$ est un facteur de w. Il suffit donc de voir que $\widehat{\gamma}$ et x ont un préfixe commun non-vide. Soit a la première lettre de A, on a alors que a est également la première lettre de x. D'un autre côté, $\overline{a}$ est la dernière lettre de $\widehat{A}$ et comme γ est un suffixe de $\widehat{A}$, $\overline{a}$ est la dernière lettre de γ. On a donc une contradiction par rapport au Lemme 13 puisque x et γ débutent par la lettre a. ∎

Il s'en suit que les algorithmes de détection des polyominos exacts présentés aux Sections 4.3.2 et 4.3.3 fonctionnent et que restent valides les bornes de complexité établies aux Corollaires 6 et 7. Cette généralisation des résultats illustre bien l'avantage d'avoir raisonné sur la structure des mots plutôt que directement sur la géométrie des objets considérés.

Étant donné un mot $w \in \{0, 1, \overline{0}, \overline{1}\}^*$, déterminer si w code un chemin de contour généralisé demande plus de travail que de tester si un mot code le contour d'un polyomino car il faut être en mesure de déterminer si les points de multiplicité 2 appartiennent à des canaux ou non. Pour ce faire, il suffit de parcourir le mot w et pour chaque point du plan (x, y) visité, d'y noter le virage correspondant. Lorsqu'on passe une deuxième fois par un point, il suffit de s'assurer que

1. le sens de parcours est inversé. Ceci permet de rejeter immédiatement le cas décrit au Lemme 13.
2. le virage effectué est du même côté (gauche ou droite) que celui effectué lors du premier passage. Il faut ensuite mémoriser ce virage jusqu'à ce que les chemins se séparent de manière à s'assurer que la condition (ii) de la Définition 48 est bien respectée.

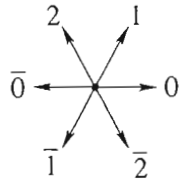
Cette vérification peut être effectuée en un temps linéaire en fonction de la longueur du mot considéré en intégrant les éléments mentionnés ci-dessus à la méthode présentée au Chapitre 2.

4.6 Grille hexagonale

Le codage de Freeman, employé pour encoder les figures géométriques, peut être utilisé sur d'autres réseaux réguliers que la grille carrée.

Sur la grille hexagonale, les polyominos sont encodés sur un alphabet à six lettres. Nous utilisons l'alphabet $\Sigma = \{0, 1, 2, \overline{0}, \overline{1}, \overline{2}\}$ associé aux pas unitaires (sur la grille hexagonale) $\{\rightarrow, \nearrow, \nwarrow, \leftarrow, \swarrow, \searrow\}$. Par exemple, en partant du point le plus à gauche, le bord du polyomino exact présenté à la Figure 4.17 est codé par le mot

$$w = 10\overline{2}010\overline{2}010\overline{2}\overline{1}0\overline{1}\overline{2}0\overline{2}\overline{1}0\overline{2}0\overline{1}\overline{0}210120\overline{1}\overline{0}2\overline{0}2,$$



et sa BN-factorisation associée au pavage illustré est

$$\begin{aligned} w &= XYZ\widehat{X}\widehat{Y}\widehat{Z}, \\ &= 10\overline{2}01 \cdot 0\overline{2}010\overline{2}\overline{1}0\overline{1}\overline{2}0 \cdot \overline{2} \cdot \overline{1}\overline{0}2\overline{0}\overline{1} \cdot \overline{0}2101201\overline{0}\overline{1}\overline{0}2\overline{0} \cdot 2. \end{aligned}$$

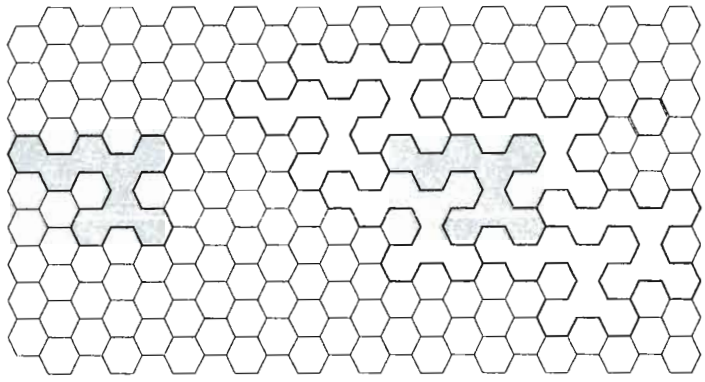


Figure 4.17 Un polyomino exact et un pavage par ce polyomino.

