UNIVERSITÉ DU QUÉBEC À MONTRÉAL

ASPECTS ALGORITHMIQUES DE LA GÉOMÉTRIE DISCRÈTE

THÈSE

PRÉSENTÉE

COMME EXIGENCE PARTIELLE

DU DOCTORAT EN MATHÉMATIQUES

PAR

HUGO TREMBLAY

DÉCEMBRE 2016

REMERCIEMENTS

Les quatre dernières années ont été semées d'embûches, de joies, de déceptions mais, surtout, d'innombrables moments inoubliables. Le parcours du doctorant, bien qu'extrêmement stimulant et enrichissant, est long et ardu. Heureusement, j'ai pu compter sur le support inconditionnel d'amis, de collègues et de membres de ma famille qui ont tous, à leur façon, contribué à l'accomplissement de ce projet un peu fou.

D'abord, je souhaite remercier sincèrement mes directeurs, Srečko Brlek et Alexandre Blondin Massé. Srečko, merci d'avoir cru en moi, de m'avoir poussé à faire de la recherche et, surtout, de me pousser hors de ma zone de confort. Alexandre, ton écoute et ta disponibilité sont sans pareilles. Tu m'as permis de réaliser ce projet dans ma terre natale et je t'en suis très reconnaissant. Merci de m'avoir appris le métier de mathématicien et, par-dessus tout, merci pour ton amitié.

Merci à tous mes amis et collègues mathématiciens de m'avoir fait oublier les longs trajets hebdomadaires Chicoutimi-Montréal. Lionel Katshingu, mon complice en toute chose, chaque moment passé en ta compagnie fait de moi une meilleure personne. Catherine Bourbeau, ta bonne humeur, ton écoute et ton amitié sont un cadeau précieux. Marco Robado, ta passion débordante pour les mathématiques contribue à alimenter la mienne. Merci aussi à Jérôme Fortier, Sébastien Labbé, Maxime Scott et Maxime Gélinas pour le bon temps passé en votre compagnie.

Merci à tout le personnel du LaCIM qui rend ce lieu si parfait pour s'épanouir mathématiquement : Johanne Patoine, dont la bonne humeur et l'efficacité empêchent tout potentiel problème administratif. Jérôme Tremblay, dont l'humour

(douteux) et la passion pour la culture scientifique sont si propices à la procrastination, mais si enrichissants. Franco Saliola, dont l'enthousiasme et la passion contagieux me poussent à me dépasser.

Merci à toute ma famille, papa, maman et Léa pour les encouragements, le soutien et, surtout, votre amour.

Merci Marie-Philip, ma compagne de tous les instants, de toutes les émotions et de tous les rêves. Voilà plus de dix ans que tu m'offres ton support inconditionnel, malgré les longues périodes passées loin l'un de l'autre. Sache que sans toi, rien de tout cela ne serait possible.

Un chapitre important de ma vie se termine, mais un autre débute à peine. Mathilde, je t'aime plus que tout. Papa sera toujours là pour toi, quoi qu'il arrive.

TABLE DES MATIÈRES

LISTE DES FIGURES

RÉSUMÉ

Cette thèse étudie diverses notions en géométrie discrète d'un point de vue algorithmique. Plus particulièrement, elle fournit un cadre d'algorithmes efficaces afin de résoudre des problèmes concernant les figures discrètes en se basant uniquement sur l'étude du contour de ces dernières. Les résultats obtenus s'appuient principalement sur la théorie des graphes et leurs représentations, ainsi que sur la combinatoire des mots. La première théorie permet d'aborder les problèmes étudiés sous un angle géométrique, tandis que la seconde fournit un modèle simple et efficace pour le traitement et l'analyse de chemins et de figures discrètes. Les résultats développés au fil du texte appartiennent principalement aux domaines de la théorie des graphes, de la combinatoire des mots et de l'informatique théorique. En plus de ces contributions théoriques, ils trouvent aussi écho dans plusieurs autres domaines liés aux technologies de l'information, notamment en traitement d'images numériques ou, encore, en cristallographie.

Ce document s'articule autour de trois sujets principaux, tous liés de près ou de loin à la notion centrale de *polyominos*, c'est-à-dire un ensemble de pixels connexes dans le plan discret $\mathbb{Z}^2$. Dans un premier temps, nous introduisons le concept de *réseau parallélogramme*, permettant ainsi d'établir de nouveaux résultats de nature algébrique sur un type particulier de morphisme appelé *morphisme parallélogramme*. Ensuite, nous étudions la notion de *polyomino premier*. Notamment, nous fournissons un algorithme polynomial de factorisation. Finalement, nous étudions diverses opérations géométriques fondamentales sur les polyominos. Notons que tous les algorithmes présentés ont été implémentés et testés sur plusieurs exemples.

Ce travail de fin d'études constitue un apport aux mathématiques discrètes par l'établissement de nouveaux résultats. Parmi ceux-ci, mentionnons un algorithme linéaire pour le calcul des enveloppes externe et convexe de tout chemin discret de $\mathbb{Z}^2$ ainsi que des algorithmes linéaires afin de calculer l'union, l'intersection et la différence de polyominos.

MOTS-CLÉS : géométrie discrète, combinatoire des mots, topologie des graphes, algorithmique, polyominos

INTRODUCTION

La place grandissante qu'occupent l'informatique et les technologies de l'information dans toutes les sphères d'activités amène à considérer plusieurs disciplines scientifiques classiques sous un nouveau jour. Parmi celles-ci, le domaine de la géométrie est sans nul doute une branche d'étude grandement stimulée par ce nouveau paradigme. Bien qu'étudié depuis plusieurs siècles, l'avènement des écrans numériques alimente les recherches effectuées en géométrie discrète, en géométrie combinatoire et en représentation des graphes.

Au coeur de ces disciplines se trouve la notion de *pixel*[1], c'est-à-dire un carré unité du plan $\mathbb{R}^2$. Plus généralement, on considère des ensembles de pixels, aussi appelés *figures discrètes* ou *images numériques*, permettant la digitalisation d'objets continus. On utilise ensuite ces objets discrets dans un large éventail d'applications allant de la robotique (Kobilarov *et al.*, 2007) à la cristallographie (Sunada, 2013) en passant par l'analyse de signaux numériques (Du *et al.*, 2015). Il est donc nécessaire de développer des algorithmes efficaces pour le traitement d'images numériques. La plupart des opérations géométriques fondamentales sur les ensembles de pixels ont déjà été étudiées. Par exemple, des algorithmes pour le calcul de rotations, de translations, de symétries, d'unions, d'intersections, de dilatations ou de segmentations de figures discrètes sont répertoriés dans (Goodman et O'Rourke, 2004). Par ailleurs, les figures discrètes sont aussi étudiées d'un point de vue combinatoire et portent alors le nom de *polyomino*. D'abord introduits formellement par S. W. Golomb dans (Golomb, 1965) et ensuite popularisés par M. Gardner

1. de l'anglais *picture element*

dans la rubrique "Mathematical Games" du journal *Scientific American*, les polyominos constituent une classe importante d'objets combinatoires. Notamment, D. H. Redelmeier donne dans (Redelmeier, 1981) un algorithme pour dénombrer et générer les polyominos selon différents critères en temps exponentiel. Même si des algorithmes améliorés existent (Jensen et Guttmann, 2000), aucun n'est asymptotiquement plus rapide.

Par contre, peu de méthodes développées jusqu'à présent sont basées sur l'étude du contour d'images numériques à l'aide de la combinatoire des mots. Pourtant, cette approche a déjà porté fruit (Blondin Massé, 2012; Provençal, 2008), mais reste peu exploitée à ce jour. Elle s'inspire du théorème de Green qui exprime une relation entre une intégrale de contour et une intégrale double (voir (Stewart, 2005)). Autrement dit, ce théorème affirme qu'il est possible d'effectuer des traitements sur la courbe décrivant le contour d'une région au lieu de la région elle-même, ce qui est souvent moins coûteux en temps de calcul. Par ailleurs, l'efficacité d'une telle approche dans un contexte discret est démontrée dans (Brlek *et al.*, 2005b; Brlek *et al.*, 2005a) où les auteurs proposent et étudient une version du théorème de Green restreinte aux courbes discrètes. Au fil du texte, nous adaptons cette idée en associant les ensembles de pixels avec ou sans trou à leur contour afin d'établir des résultats de nature géométrique concernant les figures discrètes. Il est intéressant de noter qu'il existe une généralisation en dimension trois du théorème de Green appelée théorème de Stokes (voir (Stewart, 2005)). De plus, un version discrète de ce résultat est développée par G. Labelle et A. Lacasse dans (Labelle et Lacasse, 2009). Ceci suggère que ces idées sont potentiellement transposables au contexte de la géométrie discrète tridimensionnelle où l'élément de base est le cube de volume un de l'espace $\mathbb{R}^3$ appelé *voxel*[2].

2. de l'anglais *volume element*

Parallèlement, nous utilisons la théorie des graphes, et plus particulièrement la topologie des graphes, afin d'aborder les figures discrètes d'un point de vue topologique. À l'origine, cette théorie topologique des graphes fut développée afin d'étudier la question suivante : *Étant donné un graphe, est-il possible de le dessiner dans le plan sans croisement d'arête ?* (Gross et Tucker, 1987). De tels graphes sont dits *planaires*. Il est alors évident que tout pixel constitue un graphe planaire cyclique sur quatre sommets. Par extension, tout ensemble de pixels S est aussi un graphe planaire défini par l'union des pixels de S. De la même façon, tout chemin discret γ correspond à un graphe planaire dont les sommets sont les points de γ et les arêtes relient les sommets adjacents. Ce point de vue permet d'appliquer tous les outils de la théorie des graphes à des problèmes de géométrie discrète. En particulier, ceci établit un cadre formel pour l'étude de la géométrie discrète en précisant, par exemple, les notions de connexité, de région et d'homéomorphisme. Notons que plusieurs résultats antérieurs confirment la validité de cette approche. En effet, T. Pavlidis utilise la représentation des graphes dans (Pavlidis, 1977) afin d'effectuer de la reconnaissance de motifs. Ou encore, P. Bose, V. Dujmović, F. Hurtado et P. Morin étudient dans (Bose *et al.*, 2009) les transformations d'images numériques du point de vue de la théorie des graphes. Des exemples additionnels de cette approche sont donnés dans (Gross et Yellen, 2004; Rosenfeld et Klette, 2004). De plus, le lecteur trouvera dans (Linial *et al.*, 1995) quelques algorithmes basés sur l'étude des graphes en tant qu'objets géométriques.

Cette thèse propose donc d'étudier les opérations géométriques sur les figures discrètes en combinant deux approches théoriques. D'abord, nous utilisons la théorie des graphes et, plus particulièrement, la topologie des graphes afin de considérer les figures discrètes d'un point de vue à la fois topologique et algébrique. Ainsi, nous mettons à notre disposition tous les outils de ces théories développées depuis plus d'un siècle. Puis, la combinatoire des mots est utilisée afin d'étudier le contour des

ensembles de pixels. Ceci permet, entre autres, d'obtenir des algorithmes linéaires en la longueur du périmètre des figures pour le calcul des opérations géométriques fondamentales que sont l'union, l'intersection et la différence de figures discrètes. Les sujets abordés sont divisés en quatre chapitres.

Le chapitre 1 contient les définitions et résultats de bases utilisés dans les chapitres subséquents. Nous y présentons notamment un survol de la géométrie discrète, de la topologie des graphes et de la combinatoire des mots. La notation utilisée provient de (Rosenfeld et Klette, 2004) pour la géométrie discrète, de (Diestel, 2010) pour la théorie des graphes et de (Lothaire, 1997) pour la combinatoire des mots.

Au chapitre 2, nous introduisons et étudions la notion de *réseau parallélogramme*. Cette généralisation de réseau carré est obtenue en appliquant un morphisme dit *parallélogramme* à la grille carrée $\mathbb{Z}^2$. Le résultat principal de cette section concerne la structure de tels réseaux. En effet, nous démontrons que tout réseau parallélogramme est homéomorphe au réseau carré, c'est-à-dire que l'un est obtenu de l'autre par déformation continue. Nous utilisons ensuite ce résultat pour démontrer une conjecture de (Blondin Massé *et al.*, 2012) concernant les codes circulaires :

> Tout réseau parallélogramme est un code circulaire si et seulement si ses vecteurs directeurs sont engendrés par des mots primitifs.

Remarquons que ce type de code est lié à la notion de *codes de Goppa* (aussi appelée codes de géométrie algébrique), construits à partir d'une courbe algébrique sur un corps fini (Goppa, 1977; Høholdt *et al.*, 1998). Un cas particulier des codes de Goppa, appelé *code binaire de Goppa*, est entre autre utilisé dans le système de chiffrement de McEliece (McEliece, 1978). Les résultats de ce chapitre ont été présentés dans le cadre de la $10^{\text{ième}}$ conférence internationale sur la théorie des

langages et des automates qui a eu lieu à Prague en 2016 (Blondin Massé *et al.*, 2016).

Le chapitre 3 poursuit l'étude des morphismes parallélogrammes en considérant un type de figures discrètes engendrées par ceux-ci, soit la notion de *polyomino premier*. D'abord introduit dans (Provençal, 2008), ce type particulier de figure discrète constitue en quelque sorte une réalisation géométrique de la notion de nombre premier. Après avoir prouvé l'existence d'une factorisation en polyominos premiers pour toute figure discrète, nous donnons un algorithme polynomial afin d'effectuer cette décomposition. Finalement, nous discutons brièvement d'une conjecture concernant l'unicité de la décomposition en polyominos premiers. Ces résultats ont été présentés lors de la $8^{\text{ième}}$ Conférence internationale sur la théorie des langages et des automates qui avait lieu à Madrid en 2014 (Blondin Massé *et al.*, 2014).

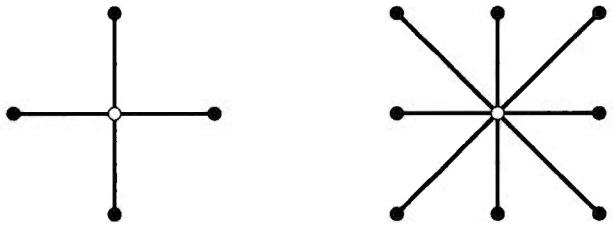
Finalement, le chapitre 4 s'attarde aux différentes opérations géométriques sur les figures discrètes. Dans un premier temps, nous généralisons à tout le plan $\mathbb{Z}^2$ une structure de données d'arbre quaternaire radix. D'abord introduite dans (Brlek *et al.*, 2011), elle permet d'accéder aux points sur le bord de figures discrètes en temps et en espace linéaire. Ensuite, cette structure est utilisée afin de développer un algorithme efficace pour le calcul de l'enveloppe externe d'images numériques. Ce faisant, nous obtenons aussi un algorithme optimal pour le calcul de l'enveloppe convexe discrète. Puis, nous étudions la notion de *graphe couvrant*, c'est-à-dire le graphe obtenu en superposant deux ou plusieurs graphes planaires. Après avoir donné un algorithme optimal pour le calculer, nous exploitons ce graphe afin d'obtenir l'union, l'intersection et la différence de figures discrètes. Ce chapitre a fait l'objet d'une publication dans la revue *Theoretical Computer Science* en 2015 (Blondin Massé *et al.*, 2015).

CHAPITRE I

DÉFINITIONS ET NOTATION

Ce chapitre contient les définitions et notation utilisées tout au long de la présente thèse. D'abord, la section 1.1 rappelle les fondements de la géométrie discrète, sujet d'étude principal de cet ouvrage. Ensuite, la section 1.2 contient un bref rappel des notions de topologie des graphes. Cette théorie fournit un cadre rigoureux afin d'étudier les différents objets digitaux d'un point de vue géométrique. Puis, la section 1.3 présente un outil particulièrement utile utilisé dans ce texte afin de décrire les figures discrètes, soit la combinatoire des mots. Enfin, la section 1.5 contient un survol des concepts importants relatifs aux pavages par translation dont les propriétés géométriques et algébriques facilitent grandement la démonstration de certains résultats.

Plusieurs références de qualité traitant des sujets énumérés plus haut sont disponibles. Parmi celles-ci, mentionnons (Rosenfeld et Klette, 2004; Provençal, 2008; Blondin Massé, 2012) pour la géométrie discrète, (Gross et Tucker, 1987; Rosenfeld et Klette, 2004; Diestel, 2010) pour la topologie des graphes, (Lothaire, 1997) pour la combinatoire des mots et (Grünbaum et Shephard, 1987; Beauquier et Nivat, 1991) pour les pavages.



(a) 4-connexité (b) 8-connexité

Figure 1.1: Deux types de connexité des points de $\mathbb{Z}^2$

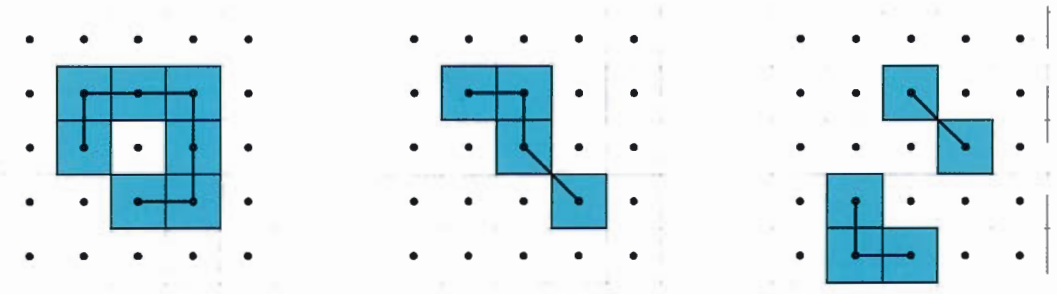
1.1 Géométrie discrète

Le *plan discret*, aussi appelé *grille discrète*, est identifié à l'ensemble $\mathbb{Z}^2 = \mathbb{Z} \times \mathbb{Z}$. Pour tout couple (a, b) de $\mathbb{Z}^2$ fixé, la *cellule*, ou encore *pixel*, $c(a, b)$ est le carré unité rectilinéaire de $\mathbb{R}^2$ dont le point au centre est (a, b). Autrement dit,

$$c(a,b) = \{(x,y) \in \mathbb{R}^2 \mid a - 1/2 \leq x \leq a + 1/2 \text{ et } b - 1/2 \leq y \leq b + 1/2\},$$

où (a, b) est dans $\mathbb{Z}^2$. On note $\mathcal{C}$ l'ensemble des cellules de $\mathbb{R}^2$. Notons qu'il est parfois important de différencier les cellules, identifiées aux points à coordonnées entières, des sommets les constituant. Conformément à la notation de (Rosenfeld et Klette, 2004), on dit qu'une cellule $c(a, b)$ est un carré unité identifié par un *point* de la grille discrète, c'est-à-dire son point central, et est constituée de quatre *sommets* : $(a - 1/2, b - 1/2)$, $(a + 1/2, b - 1/2)$, $(a - 1/2, b + 1/2)$ et $(a + 1/2, b + 1/2)$. Évidemment, tous les sommets des cellules de $\mathcal{C}$ vivent sur une grille carrée isométrique à $\mathbb{Z}^2$ via la translation $\vec{x} \mapsto \vec{x} + (1/2, 1/2)$.

La géométrie discrète (*digital geometry* en anglais) est la branche de la géométrie étudiant les propriétés de divers ensembles de cellules. Deux cellules données c_1 et c_2 sont dites 4-*adjacentes* (resp. 8-*adjacentes*) si et seulement si elles ont exacte-



(a) 4-connexe avec trou (b) 8-connexe sans trou (c) Non connexe

Figure 1.2: Représentation de divers ensembles de cellules

ment une arête (resp. un point) en commun. De plus, on considère qu'une cellule est 4-adjacente à elle-même et que toute paire de cellules 4-adjacentes sont aussi 8-adjacentes. La connexité des cellules en tant que points de la grille s'exprime géométriquement comme indiqué à la figure 1.1. Un ensemble fini ou infini de cellules $C \subseteq \mathcal{C}$ est dit 4-*connexe* si et seulement si, pour toute paire de cellules distinctes c et c', il existe une suite finie $\gamma = (c_1, c_2, \ldots, c_n)$ d'éléments de C telle que $c = c_1$, $c' = c_n$ et où c_i et c_{i+1} sont 4-adjacentes pour tout $1 \leq i \leq n-1$. On définit de manière analogue un ensemble 8-connexe. La suite γ est appelée *chemin discret de longueur* n. Notons que par définition, $\mathcal{C}$ est 8-connexe. Finalement, l'ensemble $C \subseteq \mathcal{C}$ est dit *sans trou* si son complément $\overline{C} = \mathcal{C} - C$ est 4-connexe et *avec trou* sinon. La figure 1.2 représente divers ensembles de cellules où l'adjacence des points correspondant aux cellules est marquée par une arête noire.

Dorénavant, un ensemble quelconque de n cellules est appelé *figure discrète d'aire* n. On définit alors un *polyomino* comme une figure discrète 4-connexe. Remarquons qu'un polyomino peut admettre un ou plusieurs trous. Dans la plupart des cas, les résultats développés dans cette thèse s'appliquent aux polyominos quelconques. Par contre, dans certains contextes, comme des problèmes de pavages (Blondin Massé, 2012), on se restreint à des polyominos sans trou. Par conséquent, nous spécifions explicitement au fil du texte si les polyominos considérés

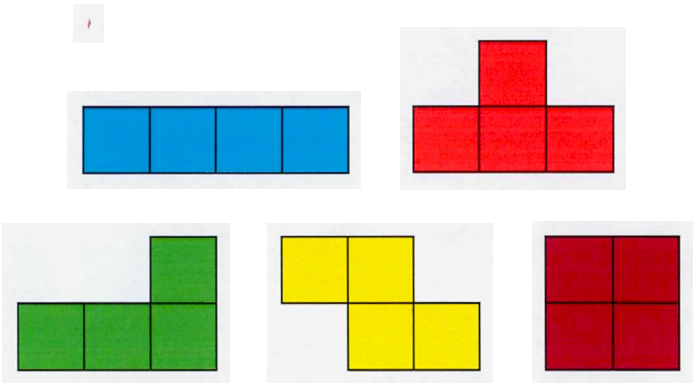


Figure 1.3: Les cinq tetrominos libres

sont non troués. Mentionnons aussi que le terme *n-omino* est employé pour décrire un polyomino d'aire n. Les préfixes *mon*, *d*, *tr*, *tetr*, *pent*, *hex*, etc. sont aussi utilisés en remplacement du nombre n. Par exemple, la figure 1.2 (a) illustre un 7-omino, ou encore heptomino.

On définit des classes d'équivalences de polyomninos à l'aide de transformations géométriques. Deux polyominos P et P' sont dits *équivalents par translation* (resp. *symétrie*, resp. *rotation*) si P' est une copie obtenue par translation (resp. *symétrie*, resp. *rotation*) de P. Les classes d'équivalences de la relation « P est équivalent par translation ou symétrie ou rotation à P' » sont appelées *polyominos libres* (voir figure 1.3).

La géométrie discrète tire son nom de la digitalisation d'objets bi ou tri-dimensionnels de l'espace euclidien. Plusieurs modèles furent proposés notamment par C. F. Gauss et par C. Jordan afin d'approximer les courbes de $\mathbb{R}^2$ et $\mathbb{R}^3$ par des figures discrètes. Étant donné un ensemble S de points de $\mathbb{R}^2$, la *digitalisation de Gauss de* S est l'ensemble

$$\textsc{Gauss}(S) = \{c(a,b) \mid (a,b) \in \mathbb{Z}^2 \cap S\}.$$

En d'autres termes, $\textsc{Gauss}(S)$ est la figure discrète formée des pixels correspon-

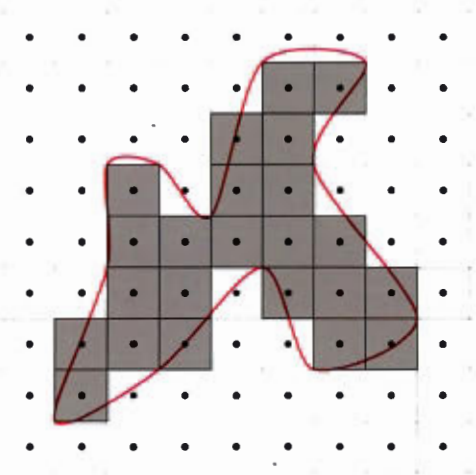


Figure 1.4: Digitalisation de Gauss du sous-ensemble de $\mathbb{R}^2$ défini par l'intérieur de la courbe rouge

dant aux points entiers de S. La figure 1.4 donne un exemple de digitalisation de Gauss d'un ensemble de $\mathbb{R}^2$.

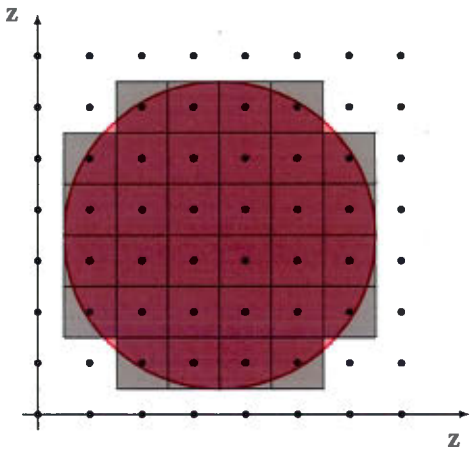
On déduit aisément que deux ensembles distincts S et S' peuvent admettre la même digitalisation de Gauss. Par exemple, considérons le disque $S = \{(x,y) \mid (x-7/2)^2+(y-7/2)^2 = 9\}$ et le disque troué $S' = S - \{(x,y) \mid (x-7/2)^2+(y-7/2)^2 = 1/4\}$. Alors, $\textsc{Gauss}(S) = \textsc{Gauss}(S')$ (voir figure 1.5).

La digitalisation de Jordan d'un ensemble $S \subseteq \mathbb{R}^2$, quant à elle, peut être calculée de deux façons : la *digitalisation interne de Jordan de* S est l'ensemble $J^-(S)$ des cellules entièrement comprises dans S et la *digitalisation externe de Jordan de* S est l'ensemble $J^+(S)$ des cellules dont l'intersection avec S est non vide. La figure 1.6 illustre ces deux parties de la digitalisation de Jordan.

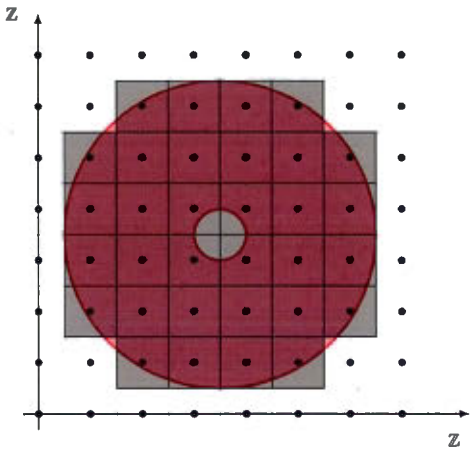
À partir des définitions précédentes, on montre la proposition 1.1.1 suivante :

Proposition 1.1.1. *Soit* $S \subseteq \mathbb{R}^2$. *Alors,*

$$J^-(S) \subseteq \textsc{Gauss}(S) \subseteq J^+(S).$$

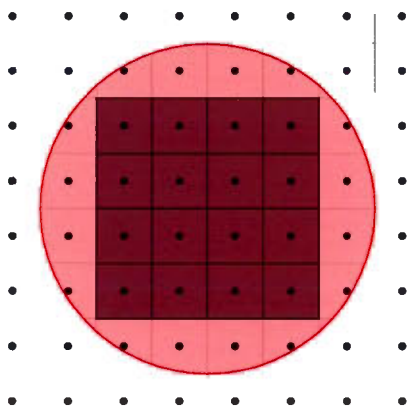


(a) Digitalisation de Gauss du disque $S = \{(x,y) \mid (x-7/2)^2 + (y-7/2)^2 = 9\}$

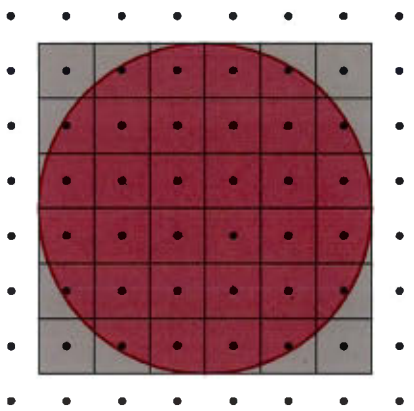


(b) Digitalisation de Gauss du disque troué $S' = S - \{(x,y) \mid (x-7/2)^2 + (y-7/2)^2 = 1/4\}$

Figure 1.5: Deux sous-ensembles de $\mathbb{R}^2$ différents admettant la même digitalisation de Gauss



(a) $J^-(S)$



(b) $J^+(S)$

Figure 1.6: Digitalisation interne et externe de Jordan d'une région $S \subseteq \mathbb{R}^2$

De plus, si S est une union de cellules, alors $J^-(S) = \text{GAUSS}(S) = J^+(S)$.

Démonstration. Soit $S \subseteq \mathbb{R}^2$ et $(a, b) \in \mathbb{Z}^2$. Par définition, $c(a, b) \in J^-(S)$ implique que le carré unité ayant (a, b) comme point milieu est entièrement inclus dans S. En particulier, (a, b) est dans S et on a bien $c(a, b) \in \text{GAUSS}(S)$. Maintenant, si $c(a, b) \in \text{GAUSS}(S)$, alors (a, b) est dans S. Par conséquent, l'intersection de $c(a, b)$ et S est non vide, c'est-à-dire que $c(a, b) \in J^+(S)$. Finalement, supposons que S soit une union de cellules. Alors, $c(a, b) \in J^+(S)$ implique $c(a, b) \in S$. Par conséquent, $c(a, b) \in J^-(S)$. De plus, (a, b) est dans S de sorte que $c(a, b) \in \text{GAUSS}(S)$. On a donc le résultat souhaité. □

1.2 Topologie des graphes

Comme mentionné à la section 1.1, toute cellule $c(a, b)$ est circonscrite par un carré unité composé de quatre sommets et d'autant d'arêtes, constituant alors un cas particulier de graphe simple. Par extension, toute figure discrète peut être vue comme un graphe simple. Cette section s'intéresse à l'aspect géométrique de la théorie des graphes, appelé *topologie des graphes*. Originalement développée afin de résoudre le problème de dessiner un graphe sur une surface donnée sans croisement, la topologie des graphes trouve de multiples applications tant en géométrie qu'en combinatoire, en topologie algébrique ou en théorie des groupes (Gross et Tucker, 1987). Nous débutons cette section par un court survol de la théorie des graphes. Le lecteur n'étant pas familier avec ces notions est invité à consulter (Diestel, 2010) pour une exposition plus détaillée de cette branche des mathématiques.

Un *graphe simple* G est un couple (V, E) où V est un ensemble de sommets et $E \subseteq V \times V$ est une relation symétrique et irréflexive sur V. On appelle *arête* tout élément de E. De plus, G est dit *fini* (resp. *infini*, *dénombrable*) si V est fini (resp. infini, dénombrable). Notons que si E est asymétrique, alors G est dit *orienté*.

Deux sommets $u, v \in V$ sont dits *adjacents* si et seulement si $\{u, v\} \in E$. Étant donné un sommet v de V, on note $\deg(v)$ le *degré de* v, c'est-à-dire le nombre de sommets adjacents à v. Soit $c = (e_1, e_2, \ldots, e_n)$ une suite de n arêtes. Alors, c est une *chaîne de longueur* n si et seulement si e_i et e_{i+1} ont exactement un sommet en commun pour tout $1 \leq i \leq n$. Toute chaîne de longueur n telle que $e_1 = e_n$ est appelée *cycle de longueur* n.

Maintenant, étant donné deux graphes $G = (V, E)$ et $H = (V', E')$, H est un *sous-graphe* de G, noté $H \subseteq G$, si $V' \subseteq V$ et $E' \subseteq E$. De plus, G et H sont dits *isomorphes* et on écrit $G \simeq H$ s'il existe une bijection $f : V \to V'$ telle que $\{u, v\} \in E$ si et seulement si $\{f(u), f(v)\} \in E'$. Il suit immédiatement de ces définitions que $G \subseteq G$ et $G \simeq G$ pour tout graphe G.

Exemple 1.2.1. *Soit* $G = (V, E)$ *où*

$$V = \{1, 2, 3, 4, 5\} \text{ et}$$
$$E = \{\{1, 5\}, \{2, 3\}, \{2, 5\}, \{3, 4\}\{3, 5\}, \{4, 5\}\}.$$

Alors, G *a un seul sommet de degré* 1, *deux sommets de degré* 2 *et deux sommets de degré* 3. *De plus, il existe deux chaînes de longueur* 2 *entre les sommets* 2 *et* 4 : $c_1 = (\{2, 3\}, \{3, 4\})$ *et* $c_2 = (\{2, 5\}, \{4, 5\})$. *Finalement, on remarque que* G *est isomorphe au graphe* $H = (V', E')$, *où* $V = \{a, b, c, d, e\}$ *et* $E = \{\{c, e\}, \{a, b\}, \{a, e\}, \{b, d\}\{b, e\}, \{d, e\}\}$ *via la bijection*

$$1 \mapsto c\,, \quad 2 \mapsto a\,, \quad 3 \mapsto b\,, \quad 4 \mapsto d \quad \text{et} \quad 5 \mapsto e.$$

Comme le montre l'exemple 1.2.1, l'étude des graphes d'un point de vue ensembliste s'avère fastidieux. Il est souvent plus intéressant de représenter un graphe G par un dessin dans le plan euclidien $\mathbb{R}^2$. Tout sommet de G est alors représenté

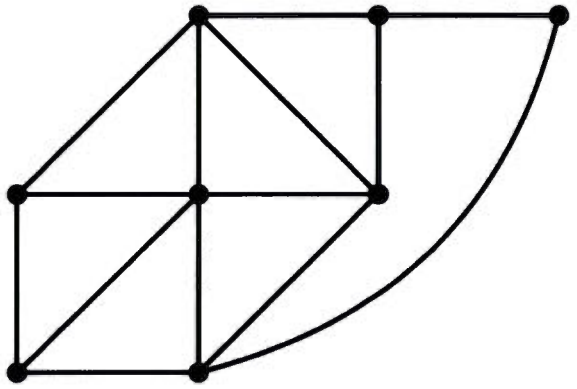


Figure 1.7: Graphe simple fini

par un point et toute arête $\{u, v\}$ par une courbe simple entre les points u et v. Par exemple, la figure 1.7 représente un graphe simple fini.

Maintenant, considérons le problème suivant :

> Est-il possible de dessiner un graphe G donné dans le plan sans croisement d'arête ?

Cette question est à la base de la *théorie topologique des graphes*, aussi appelée *topologie des graphes*. Si un tel dessin est possible, on dit que G est *planaire*. Formellement, la *représentation d'un graphe* est un couple d'ensembles finis (V, E) admettant les propriétés suivantes :

i) $V \subseteq \mathbb{R}^2$;

ii) toute arête est une courbe simple entre deux sommets ;

iii) deux arêtes différentes n'admettent pas les mêmes extrémités.

De plus, une représentation est dite *planaire* si et seulement si l'intérieur d'une arête ne contient aucun sommet et aucun point d'une autre arête. Alors, G est planaire [1] si et seulement s'il existe un isomorphisme (aussi appelé *plongement*)

1. La notion de graphe planaire peut se définir aussi à l'aide, par exemple, du théorème de

de G vers une représentation planaire d'un graphe H. On dit alors que H est une *représentation planaire* de G. Lorsque le contexte est clair, on identifie G à sa représentation.

Si $G = (V, E)$ est un graphe planaire fini, alors les régions $\mathbb{R}^2 \setminus G$ sont appelées *faces*. On note f_G le nombre de faces de G. Il existe une formule, due à L. Euler, mettant en relation f_G, $|V|$ et $|E|$:

$$|V| - |E| + f_G = 2. \tag{1.1}$$

Le graphe de la figure 1.7 admet 8 faces. On vérifie bien que $8 - 14 + 8 = 2$.

Rappelons qu'un *segment de droite dans le plan euclidien* est un sous-ensemble de $\mathbb{R}^2$ de la forme $\{p + t(q - p) \mid 0 \leq t \leq 1\}$ où p et q sont deux points distincts et où «+», «−» et «·» sont respectivement les opérations vectorielles habituelles et la multiplication par le scalaire t. Alors, un *polygone* est une région homéomorphe au cercle unité S^1 dont le bord est constitué d'une union finie de segments de droites (i.e. tout polygone est obtenu par déformation continue du cercle unité). Autrement dit, un polygone est une courbe fermée et simple constituée d'une union de segments de droites. De là, on définit un *arc polygonal* comme une union finie de segments de droites étant toutes homéomorphes à l'intervalle fermé $[0, 1]$. Comme tous les arcs considérés dans la présente thèse sont polygonaux, on utilise le terme «arc» au lieu de «arc polygonal» afin d'alléger le texte.

Intuitivement, tout polygone P sépare le plan $\mathbb{R}^2$ en deux régions distinctes : l'intérieur de P et l'extérieur de P. Ce résultat est connu sous le nom de théorème de Jordan. Malgré son apparente simplicité, il nécessita plusieurs années avant d'être démontré correctement par O. Veblen (Veblen, 1905). Nous en énonçons ici

---

Kuratowski ou de la caractéristique d'Euler (Rosenfeld et Klette, 2004).

une version restreinte aux polygones :

Théorème 1.2.2 (Théorème de Jordan pour les polygones). *Pour tout polygone $P \subseteq \mathbb{R}^2$, l'ensemble $\mathbb{R}^2 \setminus P$ admet exactement deux régions, chacune ayant P comme frontière.*

Maintenant, toute courbe simple γ est homéomorphe à une courbe polygonale (voir (Gamelin et Greene, 1999) pour une des nombreuses démonstrations de ce résultat classique en topologie). Autrement dit, on peut aplatir γ afin d'obtenir un segment de droite. Il est donc clair que toute représentation d'un graphe G est une union de points et de segments de droites, constituant ainsi un cas particulier d'espace topologique. En effet, c'est le sous-espace de $\mathbb{R}^2$ formé des points et des segments de droites de G. Notons que cet espace hérite de la topologie induite par la métrique usuelle définie par la distance dans le plan euclidien. Par conséquent, on peut définir la notion d'homéomorphisme pour les représentations de graphes : deux représentations G_1 et G_2 sont dites *homéomorphes* s'il existe une fonction bijective et continue entre les espaces topologiques associés à G_1 et G_2. Intuitivement, deux représentations de graphes sont homéomorphes s'il est possible d'obtenir G_2 par déformation de G_1. Les homéomorphismes sont les isomorphismes de la famille des espaces topologiques. Par conséquent, ils préservent toutes les propriétés topologiques : connectivité, compacité, simplicité, fermeture, etc.

Le reste de cette section est consacrée aux concepts de subdivision et d'homéomorphisme de graphes. Un résultat important concernant l'équivalence de la notion d'homéomorphisme pour les graphes et leur représentation est ensuite donné. Soit $G = (V, E)$ un graphe non-orienté. Une *subdivision* de G est un graphe obtenu de G en remplaçant des arêtes $\{u, v\} \in E$ par de nouvelles chaînes c entre u et v telles que c n'a pas de sommets internes dans V ou dans toute autre chaîne. Les

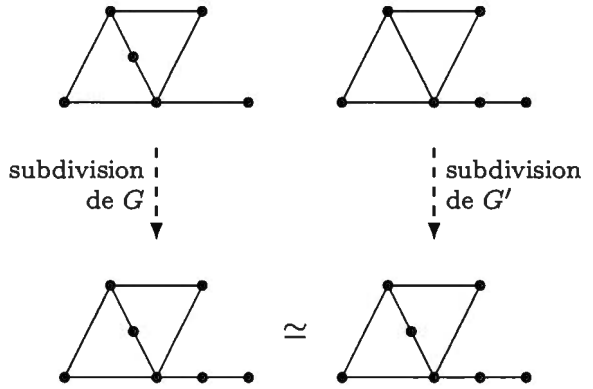


Figure 1.8: Diagramme représentant deux graphes homéomorphes et leurs subdivisions isomorphes

sommets de G sont appelés *sommets d'embranchement* et les nouveaux sommets ajoutés, *sommets internes*. Il est clair par définition que les sommets internes sont de degré 2 alors que les sommets d'embranchement conservent leur degré de G.

Finalement, on définit la notion d'*homéomorphisme de graphes* : deux graphes G et G' sont *homéomorphes* si et seulement s'il existe deux subdivisions isomorphes T et T' de G et G' respectivement. On a alors le résultat suivant. Notons que sa démonstration est omise car elle relève plutôt des domaines de la topologie classique et de la théorie des catégories. Néanmoins, le lecteur intéressé trouvera un chapitre entier y étant consacré dans (Diestel, 2010), chapitre 4.3.

Proposition 1.2.3 ((Diestel, 2010) chapitre 4.3). *Deux graphes sont homéomorphes si et seulement si leurs espaces topologiques associés (i.e. leurs représentation) sont homéomorphes.*

La figure 1.8 illustre deux graphes homéomorphes ainsi que leurs subdivisions respectives.

1.3 Combinatoire des mots

La combinatoire des mots est un des outils essentiels à l'étude de la géométrie discrète. Par exemple, les mots de Christoffel et leur généralisation, les mots Sturmiens, correspondent aux segments de droites discrètes (Berstel *et al.*, 2008) et les mots de Lyndon permettent d'étudier la convexité digitale (Brlek *et al.*, 2009b). Nous rappelons donc ici les principales définitions et notations liées à l'étude des mots.

Soit Σ un ensemble fini de lettres appelé *alphabet*. Un *mot sur* Σ est une suite $w = w_1 w_1 \cdots w_n$ de lettres de Σ. On note w_i la $i^{\text{ième}}$ lettre de w. On écrit aussi parfois $\mathrm{Fst}(w)$ et $\mathrm{Lst}(w)$ pour dénoter la première et dernière lettre de w respectivement. La *longueur de* w est le nombre de lettres de w et est notée $|w|$. De la même façon, pour toute lettre $a \in \Sigma$, $|w|_a$ dénote le nombre d'occurrences de la lettre a dans w. L'unique mot de longueur 0 est appelé *mot vide* et est noté ε. Par exemple, si $w = baabca$, alors $|w| = 6$ et $|w|_b = 2$.

On note Σ^n l'ensemble des mots de longueur n sur Σ. Par définition, $\Sigma^0 = \{\varepsilon\}$. On définit alors que

$$\Sigma^* = \bigcup_{i=0}^{\infty} \Sigma^i$$

est l'ensemble de tous les mots sur Σ. Soit $u = u_1 u_2 \cdots u_m \in \Sigma_1^*$ et $v = v_1 v_2 \cdots v_n \in \Sigma_2^*$ deux mots, possiblement sur deux alphabets Σ_1, Σ_2 différents. Alors, la *concaténation de* u *et* v est l'opération consistant en la formation d'un nouveau mot $u \cdot v$ sur $\Sigma_1 \cup \Sigma_2$ composé des lettres de u suivies des lettres de v. Autrement dit,

$$u \cdot v = u_1 u_2 \cdots u_m v_1 v_2 \cdots v_n.$$

Évidemment, $|u \cdot v| = m + n$. Remarquons que l'opération de concaténation est associative. De plus, pour tout mot w, on vérifie facilement que $w \cdot \varepsilon = \varepsilon \cdot w = w$.

Par conséquent, Σ^* muni de la concaténation est un monoïde appelé *monoïde libre*. Conformément à la notation multiplicative des opérations algébriques, on note uv la concaténation de u et v. Par extension de cette notation, on définit la $k^{\text{ième}}$ puissance de u par

$$u^k = \underbrace{uu\cdots u}_{k \text{ fois}}$$

avec $u^0 = \varepsilon$. Notons qu'un mot non vide w est dit *primitif* si $w = u^k$ implique $k = 1$ pour tout u non vide. Le résultat suivant découle directement de cette définition :

Proposition 1.3.1 (Lothaire, 1997). *Soit w un mot tel qu'il existe deux mots u et v avec $w = uv = vu$. Alors w n'est pas primitif.*

Soit w un mot de Σ^*. Alors, $u \in \Sigma^*$ est appelé *facteur de w* s'il existe deux mots x et y tels que $w = xuy$. De plus, si $x = \varepsilon$ (resp. $y = \varepsilon$), alors on dit que u est un *préfixe de w* (resp. *suffixe de w*). Par définition, u et v sont dits *conjugués* s'il existe deux mots x et y tels que $u = xy$ et $v = yx$. Autrement dit, deux mots sont conjugués s'ils sont égaux à permutation circulaire des lettres près. On montre facilement que la conjugaison est une relation d'équivalence sur l'ensemble Σ^*. On note alors $[w]$ la classe d'équivalence de w, c'est-à-dire l'ensemble des conjugués de w. Par exemple, si $w = aab$, alors $[w] = \{aab, baa, aba\}$.

Soit $\varphi : A^* \longrightarrow B^*$ une application où A et B sont deux alphabets. On dit que φ est un *morphisme* si $\varphi(uv) = \varphi(u)\varphi(v)$ pour tout u et v dans A^*. φ est un *antimorphisme* si pour toute paire de mots $u, v \in A^*$, on a $\varphi(uv) = \varphi(v)\varphi(u)$.

Soit $w = w_1 w_2 \cdots w_n$ un mot sur un alphabet Σ. L'*image miroir de w* est le morphisme défini par $\widetilde{w} = w_n w_{n-1} \cdots w_2 w_1$. Par exemple, $\widetilde{adcbba} = abbcda$.

Étant donné un alphabet Σ, on a déjà vu que Σ^* avec l'opération de concaténation constitue une structure de monoïde, c'est-à-dire un ensemble muni d'une relation

binaire associative et d'un élément identité. Si $(M, *)$ est un monoïde, alors $N \subseteq M$ est un *sous-monoïde de* M si et seulement si N contient l'élément identité et l'opération $*$ est fermée sur N. Par exemple, étant donné un ensemble $X \subset \Sigma^*$, les mots de X engendrent un sous-monoïde $X^* = \{x_1 x_2 \cdots x_n \mid n \geq 0 \text{ et } x_i \in X\}$ de Σ^*. On dit alors que X est un *ensemble de générateurs* de X^*. De plus, X est dit *minimal* si $X \subseteq Y$ pour tout ensemble de générateurs Y. La théorie des codes s'intéresse, entre autre, à la factorisation des éléments de X^*. Les paragraphes suivants constituent un survol rapide des principales notions de cette théorie, telle que présentée dans (Berstel *et al.*, 2009).

Soit Σ un alphabet et $X \subseteq \Sigma^*$. Alors, X est un *code sur* Σ si pour tout $m, n \geq 1$ et $x_1, x_2, \ldots, x_m, y_1, y_2, \ldots, y_n \in X$, la condition $x_1 x_2 \cdots x_m = y_1 y_2 \cdots y_n$ implique $m = n$ et $x_i = y_i$ pour $i = 1, 2, \ldots, n$. Autrement dit, X est un code si tout mot de X^* s'écrit de façon unique comme un produit de mots de X. De façon similaire, on dit que X est un *code circulaire* si pour tout $m, n \geq 1$ et $x_1, x_2, \ldots, x_m, y_1, y_2, \ldots, y_n \in X$, $p \in \Sigma^*$ et $s \in \Sigma^+$, les relations $s x_2 x_3 \cdots x_m p = y_1 y_2 \cdots y_n$ et $x_1 = ps$ impliquent $m = n$, $p = \varepsilon$ et $x_i = y_i$ pour $i = 1, 2, \ldots, n$. En d'autres termes, X est un code circulaire si tout mot $w \in X^*$ admet une unique factorisation en mots de X à permutation circulaire des facteurs près.

Une autre classe importante de codes est celle des codes préfixes. Un ensemble $X \subseteq \Sigma^*$ est dit *préfixe* si aucun élément de X n'est un préfixe propre d'un autre élément de X. Alors, on démontre que tout ensemble préfixe $X \neq \{\varepsilon\}$ est un code, appelé *code préfixe*.

Par exemple, Σ est un code et un code circulaire sur Σ pour tout alphabet Σ. Par contre, $X = \{a, ab, ba\}$ n'est pas un code sur $\{a, b\}$ car $w = (ab)a = a(ba)$ admet deux factorisations distinctes. Remarquons que tout code circulaire est aussi un code, mais l'inverse n'est pas vrai. Par exemple, $X = \{ab, aab, baaba\}$

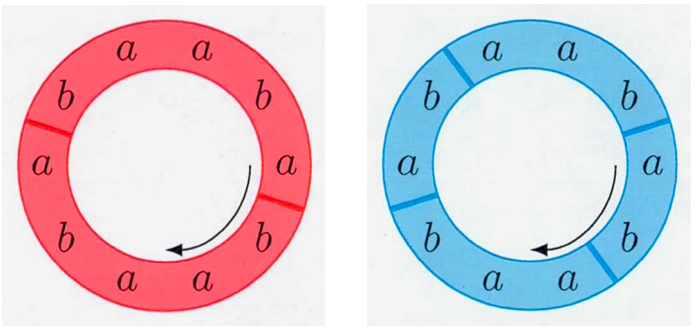


Figure 1.9: Le mot *abaababaab* admet deux factorisations circulaires distinctes

est un code préfixe sur $\{a,b\}^*$. Or, X n'est pas circulaire car pour $m = n = 2$, $x_1 = x_2 = baaba$, $y_1 = y_2 = abaab$, $p = baab$ et $s = a$, les relations

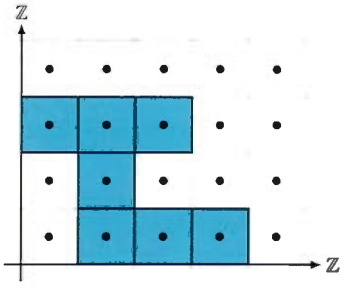
$$sx_2p = y_1y_2$$

$$x_1 = ps$$

n'impliquent pas $p = \varepsilon$ et $x_i = y_i$ (voir figure 1.9).