Tout comme dans le cas de la grille carrée, dans un mot codant un chemin sur la grille hexagonale, une lettre ne peut jamais être suivie de son barré. De plus, lorsqu'on se déplace sur la grille hexagonale, d'autres paires de lettres ne peuvent se suivre. L'ensemble des paires interdites est

$$\Phi = \{\alpha\alpha | \alpha \in \Sigma\} \cup \{\alpha\overline{\alpha} | \alpha \in \Sigma\} \cup \{02, 0\overline{1}, 1\overline{0}, 1\overline{2}, 20, 2\overline{1}, \overline{0}1, \overline{02}, \overline{1}0, \overline{1}2, \overline{2}1, \overline{20}\}.$$

De sorte que l'ensemble des chemins sur la grille hexagonale correspond aux $w \in \mathcal{P}$ où

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

Une différence notable entre la grille hexagonale et la grille carrée est établie dans (Brlek, Labelle et Lacasse, 2005; Brlek, Labelle et Lacasse, 2006) généralisant la Propriété 2 et donc par le fait même le résultat de Daurat et Nivat sur le nombre de coins *saillants* et *rentrants* d'un polyomino (Daurat et Nivat, 2003). Cette fois-ci, on définit l'ensemble des virages à gauche sur s'alphabet Σ comme l'ensemble $\mathcal{L} = \{01, 12, 2\overline{0}, \overline{01}, \overline{12}\}$ et l'ensemble des virages à droite $\mathcal{R} = \{0\overline{2}, 10, 21, \overline{0}2, \overline{10}, \overline{21}\}$.

Propriété 5 (Brlek, Labelle, Lacasse). *Soit $w \in \Sigma^*$ un mot codant un chemin fermé qui ne se croise pas sur la grille hexagonale, alors $\Delta_{\mathcal{C}}(w) = 6$.*

Cette propriété à une conséquence directe sur les pièces qui pavent le plan par translation.

Proposition 8. *Sur la grille hexagonale, une polyomino est exact si et seulement s'il est un pseudo-hexagone.*

Preuve. Puisque le théorème de Beauquier-Nivat (Théorème 9) s'applique également à la grille hexagonale, il suffit de voir qu'un polyomino ne peut admettre une BN-factorisation de type pseudo-carré. Par contradiction, supposons que le bord d'un polyomino est codé par le mot $w = XY\widehat{X}\widehat{Y}$. En appliquant la fonction $\Delta_{\mathcal{C}}$, on obtient

$$\begin{aligned}\Delta_{\mathcal{C}} w = \Delta(X) + \Delta\big(l(X)f(Y)\big) + \Delta(Y) + \Delta\big(l(Y)f(\widehat{X})\big) \\ + \Delta\big(\widehat{X}\big) + \Delta\big(l(\widehat{X})f(\widehat{Y})\big) + \Delta\big(\widehat{Y}\big) + \Delta\big(l(\widehat{Y})f(X)\big), \\ \leq 4.\end{aligned}$$

Ce qui contredit la Propriété 5. ∎

Du reste, tous les autres résultats énoncés au cours de ce chapitre ainsi que leurs démonstrations s'appliquent directement à la grille hexagonale. Notons que la définition d'un facteur admissible ne tient pas compte de la cardinalité de l'alphabet, uniquement de la relation entre une lettre et son barré. La Proposition 6 tient toujours, la seule différence est que dans la démonstration au

lieu de citer la Propriété 2, il faut employer la version qui s'applique à la grille hexagonale, soit la Propriété 5.

Il s'ensuit que les algorithmes de détection des pseudo-hexagones présentés aux Sections 4.3.2 et 4.3.3 fonctionnent et que restent valides les bornes de complexité établies aux Corollaires 6 et 7. La généralisation aux chemins de contour généralisés s'effectue tout aussi bien sur la grille hexagonale. Cette transposition des résultats à une géométrie différente illustre encore une fois l'avantage d'avoir raisonné sur la structure des mots plutôt que sur les objets eux-mêmes. On peut ainsi réutiliser directement les résultats établis.