Soit Σ un alphabet et $M \subseteq \Sigma^*$ un sous-monoïde de Σ^*. M est dit *pur* si pour tout $x \in \Sigma^*$ et $n \geq 1$, $x^n \in M$ implique $x \in M$. De plus, M est dit *très pur* si pour tout $u, v \in \Sigma^*$, les relations $uv \in M$ et $vu \in M$ impliquent $u, v \in M$. Il est aisé de démontrer par induction que tout sous-monoïde très pur est aussi pur. Par contre, l'inverse n'est pas vrai. Par exemple, le sous-monoïde de $\{a,b\}$ engendré par $\{ab, ba\}$ est pur mais pas très pur. Le lecteur est invité à consulter (Berstel *et al.*, 2009) pour un exposé plus détaillée de la théorie des codes. Toutefois, le résultat suivant donnant une caractérisation des sous-monoïdes purs en terme de codes circulaires est essentielle pour la suite.

Théorème 1.3.2 .(Proposition 1.1 de (Berstel *et al.*, 2009)). *Un sous-monoïde M de Σ^* est très pur si et seulement si son ensemble minimal de générateurs est un code circulaire.*



(a) Un heptomino P

$$\begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

(b) Représentation matricielle de P

Figure 1.10: Un polyomino et sa représentation matricielle

1.4 Figures et chemins discrets

Soit S une figure discrète. Alors, il existe plusieurs représentations possibles pour S. Par exemple, si S est entièrement située dans le premier quadrant et touche les deux axes de coordonnées, on peut utiliser la matrice booléenne $\mathbb{B}(S)_{m\times n} = [\beta_{i,j}]$ définie par

$$\beta_{i,j} = \begin{cases} 1 & \text{si} \quad c(i,j) \in S \\ 0 & \text{si} \quad c(i,j) \notin S \end{cases}$$

où m et n sont respectivement la longueur et la largeur de S. Parmi les autres représentations possibles, mentionnons l'énumération des pixels de S et l'utilisation d'une fonction définie sur $\mathbb{Z}^2$ et assignant une valeur à chaque pixel (Rosenfeld et Klette, 2004). La figure 1.10 illustre une figure discrète ainsi que sa représentation matricielle.

Maintenant, il est clair que tout polyomino sans trou est uniquement déterminé par son contour : soit P un polyomino sans trou et (a, b) un point sur le bord de P. Alors, il existe uniquement deux suites partant du point (a, b) et formées des pas unitaires haut, bas, droite et gauche décrivant le contour de P ; l'une dans le sens horaire, l'autre dans le sens antihoraire. Autrement dit, pour décrire tout polyomino sans trou P, il est suffisant de fixer un point sur le contour de P et

une suite de pas unitaires. Par exemple, le polyomino de la figure 1.10 (a) est complètement décrit par le couple $((0,2),(\uparrow,\rightarrow,\rightarrow,\rightarrow,\downarrow,\leftarrow,\downarrow,\rightarrow,\rightarrow,\downarrow,\leftarrow,\leftarrow,\leftarrow,\uparrow,\uparrow,\leftarrow))$ pour l'orientation horaire, ou $((0,2),(\rightarrow,\downarrow,\downarrow,\rightarrow,\rightarrow,\rightarrow,\uparrow,\leftarrow,\leftarrow,\uparrow,\rightarrow,\uparrow,\leftarrow,\leftarrow,\leftarrow,\downarrow))$ pour l'orientation antihoraire.

En 1961, H. Freeman a proposé dans (Freeman, 1961) un codage des figures discrètes en spécifiant leur contour via la bijection

$$\begin{aligned} \mathbf{0} &\mapsto \rightarrow \\ \mathbf{1} &\mapsto \uparrow \\ \mathbf{2} &\mapsto \leftarrow \\ \mathbf{3} &\mapsto \downarrow \end{aligned}$$

L'alphabet $\mathcal{F} = \{\mathbf{0},\mathbf{1},\mathbf{2},\mathbf{3}\}$ est appelé *alphabet de Freeman*. En utilisant cet encodage, tout polyomino sans trou P est représenté par une coordonnée sur son bord ainsi que par un mot $w \in \mathcal{F}^*$ appelé *mot de contour* ou *code de Freeman*. Par exemple, le polyomino de la figure 1.10 (a) est décrit par

$((0,2),\mathbf{1000323003222112})$ pour l'orientation horaire, ou

$((0,2),\mathbf{0330001221012223})$ pour l'orientation antihoraire.

Notons que si w est un mot de contour pour P, alors $[w]$ est l'ensemble de tous les mots de contour de P pour une orientation donnée. Par conséquent, nous décrivons dorénavant tout polyomino sans trou uniquement par sa classe $[w]$, sans spécifier de point sur son bord. Cette approche est justifiée par le fait que l'on considère aussi les polyominos comme des objets combinatoires et pas seulement comme des objets géométriques. De plus, nous parlons par abus de langage du « mot de contour de P » au lieu de la « classe d'équivalence des mots de contour de P » afin d'alléger le texte. Ceci n'affecte pas les résultats obtenus ultérieurement

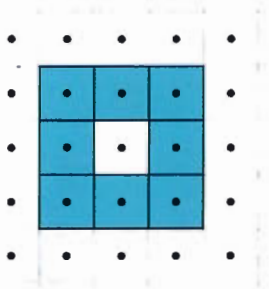


Figure 1.11: Un polyomino avec trou décrit par deux mots de contour

car tous les mots de contour d'une même classe d'équivalence décrivent le même polyomino sans trou.

Remarque. *Il est pertinent d'identifier les polyominos avec trous par une famille de mots de contour. Il faut alors aussi spécifier le contour des trous ainsi que leur emplacement au sein de la figure. Par exemple, le polyomino de la figure 1.11 est décrit par* $\{((0,0),u),((1,1),v)\}$ *où* $u = \mathbf{000111222333}$ *et* $v = \mathbf{0123}$.

Le principal avantage de cette description, outre son unicité à orientation et conjugaison près, est la possibilité d'utiliser tous les outils de la combinatoire des mots afin d'étudier les figures discrètes. Par exemple, plusieurs transformations géométriques de base sur les polyominos sans trou s'expriment à l'aide de morphismes sur les mots de contour. En effet, considérons les morphismes suivants :

$$\rho^i : \mathcal{F}^* \longrightarrow \mathcal{F}^* : x \mapsto (x+i) \mod 4 \tag{1.2}$$

et

$$\sigma^i : \mathcal{F}^* \longrightarrow \mathcal{F}^* : x \mapsto (i-x) \mod 4. \tag{1.3}$$

Alors, ρ^i correspond respectivement aux rotations d'angle 0, $\pi/2$, π et $3\pi/2$ lorsque $i = 0,1,2,3$ tandis que σ^i exprime respectivement des réflexions par rapport aux axes d'angle 0, $\pi/4$, $\pi/2$ et $3\pi/4$ pour $i = 0,1,2,3$ (voir figure 1.12).

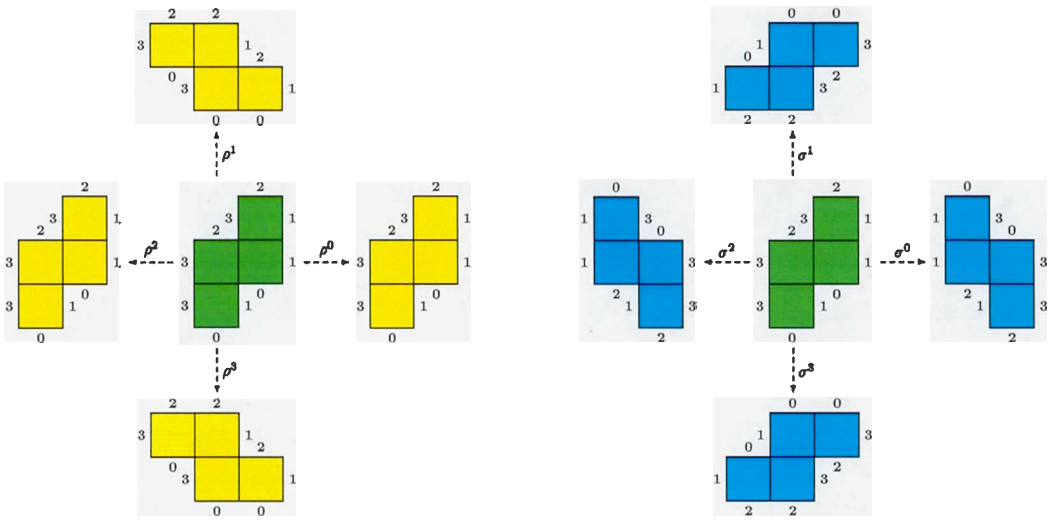


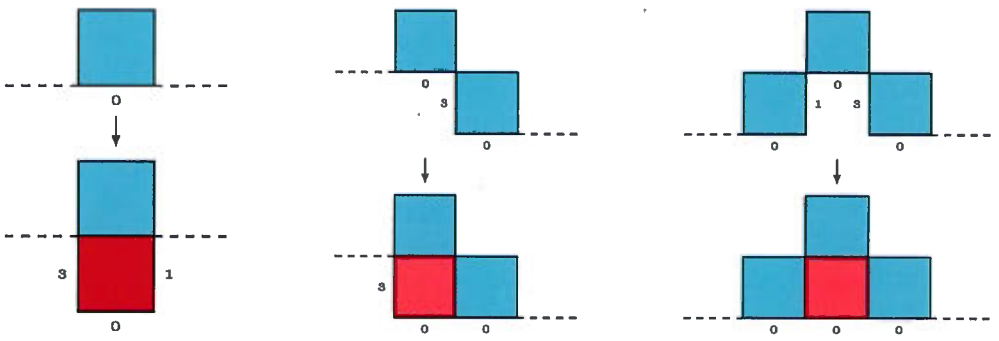
Figure 1.12: Interprétations géométriques des morphismes (a) ρ^i et (b) σ^i appliqués à $w = \mathbf{0101123233}$ pour $i = 0, 1, 2, 3$

Étant donné un polyomino sans trou P, son mot de contour w décrit une courbe fermée de $\mathbb{R}^2$ puisque P est, par définition, un ensemble 4-connexe de pixels. Cette propriété de fermeture s'exprime aisément en fonction du mot de contour w de la façon suivante : un mot w sur $\mathcal{F}^*$ est dit *fermé* si et seulement si $|w|_0 = |w|_2$ et $|w|_1 = |w|_3$. Géométriquement, cette définition exprime le fait qu'un mot fermé décrit une courbe contenant autant de pas *est* que de pas *ouest* et autant de pas *nord* que de pas *sud*. Un mot n'admettant aucun facteur propre fermé est dit *simple* ou *auto-évitant*.

La proposition 1.4.1 suivante affirme que tout mot de contour d'un polyomino est fermé et simple[2]. De plus, elle s'applique autant aux polyominos avec trous que sans trou.

Proposition 1.4.1. *Tout mot de contour d'un polyomino est fermé et simple.*

2. On appelle également *courbe de Jordan* toute courbe fermée et simple.



(a) Deuxième cas (b) Troisième cas (c) Quatrième cas

Figure 1.13: Différents cas lors de l'ajout d'un pixel à un polyomino

De plus, si $w \in \mathcal{F}^*$ est fermé et simple, alors il existe un polyomino admettant w comme mot de contour.

Démonstration. Soit P un polyomino d'aire n. Il est clair que pour $n = 0$ et $n = 1$, P admet respectivement les mots de contour $w = \varepsilon$ et $w = \mathbf{0123}$ qui sont fermés et simples. Supposons maintenant par induction que tout polyomino P d'aire n admette un mot de contour w fermé et simple et considérons P', un polyomino d'aire $n+1$ et w' son mot de contour. Par construction, P' est obtenu de P en lui ajoutant un pixel p. On a alors un des quatre cas suivants à rotation près :

1. Si p est ajouté dans un trou de P tel que w n'est pas le mot de contour de ce trou, alors $w = w'$;
2. Si p est 4-adjacent à exactement un pixel de P, alors w' est obtenu de w en remplaçant une lettre **0** par le facteur $u = \mathbf{301}$ (voir figure 1.13 (a))
3. Si p est 4-adjacent à exactement deux pixels de P, alors w' est obtenu de w en remplaçant un facteur **030** par le facteur $u = \mathbf{300}$ (voir figure 1.13 (b))
4. Si p est 4-adjacent à exactement trois pixels de P, alors w' est obtenu de w en remplaçant un facteur **103** par le facteur $u = \mathbf{0}$ (voir figure 1.13 (c))

Comme l'ajout d'un pixel ne crée pas de croisement ni d'ouverture, le résultat

suit par induction. Maintenant, soit $w \in \mathcal{F}^*$ fermé et simple. Alors, on construit aisément un polyomino admettant w comme mot de contour : il suffit de considérer l'ensemble des pixels situés à l'intérieur de la courbe discrète de $\mathbb{Z}^2$ déterminée par w. On a donc le résultat souhaité. □

Le corollaire suivant découle directement de la démonstration de la seconde partie de la proposition 1.4.1.

Corollaire 1.4.2. *Soit $w \in \mathcal{F}^*$ fermé et simple. Alors, il existe un et un seul polyomino sans trou admettant w comme mot de contour.*

Démonstration. Soit $w \in \mathcal{F}^*$ fermé et simple. Par la proposition 1.4.1, il existe au moins un polyomino sans trou P ayant w comme mot de contour. Supposons par contradiction qu'il existe un polyomino sans trou Q, différent de P et admettant w comme mot de contour. Alors, P et Q diffèrent d'au moins une cellule sur leur contour, tel qu'illustré à la figure 1.13. Par la preuve de la proposition 1.4.1, P et Q n'ont pas le même mot de contour, une contradiction. □

La proposition 1.4.1 caractérise les mots décrivant le contour de polyominos. Étant donné un mot $w \in \mathcal{F}^*$ fermé et simple, on note $\mathrm{POLY}(w)$ l'unique polyomino sans trou associé à w. Or, plusieurs mots sur $\mathcal{F}^*$ ne décrivent pas le bord d'un polyomino. Par exemple, $u = \mathbf{00122}$ n'est pas fermé puisque $|u|_1 = 1 \neq 0 = |u|_3$, tandis que $v = \mathbf{01103223}$ est fermé mais non simple car il admet le facteur fermé $v' = \mathbf{1032}$.

En général, un mot $w \in \mathcal{F}^*$ décrit un *chemin discret*, c'est-à-dire une suite

$$\gamma = (p_1, p_2, \ldots, p_n)$$

de points adjacents de $\mathbb{Z}^2$. Deux points p_i et p_{i+1} sont dits adjacents si $|p_i - p_{i+1}| =$

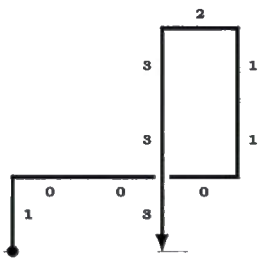


Figure 1.14: Le chemin non simple et ouvert $w = \mathbf{1000112333}$

1. On montre facilement que les chemins discrets sont liés aux mots sur $\mathcal{F}$ via la bijection

$$(p_1, p_2, \ldots, p_n) \simeq (p_1, w = w_1 w_2 \cdots w_{n-1}),$$

où w_i est défini par

$$w_i = \begin{cases} \mathbf{0} & \text{si} \quad p_{i+1} - p_i = (1,0) \\ \mathbf{1} & \text{si} \quad p_{i+1} - p_i = (0,1) \\ \mathbf{2} & \text{si} \quad p_{i+1} - p_i = (-1,0) \\ \mathbf{3} & \text{si} \quad p_{i+1} - p_i = (0,-1) \end{cases}$$

pour tout $1 \leq i \leq n-1$. Par conséquent, on parle par abus de langage du « chemin w » au lieu du « mot associé au chemin γ ». Par exemple, le chemin $w = \mathbf{1000112333}$ est illustré à la figure 1.14.

Il est souvent utile de considérer le chemin $\gamma = (p_1, p_2, \ldots, p_n)$ d'un point de vue graphique (c'est-à-dire en tant que sous-graphe de $\mathbb{Z}^2$). Par conséquent, on associe à γ le graphe $G(\gamma) = (V, E)$ où $V = \{p_1, p_2, \ldots . p_n\}$ et $E = \{\{p_i, p_{i+1}\} \mid 1 \leq i < n\}$. Intuitivement, $G(\gamma)$ est le graphe associé à la représentation graphique de γ.

Par construction, un chemin discret engendre un déplacement. Par exemple, le

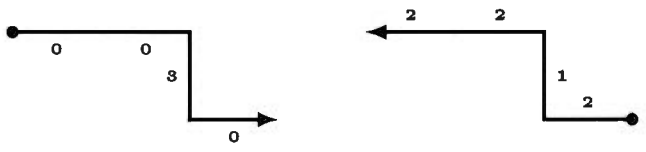


Figure 1.15: Les chemins homologues $w = \mathbf{0030}$ et $\widehat{w} = \mathbf{2122}$

chemin $((a,b),w)$ admet (a,b) comme point d'origine et $(a,b)+(|w|_0-|w|_2, |w|_1-|w_3|)$ comme point d'arrivée. Le vecteur défini par ces deux points est appelé *vecteur déplacement de* $((a,b),w)$ et est noté $\overrightarrow{w}$. Par exemple, si w est le chemin $((1,2),\mathbf{001012})$, alors $\overrightarrow{w} = (2,2)$.

Dans certains cas, étant donné un chemin $w \in \mathcal{F}^*$, il est intéressant de considérer le chemin $\widehat{w}$ obtenu en parcourant w dans le sens opposé. Considérons le morphisme *complément de* w défini par

$$\bar{\cdot} : \mathcal{F}^* \longrightarrow \mathcal{F}^* : \mathbf{0} \leftrightarrow \mathbf{2} \text{ et } \mathbf{1} \leftrightarrow \mathbf{3}.$$

Géométriquement, $\overline{w}$ est obtenu de w en échangeant entre eux les pas horizontaux et les pas verticaux. On définit maintenant l'antimorphisme $\widehat{\cdot}$ comme la composition $\widetilde{\cdot} \circ \bar{\cdot}$. Il est aisé de vérifier que $\widehat{w}$ correspond à w parcouru en sens inverse (voir figure 1.15). On dit alors que w et $\widehat{w}$ sont *homologues*.

Mentionnons aussi que l'alphabet $\mathcal{F}^*$ peut être utilisé afin de coder les chemins discrets selon la notion de virages. Considérons l'opérateur Δ défini pour tout mot $w = w_1 w_2 \cdots w_n \in \mathcal{F}^*$ comme suit :

$$\Delta(w) = (w_2 - w_1)(w_3 - w_2) \cdots (w_n - w_{n-1}),$$

(a) Le chemin $w = \mathbf{001100322223}$ codé par des pas unitaires

(b) Un codage du même chemin à l'aide du mot des différences finies $\Delta(w) = \mathbf{01030330001}$ représentant les virages

Figure 1.16: Codage d'un chemin à l'aide de pas unitaires et de virages

où les soustractions sont effectuées modulo 4. Alors, $\Delta(w) \in \mathcal{F}^*$ est le *mot des différences finies de* w. On interprète géométriquement $\Delta(w)$ via la correspondance

0 : aller tout droit,

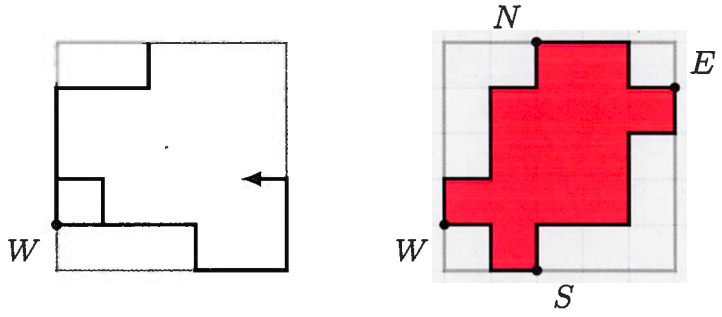
1 : tourner à gauche,

2 : faire demi-tour,

3 : tourner à droite

de telle sorte que $\Delta(w)$ code le chemin w à l'aide de virages. Par exemple, la figure 1.16 représente le chemin $w = \mathbf{0011223323}$ ainsi que $\Delta(w)$.

Étant donné un chemin discret $\gamma = (p, w)$, on nomme *rectangle englobant* le rectangle d'aire minimale contenant tous les points de γ. Ceci permet de définir les



(a) Rectangle englobant d'un chemin discret et le point W

(b) Rectangle englobant d'un chemin discret fermé et simple ainsi que les quatre points extrémaux

Figure 1.17: Rectangles englobants

quatre points extrémaux sur le bord du rectangle :

W : le plus bas sur le côté ouest,

N : le plus à gauche sur le côté nord,

E : le plus haut sur le côté est et

S : le plus à droite sur le côté sud,

comme illustré à la figure 1.17 (a). Clairement, ces quatre points appartiennent à γ. De plus, il existe un algorithme linéaire en $|w|$ permettant de les calculer : il suffit de lire le mot de contour w en gardant en mémoire les coordonnées extrémales. Il est intéressant de noter, que dans le cas d'un chemin fermé et simple, ces points induisent une factorisation du mot w, dite *standard*[3], en quatre facteurs : $w = WS \cdot SE \cdot EN \cdot NW$ où, par exemple, NW représente le facteur de w codant le chemin allant du point N au point W (voir figure 1.17 (b)).

3. Cette terminologie est tirée de (Brlek *et al.*, 2009b).

1.5 Pavages par translation

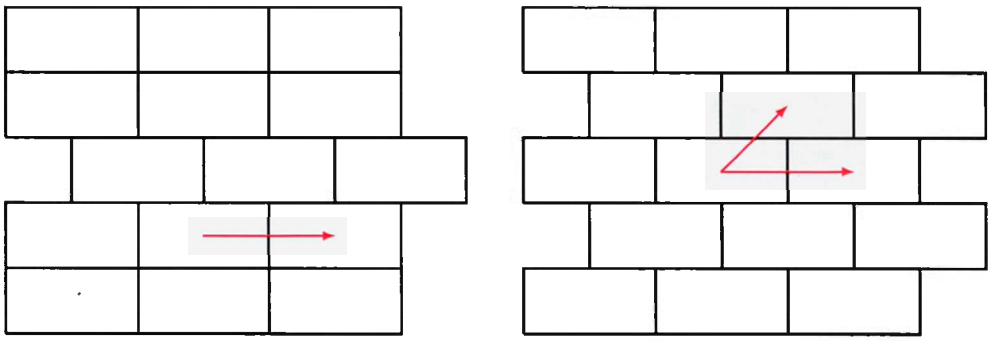
Le problème du pavage se résume ainsi : étant donné une surface S et un ensemble fini de tuiles $T = \{s_i \subset S \mid s_i \cap s_j = \emptyset\}$, est-il possible de recouvrir entièrement S par des copies d'éléments de T sans créer de chevauchement ? Évidemment, ce problème se décompose en une multitude de sous-problèmes selon les contraintes imposées. Par exemple, le problème de pavage est résoluble en temps linaire lorsque $T = \{\mathbf{01^k23^k}, \mathbf{0^k12^k3}\}$ avec $k > 0$ et S est un ensemble fini de cellules sans trou (Kenyon et Kenyon, 1992), tandis qu'il est NP-complet si $T = \{\mathbf{0^k12^k3}, \mathbf{01^l23^l}\} \setminus \{\mathbf{001223}, \mathbf{011233}\}$ avec $k, l > 0$ et S est un ensemble fini de cellules admettant un nombre arbitraire de trous (Beauquier *et al.*, 1995). Afin de répondre aux nécessités de la présente thèse, nous nous restreignons au contexte suivant : S est le plan euclidien $\mathbb{R}^2$ et T contient des copies translatées d'un seul polyomino sans trou P (et donc les rotations et symétries de P sont interdites). Formellement, on note $P + \vec{u}$ la copie translatée de P par le vecteur $\vec{u} \in \mathbb{R}^2$, c'est-à-dire le polyomino obtenu en translatant chacune des cellules de P par le vecteur $\vec{u}$. On a alors la définition 1.5.1 suivante. Notons que la terminologie utilisée est tirée de (Beauquier et Nivat, 1991).

Définition 1.5.1. *L'ensemble* $T = \{(P, \vec{u}) \mid \vec{u} \in U \subset \mathbb{Z}^2\}$ *constitue un* pavage du plan par translation par le polyomino P *si les deux conditions suivantes sont respectées :*

1. $\mathbb{R}^2$ *est l'union de tous les polyominos* $P + \vec{u}$ *pour tout* $(P, \vec{u}) \in T$,
2. *pour toute paire distincte* $(P, \vec{u}) \neq (P, \vec{v})$, *l'intersection de* $P + \vec{u}$ *avec* $P + \vec{v}$ *est d'aire nulle.*

On dit alors que P *est une* tuile.

Par exemple, la cellule et le rectangle $\mathbf{0^312^33}$ sont tous deux des tuiles. Un pavage par translation T est dit *k-périodique* s'il existe un entier $k > 0$ et un ensemble



(a) 1-périodique (b) Régulier

Figure 1.18: Différents types de pavage du plan par translation

de k vecteurs linéairement indépendants tel que la translation par ces vecteurs ne modifie pas l'ensemble T. De plus, T est dit *régulier* s'il existe deux vecteurs $\vec{u}$ et $\vec{v}$ tels que

$$T = \{(P, a\vec{u} + b\vec{v}) \mid a, b \in \mathbb{Z}^2\}.$$

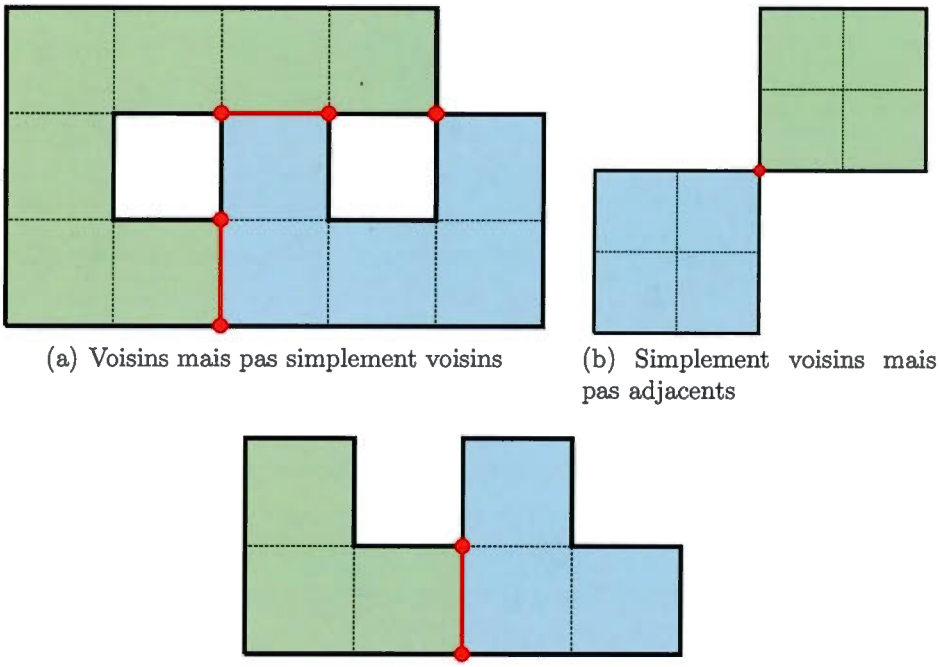
La figure 1.18 illustre différents types de pavage du plan par translation.

Il existe un lien entre les pavages et la combinatoire des mots. En effet, D. Beauquier et M. Nivat donnent dans (Beauquier et Nivat, 1991) la caractérisation suivante des polyominos pavant le plan par translation.

Proposition 1.5.2 (Théorème 3.2 de (Beauquier et Nivat, 1991)). *Soit P un polyomino admettant $w \in \mathcal{F}^*$ comme mot de contour. Alors, P est une tuile si et seulement si $w = XYZ\widehat{X}\widehat{Y}\widehat{Z}$ où au plus un des mots X, Y, Z est vide.*

Une tuile admettant exactement un des facteurs X, Y ou Z vide est dite *parallélogramme*. Sinon, on dit que P est un polyomino *hexagone*. Un autre résultat remarquable du même article concerne le type de pavage induit par les tuiles.

Proposition 1.5.3 (Proposition 3.1 et théorème 4.1 de (Beauquier et Nivat, 1991)). *Soit P une tuile. Alors,*



(a) Voisins mais pas simplement voisins

(b) Simplement voisins mais pas adjacents

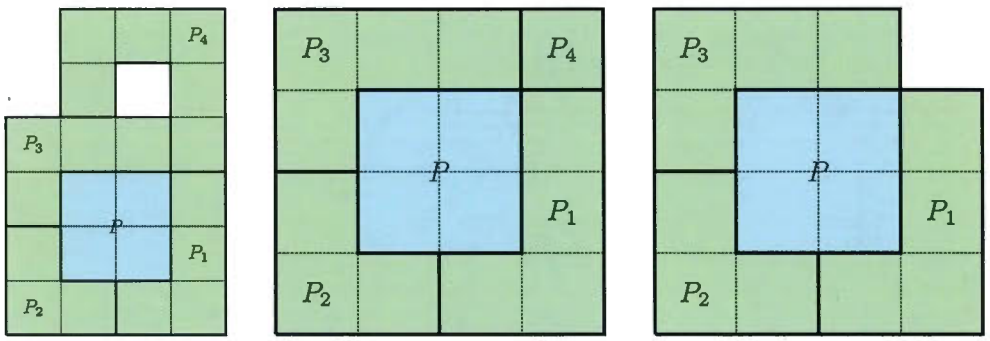
(c) Adjacents

Figure 1.19: Différents types de voisinage de polyominos

1. *tout pavage par translation du plan par P est 1-périodique et*

2. *il existe un pavage régulier du plan par P.*

En utilisant la proposition 1.5.3, il est possible de décrire explicitement tous les pavages réguliers par translation du plan par un parallélogramme ou un hexagone en étudiant l'entourage de chaque tuile (voir (Beauquier et Nivat, 1991), théorème 5.1). Nous terminons cette section en explicitant ces notions. Deux polyominos P et Q sont dits *voisins* si leur intersection est non-vide et d'aire nulle. Deux polyominos voisins sont dits *simplement voisins* si leur intersection est connexe. Finalement, deux polyominos simplement voisins sont dits *adjacents* si leur intersection n'est pas réduite à un point. La figure 1.19 illustre ces trois types de voisinage.



(a) Entourage de P (b) Entourage complet de P (c) Entourage minimal de P

Figure 1.20: Différents types d'entourage de P

Maintenant, étant donné un polyomino P, un cycle $\langle P_0, \ldots, P_{k-1} \rangle$ de polyominos tels que P et P_i sont simplement voisins pour tout $0 \leq i < k$ et P_i et P_{i+1} sont simplements voisins pour tout $i \geq 0$ est appelé *entourage de* P. Notons que les indices sont considérés modulo k. De plus, un entourage est dit *complet* si P_i et P_{i+1} sont adjacents et *minimal* si P et P_i sont adjacents. La figure 1.20 illustre ces trois types d'entourage.

Dans (Beauquier et Nivat, 1991), les auteurs donnent une description de tous les pavages réguliers possibles induits par des tuiles en spécifiant leur entourage. En particulier, si $P = XY\widehat{X}\widehat{Y}$ est un parallélogramme, alors on obtient un pavage régulier du plan par translation en considérant l'entourage minimal suivant de P :

$$\left(P + \overrightarrow{X}, P + \overrightarrow{\widehat{Y}}, P + \overrightarrow{\widehat{X}}, P + \overrightarrow{Y}\right) = \left(P + \overrightarrow{X}, P - \overrightarrow{Y}, P - \overrightarrow{X}, P + \overrightarrow{Y}\right), \quad (1.4)$$

c'est-à-dire en entourant P par quatre copies translatées de lui-même en faisant correspondre les côtés homologues entre-eux, tel qu'illustré à la figure 1.21. En étendant cet entourage à tout le plan, on obtient un pavage régulier de $\mathbb{R}^2$.

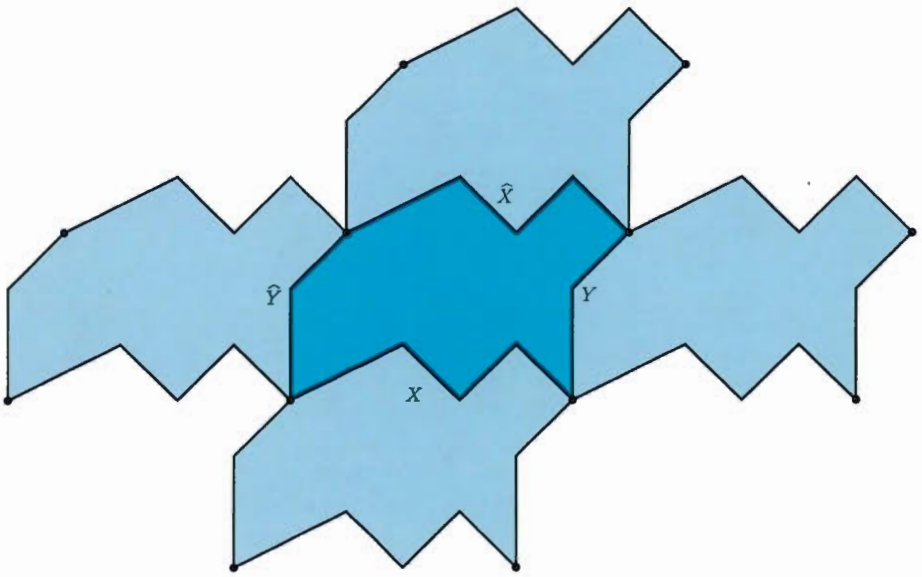


Figure 1.21: L'entourage minimal (1.4) d'un parallélogramme engendre un pavage régulier du plan

CHAPITRE II

RÉSEAUX PARALLÉLOGRAMMES ET CODES CIRCULAIRES

Dans ce chapitre, nous développons la notion de réseau parallélogramme, un outil puissant établissant un pont entre la combinatoire des mots, la géométrie discrète et la théorie des graphes. Après l'énoncé des définitions nécessaires à la section 2.1, nous démontrons à la section 2.2 le résultat principal de ce chapitre : les réseaux parallélogrammes constituent une généralisation du réseau carré $\mathbb{Z}^2$. Ce type de réseaux a cependant un intérêt théorique hors de la géométrie discrète puisque nous démontrons à la section 2.3, à l'aide des réseaux parallélogrammes, une conjecture de A. Blondin Massé et al. concernant la théorie des codes. Par ailleurs, comme nous verrons au chapitre 3, cette notion permet de jeter un regard nouveau sur l'étude de la décomposition de polyominos. Finalement, notons qu'une version primitive de ce chapitre a été présentée dans le cadre de la $10^{\text{ième}}$ conférence internationale sur la théorie des langages et des automates (Blondin Massé *et al.*, 2016).

2.1 Réseaux parallélogrammes

Il existe plusieurs définitions équivalentes de réseau carré. Par exemple, on définit algébriquement un réseau carré par l'union de deux familles infinies de droites parallèles équidistantes, telles que toute paire de droites appartenant à deux familles différentes sont perpendiculaires (Bern *et al.*, 1999). Cependant, nous privilégions

une approche géométrique axée sur la théorie des graphes. Par définition, le réseau carré $G(\mathbb{Z}^2)$, aussi appelé *grille carrée*, est le graphe infini et simple (V, E) où $V = \mathbb{Z}^2$ et où $\{p, q\} \in E$ si et seulement si la distance euclidienne entre p et q est 1 (i.e. $d(p, q) = 1$). On note $\Gamma(\mathbb{Z}^2)$ l'ensemble de tous les chemins de $G(\mathbb{Z}^2)$. Remarquons que tout chemin $\gamma \in \Gamma(\mathbb{Z}^2)$ induit un sous-graphe de $G(\mathbb{Z}^2)$, noté $G(\gamma)$. Par extension, on note $G(P) := G(\gamma)$ le graphe du polyomino P dont le contour est le chemin γ.

Maintenant, considérons un chemin (p, w) de la grille carrée $G(\mathbb{Z}^2)$, où $p = (a, b) \in \mathbb{Z}^2$ et $w \in \mathcal{F}^*$ ainsi qu'un morphisme $\varphi : \mathcal{F}^* \longrightarrow \mathcal{F}^*$. Rappelons que $\mathcal{F}$ est l'alphabet de Freeman introduit à la section 1.4. On étend le domaine de φ à l'ensemble des chemins de $\mathbb{Z}^2$ de la façon suivante [1] :

$$\varphi(p, w) = (\varphi(p), \varphi(w)), \tag{2.1}$$

où

$$\varphi(p) := a\overrightarrow{\varphi(0)} + \overrightarrow{b\varphi(1)}.$$

Par exemple, si φ est défini par $\varphi(\mathbf{0}) = \mathbf{32}$, $\varphi(\mathbf{1}) = \mathbf{000}$, $\varphi(\mathbf{2}) = \mathbf{1}$ et $\varphi(\mathbf{3}) = \mathbf{3}$, alors $\varphi((-1, 3), \mathbf{0012}) = ((10, 1), \mathbf{32320001})$.

La définition de réseau parallélogramme découle de deux classes particulières de morphismes : les morphismes homologues et les morphismes parallélogrammes (Blondin Massé, 2012).

Définition 2.1.1. *Un morphisme $\varphi : \mathcal{F}^* \longrightarrow \mathcal{F}^*$ est dit* homologue *si $\varphi(a) = \widehat{\varphi(\overline{a})}$ pour tout $a \in \mathcal{F}$.*

---

1. Cet abus de notation est justifié par le fait qu'il existe une bijection entre les chemins discrets et les mots sur $\mathcal{F}^*$.

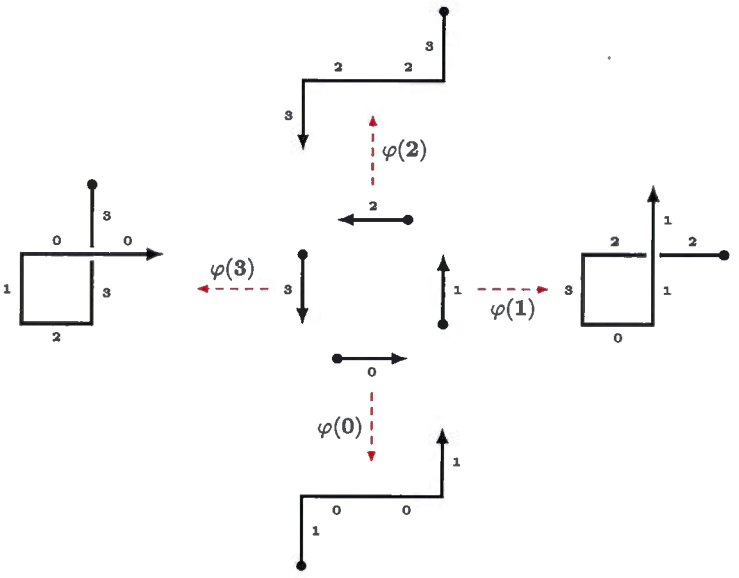


Figure 2.1: Application du morphisme (2.2) sur les quatre pas élémentaires

Les morphismes homologues sont en quelque sorte une généralisation de l'antimorphisme $\widehat{\cdot}$. En effet, un morphisme φ homologue remplace les deux pas horizontaux homologues **0** et **2** par deux chemins homologues $\varphi(\mathbf{0})$ et $\varphi(\mathbf{2})$, et de même pour les pas verticaux **1** et **3**. Par exemple, on vérifie facilement que le morphisme suivant est homologue :

$$\varphi(\mathbf{0}) = \mathbf{1001},\ \varphi(1) = \mathbf{223011},\ \varphi(\mathbf{2}) = \mathbf{3223},\ \varphi(\mathbf{3}) = \mathbf{332100}. \tag{2.2}$$

La figure 2.1 illustre l'effet du morphisme (2.2) sur les quatre pas élémentaires.

De la définition 2.1.1, on déduit immédiatement que

$$\varphi(\widehat{w}) = \widehat{\varphi(w)} \tag{2.3}$$

pour tout morphisme homologue et tout mot $w \in \mathcal{F}^*$. En effet, on a

$$\begin{aligned}
\varphi(\widehat{w}) &= \varphi(\overline{w_n w_{n-1} \cdots w_1}) && \text{par définition de } \widehat{\cdot} \\
&= \varphi(\overline{w_n})\varphi(\overline{w_{n-1}}) \cdots \varphi(\overline{w_1}) && \text{puisque } \varphi \text{ et } \bar{\cdot} \text{ sont des morphismes} \\
&= \widehat{\varphi(w_n)}\widehat{\varphi(w_{n-1})} \cdots \widehat{\varphi(w_1)} && \text{puisque } \varphi \text{ est homologue} \\
&= \widehat{\varphi(w)} && \text{puisque } \widehat{\cdot} \text{ est un antimorphisme.}
\end{aligned}$$

De là, on définit la notion de morphisme parallélogramme. Rappelons que $\mathrm{Fst}(w)$ et $\mathrm{Lst}(w)$ dénotent respectivement la première et dernière lettre de w.

Définition 2.1.2. *Un morphisme homologue $\varphi : \mathcal{F}^* \longrightarrow \mathcal{F}^*$ est dit* parallélogramme *si*

(i) $\varphi(\mathbf{0123})$ *est simple et fermé ;*

(ii) $\mathrm{Fst}(\varphi(a)) = a$ *pour tout* $a \in \mathcal{F}$.

Par définition, tout morphisme parallélogramme est aussi homologue. Cependant, l'inverse n'est pas vrai puisque le morphisme défini dans l'équation (2.2) n'est pas parallélogramme. La proposition suivante découle directement des définitions précédentes :

Proposition 2.1.3. *Soit φ un morphisme parallélogramme. Alors,* $\mathrm{Fst}(\varphi(a)) = a = \mathrm{Lst}(\varphi(a))$ *pour tout* $a \in \mathcal{F}$.

Démonstration. Soit $a \in \mathcal{F}$. Comme φ est parallélogramme, il est homologue et on a $a = \mathrm{Fst}(\varphi(a)) = \mathrm{Fst}(\widehat{\varphi(\overline{a})})$. Maintenant, remarquons que $\mathrm{Fst}(\widehat{w}) = \overline{\mathrm{Lst}(w)}$ par définition. Par conséquent, $a = \overline{\mathrm{Lst}(\varphi(\overline{a}))}$. Comme $\bar{\cdot} \circ \bar{\cdot} = \mathrm{Id}$, on déduit que

$$\mathrm{Fst}(\varphi(\overline{a})) = \overline{a} = \mathrm{Lst}(\varphi(\overline{a})),$$

d'où le résultat. □

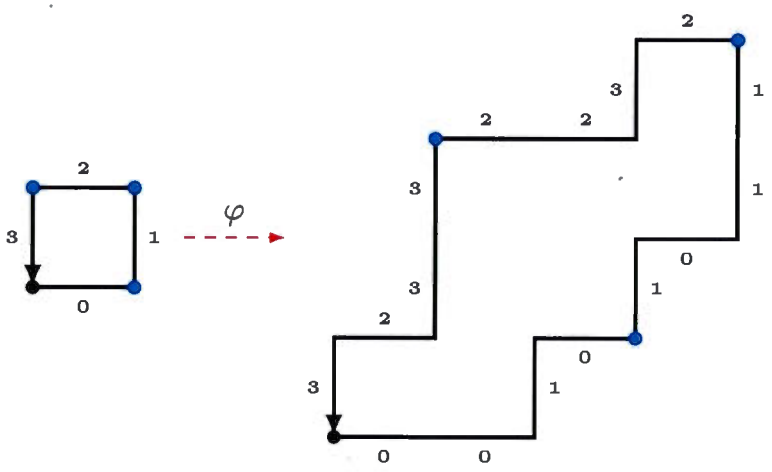


Figure 2.2: Effet du morphisme φ de l'exemple 2.1.4 sur le chemin **0123**

Exemple 2.1.4. *Considérons le morphisme homologue* $\varphi : \mathcal{F}^* \longrightarrow \mathcal{F}^*$ *défini par :*

$$\varphi(a) = \begin{cases} \mathbf{0010} & si \quad a = 0 \\ \mathbf{1011} & si \quad a = 1 \\ \mathbf{2322} & si \quad a = 2 \\ \mathbf{3323} & si \quad a = 3 \end{cases}$$

On vérifie aisément que $\varphi(\mathbf{0123}) = \mathbf{0010101123223323}$ *est simple et fermé (voir figure 2.2). De plus,* $\mathrm{Fst}(\varphi(a)) = a$ *pour tout* $a \in \mathcal{F}$. *Par conséquent,* φ *est un morphisme parallélogramme.*

Il est intéressant de noter que les morphismes parallélogrammes tirent leur nom des tuiles parallélogrammes (voir section 1.5). En effet, si φ est un morphisme parallélogramme, alors $\varphi(\mathbf{0123})$ est un polyomino parallélogramme par définition. Nous sommes maintenant prêts à définir une notion nouvelle et centrale de la présente thèse, soit celle de réseau parallélogramme.

Définition 2.1.5. *Soit* φ *un morphisme parallélogramme. Alors, l'union des graphes des images sous* φ *de tous les chemins* γ *de* $G(\mathbb{Z}^2)$ *est appelé* réseau parallélo-

gramme associé à φ *et est noté* $G(\varphi)$. *Formellement,*

$$G(\varphi) = \bigcup_{\gamma \in \Gamma(\mathbb{Z}^2)} G(\varphi(\gamma)).$$

La proposition suivante fournit une définition équivalente de réseau parallélogramme.

Proposition 2.1.6. *Soit* φ *un morphisme parallélogramme. Alors,*

$$G(\varphi) = \bigcup_{p \in \mathbb{Z}^2} G(\varphi(p, \mathbf{0123})).$$

Démonstration. On a bien que $\bigcup_{p\in\mathbb{Z}^2} G(\varphi(p, \mathbf{0123})) \subseteq \bigcup_{\gamma\in\Gamma(\mathbb{Z}^2)} G(\varphi(\gamma))$ puisque $(p, \mathbf{0123})$ est un chemin de $G(\mathbb{Z}^2)$. Soit $\gamma \in \Gamma(\mathbb{Z}^2)$. Alors, γ est une union de chemins de longueur 1 :

$$\begin{aligned} \gamma &= p_1, p_2, \ldots, p_n \\ &= (p_1, a_1) \cup (p_2, a_2) \cup \cdots \cup (p_{n-1}, a_{n-1}) \qquad \text{où } a_i \in \mathcal{F}. \end{aligned}$$

Or,

$$(p_i, a_i) \subseteq \begin{cases} (p_i, \mathbf{0123}) & \text{si} \quad a_i = \mathbf{0} \text{ ou } \mathbf{1} \\ (p_i - (1,1), \mathbf{0123}) & \text{sinon} \end{cases}$$

d'où le résultat. □

La proposition 2.1.6 affirme que tout réseau parallélogramme est obtenu de la grille carrée en remplaçant chaque cellule de $\mathbb{Z}^2$ par le polyomino ayant $\varphi(\mathbf{0123})$ comme mot de contour. La figure 2.3 illustre le réseau parallélogramme obtenu par l'application du morphisme φ de l'exemple 2.1.4. On remarque que tout morphisme parallélogramme induit un pavage régulier du plan par la tuile $\varphi(\mathbf{0123})$.

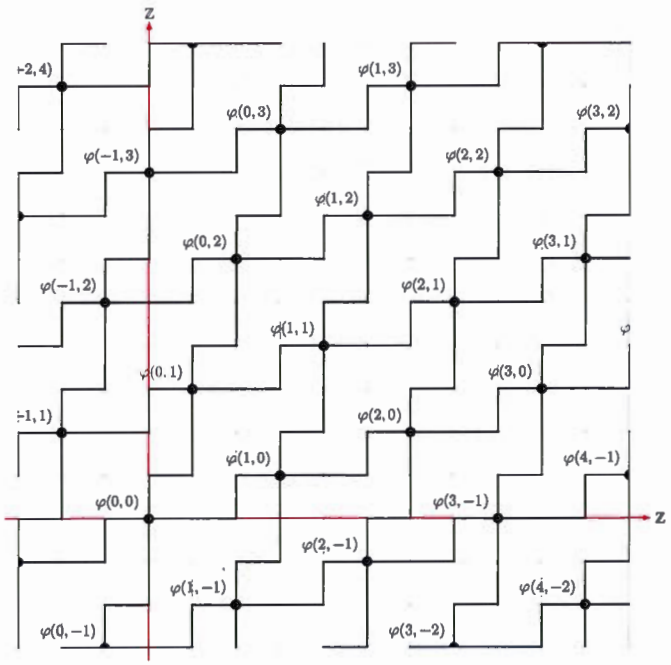


Figure 2.3: Réseau parallélogramme $G(\varphi)$ associé au morphisme φ de l'exemple 2.1.4

La proposition 2.1.7 formalise cette idée en utilisation la notation de la section 1.5.