4.7 Passage de la grille carrée à l'hexagonale

Il existe une bijection bien connue entre la grille carrée et l'hexagonale. La Figure 4.18 illustre la transformation d'un polyomino sur la grille carrée composée en un polyomino sur la grille hexagonale. Cette transformation s'effectue en trois étapes :

1. Étirer verticalement chacune des cases en un rectangle de dimensions 2×1.
2. Décaler chacune des colonnes d'une case vers le bas.
3. Étirer horizontalement chacun des rectangles de manière à obtenir un hexagone.

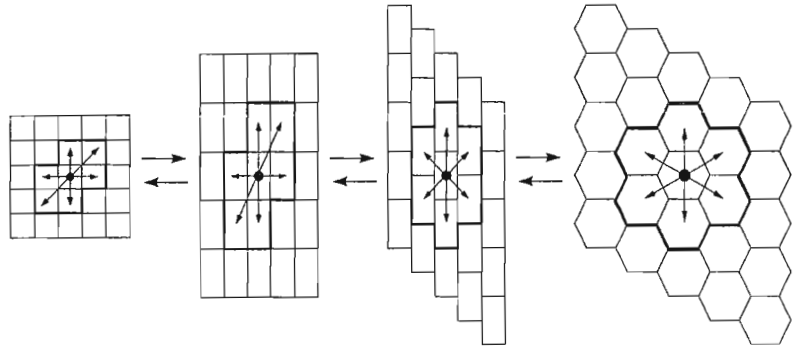


Figure 4.18 Bijection entre la grille hexagonale et la grille carrée

Il s'agit clairement d'un processus réversible qui transforme chacune des cases du réseau carré en une case du réseau hexagonal. Puisque le passage de la grille carrée à la grille hexagonale préserve la 4-connexité, l'image d'un polyomino sur la grille carrée sera toujours un polyomino

sur la grille hexagonale. Cependant, les polyominos ne sont pas préservés lors du passage de la grille hexagonale à la grille carrée, tel que l'illustre la Figure 4.19

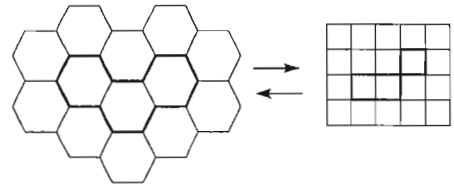


Figure 4.19 Un polyomino dont l'image n'est pas un polyomino.

D'un autre côté, les chemins de contour généralisés sont préservés lors de cette transformation du plan. Il en va donc de même avec les polyominos généralisés. La Figure 4.20 illustre la transformation du polyomino généralisé sur la grille carrée dont le mot de contour, à partir du point S, est donné par

$$w_{\square} = 0\,1\,\overline{0}\,1\,0\,\overline{1}\,0\,1\,0\,\overline{1}\,\overline{0}\,\overline{1}\,0\,\overline{1}\,0\,1\,\overline{0}\,\overline{1}\,\overline{0}\,1,$$

et le polyomino généralisé sur la grille hexagonale dont le mot de contour est

$$w_{\hexagon} = 0\,1\,2\,\overline{0}\,2\,1\,0\,\overline{2}\,\overline{1}\,2\,0\,1\,0\,\overline{2}\,\overline{1}\,\overline{0}\,\overline{1}\,2\,0\,\overline{2}\,\overline{1}\,0\,2\,1\,2\,0\,\overline{1}\,\overline{0}\,2\,1.$$

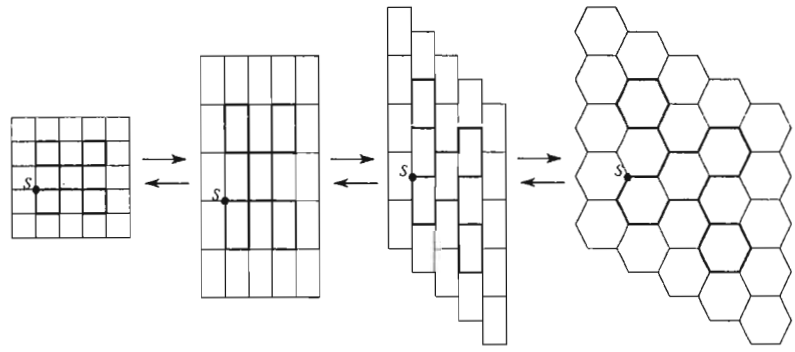


Figure 4.20 Transformation d'un polyomino généralisé de la grille carrée à la grille hexagonale.