Proposition 2.1.7. *Si φ est un morphisme parallélogramme, alors*

$$\left\{(\varphi(\mathbf{0123}), a\overrightarrow{\varphi(\mathbf{0})} + b\overrightarrow{\varphi(\mathbf{1})}) \mid a, b \in \mathbb{Z}\right\}$$

est un pavage régulier du plan.

Démonstration. Si $G(\varphi)$ est un réseau parallélogramme, alors φ est un morphisme parallélogramme. Posons $P = \varphi(\mathbf{0123})$ On a donc par définition que

$$P = \varphi(\mathbf{0})\varphi(\mathbf{1})\varphi(\mathbf{2})\varphi(\mathbf{3}) = \varphi(\mathbf{0})\varphi(\mathbf{1})\widehat{\varphi(\mathbf{0})\varphi(\mathbf{1})}.$$

Puisque P est fermé et simple, c'est un polyomino exact par la proposition 1.5.2.

Maintenant, par la proposition 1.5.3 et l'équation (1.4), P pave le plan de façon régulière et son entourage est

$$\left(P + \overrightarrow{\varphi(\mathbf{0})}, P - \overrightarrow{\varphi(\mathbf{1})}, P - \overrightarrow{\varphi(\mathbf{0})}, P + \overrightarrow{\varphi(\mathbf{1})}\right).$$

On a donc bien que $\left\{(\varphi(\mathbf{0123}), a\overrightarrow{\varphi(\mathbf{0})} + b\overrightarrow{\varphi(\mathbf{1})}) \mid a, b \in \mathbb{Z}\right\}$ est un pavage régulier du plan. □

Il est aisé de vérifier que le morphisme identité Id est parallélogramme. Par conséquent, $G(\mathbb{Z}^2)$ est le réseau parallélogramme associé à Id et correspond au pavage du plan par un seul pixel. Nous démontrons maintenant une propriété des réseaux parallélogrammes concernant la correspondance entre les sommets de $G(\mathbb{Z}^2)$ et ceux de $G(\varphi)$.

Proposition 2.1.8. *Soit φ un morphisme parallélogramme et $\overrightarrow{e} \in \{(1,0),(0,1),(-1,0),(0,-1)\}$. Alors,*

(i) *p est un sommet de $G(\mathbb{Z}^2)$ si et seulement si $\varphi(p)$ est un sommet de $G(\varphi)$;*

(ii) *$\{p, p + \overrightarrow{e}\}$ est une arête de $G(\mathbb{Z}^2)$ si et seulement si $\{\varphi(p), \varphi(p) + \overrightarrow{e}\}$ est une arête de $G(\varphi)$.*

Démonstration. Remarquons que l'arête $\{p, p + \overrightarrow{e}\}$ induit le chemin (p, a) où $a \in \mathcal{F}$. Alors, le résultat suit de l'équation (2.1) et de la définition 2.1.5. □

Corollaire 2.1.9. *Soit $p \in \mathbb{Z}^2$ et φ un morphisme parallélogramme. Alors, $\varphi(p)$ est de degré 4 dans $G(\varphi)$.*

2.2 Homéomorphisme de réseaux parallélogrammes

Dans cette section, nous démontrons plusieurs propriétés géométriques et algébriques des réseaux parallélogrammes, culminant avec la démonstration d'un

théorème fondamental : tout réseau parallélogramme est homéomorphe au réseau carré, c'est-à-dire que tout réseau parallélogramme est une déformation continue de $G(\mathbb{Z}^2)$. Nous démontrons d'abord certaines propriétés des morphismes homologues utiles pour la suite.

Proposition 2.2.1. *Soit φ un morphisme homologue et $w \in \mathcal{F}^*$.*

(i) *Pour tout $a \in \mathcal{F}$, $\overrightarrow{\varphi(a)} + \overrightarrow{\varphi(\overline{a})} = \overrightarrow{0}$.*

(ii) *Si $\overrightarrow{w} = (x, y)$, alors $\overrightarrow{\varphi(w)} = x\overrightarrow{\varphi(\mathbf{0})} + y\overrightarrow{\varphi(\mathbf{1})}$.*

Démonstration. (i) Puisque φ est homologue, pour tout $a \in \mathcal{F}$, $\varphi(\overline{a})$ est le chemin $\varphi(a)$ parcouru dans le sens inverse. On a alors

$$\overrightarrow{\varphi(a)} = \overrightarrow{\widetilde{\varphi(\overline{a})}} = -\overrightarrow{\varphi(\overline{a})}.$$

(ii) Posons $w = w_1 w_2 \cdots w_n$. Alors,

$$\begin{aligned}
\overrightarrow{\varphi(w)} \quad &= \sum_{i=1}^{n} \overrightarrow{\varphi(w_i)} \\
&= \sum_{a \in \mathcal{F}} |w|_a \overrightarrow{\varphi(a)} \\
&= \sum_{a \in \{\mathbf{0},\mathbf{1}\}} (|w|_a \overrightarrow{\varphi(a)} + |w|_{\overline{a}} \overrightarrow{\varphi(\overline{a})}) && \text{car } \varphi \text{ est homologue} \\
&= \sum_{a \in \{\mathbf{0},\mathbf{1}\}} (|w|_a - |w|_{\overline{a}}) \overrightarrow{\varphi(a)} && \text{par (i)} \\
&= x\overrightarrow{\varphi(\mathbf{0})} + y\overrightarrow{\varphi(\mathbf{1})} && \text{par définition de } \overrightarrow{w},
\end{aligned}$$

ce qu'il fallait démontrer. □

Nous pouvons maintenant démontrer une propriété algébrique fondamentale des réseaux parallélogrammes.

Proposition 2.2.2. *Soit φ un morphisme parallélogramme. Alors $\left\{\overrightarrow{\varphi(\mathbf{0})}, \overrightarrow{\varphi(\mathbf{1})}\right\}$ est une base de l'espace vectoriel $\mathbb{R}^2$.*

Démonstration. Posons $\vec{u} = \overrightarrow{\varphi(\mathbf{0})}$ et $\vec{v} = \overrightarrow{\varphi(\mathbf{1})}$. Il suffit de montrer que $\vec{u}$ et $\vec{v}$ sont linéairement indépendants. Par contradiction, supposons qu'ils sont linéairement dépendants. Par la proposition 2.1.7, $T = \{(\varphi(\mathbf{0123}), a\vec{u} + b\vec{v}) \mid a, b \in \mathbb{Z}\}$ est un pavage régulier du plan.

Maintenant, considérons la bande B de $\mathbb{R}^2$ délimitée par les droites

$$d_1 = (0, |\varphi(\mathbf{0123})|_1 + 1) + t\vec{u}$$

et

$$d_2 = (0, -|\varphi(\mathbf{0123})|_3 - 1) + t\vec{v},$$

où $t \in \mathbb{R}$. Comme $\vec{u}$ et $\vec{v}$ sont linéairement dépendants, tout élément du pavage est entièrement compris dans B. Or, $B \neq \mathbb{R}^2$ car le point $(0, |\varphi(\mathbf{0123})|_1 + 2)$ n'est pas dans B, contredisant ainsi le fait que T est un pavage du plan. □

Une conséquence immédiate de la proposition 2.2.2 est que les morphismes parallélogrammes préservent les chemins fermés.

Proposition 2.2.3. *Soit φ un morphisme parallélogramme et $w \in \mathcal{F}^*$. Alors w est fermé si et seulement si $\varphi(w)$ est fermé.*

Démonstration. ($\Rightarrow$) Supposons que w est fermé de sorte que $\vec{w} = \vec{0}$. Par la proposition 2.2.1(ii), on conclut que $\overrightarrow{\varphi(w)} = 0\varphi(\mathbf{0}) + 0\varphi(\mathbf{1}) = \vec{0}$. Donc, $\varphi(w)$ est fermé.

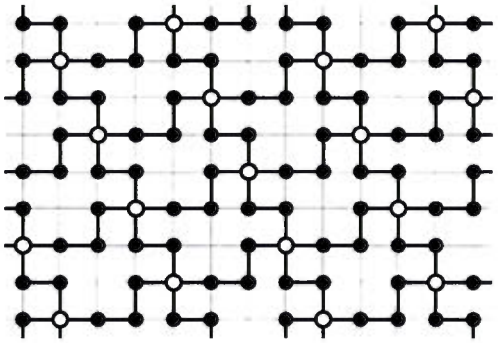


Figure 2.4: Le réseau parallélogramme $G(\varphi)$ engendré par le morphisme défini par $\varphi(\mathbf{0}) = \mathbf{0010}$ et $\varphi(\mathbf{1}) = \mathbf{121}$. Les points blancs représentent les carrefours

($\Leftarrow$) Par contraposée, supposons que w n'est pas fermé. Alors $\overrightarrow{w} = (x, y)$, où $xy \neq 0$. Par conséquent, $\overrightarrow{\varphi(w)} = x\overrightarrow{\varphi(\mathbf{0})} + y\overrightarrow{\varphi(\mathbf{1})}$. Or, nous savons par la proposition 2.2.2 que $\overrightarrow{\varphi(\mathbf{0})}$ et $\overrightarrow{\varphi(\mathbf{1})}$ sont linéairement indépendants. Ceci implique que $\overrightarrow{\varphi(w)} \neq \overrightarrow{0}$ et donc, que $\varphi(w)$ n'est pas fermé. □

Une autre propriété remarquable des morphismes parallélogrammes est qu'ils préservent aussi les chemins simples. Ce résultat est obtenu comme corollaire du théorème 2.2.6 et le reste de cette section est consacré à sa démonstration.

Considérons le réseau parallélogramme illustré à la figure 2.4. Chaque sommet $x\overrightarrow{\varphi(\mathbf{0})} + y\overrightarrow{\varphi(\mathbf{1})}$ de $G(\varphi)$, où $x, y \in \mathbb{Z}$, est appelé un *carrefour*. Un non-carrefour p est appelé *sommet interne de type* a s'il existe un chemin discret $(p', \varphi(a))$ visitant p tel que p' est un carrefour. Notons que si p est un sommet interne de type a, alors c'est aussi un sommet interne de type $\overline{a}$. La proposition suivante est une conséquence de la correspondance entre réseau parallélogramme et pavage régulier (proposition 2.1.7) et donne une description des sommets de $G(\varphi)$ en fonction de leur degré.

Proposition 2.2.4. *Soit φ un morphisme parallélogramme et $p \in G(\varphi)$. Alors,*

$$\deg(p) = \begin{cases} 4, & \text{si } p \text{ est un carrefour;} \\ 2, & \text{sinon.} \end{cases}$$

Démonstration. D'abord, comme $G(\varphi)$ est un sous-graphe de $G(\mathbb{Z}^2)$, on a que $1 \leq \deg(p) \leq 4$ pour tout $p \in G(\varphi)$. De plus, $\deg(p) \neq 1$ car par la proposition 2.1.6, il existe un $p_0 \in \mathbb{Z}^2$ tel que p est dans le chemin $\varphi(p_0, \mathbf{0123})$. Comme $\varphi(\mathbf{0123})$ est fermé, tous les points du chemin sont de degré au moins 2.

Maintenant, supposons par contradiction qu'il existe un sommet $q \in G(\varphi)$ tel que $\deg(q) = 3$. Notons $\vec{a} = \overrightarrow{\varphi(\mathbf{0})}$ et $\vec{b} = \overrightarrow{\varphi(\mathbf{1})}$. Alors, il existe $p \in G(\varphi)$ tel que $\overrightarrow{q-p} = k_1\vec{a} + k_2\vec{b}$ où $k_1, k_2 \in \mathbb{Q} \setminus \mathbb{Z}$ (voir figure 2.5). Considérons les polyominos $P = (p, \varphi(\mathbf{0123}))$ et $Q = (q, \varphi(\mathbf{0123}))$. Par la proposition 2.1.7, P pave la plan par translation et Q est dans l'entourage de P. Donc, $\overrightarrow{q-p}$ est une $\mathbb{Z}$-combinaison linéaire des vecteurs $\vec{a}$ et $\vec{b}$, une contradiction. On a donc le résultat souhaité.

□

À présent, nous sommes en mesure de démontrer que $G(\mathbb{Z}^2)$ et $G(\varphi)$ sont homéomorphes, c'est-à-dire qu'ils ont essentiellement le même dessin. Observons d'abord que tout morphisme parallélogramme φ induit une subdivision T_φ de $\mathbb{Z}^2$: subdivisons les arêtes horizontales $\{u, v\}$ de $G(\mathbb{Z}^2)$ en ajoutant $|\varphi(\mathbf{0})| - 1$ sommets internes entre les deux carrefours u et v. De façon similaire, les arêtes verticales sont subdivisées en ajoutant $|\varphi(\mathbf{1})|-1$ nouveaux sommets. Par conséquent, les nouvelles chaînes horizontales (*resp.* verticales) liant deux carrefours adjacents dans le graphe original sont de longueur $|\varphi(\mathbf{0})|$ (*resp.* $|\varphi(\mathbf{1})|$), puisque $|\varphi(\mathbf{0})| = |\varphi(\mathbf{2})|$ et $|\varphi(\mathbf{1})| = |\varphi(\mathbf{3})|$.

Exemple 2.2.5. *Le graphe T_φ est représenté à la figure 2.6, où φ est le morphisme*

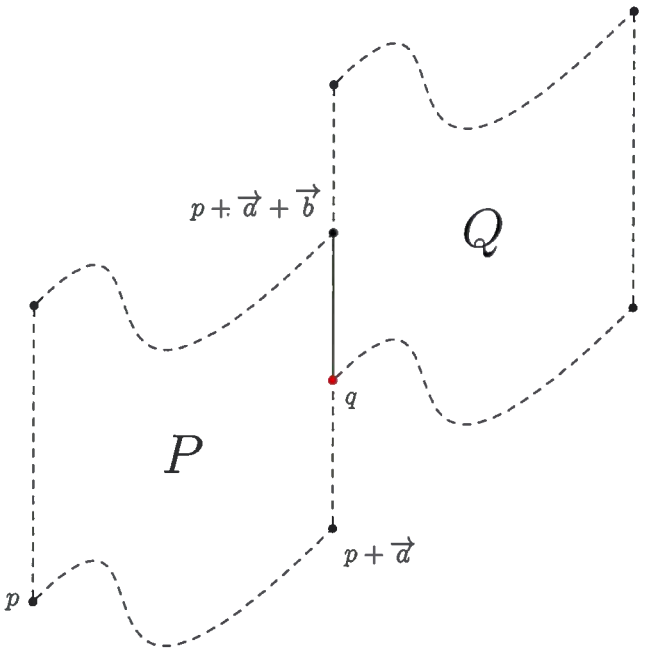
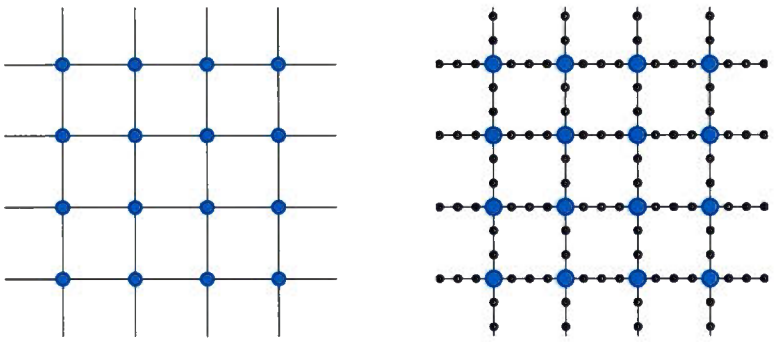


Figure 2.5: Situation décrite dans la preuve de la proposition 2.2.4.



(a) $G(\mathbb{Z}^2)$ (b) T_φ

Figure 2.6: Le graphe T_φ où $\varphi(\mathbf{0}) = \mathbf{0100}$ et $\varphi(\mathbf{1}) = \mathbf{121}$, obtenu par subdivision de $G(\mathbb{Z}^2)$

parallélogramme défini par $\varphi(\mathbf{0}) = \mathbf{0100}$ *et* $\varphi(\mathbf{1}) = \mathbf{121}$.

De là, on a le résultat principal de cette section :

Théorème 2.2.6. *Soit φ un morphisme parallélogramme. Alors, $G(\mathbb{Z}^2)$ et $G(\varphi)$ sont homéomorphes.*

Démonstration. Par définition, il suffit de démontrer que T_φ et $G(\varphi)$ sont isomorphes en tant que graphes. Posons $V(T_\varphi)$ et $V(G(\varphi))$ l'ensemble des sommets de T_φ et $G(\varphi)$ respectivement. De plus, soit (x, y) un sommet de T_φ. Par construction, on a

$$(x, y) \in \left\{ (\lfloor x \rfloor, \lfloor y \rfloor) + \left(\frac{k_1}{|\varphi(\mathbf{0})|}, \frac{k_2}{|\varphi(\mathbf{1})|} \right) \right\}$$

où $0 \leq k_1 < |\varphi(\mathbf{0})|$, $0 \leq k_2 < |\varphi(\mathbf{1})|$ et $k_1 k_2 = 0$. Maintenant, considérons la fonction $f : V(T_\varphi) \to V(G(\varphi))$ définie par

$$f(x, y) = \varphi(\lfloor x \rfloor, \lfloor y \rfloor) + \begin{cases} \overrightarrow{\mathrm{Pref}_{k_1}(\varphi(\mathbf{0}))}, & \text{si } k_2 = 0\,; \\ \overrightarrow{\mathrm{Pref}_{k_2}(\varphi(\mathbf{1}))}, & \text{si } k_1 = 0. \end{cases}$$

Intuitivement, la fonction f trouve le carrefour en bas ou à gauche $(\lfloor x \rfloor, \lfloor y \rfloor)$ le

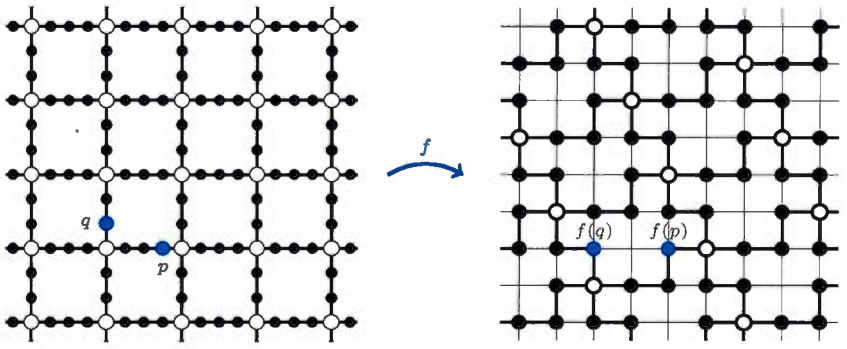


Figure 2.7: Effet de la fonction f sur deux sommets de T_φ

plus près de tout sommet (x, y) et puis considère le $k^{\text{ième}}$ sommet dans le chemin $\varphi((x, y), a)$ de $G(\varphi)$, où $k \in \{k_1, k_2\}$ et $a \in \{\mathbf{0}, \mathbf{1}\}$ (voir figure 2.7). On a que f est une bijection par construction. Il reste à montrer que $p, q \in \mathbb{Z}^2$ sont voisins dans T_φ si et seulement si $f(p)$ et $f(q)$ sont voisins dans $G(\varphi)$.

D'abord, pour tout $p \in \mathbb{Z}^2$ et $a \in \mathcal{F}$, soit s la suite de points où le $i^{\text{ième}}$ élément est $p + i\overrightarrow{a}$, pour $i = 0, 1, \ldots, |\varphi(a)|$ et considérons la suite $f(s)$ où le $i^{\text{ième}}$ élément est $f(p + i\overrightarrow{a})$, pour $i = 0, 1, \ldots, |\varphi(a)|$. Alors,

$$f(p + i\overrightarrow{a}) = \varphi(p) + \overrightarrow{\text{Pref}_i(\varphi(a))}.$$

Par conséquent, s est une chaîne de T_φ si et seulement si $f(s)$ est une chaîne de $G(\varphi)$, puisque (p, a) est un chemin discret de T_φ si et seulement si $(p, \varphi(a))$ est un chemin discret de $G(\varphi)$.

En d'autres termes, les chemins reliant les sommets de coordonnées entières dans T_φ sont isomorphes aux chemins reliant les carrefours dans $G(\varphi)$. Par la proposition 2.2.4, les degrés des sommets concordent et on a donc considéré tous les voisins possibles. □

Un corollaire immédiat du théorème 2.2.6 est que tout morphisme parallélo-

gramme φ préserve les chemins fermés et simples. Par conséquent, les réseaux parallélogrammes sont simplement une image déformée du réseau carré $G(\mathbb{Z}^2)$ et constituent une généralisation de ce dernier.

2.3 Application à la théorie des codes

Bien que le théorème 2.2.6 soit de nature topologique, il présente aussi un intérêt pour l'algèbre et la combinatoire des mots. En effet, cette section fournit une démonstration d'une conjecture de (Blondin Massé *et al.*, 2012), établissant ainsi l'équivalence des réseaux parallélogrammes et des codes circulaires. Avant de démontrer la conjecture, il convient de démontrer le résultat suivant concernant la distance entre deux sommets d'un réseau parallélogramme.

Lemme 2.3.1. *Soit φ un morphisme parallélogramme et p un sommet de $G(\varphi)$. De plus, soit $q = p + k\overrightarrow{\varphi(a)}$ pour un certain $a \in \mathcal{F}$ et un entier positif k.*

(i) *Si p et q sont des carrefours, alors $(p, \varphi(a)^k)$ est l'unique plus court chemin de p à q et $\mathrm{dist}_{G(\varphi)}(p, q) = k|\varphi(a)|$.*

(ii) *Si p et q sont des sommets internes de type b, où $b \in \mathcal{F}$ et $b \neq a, \overline{a}$, alors $\mathrm{dist}_{G(\varphi)}(p, q) > k|\varphi(a)|$.*

Démonstration. (i) Par définition de réseau parallélogramme, il existe un chemin de p à q dans $G(\varphi)$ donné par $\varphi(a)^k$. Ce chemin est aussi le plus court. En effet, tout chemin de p à q est de la forme $\varphi(a)^m \cdot \varphi(\overline{a})^n \cdot \varphi(b)^x \cdot \varphi(\overline{b})^y$ où $b = (a+1) \bmod 4$, $m + n = k$, $x + y = 0$ et $m, n, x, y \geq 0$. Comme les valeurs de m, n, x et y sont positives, le chemin le plus court est réalisé lorsque $n = x = y = 0$ et $m = k$.

(ii) Un plus court chemin de p à q s'obtient en allant vers le carrefour le plus près, puis en suivant $\varphi(b)^k$ jusqu'à q. Comme p et q sont des sommets internes, le nombre de sommets dans ce chemin le plus court est strictement plus grand que $k|\varphi(a)|$. □

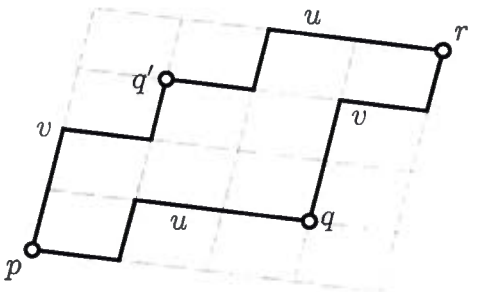


Figure 2.8: Représentation géométrique des chemins u et v

Rappelons que tout sous-ensemble $X \subseteq \Sigma^*$ est un code circulaire si et seulement si tout mot de X^* admet une unique factorisation en mots de X à permutation des facteurs près (section 1.3). Nous démontrons à présent la conjecture 37 de (Blondin Massé *et al.*, 2012) afin d'établir une condition nécessaire et suffisante pour que l'ensemble $\varphi(\mathcal{F}) = \{\varphi(\mathbf{0}), \varphi(\mathbf{1}), \varphi(\mathbf{2}), \varphi(\mathbf{3})\}$ soit un code circulaire Notons que nous avons reformulé l'énoncé de la conjecture afin de rendre la notation compatible avec celle de la présente thèse.

Théorème 2.3.2 (Conjecture 37 de (Blondin Massé *et al.*, 2012)). *Soit φ un morphisme parallélogramme. Alors, $\varphi(\mathcal{F})$ est un code circulaire si et seulement si $\varphi(\mathbf{0})$ et $\varphi(\mathbf{1})$ sont tous deux primitifs.*

Démonstration. ($\Rightarrow$) Si $\varphi(\mathcal{F})$ est un code circulaire, alors chacun de ses éléments doit être primitif. En effet, supposons qu'il existe un code circulaire X et un mot non-simple $y = x^k \in X$. Alors, par le théorème 1.3.2, $x \in X$. Ceci est une contradiction car y se factorise alors de plusieurs façons différentes. Donc, en particulier, $\varphi(\mathbf{0})$ et $\varphi(\mathbf{1})$ sont primitifs.

($\Leftarrow$) Soit $M = \varphi(\mathcal{F})^*$. Montrons que M est très pur. Supposons par contradiction que ce n'est pas le cas, c'est-à-dire qu'il existe $u, v \in \mathcal{F}^*$ tels que $uv, vu \in M$ mais $u, v \notin M$.

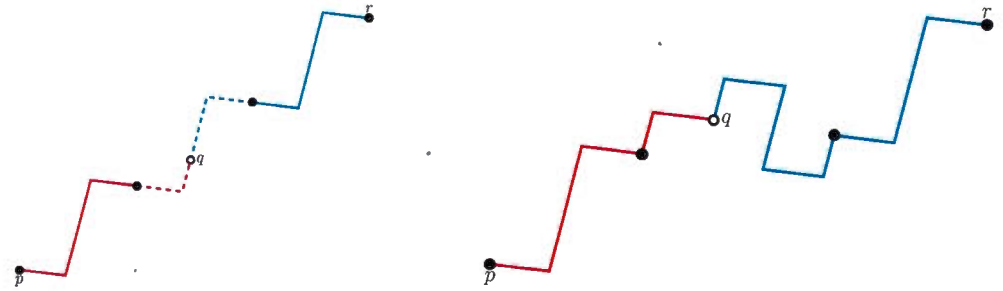
Clairement, $\overrightarrow{u\widehat{v}} = \overrightarrow{v\widehat{u}}$, ce qui implique que les chemins discrets (p, uv) et (p, vu) de $G(\varphi)$ terminent au même point, pour tout $p \in \mathbb{Z}^2$. De plus, il existe des carrefours $p, r \in \mathbb{Z}^2$ de $G(\varphi)$ et des sommets internes q, q' de type a, a' de $G(\varphi)$ tels que les chemins discrets (p, u) et $(r, \widehat{v})$ terminent tous deux à q et les chemins discrets (p, v) et $(r, \widehat{u})$ terminent tous deux à q'. Cette situation est illustrée à la figure 2.8. Il y a deux cas à considérer.

D'abord, supposons que $uv = \varphi(b)^k$ pour un certain $b \in \mathcal{F}$, c'est-à-dire que uv est un plus court chemin de p à r. Comme $|uv| = |vu|$ et comme $(p, \varphi(b)^k)$ est l'unique plus court chemin de p à r (lemme 2.3.1(i)), on déduit que $uv = \varphi(b)^k = vu$. On a alors la situation illustrée à la figure 2.9 (a). Posons $u = u'u''$ et $v = v'v''$, où $u', v'' \in M$ et $u''v' = \varphi(b)$. Notons que cette décomposition existe car $uv \in M$. De plus, comme q est un sommet interne, on a nécessairement que u et v ne sont pas dans M. Ceci entraîne que cette décomposition est unique. Maintenant, $\varphi(b)^k = uv = vu = v'v''u'u''$, ce qui implique que v' est un préfixe de $\varphi(b)$ et u'' est un suffixe de $\varphi(b)$. Par conséquent, $u''v' = \varphi(b) = v'u''$, et on a par la proposition 1.3.1 que $\varphi(b)$ n'est pas primitif, contredisant ainsi notre hypothèse.

Sinon, $uv \neq \varphi(b)^k$ pour un certain $b \in \mathcal{F}$ et on a la situation illustrée à la figure 2.9 (b). Notons que, par le même argument que précédemment, u et v ne sont pas dans M. Soit u' et v' les mots maximaux de $\mathcal{F}^*$ tels que $\varphi(u')$ est un préfixe de u et $\varphi(v')$ est un suffixe de v. Posons

$$Q = \left\{ q + \overrightarrow{\varphi(u'')} \mid u'' \in \mathrm{Pref}(u') \right\} \cup \left\{ q' + \overrightarrow{\varphi(\widehat{v''})} \mid v'' \in \mathrm{Suff}(v') \right\}.$$

Autrement dit, Q est l'ensemble des points atteignables à partir de q et q' en considérant des chemins entre carrefours de $G(\varphi)$. Comme $u, v \notin M$ et comme q et q' sont des sommets internes, tous les éléments de Q sont des sommets internes. De plus, ils sont tous de type a ou a' c'est-à-dire du même type que q ou q'.



(a) Cas 1 : le chemin u en rouge et le chemin v en bleu. Les sous-chemins u'' et v' sont pointillés. Les points noirs sont des carrefours

(b) Cas 2 : le chemin u en rouge et le chemin v en bleu. Les points noirs sont des carrefours

Figure 2.9: Représentation géométrique du chemin uv.

Cependant, il doit exister deux éléments distincts $s, s' \in Q$ tels que $s' = s + \overrightarrow{\varphi(b)}$, où $b \neq a', \overline{a'}$: sinon, on aurait que $uv = \varphi(a') = vu$ qui est le cas considéré dans le paragraphe précédent. Mais le lemme 2.3.1 s'applique à s et s', de sorte que $\mathrm{dist}_{G(\varphi)}(s, s') > |\varphi(b)|$, contredisant le fait que s' peut être atteint depuis s en suivant le chemin $(s, \varphi(b))$. □

Par exemple, si φ est le morphisme parallélogramme défini par $\varphi(\mathbf{0}) = \mathbf{0100}$ et $\varphi(\mathbf{1}) = \mathbf{11}$, alors $\varphi(\mathcal{F}) = \{\mathbf{0100}, \mathbf{11}, \mathbf{2232}, \mathbf{33}\}$ est un code circulaire puisque $\mathbf{0100}$ et $\mathbf{11}$ sont tous deux primitifs.

CHAPITRE III

ARITHMÉTIQUE DES POLYOMINOS

Ce chapitre a pour but l'étude d'une classe fondamentale de polyominos sans trou appelés *polyominos premiers*. De façon analogue aux nombres premiers, ces figures discrètes particulières constituent en quelque sorte les éléments atomiques à partir desquels tous les ensembles de pixels sont formés. Cette théorie trouve son origine dans la thèse de doctorat de X. Provençal (Provençal, 2008) ainsi que dans l'étude des pavages du plan par des polyominos, mais nous l'étendons en développant de nouveaux résultats. Dans un premier temps, nous définissons le concept de polyomino premier. Ensuite, nous étudions l'existence et l'unicité de la décomposition en polyominos premiers. Finalement, nous donnons un algorithme de factorisation de tout polyomino P sans trou, polynomial en le périmètre de P. Notons qu'une partie du présent chapitre a été présentée lors de la $8^{\text{ième}}$ Conférence internationale sur les langages et la théorie des automates et leurs applications (LATA 2014), qui avait lieu du 10 au 14 mars 2014 à Madrid en Espagne (Blondin Massé *et al.*, 2014).

3.1 Polyominos premiers

Rappelons qu'étant donné un morphisme parallélogramme φ, le réseau parallélogramme $G(\varphi)$ est homéomorphe au réseau carré $G(\mathbb{Z}^2)$ et donc préserve les chemins fermés et simples. Par conséquent, si w est le mot de contour d'un poly-

omino sans trou, $\varphi(w)$ est le mot de contour du polyomino obtenu en remplaçant chaque pas unitaire de w par son image sous φ (voir figure 3.1). Inversement, tout chemin fermé et simple de $G(\varphi)$ est obtenu par un unique chemin fermé et simple de $\mathbb{Z}^2$ en lui appliquant le morphisme φ. Ce résultat est à la base du concept de polyomino premier, introduit pour la première fois dans (Provençal, 2008).

Définition 3.1.1. *Soit Q un polyomino admettant w comme mot de contour et $P = \varphi(\mathbf{0123})$ le polyomino obtenu du morphisme parallélogramme φ. Le* produit *de P par Q, noté $P \circ_\varphi Q$, est le polyomino sans trou dont le mot de contour est $\varphi(w)$.*

Remarque. *Comme nous verrons plus loin, il est possible d'étendre l'opération de produit à n polyominos. On a alors des morphismes parallélogrammes $\varphi_1, \varphi_2, \ldots, \varphi_{n-1}$ et un mot w tels que $(\varphi_1 \circ \varphi_2 \cdots \circ \varphi_{n-1})(w)$, où $\circ$ dénote la composition de morphisme. Par conséquent, on dit parfois que $P \circ_\varphi Q$ est la* composition des polyominos P et Q. *Par abus de langage, on parle également de la* composition $\varphi(w)$ *afin d'accentuer cette interprétation.*

Un polyomino R différent de la cellule unité est dit *premier* si pour toute paire de polyominos P et Q, l'égalité $R = P \circ_\varphi Q$ implique $R = P$ ou $R = Q$. Sinon, on dit que R est *composé*. De plus, on dit que P et Q sont des *facteurs de* R, conformément à la terminologie usuelle utilisée en arithmétique.

Tout comme le chiffre 1 n'est ni premier, ni composé, la cellule unité engendrée par le morphisme parallélogramme Id n'est ni première, ni composée. En effet, soit $\varphi = \mathrm{Id}$, $C = \varphi(\mathbf{0123}) = \mathbf{0123}$ et Q un polyomino quelconque. Alors, on a $C \circ_\varphi Q = Q$ par définition. Par conséquent, C est facteur de tout polyomino. De plus, si on pose $P^n := \underbrace{P \circ_\varphi P \circ_\varphi \cdots \circ_\varphi P}_{n \text{ fois}}$ pour tout $n \geq 0$ et $P^0 = C$, alors $C^n \circ_\varphi Q = Q$. La cellule unité a donc un comportement analogue au chiffre 1 en regard de l'opération de composition.

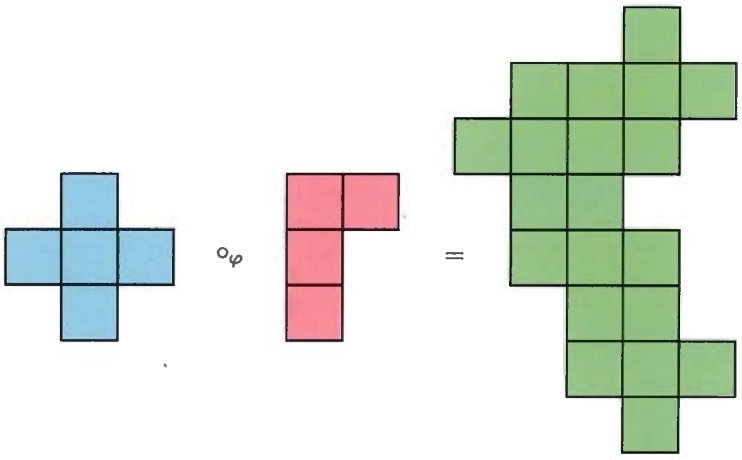


Figure 3.1: Produit du polyomino parallélogramme $\varphi(\mathbf{0123})$ où $\varphi(\mathbf{0}) = \mathbf{010}$ et $\varphi(1) = \mathbf{121}$ avec le polyomino $\mathbf{01^2012^23^3}$. Le résultat est le polyomino $\mathbf{010 \cdot 121 \cdot 121 \cdot 010 \cdot 121 \cdot 232 \cdot 232 \cdot (303)^3}$

Afin de simplifier certains résultats, il est utile de définir le concept de primalité de polyomino en fonction de réseaux parallélogramme.

Définition 3.1.2. *Un polyomino sans trou P est* composé *si et seulement s'il existe un morphisme parallélogramme φ différent du morphisme identité tel que $G(P)$ est différent de $G(\varphi(\mathbf{0123}))$ et tel qu'il est un sous-graphe de $G(\varphi)$. Autrement, P est dit* premier.

Remarquons aussi que la définition 3.1.2 est équivalente à celle donnée dans (Provençal, 2008) : soit P un polyomino admettant w comme mot de contour et φ un morphisme parallélogramme. Alors, le produit $\varphi(w)$ est bien un sous-graphe de $G(\varphi) = \bigcup_{\gamma \in \Gamma(\mathbb{Z}^2)} G(\varphi(\gamma))$ par définition. Inversement, comme φ préserve les chemins fermés et simples, si $G(P)$ est un sous-graphe de $G(\varphi)$, alors P est le produit de Q et $\varphi(\mathbf{0123})$ où Q admet le mot de contour u tel que $w = \varphi(u)$.

Un avantage de la définition 3.1.2 est son interprétation géométrique. En effet, rappelons que le réseau parallélogramme $G(\varphi)$ induit un pavage régulier du plan par la tuile parallélogramme $\varphi(\mathbf{0123})$ (voir proposition 2.1.7). Alors, P est com-

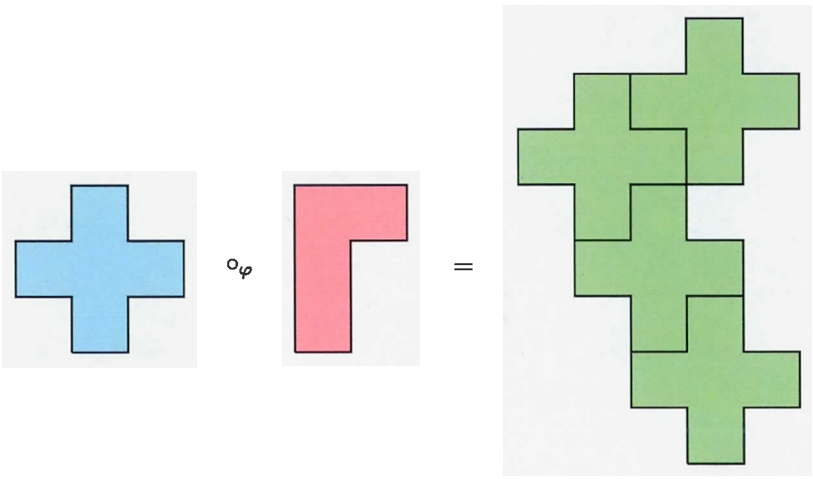
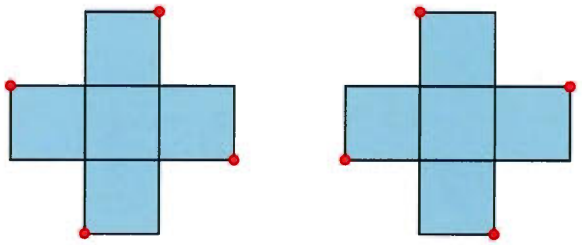


Figure 3.2: Le polyomino P codé par $\mathbf{010 \cdot 121 \cdot 121 \cdot 010 \cdot 121 \cdot 232 \cdot 232 \cdot (303)^3}$ est composé puisqu'il existe un pavage par translation de P par la tuile parallélogramme définie par le morphisme $\varphi(\mathbf{0}) = \mathbf{010}$ et $\varphi(1) = \mathbf{121}$

posé s'il existe une tuile parallélogramme Q différente du polyomino d'aire 1 formé d'un seul pixel telle que Q pave P par translation. La figure 3.2 illustre bien cette interprétation.

Remarque. *Par le théorème 1.5.2, pour tout morphisme parallélogramme, on associe une unique tuile parallélogramme* $\varphi(\mathbf{0123})$. *Par contre, l'inverse n'est pas vrai. En effet, certaines tuiles, appelées* doubles parallélogrammes, *admettent deux factorisations* $XY\widehat{X}\widehat{Y}$ *distinctes, c'est-à-dire qu'elles sont engendrées par deux morphismes parallélogrammes différents. Par exemple, les morphismes parallélogrammes définis par* $\varphi(0) = 010$ *et* $\varphi(1) = 121$ *et* $\varphi'(0) = 030$ *et* $\varphi'(1) = 101$ *engendrent tous les deux la tuile* $\mathbf{010121232303}$. *La figure 3.3 illustre ces deux factorisations. Ce type particulier de polyominos est étudié en détail dans (Blondin Massé* et al., *2011; Blondin Massé* et al., *2012; Blondin Massé, 2012).*

Par extension de la définition 3.1.2, on dit que $w \in \mathcal{F}^*$ est *premier* s'il code le contour d'un polyomino premier. De même, φ est premier si $\varphi(\mathbf{0123})$ est le mot de contour d'un polyomino premier. La proposition suivante donne une définition



(a) **010·121·232·303** (b) **030·101·212·323**

Figure 3.3: Le polyomino codé par **010121232303** admet deux factorisations $XY\widehat{X}\widehat{Y}$

équivalente de primalité.

Proposition 3.1.3. *Un polyomino P est composé s'il existe un mot de contour $u \neq \mathbf{0123}$ et un morphisme parallélogramme φ différent du morphisme identité tel que $\varphi(u)$ soit le mot de contour de P.*

Démonstration. Par définition, si P est composé, alors il existe un morphisme parallélogramme $\varphi \neq \mathrm{Id}$ tel que

$$G(P) \subseteq G(\varphi) = \bigcup_{\gamma \in \Gamma(\mathbb{Z}^2)} G(\varphi(\gamma)).$$

Puisque φ préserve les chemins fermés et simples et que $G(P)$ est un cycle, il existe un mot de contour correspondant à un chemin $\gamma \in \Gamma(\mathbb{Z}^2)$ tel que $G(\varphi(\gamma)) = G(P)$. Inversement, s'il existe un morphisme parallélogramme $\varphi \neq \mathrm{Id}$ et un mot de contour $u \neq \mathbf{0123}$ tels que $\varphi(u)$ soit le mot de contour de P, alors on a bien que

$$G(P) = G(\varphi(u)) \subseteq G(\varphi).$$

□

Nous tournons à présent notre attention vers la décomposition de polyominos

composés. D'abord, l'exemple suivant démontre que décider de la primalité d'un polyomino est au moins aussi difficile que décider si un nombre est premier.

Exemple 3.1.4. *Soit $w = 0^n12^n3$. Alors, le polyomino codé par w est premier si et seulement si n est premier. En effet, par la proposition 3.1.3, si $n = mk$, alors il suffit de choisir le morphisme $\varphi(\mathbf{0}) = \mathbf{0}^k$ et $\varphi(\mathbf{1}) = \mathbf{1}$ avec $u = \mathbf{0}^m\mathbf{1}\mathbf{2}^m\mathbf{3}$, de sorte que w code le contour d'un polyomino composé. Inversement, il doit exister un morphisme parallélogramme $\varphi \neq \mathrm{Id}$ et un mot $u \neq \mathbf{0123}$ tels que $\varphi(u) = w$. Nécessairement, $\varphi(\mathbf{1}) = \mathbf{1}$ par la proposition 2.1.3. Ceci implique que $\varphi(\mathbf{0}) = \mathbf{0}^k$ avec $k > 1$. Finalement, $u = \mathbf{0}^m\mathbf{1}\mathbf{2}^m\mathbf{3}$ où $m > 1$ et on a bien $n = mk$.*

Le résultat développé dans l'exemple 3.1.4 s'étend à tous les polyominos afin de fournir un critère de primalité.

Proposition 3.1.5. *Soit P un polyomino d'aire n. Alors, l'aire de tout facteur de P divise n.*

Démonstration. Soit P un polyomino composé d'aire n. Alors, il existe un polyomino Q (d'aire m) pavant P par translation. Autrement dit, P est composé de k copies de Q. Par conséquent, l'aire de P est $mk = n$, d'où le résultat. □

On en déduit immédiatement le corollaire suivant.

Corollaire 3.1.6. *Soit p un nombre premier. Alors, tout polyomino d'aire p est premier.*

Remarquons que l'inverse du corollaire 3.1.6 est faux. En effet, nous verrons à la fin du présent chapitre qu'il existe plusieurs polyominos premiers dont l'aire est un nombre composé.

3.2 Théorème fondamental de l'arithmétique pour les polyominos

Il est maintenant naturel de se demander si un analogue du théorème fondamental de l'arithmétique est valide pour les polyominos : étant donné un polyomino sans trou Q, existe-t-il une unique factorisation de Q ? Autrement dit, existe-t-il d'uniques polyominos premiers $P_1, P_2, \ldots, P_k$ tels que $Q = P_1 \circ_{\varphi_1} P_2 \circ_{\varphi_2} \cdots \circ_{\varphi_k} P_k$? Dans un premier temps, la question de l'existence est étudiée.

D'abord, remarquons que l'opération $\circ_\varphi$ n'est pas commutative puisque le fait que $P \circ_\varphi R$ soit bien définie n'implique pas nécessairement que $R \circ_\varphi P$ soit définie. En effet, si R n'est pas un parallélogramme, alors il n'existe pas de morphisme parallélogramme φ tel que $\varphi(\mathbf{0123})$ est le mot de contour de R. Il est alors impossible de définir $R \circ_\varphi P$ pour tout polyomino P.

Par contre, l'opération de composition est associative lorsque restreinte aux polyomninos parallélogrammes : posons respectivement $\mathcal{M}$, $\mathcal{H}$ et $\mathcal{P}$ l'ensemble des morphismes, des morphismes homologues et des morphismes parallélogrammes sur $\mathcal{F}$. On rappelle que le morphisme identité Id est parallélogramme. De plus, la composition de morphismes est associative. Par conséquent, $\mathcal{M}$ muni de l'opération de composition est un monoïde, c'est-à-dire un ensemble muni d'une opération binaire associative et d'un élément identité. Nous montrons alors que $\mathcal{H}$ et $\mathcal{P}$ sont des sous-monoïdes de $\mathcal{M}$, de façon à obtenir comme corollaire l'associativité de $\circ_\varphi$ pour les polyominos parallélogrammes.

Théorème 3.2.1. *En regard de l'opération de composition, $\mathcal{H}$ est un sous-monoïde de $\mathcal{M}$ et $\mathcal{P}$ est un sous-monoïde de $\mathcal{H}$.*

Démonstration. On a déjà remarqué que Id est un morphisme parallélogramme.

Montrons que $\mathcal{H}$ est fermé pour $\circ$. Soit φ et φ' dans $\mathcal{H}$. Alors, pour tout $a \in \mathcal{F}$,

$$\begin{aligned}
(\varphi \circ \varphi')(a) &= \varphi(\varphi'(a)) && \text{par définition} \\
&= \varphi(\widehat{\varphi'(\overline{a})}) && \text{car } \varphi' \text{ est homologue} \\
&= \widehat{\varphi(\varphi'(\overline{a}))} && \text{car } \varphi \text{ est homologue et par (2.3)} \\
&= \widehat{(\varphi \circ \varphi')}(\overline{a}) && \text{par définition.}
\end{aligned}$$

Par conséquent, $\varphi \circ \varphi'$ est un morphisme homologue. $\mathcal{H}$ est donc bien un sous-monoïde de $\mathcal{M}$. Maintenant, soit φ et φ' dans $\mathcal{P}$. Alors, pour tout $a \in \mathcal{F}$,

$$\mathrm{Fst}((\varphi \circ \varphi')(a)) = \mathrm{Fst}(\varphi(\varphi'(a))) = a.$$

En effet, φ et φ' étant homologues, on a par définition que $\varphi(\varphi'(a)) = \varphi(au) = aw$ où $u,\ w \in \mathcal{F}^*$. Il reste à montrer que $(\varphi \circ \varphi')(\mathbf{0123})$ est fermé et simple. Or, le théorème 2.2.6 implique que les morphismes parallélogrammes préservent les chemins fermés et simples. On a donc le résultat souhaité. □

Le théorème suivant garantit l'existence d'une telle factorisation pour tout polyomino sans trou.

Théorème 3.2.2 (Existence d'une factorisation). *Soit Q un polyomino sans trou différent de la cellule unité. Alors, il existe des polyominos parallélogrammes premiers $P_1, P_2, \ldots, P_k$ et un polyomino premier P tels que $Q = (P_1 \circ_{\varphi_1} P_2 \circ_{\varphi_2} \cdots P_k) \circ_{\varphi_k} P$.*

Le lemme suivant est utile afin de démontrer le théorème 3.2.2.