Cette transformation des mots de contour généralisés s'effectue tout simplement par le transducteur illustré à la Figure 4.21. L'état initial est I et afin d'alléger la présentation, on suppose que le mot lu débute par les lettres 0 ou $\overline{0}$. On remarque que ce transducteur réécrit exactement le

même mot en y ajoutant, aux bons endroits, les lettres 2 et $\overline{2}$. Conséquemment, le passage de la grille hexagonale à la grille carrée s'effectue tout simplement en effaçant toutes les occurrences de 2 et $\overline{2}$.

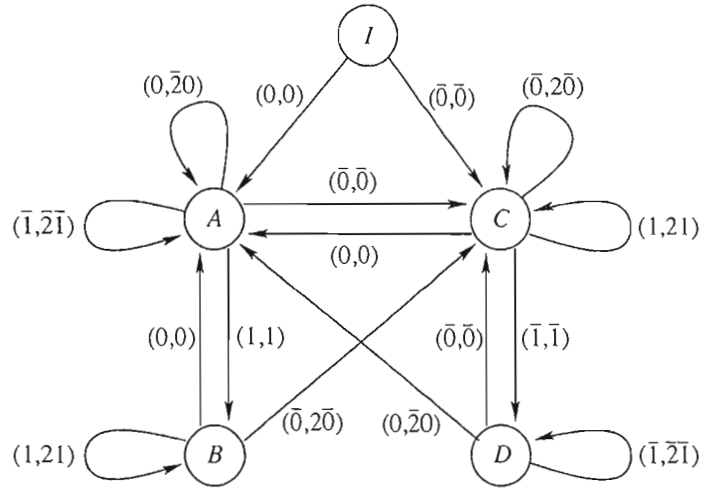


Figure 4.21 Transducteur qui traduit un chemin sur la grille carrée en un chemin sur la grille hexagonale.

CONCLUSION

Nous avons tenté au cours de cette thèse de montrer que le point de vue offert par la combinatoire des mots permet d'élaborer des solutions algorithmiques élégantes et efficaces à différents problèmes posés par la géométrie discrète. L'étude des liens entre la combinatoire des mots et la géométrie discrète apparaît prometteuse et permet d'enrichir mutuellement les deux domaines.

L'algorithme de détection des chemins auto-évitants présenté au chapitre 2 propose en fait une représentation compacte et efficace des ensembles 4-connexes. Il se prête donc à de nombreuses généralisations et applications. Par exemple, il serait intéressant de voir comment on peut l'adapter à la génération aléatoire de chemins auto-évitants, au calcul de l'enveloppe convexe d'un chemin qui se croise et à la représentation d'ensembles décrits implicitement.

Au chapitre 3, on propose un algorithme pour calculer l'enveloppe convexe d'un polyomino. Comme il l'a été mentionné, dans ce cas précis les mots ne font que masquer l'arithmétique du problème. Il serait intéressant de développer un algorithme basé principalement sur des arguments combinatoires.

Au chapitre 4, il semble possible d'abaisser encore plus la borne de complexité $\mathcal{O}\left(n(\log n)^3\right)$. Une analyse plus fine de l'algorithme suggéré devrait être possible. Notons également les deux conjectures relatives aux doubles pseudo-carrés et la possibilité de généraliser davantage la notion de *mots de contour*.

Finalement, une autre voie à explorer est le passage du plan discret $\mathbb{Z}^2$ à l'espace discret $\mathbb{Z}^3$ où les figures géométriques peuvent être décomposées en un arrangement de plan discrets. Ces plans peuvent alors être codés par des mots bidimensionnels dont la structure combinatoire reflète, encore une fois, la géométrie des objets codés. À l'instar du cas bidimensionnel, on peut alors s'intéresser aux tests de convexité discrète ainsi qu'aux problèmes de pavages de l'espace par translation.