Lemme 3.2.3. *Si Q est un polyomino parallélogramme composé, alors l'égalité $Q = P \circ_\varphi R$ implique que R est un polyomino parallélogramme.*

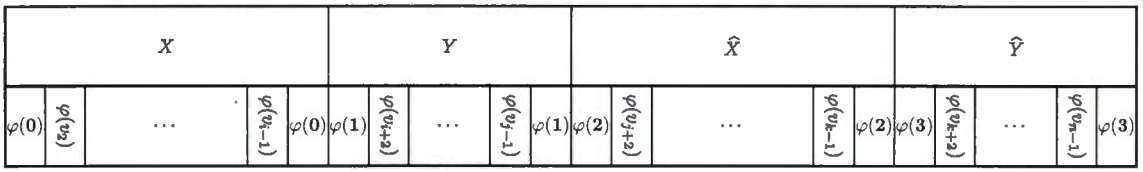


Figure 3.4: Situation décrite dans la preuve du lemme 3.2.3

Démonstration. Posons w, u, v les mots de contour de Q, P, R respectivement. Posons également φ un morphisme parallélogramme associé à P de sorte que $\varphi(\mathbf{0123}) = u$. On a par hypothèse que

$$w \equiv XY\widehat{X}\widehat{Y} \equiv \varphi(v),$$

où $X, Y \in \mathcal{F}^*$. Cette situation est illustrée à la figure 3.4. Comme les éléments

$$\begin{aligned} X &= \varphi(\mathbf{0})\varphi(v_2)\cdots\varphi(v_{i-1})\varphi(\mathbf{0}) & Y &= \varphi(\mathbf{1})\varphi(v_{i+2})\cdots\varphi(v_{j-1})\varphi(\mathbf{1}) \\ \widehat{X} &= \varphi(\mathbf{2})\varphi(v_{j+2})\cdots\varphi(v_{k-1})\varphi(\mathbf{2}) & \widehat{Y} &= \varphi(\mathbf{3})\varphi(v_{k+2})\cdots\varphi(v_{n-1})\varphi(\mathbf{3}) \end{aligned}$$

sont respectivement homologues, on a

$$\begin{aligned} \widehat{X} &= \varphi(\mathbf{0})\varphi(v_2)\widehat{\cdots\varphi(v_{i-1})}\varphi(\mathbf{0}) \\ &= \widehat{\varphi(\mathbf{0})}\widehat{\varphi(v_{i-1})}\cdots\widehat{\varphi(v_2)}\widehat{\varphi(\mathbf{0})} \\ &= \varphi(\widehat{\mathbf{0}})\varphi(\widehat{v_{i-1}})\cdots\varphi(\widehat{v_2})\varphi(\widehat{\mathbf{0}}) \\ &= \varphi(\mathbf{2})\varphi(v_{j+2})\cdots\varphi(v_{k-1})\varphi(\mathbf{2}). \end{aligned}$$

Par conséquent, on a $\mathbf{0}v_2\widehat{\cdots v_{i-1}}\mathbf{0} = \widehat{\mathbf{0}}\widehat{v_{i-1}}\cdots\widehat{v_2}\widehat{\mathbf{0}} = \mathbf{2}v_{j+2}\cdots v_{k-1}\mathbf{2}$. De manière analogue, on calcule que $\mathbf{1}v_{i+2}\widehat{\cdots v_{j-1}}\mathbf{1} = \widehat{\mathbf{1}}\widehat{v_{j-1}}\cdots\widehat{v_{i+2}}\widehat{\mathbf{1}} = \mathbf{3}v_{k+2}\cdots v_{n-1}\mathbf{3}$. On conclut alors que $v = AB\widehat{A}\widehat{B}$ où $A = \mathbf{0}v_2\cdots v_{i-1}\mathbf{0}$ et $B = \mathbf{1}v_{i+2}\cdots v_{j-1}\mathbf{1}$, d'où le résultat. □

Nous sommes maintenant prêts à démontrer le théorème 3.2.2. De façon analogue à la preuve d'existence du théorème fondamental de l'arithmétique, nous procédons par induction sur l'aire des polyominos.

Démonstration. Soit Q un polyomino sans trou d'aire strictement supérieure à 1. Si Q est d'aire 2 ou 3, alors il est premier par le corollaire 3.1.6 et le résultat est vérifié. Maintenant, supposons que le résultat soit vrai pour tout polyomino d'aire comprise entre 1 et n exclusivement. Soit Q un polyomino d'aire n. Si Q est premier, il n'y a rien à démontrer. Sinon, il existe par définition deux polyominos P et R différents du carré unité, où P est parallélogramme, tels que

$$Q = P \circ_{\varphi} R.$$

Par la proposition 3.1.5, l'aire de P et celle de R sont strictement inférieures à celle de Q. Donc, par hypothèse d'induction, il existe des polyominos parallélogrammes premiers $P_1, P_2, \cdots P_k, R_1, R_2, \ldots, R_l$ et deux polyominos premiers P' et R' tels que

$$Q = P \circ_{\varphi} R = (P_1 \circ_{\varphi_1} P_2 \circ_{\varphi_2} \cdots P_k \circ_{\varphi_k} P') \circ_{\varphi_{k+1}} (R_1 \circ_{\varphi_{k+2}} R_2 \circ_{\varphi_{k+3}} \ldots R_l \circ_{\varphi_{k+l}} R').$$

Or, par le lemme 3.2.3, P' est parallélogramme. On conclut donc grâce au théorème 3.2.1 que $Q = (P_1 \circ_{\varphi_1} P_2 \circ_{\varphi_2} \cdots P_k \circ_{\varphi_k} P' \circ_{\varphi_{k+1}} R_1 \circ_{\varphi_{k+2}} R_2 \circ_{\varphi_{k+3}} \ldots R_l) \circ_{\varphi_{k+l}} R'$, d'où le résultat. □

Exemple 3.2.4. *Considérons le polyomino composé Q codé par*

$$w = (\mathbf{0010110010})^2\mathbf{1123221123223323221123221123223323 22}(\mathbf{33001033})^2.$$

Alors, $Q = (P_1 \circ_{\varphi_1} P_2) \circ_{\varphi_2} R$ où P_1 et P_2 sont respectivement donnés par les morphismes parallélogrammes $\varphi_1(\mathbf{0}) = \mathbf{0010}$ et $\varphi_1(\mathbf{1}) = \mathbf{11}$, et $\varphi_2(\mathbf{0}) = \mathbf{010}$

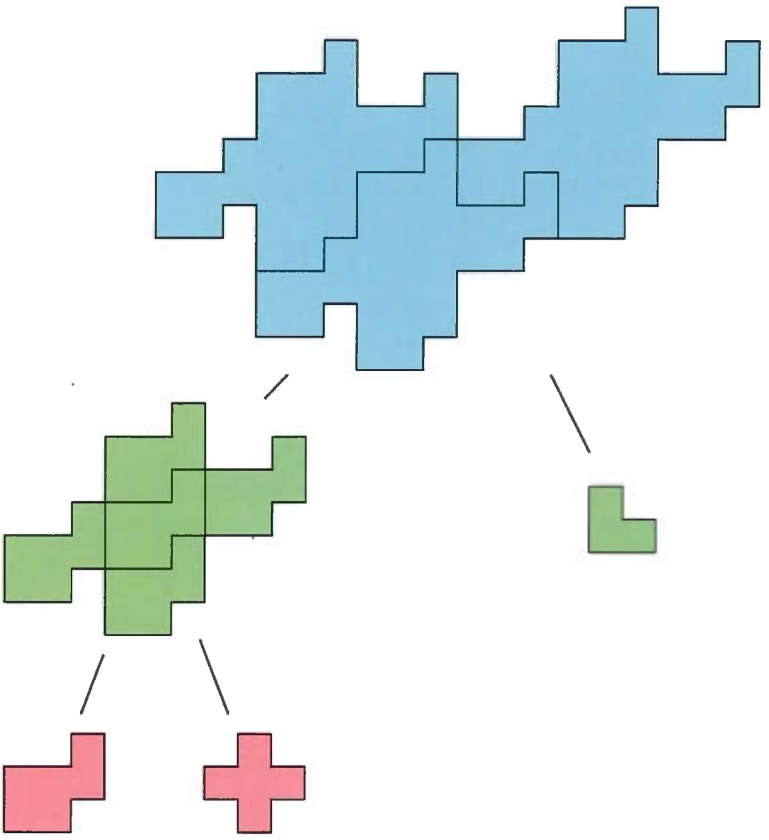


Figure 3.5: Factorisation du polyomino Q

et $\varphi_2(\mathbf{1}) = \mathbf{121}$, *avec* R *codé par* $u = \mathbf{00121233}$. *La figure 3.5 illustre cette décomposition.*

La question de l'unicité demande quant à elle un peu plus d'attention et demeure ouverte. Cependant, nous croyons que le concept de réseau parallélogramme constitue une étape importante pour son éventuelle démonstration. Notons aussi, qu'à ce jour, aucun calcul informatique a fourni de décomposition multiple pour aucun polyomino. Nous posons donc la conjecture suivante :

Conjecture 3.2.5 (Unicité de la décomposition). *Soit* Q *un polyomino sans trou d'aire strictement supérieure à* 1. *Alors, il existe d'uniques polyominos parallélogrammes premiers* $P_1, P_2, \ldots, P_k$ *et un unique polyomino premier* P *tels que*

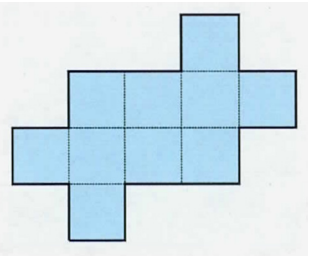


Figure 3.6: Un polyomino composé Q admettant un double parallélogramme comme facteur premier

$$Q = (P_1 \circ_{\varphi_1} P_2 \circ_{\varphi_2} \cdots P_k) \circ_{\varphi_k} P.$$

Il est intéressant de noter que les polyominos double parallélogrammes, c'est-à-dire les polyominos admettant deux factorisations $XY\widehat{X}\widehat{Y}$ différentes, ne semblent pas compromettre l'unicité de la factorisation. À titre d'exemple, considérons le polyomino Q de la figure 3.6. Ce polyomino est composé car $Q = P \circ_\varphi R$, où P est induit par le morphisme parallélogramme φ défini par $\varphi(0) = \mathbf{010}$ et $\varphi(1) = \mathbf{121}$ et R est codé par le mot de contour $u = \mathbf{001223}$. Comme illustré à la figure 3.3, P est un double parallélogramme car il est aussi induit par le morphisme parallélogramme φ' défini par $\varphi'(0) = \mathbf{030}$ et $\varphi'(1) = \mathbf{101}$. Or, il n'existe pas de mot de contour u' tel que $\varphi'(u')$ est le mot de contour de Q. Ceci suggère que $Q = P \circ_\varphi R$ est l'unique factorisation de Q.

3.3 Algorithme de factorisation

À présent, l'aspect algorithmique de la factorisation de polyominos est abordé : étant donné un mot de contour w, trouver des morphismes parallélogrammes $\varphi_1, \varphi_2, \ldots, \varphi_n$ et un mot de contour u tels que

$$w = (\varphi_1 \circ \varphi_2 \circ \cdots \circ \varphi_n)(u).$$

À cette fin, nous développons un algorithme polynomial effectuant cette factorisation. Sa construction repose sur le résultat suivant, démontré dans (Blondin Massé *et al.*, 2012) :

Proposition 3.3.1 (Blondin Massé *et al.*, 2012). *Soit* $w = xy\widehat{x}\widehat{y}$ *le mot de contour d'un polyomino parallélogramme sans trou. Alors,* $\mathrm{Fst}(x) = \mathrm{Lst}(x)$ *et* $\mathrm{Fst}(y) = \mathrm{Lst}(y)$. *De plus, les premières lettres de* x, y, $\widehat{x}$ *et* $\widehat{y}$ *sont mutuellement distinctes, c'est-à-dire que*

$$\{\mathrm{Fst}(x), \mathrm{Fst}(y), \mathrm{Fst}(\widehat{x}), \mathrm{Fst}(\widehat{y})\} = \{\mathbf{0}, \mathbf{1}, \mathbf{2}, \mathbf{3}\}.$$

Intuitivement, la proposition 3.3.1 stipule que tout polyomino parallélogramme est obtenu par déformation des quatres côtés d'un pixel, conformément aux notions établies dans le chapitre 2. À partir de là, on utilise la définition 3.1.2 afin de construire un algorithme naïf de factorisation d'un mot de contour w :

1. Calculer $F_{\mathbf{0}}$ et $F_{\mathbf{1}}$, l'ensemble de tous les facteurs de w débutant et se terminant par $\mathbf{0}$ et $\mathbf{1}$ respectivement.
2. Soit $x \in F_{\mathbf{0}}$ et $y \in F_{\mathbf{1}}$.
3. Posons φ le morphisme parallélogramme induit par x et y.
4. S'il existe un conjugué w' de w et un mot de contour u tels que $w' = \varphi(u)$, alors retourner (φ, u).
5. Sinon, répéter les étapes 2 à 4 pour tous les x et y possibles.

Remarquons que cet algorithme est polynomial. En effet, supposons que w est de longueur n. Alors, il existe au plus $n + (n-1) + \cdots + 2 + 1 \in \mathcal{O}(n^2)$ facteurs dans $F_{\mathbf{0}}$ et dans $F_{\mathbf{1}}$. Par conséquent, les étapes 2 à 4 sont répétées $\mathcal{O}(n^4)$ fois. L'étape 3 est calculée en temps constant puisque φ est complètement déterminé par x et y. Comme il existe n conjugués de w et que u s'obtient en décodant w' par les

facteurs x et y, l'étape 4 s'effectue en $\mathcal{O}(n^2)$. L'algorithme est donc dans $\mathcal{O}(n^6)$. On vient de démontrer le résultat suivant :

Proposition 3.3.2. *Le problème de factorisation d'un polyomino est dans la classe de complexité P.*

Cependant, il est possible de faire beaucoup mieux en étudiant les facteurs de F_0 et F_1. Considérons l'exemple suivant : si $w = \mathbf{0010122233}$, alors $x = \mathbf{0010}$ et $y = \mathbf{101}$ sont deux choix valides pour l'étape 2. Pourtant, ces facteurs ne peuvent constituer une factorisation de w puisque x et y se chevauchent. Nous développons maintenant cette idée afin de fournir un algorithme amélioré de factorisation comme suit : considérons un polyomino Q codé par le mot de contour w. Soit u un conjugué de w commençant par la lettre **0**. D'abord, on calcule un facteur $b_0 = u_1 u_2 \cdots u_k$ où $k \leq n$, et $u_1 = u_k = \mathbf{0}$. Une fois ce bloc fixé, on décode u jusqu'à rencontrer un **1** ou un **3** (notons qu'il est impossible de rencontrer la lettre **2**, autrement u contiendrait le facteur fermé **02**). Ensuite, on construit un bloc b_1 (ou b_3). Puis, on termine le décodage de u par les blocs ainsi construits. Si le décodage est valide, on retourne la factorisation. Sinon, on répète les étapes précédentes pour tous les conjugués de w. On a alors l'algorithme 1 suivant. La fonction DÉCODEBLOC est décrite dans l'algorithme 2.

Par la définition 3.1.2 et la proposition 3.3.1, les algorithmes 1 et 2 sont corrects. Nous démontrons maintenant que l'algorithme 1 est asymptotiquement plus rapide que celui de factorisation naïf. D'abord, il est clair que l'algorithme 2 s'effectue au pire cas en $|b|$ étapes. Maintenant, les instructions 3 à 29 de l'algorithme 1 sont répétées n fois puisque w admet exactement n conjugués. Ensuite, la boucle **pour** de la ligne 4 est elle aussi effectuée n fois. La construction de b_0 et b_1 aux lignes 7 et 8 s'effectue en i opérations élémentaires. Puis, la boucle **tant que** aux lignes 10 à 14 est répétée au plus $n - i$ fois puisqu'à chaque appel de la fonction

Algorithme 1 Factorisation d'un polyomino Q

Entrée: Un polyomino Q codé par le mot de contour $w = w_1 w_2 \cdots w_n$
Sortie: Deux polyominos P et R tels que $Q = P \cdot R$ si Q est composé

1: **Fonction** FACTORISATION(w : mot de contour)
2: **pour** chacun des conjugués u de w débutant par $\mathbf{0}$
3: $v \leftarrow \mathbf{0}$
4: **pour** $1 \leq i \leq n$
5: *valide* $\leftarrow$ **vrai**
6: **si** $u[i] = \mathbf{0}$ **alors** ▷ Construction de b_0 et b_2
7: $b_0 \leftarrow u[1] \cdots u[i]$
8: $b_2 \leftarrow \widehat{b_0}$
9: $c \leftarrow i + 1$
10: **tant que** $u[c] = \mathbf{0}$ **et** *valide*
11: *valide* $\leftarrow$ DÉCODEBLOC(u, b_0, c)
12: $v \leftarrow v \cdot \mathbf{0}$
13: $c \leftarrow c + |b_0|$
14: **fin tant que**
15: **si** *valide* **alors**
16: **pour** $c \leq j \leq n$
17: **si** $u[j] = u[c]$ **alors** ▷ Construction de b_1 et b_3
18: $b_{u[c]} \leftarrow u[c] \cdots u[j]$
19: $b_{\overline{u[c]}} \leftarrow \widehat{b_{u[c]}}$
20: $k \leftarrow j + 1$
21: **tant que** $k \leq n$ **et** *valide*
22: *valide* $\leftarrow$ DÉCODEBLOC($u, b_{u[k]}, k$)
23: $v \leftarrow v \cdot u[k]$
24: $k \leftarrow k + |b_{u[k]}|$
25: **fin tant que**
26: **si** $b_0 \cdot b_1 \cdot b_2 \cdot b_3 \neq \mathbf{0123}$ **et** *valide* **alors**
27: **retourner** $(b_0 \cdot b_1 \cdot b_2 \cdot b_3, v)$
28: **fin pour**
29: **fin pour**
30: **fin pour**
31: **retourner** w

Algorithme 2 Décodage du bloc b dont la première lettre est α à partir de w_k

```
1: Fonction DÉCODEBLOC(w : mot de contour, b ∈ αF*α, k : entier)
2:     pour 0 ≤ i < |b|
3:         si w[k + i] ≠ b[i + 1] alors
4:             retourner faux
5:     fin pour
6:     retourner vrai
```

DÉCODEBLOC, l'indice c est augmenté de $|b_0|$. Après cela, on construit les blocs b_1 et b_2 aux lignes 17 à 20. Comme pour la construction de b_0 et b_1, ces opérations utilisent $j - c$ unités de temps. Finalement, on procède au reste du décodage du mot aux lignes 21 à 25 avant de retourner la factorisation. Les lignes 16 à 28 sont répétées $n - c$ fois car on doit possiblement vérifier toutes les constructions de blocs b_1 et b_3 possibles. Notons que toutes les autres instructions sont effectuées en temps constant. Au final, on calcule que le nombre d'opérations de l'algorithme 1 est

$$n(1 + n(4n^2 + \kappa_1 n) + \kappa_2) = 4n^4 + \kappa_1 n^3 + \kappa_2 n^2 + n,$$

où $\kappa_1 = 3 + 2j - 6c - 4k$ et $\kappa_2 = (4k - 2j - 2)c + 2c^2 + 4 + 2i$. De plus, puisque les indices j, k et c sont tous bornés par n, κ_1 et κ_2 sont respectivement bornés par un multiple de n et de n^2. De cette façon, on conclut que cet algorithme a une complexité temporelle dans $\mathcal{O}(n^4)$.

L'algorithme 1 a été implémenté en langage *Python* à l'aide du logiciel de calcul *Sage* (Stein *et al.*, 2012) et testé pour tous les polyominos d'aire 1 à 10 inclusivement. Le code source est disponible sur BitBucket à l'adresse `https://bitbucket.org/htremblay/sage-doctorat`. La figure 3.7 illustre tous les polyominos composés d'aire inférieure à 10.

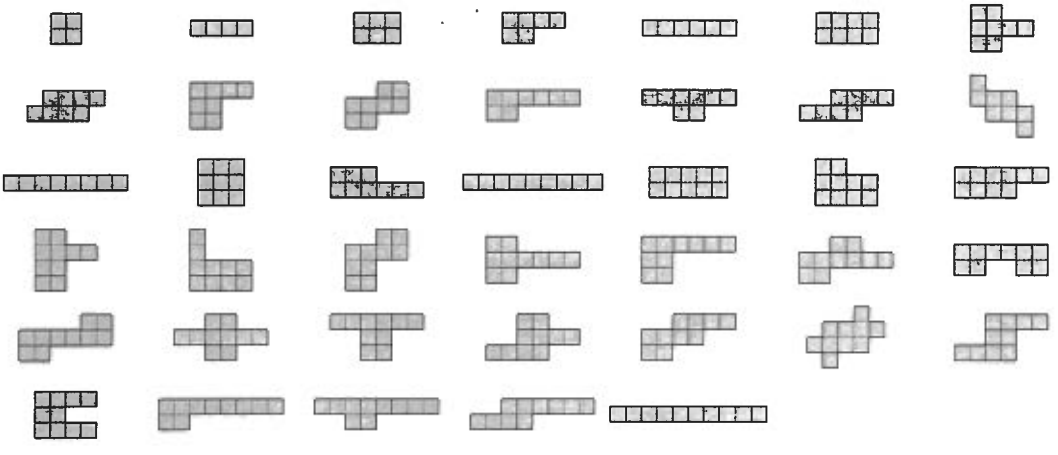


Figure 3.7: Polyominos composés d'aire inférieure ou égale à 10

CHAPITRE IV

OPÉRATIONS GÉOMÉTRIQUES SUR LES POLYOMINOS

Le présent chapitre vise à établir un cadre d'algorithmes linéaires en temps et en espace afin d'effectuer diverses opérations fondamentales sur les chemins et figures discrètes, notamment l'enveloppe externe, l'enveloppe convexe, l'union, l'intersection et la différence ensembliste. Afin d'y parvenir, la «règle de la main droite» est utilisée, permettant ainsi de parcourir, pour un chemin γ, le graphe $G(\gamma)$ en choisissant à chaque sommet la prochaine arête à emprunter. Par ailleurs, l'efficacité des algorithmes présentés repose sur une structure de données d'arbre quaternaire superposée à celle d'un graphe d'adjacence dans $\mathbb{Z}^2$. D'abord introduite par S. Brlek, M. Koskas et X. Provençal dans (Brlek *et al.*, 2011) afin de détecter l'intersection de chemins discrets, nous généralisons cette structure afin d'effectuer les opérations proposées.

Le chapitre est divisé comme suit : d'abord, la section 4.1 étend les notions présentées à la section 1.2 en développant le concept de graphe d'adjacence, fournissant ainsi une base théorique solide pour la suite de l'exposition. Ensuite nous présentons, à la section 4.2, la structure de données utilisée. Puis, un algorithme pour le calcul de l'enveloppe externe est construit à la section 4.3. Le calcul de l'enveloppe convexe est alors obtenu comme corollaire de ce résultat. Ensuite, nous étudions le concept de graphe couvrant à la section 4.4. Finalement, nous utilisons

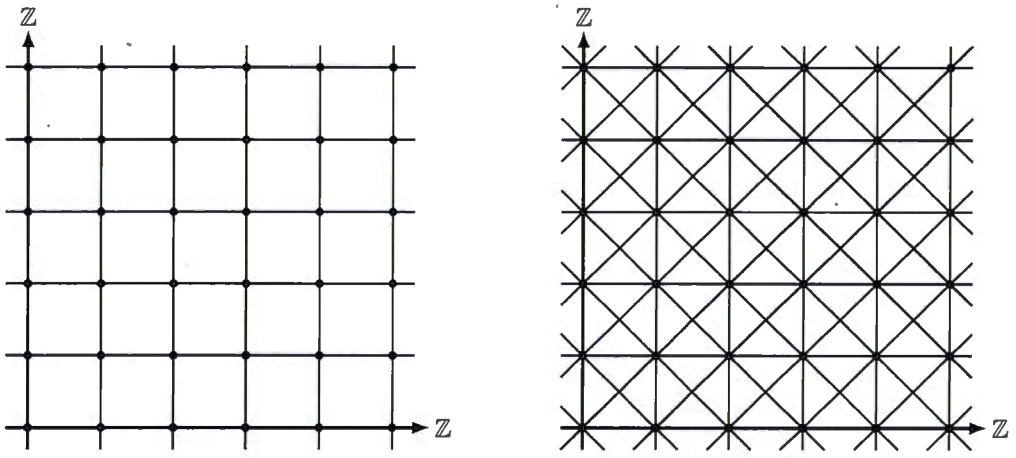
cette construction afin de calculer l'union, l'intersection et la différence de figures discrètes à la sous-section 4.4.2.

Notons que ce chapitre est une version légèrement étendue des travaux présentés à la 18$^{\text{ième}}$ Conférence internationale sur la géométrie discrète pour l'imagerie numérique (DGCI 2014), qui avait lieu du 10 au 12 septembre 2014 à Sienne en Italie (Brlek *et al.*, 2014) et publiés dans la revue Theoretical Computer Science en 2015 (Blondin Massé *et al.*, 2015).

4.1 Graphes d'adjacences

Le concept de graphe d'adjacence permet la description formelle d'un chemin discret de point de vue de la théorie des graphes. Autrement dit, il fournit un cadre théorique pour l'analyse d'images numériques bi et tridimensionnelles. Concrètement, ce type de graphe est utilisé afin de généraliser la notion de connexité déjà présentée à la section 1.1. Enfin, les graphes d'adjacences permettent de définir rigoureusement l'idée intuitive de «règle de la main droite». Notons que la terminologie utilisée dans cette section est tirée de (Rosenfeld et Klette, 2004).