INDEX

RÉFÉRENCES

Aloupis, G. Site web. A history of linear-time convex hull algorithms for simple polygons. Available electronically at http ://cgm.cs.mcgill.ca/~athens/cs601/.

Apostolico, A. 1985. *The myriad virtues of subword trees.* Coll. « Combinatorial algorithms on words (Maratea, 1984) ». T. 12, série *NATO Adv. Sci. Inst. Ser. F Comput. Systems Sci.*, p. 85–96. Berlin : Springer.

Beauquier, D. et M. Nivat. 1991. « On translating one polyomino to tile the plane », *Discrete Comput. Geom.*, vol. 6, p. 575–592.

Beauquier, D., M. Nivat, É. Remila, et M. Robson. 1995. « Tiling figures of the plane with two bars », *Comput. Geom.*, vol. 5, no. 1, p. 1–25.

Berger, R. 1966. « The undecidability of the domino problem », *Mem. Amer. Math. Soc.*, vol. 66.

Berstel, J. et A. de Luca. 1997. « Sturmian words, Lyndon words and trees », *Theoret. Comput. Sci.*, vol. 178, no. 1-2, p. 171–203.

Berstel, J., A. Lauve, C. Reutenauer, et F. Saliola. 2008. *Combinatorics on Words.* À paraître dans la série CRM proceedings and Lecture Notes.

Borel, J.-P. et F. Laubie. 1993. « Quelques mots sur la droite projective réelle », *J. Théor. Nombres Bordeaux*, vol. 5, no. 1, p. 23–51.

Brlek, S., J.-M. Fédou, et X. Provençal. 2008. « On the tiling by translation problem », *Discrete Applied Math.* Accepté.

Brlek, S., M. Koskas, et X. Provençal. 2008. « A linear time algorithm for detecting path intersection ». Article soumis.

Brlek, S., G. Labelle, et A. Lacasse. 2005. « A note on a result of Daurat and Nivat ». In de Felice, C. et A. Restivo, éditeurs, *Proc. DLT 2005, 9-th International Conference on Developments in Language Theory*. Coll. « LNCS », no 3572, p. 189–198, Palermo, Italia. Springer-Verlag.

———. 2006. « Properties of the contour path of discrete sets », *Int. J. Found. Comput. Sci.*, vol. 17, no. 3, p. 543–556.

Brlek, S., J.-O. Lachaud, et X. Provençal. 2008. « Combinatorial view of convexity. ». In Coeurjolly, D., I. Sivignon, L. Tougne, et F. Dupont, éditeurs, *Discrete Geometry for Computer Imagery, 14th International Conference, DGCI 2008, Lyon, France, April 16-18, 2008. Proceedings.* T. 4992, série *Lecture Notes in Computer Science*, p. 57–68. Springer.

Brlek, S., J.-O. Lachaud, X. Provençal, et C. Reutenauer. 2008. « Lyndon + Christoffel = convex », *Pattern Recognition*. Accepté.

Brlek, S. et X. Provençal. 2006a. « A fast algorithm for detecting pseudo-hexagons ». In *Proceedings of the International school and conference on Combinatorics, Automata and Number Theory*, Liège, Belgique. Université de Liège.

———. 2006b. « On the problem of deciding if a polyomino tiles the plane by translation ». In Holub, J. et J. Žďárek, éditeurs, *Proceedings of the Prague Stringology Conference '06*. Coll. « ISBN80-01-03533-6 », p. 65–76, Prague, Czech Republic. Czech Technical University in Prague.

———. 2006c. « An optimal algorithm for detecting pseudo-squares. ». In Kuba, A., L. G. Nyúl, et K. Palágyi, éditeurs, *Discrete Geometry for Computer Imagery. 13th International Conference, DGCI 2006, Szeged, Hungary, October 25-27, 2006, Proceedings*. T. 4245, série *Lecture Notes in Computer Science*, p. 403–412. Springer.

Chen, K.-T., R. H. Fox, et R. C. Lyndon. 1958. « Free differential calculus. IV. The quotient groups of the lower central series », *Ann. of Math. (2)*, vol. 68, p. 81–95.

Christoffel, E. B. 1875. « Observatio arithmetica », *Annali di Mathematica*, vol. 6, p. 145–152.

Daurat, A. et M. Nivat. 2003. « Salient and reentrant points of discrete sets ». In del Lungo, A., V. di Gesu, et A. Kuba, éditeurs, *Proc. IWCIA'03, International Workshop on Combinatorial Image Analysis*. Coll. « Electronic Notes in Discrete Mathematics », Palermo, Italia. Elsevier Science.

Debled-Rennesson, I., J.-L. Rémy, et J. Rouyer-Degli. 2003. « Detection of the discrete convexity of polyominoes », *Discrete Appl. Math.*, vol. 125, no. 1, p. 115–133. 9th International Conference on Discrete Geometry for Computer Imagery (DGCI 2000) (Uppsala).

Duval, J.-P. 1980. « Mots de Lyndon et périodicité », *RAIRO Inform. Théor.*, vol. 14, no. 2, p. 181–191.

———. 1983. « Factorizing words over an ordered alphabet », *J. Algorithms*, vol. 4, no. 4, p. 363–381.

Eckhardt, U. 2001. « Digital lines and digital convexity », *Digital and image geometry : advanced lectures*, p. 209–228.

Ehrenfeucht, A., J. Haemer, et D. Haussler. 1987. « Quasi-monotonic sequences : theory, algorithms and applications », *SIAM J. Algebraic Discrete Methods*, vol. 8, no. 3, p. 410–429.

Fine, N. J. et H. S. Wilf. 1965. « Uniqueness theorems for periodic functions », *Proc. Amer. Math. Soc.*, vol. 16, p. 109–114.

Fredricksen, H. et J. Maiorana. 1978. « Necklaces of beads in k colors and k-ary de Bruijn

sequences », *Discrete Math.*, vol. 23, no. 3, p. 207–210.

Freeman, H. 1961. « On the encoding of arbitrary geometric configurations », *IRE Trans. Electronic Computer*, vol. 10, p. 260–268.

———. 1970. « Boundary encoding and processing ». In Lipkin, B. et A. Rosenfeld, éditeurs, *Picture Processing and Psychopictorics*, p. 241–266. Academic Press, New York.

Gambini, I. et L. Vuillon. 2003. An algorithm for deciding if a polyomino tiles the plane by translations. Rapport, LAMA.

Gardner, M. 1958. « Mathematical games », *Scientific American*, p. Sept. 182–192, Nov. 136–142.

Giancarlo, R. et F. Mignosi. 1994. *Generalizations of the periodicity theorem of Fine and Wilf.* Coll. « Trees in algebra and programming—CAAP '94 (Edinburgh, 1994) ». T. 787, série *Lecture Notes in Comput. Sci.*, p. 130–141. Berlin : Springer.

Giegerich, R. et S. Kurtz. 1997. « From Ukkonen to McCreight and Weiner : a unifying view of linear-time suffix tree construction », *Algorithmica*, vol. 19, no. 3, p. 331–353.

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

Gurevich, Y. et I. Koriakov. 1972. « A remark on Berger's paper on the domino problem », *Siberian Journal of Mathematics*, vol. 13, p. 459–463. (in Russian).

Gusfield, D. 1997. *Algorithms on strings, trees, and sequences.* Cambridge : Cambridge University Press. Computer science and computational biology.

Harel, D. et R. E. Tarjan. 1984. « Fast algorithms for finding nearest common ancestors », *SIAM J. Comput.*, vol. 13, no. 2, p. 338–355.

Ilie, L. 2005. « A note on the number of distinct squares in a word ». In Brlek, S. et C. Reutenauer, éditeurs, *Proc. Words2005, 5-th International Conference on Words.* T. 36, p. 289–294, Montreal, Canada. Publications du LaCIM.

Kenyon, C. et R. Kenyon. 1992. « Tiling a polygon with rectangles ». In *IEEE Symposium on Foundations of Computer Science*, p. 610–619.

Kim, C. 1981. « On the cellular convexity of complexes », *Pattern Analysis and Machine Intelligence*, vol. 3, no. 6, p. 617–625.

———. 1982. « Digital convexity, straightness, and convex polygons », *Pattern Analysis and Machine Intelligence*, vol. 4, no. 6, p. 618–626.

Kim, C. et A. Rosenfeld. 1982. « Digital straight lines and convexity of digital regions », *Pattern Analysis and Machine Intelligence*, vol. 4, no. 2, p. 149–153.

Klette, R. et A. Rosenfeld. 2004. « Digital straightness—a review », *Discrete Appl. Math.*,

vol. 139, no. 1-3, p. 197–230.

Lothaire, M. 1997. *Combinatorics on words*. Coll. « Cambridge Mathematical Library ». Cambridge : Cambridge University Press. Première édition chez Addison-Wesley, 1983.

———. 2002. *Algebraic combinatorics on words*. T. 90, série *Encyclopedia of Mathematics and its Applications*. Cambridge : Cambridge University Press.

———. 2005. *Applied combinatorics on words*. T. 105, série *Encyclopedia of Mathematics and its Applications*. Cambridge : Cambridge University Press.

Lyndon, R. C. 1954. « On Burnside's problem », *Trans. Amer. Math. Soc.*, vol. 77, p. 202–215.

———. 1955. « On Burnside's problem. II », *Trans. Amer. Math. Soc.*, vol. 78, p. 329–332.

McCallum, D. et D. Avis. 1979. « A linear algorithm for finding the convex hull of a simple polygon », *Information Processing Letters*, vol. 9, p. 201–206.

McCreight, E. M. 1976. « A space-economical suffix tree construction algorithm », *J. Assoc. Comput. Mach.*, vol. 23, no. 2, p. 262–272.

Melkman, A. A. 1987. « On-line construction of the convex hull of a simple polyline », *Inf. Process. Lett.*, vol. 25, no. 1, p. 11–12.

Minsky, M. et S. Papert. 1969. *Perceptrons : An Introduction to Computational Geometry*. Cambridge, Mass. : MIT Press.

Reveillès, J.-P. 1991. « Géométrie discrète, calcul en nombres entiers et algorithmique ». Thèse de Doctorat, Université Louis Pasteur, Strasbourg.

Ronse, C. 1985. « Definition of convexity and convex hull in digital images », *Bulletin de la Société Mathématique de Belgique*, no. 2, p. 71–85.

Schieber, B. et U. Vishkin. 1988. « On finding lowest common ancestors : simplification and parallelization », *SIAM J. Comput.*, vol. 17, no. 6, p. 1253–1262.

Sklansky, J. 1970. « Recognition of convex blobs », *Pattern Recognition*, vol. 2, no. 1, p. 3–10.

Spitzer, F. 1956. « A combinatorial lemma and its application to probability theory », *Trans. Amer. Math. Soc.*, vol. 82, p. 323–339.

Thiant, N. 2003. « An $O(n \log n)$-algorithm for finding a domino tiling of a plane picture whose number of holes is bounded », *Theoret. Comput. Sci.*, vol. 303, no. 2-3, p. 353–374. Tilings of the plane.

Ukkonen, E. 1995. « On-line construction of suffix trees », *Algorithmica*, vol. 14, no. 3, p. 249–260.

Viennot, G. 1978. *Algèbres de Lie libres et monoïdes libres*. T. 691, série *Lecture Notes in Mathematics*. Berlin : Springer. Bases des algèbres de Lie libres et factorisations des monoïdes libres.

Vuillon, L. 2008. Communication privée.

Wang, H. 1961. « Proving theorems by pattern recognition - II », *Bell System Technical Journal*, vol. 40, p. 1–41.

Weiner, P. 1973. *Linear pattern matching algorithms*. Coll. « 14th Annual IEEE Symposium on Switching and Automata Theory (Univ. Iowa, Iowa City, Iowa, 1973) », p. 1–11. IEEE Comput. Soc., Northridge, Calif.

Wijshoff, H. A. G. et J. van Leeuven. 1984. « Arbitrary versus periodic storage schemes and tesselations of the plane using one type of polyomino », *Inform. Control*, vol. 62, p. 1–25.