On appelle *structure d'adjacence* tout graphe (V, E) tel que V est dénombrable. Par définition, l'ensemble des arêtes E est une relation appelée *relation d'adjacence* et est notée A. Étant donné un sommet $v \in V$, $A(v) = \{u \in V \mid uAv\}$ est l'ensemble des sommets adjacents à v, c'est-à-dire que u est dans $A(v)$ si et seulement si le graphe G contient l'arête $\{u, v\}$. Le *voisinage de* v est l'ensemble $A(v) \cup \{v\}$ et est noté $N(v)$. Le *degré de* v est la cardinalité de $A(v)$, c'est-à-dire $\deg(v) = |A(v)|$. Considérons le graphe infini $(\mathbb{Z}^2, A_4)$ où A_4 dénote la relation de 4-adjacence des points de $\mathbb{Z}^2$ (cf. figure 1.1 (a)). Alors, $(\mathbb{Z}^2, A_4)$ est une structure d'adjacence puisque $\mathbb{Z}^2$ est dénombrable. Par conséquent, le plan discret considéré en tant que graphe est une structure d'adjacence. Remarquons que, par le même argument, $(\mathbb{Z}^2, A_8)$ est lui aussi une structure d'adjacence. La figure 4.1 représente



(a) $(\mathbb{Z}^2, A_4)$ (b) $(\mathbb{Z}^2, A_8)$

Figure 4.1: Le plan discret en tant que structure de 4 et 8-adjacence

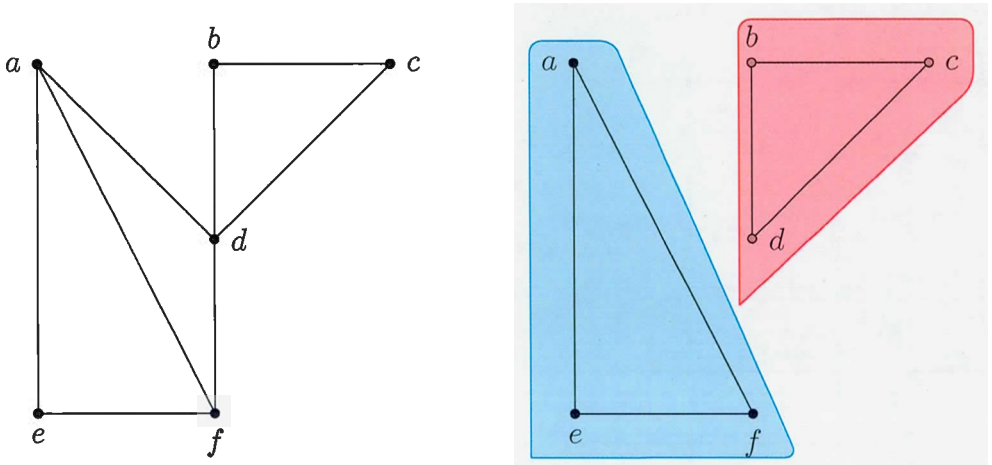
le plan discret en tant que structure de 4 et 8-adjacence.

La notion de connexité présentée à la section 1.1 se généralise à l'aide de celle de sous-graphe : considérons un graphe $G = (V, E)$ et un sous-ensemble $M \subseteq V$. Alors, M induit un sous-graphe (M, E_M) où $E_M = \{\{p, q\} \in E \mid p, q \in M\}$ est appelé *sous-graphe induit par* M. Deux sommets $p, q \in V$ sont alors *connexes par rapport à* $M \subseteq V$ si et seulement s'il existe une suite de sommets $(p_0, p_2, \ldots, p_n)$ de V où $p = p_0$ et $q = p_n$, p_i est adjacent à p_{i+1} pour tout $0 \leq i < n$, et les p_i sont soit tous dans M, soit tous dans $\overline{M} = V - M$. Notons que la suite $(p_1, p_2, \ldots, p_n)$ correspond à un chemin de longueur n (voir section 1.2). Un graphe (V, E) est dit *connexe* si toute paire de sommets est connexe par rapport à V.

Si M est un sous-ensemble de V, alors la relation R_M définie par

pR_Mq si et seulement si p et q sont connexes par rapport à M.

est clairement une relation d'équivalence. Les classes d'équivalence $R_M(p)$ sont alors appelées respectivement *composantes connexes de* M si $R_M(p) \subseteq M$ et *composantes connexes complémentaires de* M si $R_M(p) \subseteq \overline{M}$. Dans le cas d'un



(a) Le graphe (V, E) (b) Le sous-graphe induit (M, E_M)

Figure 4.2: (a) Un graphe (V, E) et (b) un sous-graphe induit (M, E_M) où $M = \{a, e, f\}$. La composante connexe est représentée en bleu et la composante connexe complémentaire, en rouge

graphe fini, ces composantes s'obtiennent facilement en effectuant un parcours en largeur du graphe (V, E) (pour les détails de tels algorithmes, voir (Cormen *et al.*, 1990) et (Tarjan, 1972) respectivement pour le cas non-orienté et orienté). Par exemple, la figure 4.2 illustre un graphe connexe (V, E), un sous-graphe induit (M, E_M) non-connexe ainsi que ses composantes connexes correspondant aux classes d'équivalences $R_M(a) = \{a, e, f\} \subseteq M$ et $R_M(b) = \{b, c, d\} \subseteq \overline{M}$.

La notion de structure d'adjacence est insuffisante afin de bien étudier la notion d'adjacence. Par exemple, on aimerait qu'une structure d'adjacence soit obligatoirement connexe, ce qui n'est pas le cas en général. En effet, il suffit de remarquer que le graphe formé de deux sommets et d'aucune arête est une structure d'adjacence. Pour remédier à cela, on introduit la définition suivante :

Définition 4.1.1. *Une structure d'adjacence (V, A) est appelée* graphe d'adjacence *si et seulement si elle admet les trois propriétés suivantes :*

P1 : *$A(p)$ est fini pour tout $p \in V$;*

P2 : *V est connexe par rapport à la relation A ;*

P3 : *Tout sous-ensemble fini $M \subseteq V$ a au plus une composante complémentaire infinie.*

Toute composante connexe finie d'un graphe d'adjacence est appelée *région* ou *face*. Notons que cette définition coïncide avec celle déjà donnée à la section 1.2. La proposition suivante montre que la grille discrète munie de la relation de α-adjacence est un graphe d'adjacence pour $\alpha \in \{4, 8\}$.

Proposition 4.1.2. *$(\mathbb{Z}^2, A_\alpha)$ est un graphe d'adjacence pour $\alpha \in \{4, 8\}$.*

Démonstration. Nous avons déjà remarqué que $(\mathbb{Z}^2, A_\alpha)$ est une structure d'adjacence. Soit $p \in \mathbb{Z}^2$. Alors, p a exactement α voisins par définition de sorte que **P1** est respectée. De plus, **P2** est vraie. En effet, soit $(a, b), (a', b') \in Z^2$. Supposons sans perte de généralité que (a, b) est situé plus en bas et à gauche que (a', b'). Alors, il existe un α-chemin de (a, b) à (a', b') :

$$(a, b), (a+1, b), \ldots, (a+|a'-a|, b), (a+|a'-a|, b+1), \ldots, (a+|a'-a|, b+|b'-b|).$$

V est donc connexe par rapport à A_α. Il reste à montrer que **P3** est vérifiée. Soit $M \subseteq \mathbb{Z}^2$ un ensemble fini et $b(M)$ son rectangle englobant. Par construction, $b(M)$ est fini car M est fini. Ceci implique que toute composante complémentaire infinie de M doit nécessairement sortir du rectangle $b(M)$, c'est-à-dire qu'elle a une intersection non-vide avec $\mathbb{Z}^2 \setminus b(M)$. Soit p et p' deux points à l'extérieur de $b(M)$. Par l'argument précédent, p et p' sont dans des composantes complémentaires infinies. Alors, il existe un chemin extérieur à $b(M)$ reliant ces deux points : il suffit de contourner le rectangle englobant. Par conséquent, p et p' sont connexes par rapport à M. La propriété **P3** est donc vérifiée.

□

À présent, il est clair que toute figure discrète s'exprime comme un sous-graphe de $(\mathbb{Z}^2, A_4)$: soit S la figure discrète définie par l'ensemble de pixels $\{p_1, p_2, \ldots, p_n\}$, où $p_i \in \mathbb{Z}^2$. Alors, S est équivalente au sous-graphe induit (S, A_4) où A_4 est la relation de 4-adjacence restreinte aux points de S. Autrement dit, $\{p_i, p_j\}$ est une arête de (S, A_4) si et seulement si les pixels correspondants sont 4-adjacents dans $\mathbb{Z}^2$. La figure 4.3 illustre cette correspondance. Notons toutefois que (S, A_4) est différent de $G(S)$. En effet, (S, A_4) représente l'adjacence de cellules alors que $G(S)$ considère chaque côté d'un pixel comme une arête, e.g. si S est composée d'un seul pixel, alors (S, A_4) a un sommet et aucune arête alors que $G(S)$ a quatre sommets et autant d'arêtes.

Géométriquement, **P3** exprime le fait que toute figure discrète M a un nombre fini de composantes connexes complémentaires finies de M, appelées *trous* et une seule composante connexe infinie, l'*extérieur de* M (voir exemple 4.1.3).

On classe les sommets des sous-ensembles $M \subseteq V$ comme suit : $p \in M$ est un *sommet interne* si et seulement si $A(p) \subseteq M$. Sinon, p est un *sommet sur le bord*. Les ensembles de sommets internes et sur le bord sont respectivement notés $M^{\Delta} = \{p \in M \mid A(p) \subseteq M\}$ et $\delta(M) = M - M^{\Delta}$. Évidemment, $M^{\Delta} \cup \delta(M) = M$ et $M^{\Delta} \cap \delta(M) = \emptyset$.

Exemple 4.1.3. *Considérons le plan discret muni de la relation de* 4*-adjacence* $(\mathbb{Z}^2, A_4)$. *Soit* $M \subseteq \mathbb{Z}^2$, *la figure discrète d'aire* 14 *représentée par l'ensemble des cellules bleues de la figure 4.3.*

a) M a deux composantes connexes finies $R_M(c(1,1))$ *et* $R_M(c(3,3))$ *(en bleu), deux composantes connexes complémentaires finies* $R_M(c(3,2))$ *et* $R_M(c(4,3))$ *(en gris), et une composante connexe complémentaire infinie* $R_M(c(0,0))$ *(l'ensemble des cellules non colorées).*

b) M est constituée d'un seul sommet interne $M_4^{\Delta} = \{c(5,2)\}$ *(en bleu foncé)*

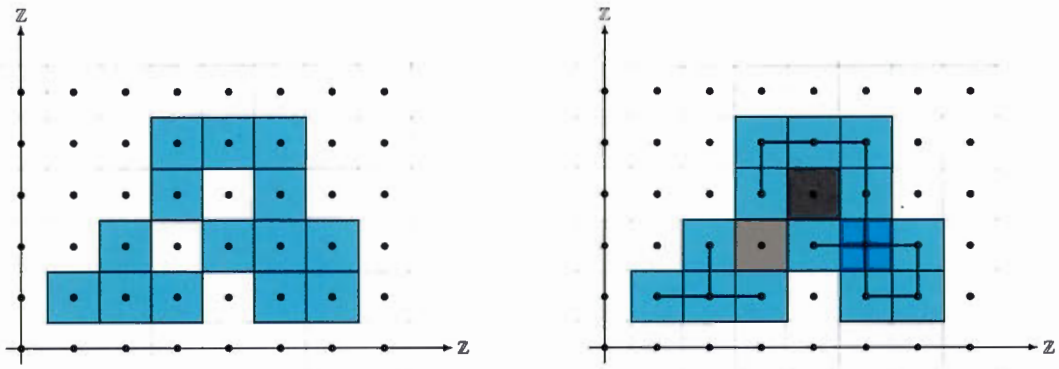


Figure 4.3: (a) Une figure discrète et (b) ses différentes composantes connexes et ses sommets internes

et de 13 *sommets sur le bord* $\delta_4(M) = M - M_4^{\Delta}$ *(en bleu pâle).*

Rappelons qu'un graphe fini est dit *planaire* s'il est possible de le dessiner dans le plan euclidien $\mathbb{R}^2$ sans croisement d'arêtes. Or, il existe une infinité de façons de dessiner un graphe planaire donné, e.g. en changeant la courbure des arêtes. Il est alors naturel de se demander comment déterminer si deux graphes planaires donnés sont équivalents. Afin de répondre à cette question, la notion d'orientation pour les graphes d'adjacence est développée. Notons que cette construction se généralise aisément à tout type de graphe (voir (Gross et Tucker, 1987)).

Étant donné un graphe d'adjacence (V, A) et un sommet $p \in V$, un *ordre circulaire pour* p, noté $\sigma(p)$ est une permutation de $A(p)$, c'est-à-dire une bijection de l'ensemble $A(p)$ vers lui-même. Nous utilisons la notation usuelle $\langle x_1, x_2, \ldots, x_n \rangle$ pour représenter les permutations circulaires. Par exemple, tout sommet (x, y) de

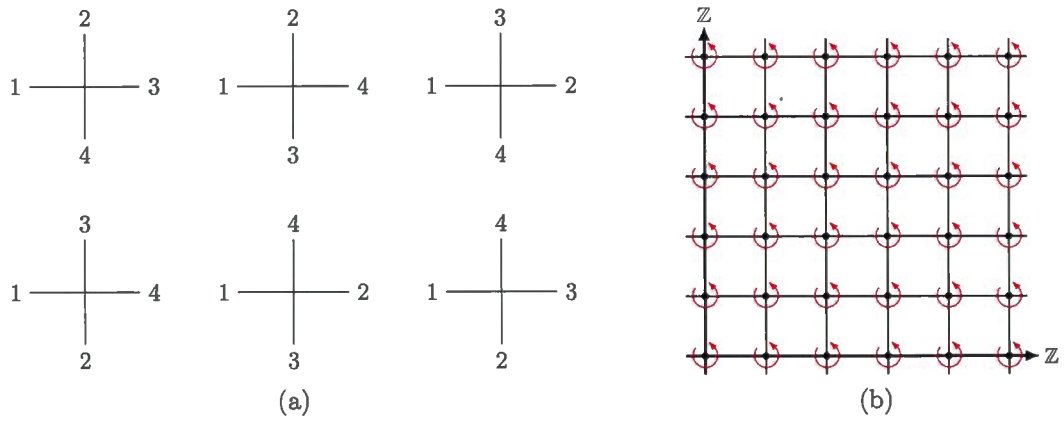


Figure 4.4: (a) Les six ordres circulaires pour A_4 ; (b) Le graphe d'adjacence orienté $(\mathbb{Z}^2, A_4)$ dans le sens antihoraire

$(\mathbb{Z}^2, A_4)$ admet six ordres circulaires :

$$\langle (x-1,y),(x,y+1),(x+1,y),(x,y-1)\rangle,$$
$$\langle (x-1,y),(x,y+1),(x,y-1),(x+1,y)\rangle,$$
$$\langle (x-1,y),(x+1,y),(x,y+1),(x,y-1)\rangle,$$
$$\langle (x-1,y),(x,y-1),(x+1,y),(x+1,y)\rangle,$$
$$\langle (x-1,y),(x+1,y),(x,y-1),(x,y+1)\rangle,$$
$$\langle (x-1,y),(x,y-1),(x+1,y),(x,y+1)\rangle,$$

illustrés à la figure 4.4 (a). Une fonction qui, à chaque sommet, lui assigne un ordre circulaire constitue une *orientation de* (V, A). Par exemple, en fixant l'ordre circulaire antihoraire pour tout sommet de $(\mathbb{Z}^2, A_4)$, on obtient le graphe d'adjacence orienté illustré à la figure 4.4 (b). Il est démontré dans (Gross et Tucker, 1987) que toute orientation sur un graphe donné est équivalente à considérer un plongement sur ce dernier (voir section 1.2).

Maintenant, toute orientation permet de définir des chemins dans un graphe de la

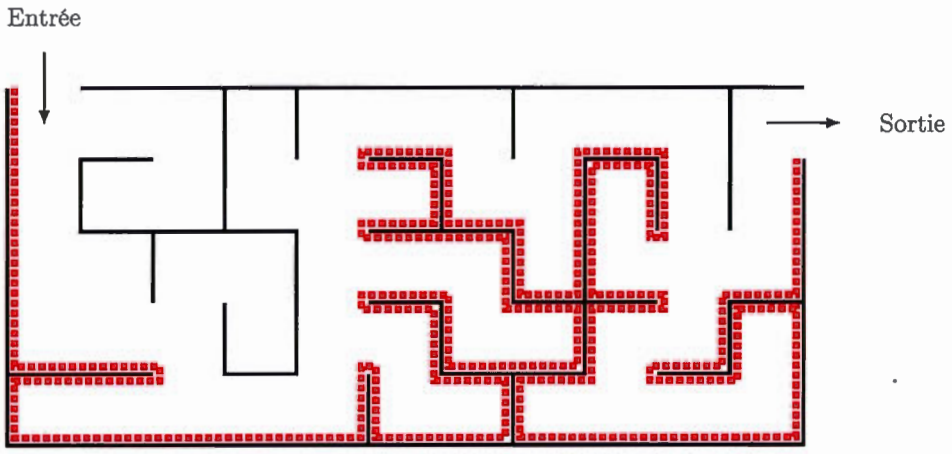


Figure 4.5: Algorithme de la main droite

façon suivante : soit $p \in V$ et $\sigma(p) = \langle q_0, q_1, \ldots, q_n \rangle$. Si on arrive au sommet p à partir du sommet $q_i \in A(p)$, alors le prochain sommet du chemin est $q_k \in A(p) \cap M$ où $k = (i+1) \mod n$. On note $p \to q_k$ l'arête orientée, c'est-à-dire l'*arc* allant de p vers q_k. De plus, si $P : p_0 \to p_1 \to p_2 \to \cdots \to p_m$ est un tel chemin, alors on dit que $p_0 \to p_1$ *initie* ou *engendre* le chemin P. Cette idée est à la base d'un algorithme appelé *algorithme de la main droite*, mieux connu sous le nom de *wall follower algorithm* en anglais et est illustrée à la figure 4.5 : en partant de l'entrée d'un labyrinthe, il existe au moins un chemin permettant d'en sortir. Il suffit de suivre le mur droit du labyrinthe en gardant sa main droite dessus jusqu'à atteindre une sortie. La discussion précédente confirme l'exactitude de cette méthode.

L'observation 4.1.4 suivante découle de l'injectivité de $\sigma(p)$. Elle concerne les arcs incidents à un sommet p et leur ordonnancement selon l'ordre local de p. Notamment utilisée dans la démonstration de la proposition 4.1.6, elle constitue une idée phare pour les algorithmes développés dans la suite du texte.

Observation 4.1.4. *Soit (V, A) un graphe d'adjacence et $\sigma(p) = \langle q_0, q_1, \ldots, q_n \rangle$ un ordre circulaire sur $A(p) \in V$. Observons que pour tout $0 \leq j \leq n$, le chemin*

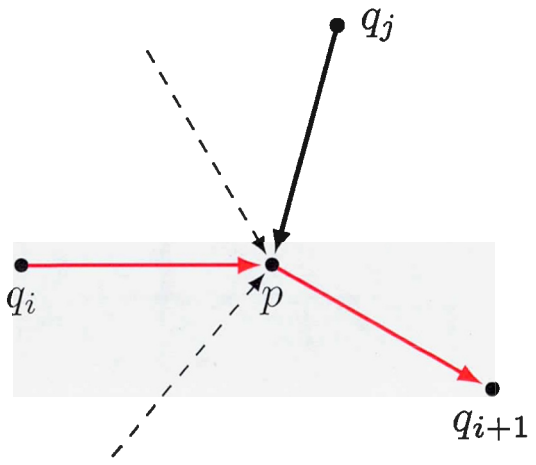


Figure 4.6: Arcs incidents au sommet p dans un graphe d'adjacence pour une orientation donnée. $q_i \to p \to q_{i+1}$ est le seul chemin valide

$q_j \to p \to q_{i+1}$ est engendré par $q_j \to p$ si et seulement si $j = i$ (voir figure 4.6). Autrement dit, la seule façon d'emprunter l'arc $p \to q_{i+1}$ est d'abord d'emprunter l'arc $q_i \to p$.

Exemple 4.1.5. *Considérons $(\mathbb{Z}^2, A_4)$ muni de l'orientation horaire : $\sigma(p) = \langle (x-1,y), (x,y+1), (x+1,y), (x,y-1) \rangle$ pour tout $p \in \mathbb{Z}^2$. Alors, tout chemin de longueur 4 suivant la construction du paragraphe précédent et initié par $p \to q$, où $q \in A(p)$ est un cycle. En effet, si $p = (x,y)$, alors le chemin*

$$(x,y) \to (x,y-1) \to (x-1,y-1) \to (x-1,y) \to (x,y)$$

est bien un cycle.

On vérifie facilement à partir de l'exemple 4.1.5 que pour $(\mathbb{Z}^2, A_4)$, seules les orientations horaire et antihoraire mènent à des cycles finis. Cette propriété est à la base de la définition suivante : un *graphe d'adjacence orienté*[1] (V, A, σ) est un

1. La notion de graphe d'adjacence orienté est équivalente à celle de *carte combinatoire* (*combinatorial map* en anglais) qui a l'avantage de bien se généraliser à trois dimensions (Rosenfeld

graphe d'adjacence (V, A) muni d'une orientation telle que

P4 : Tout arc initie un cycle (et non un chemin infini).

Il est clair par l'exemple 4.1.5 que $(\mathbb{Z}^2, A_4, \sigma)$ où σ est respectivement l'orientation horaire et antihoraire est un graphe d'adjacence orienté.

Étant donné un sous-ensemble $M \subseteq V$ où (V, A, σ) est un graphe d'adjacence orienté, il est naturel de considérer la sous-structure de graphe d'adjacence orienté (M, A_M, σ_M) induite par M où A_M est la relation d'adjacence A restreinte à M et σ_M est l'orientation obtenue de σ en supprimant les sommets de $V \setminus M$. Par exemple, si $M = \{(0,0), (0,1), (1,1)\} \subset \mathbb{Z}^2$ et σ est l'orientation antihoraire, alors σ_M autour de $(0,1)$ est $\langle (0,0), (1,1) \rangle$. Cette sous-structure est elle-même un graphe d'adjacence orienté si et seulement si M est connexe par rapport à A_M, comme le montre la proposition suivante :

Proposition 4.1.6. *Soit (V, A, σ) un graphe d'adjacence orienté et (M, A_M, σ_M) la sous-structure induite par $M \subseteq V$. Alors, (M, A_M, σ_M) est un graphe d'adjacence orienté si et seulement si M est fini et connexe par rapport à A_M.*

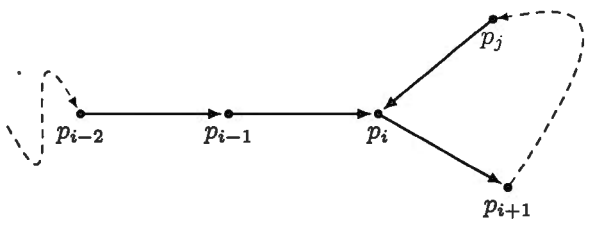
Démonstration. Si (M, A_M, σ_M) est un graphe d'adjacence orienté, alors la propriété **P2** est respectée et M est bien connexe par rapport à A_M. Inversement, considérons (M, A_M, σ_M) telle que **P2** est vraie. Alors, **P1** est respectée car $M \subseteq V$. De plus, **P3** est respectée car tout sous-ensemble de M est un sous-ensemble de V par transitivité. Finalement, montrons que **P4** est respectée. Soit $p_0 \to p_1$ un arc de M et considérons le chemin (infini) γ engendré par celui-ci :

$$\gamma : p_0 \to p_1 \to p_2 \to \cdots \to p_i \to \cdots .$$

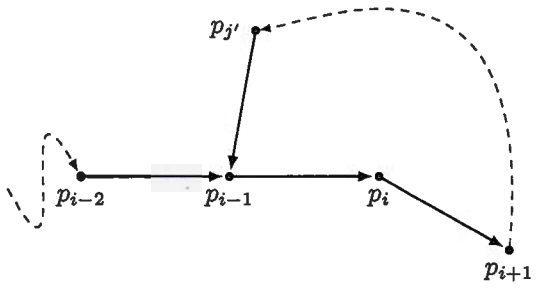
Comme M est fini, il y a un nombre fini d'arcs apparaissant dans γ. Par le principe

---

et Klette, 2004).



(a) Si $p_i \to p_{i+1}$ le premier arc répété du chemin γ, ...



(b) ... Alors on déduit par l'observation 4.1.4 que $p_{i-1} \to p_i$ est le premier arc répété, une contradiction

Figure 4.7: Illustration de la preuve de la proposition 4.1.6

des tiroirs, au moins un arc de la suite γ se répète. Soit $p_i \to p_{i+1}$ le premier tel arc. Nous allons montrer que $i = 0$ de sorte que γ est un cycle. Supposons par contradiction que $i \neq 0$. Alors, on a la situation illustrée à la figure 4.7 (a). Or, il suit de l'observation 4.1.4 que $p_{i-1} \to p_i$ est le premier arc de γ qui se répète, une contradiction (voir figure 4.7 (b)). Par conséquent, **P4** est vérifiée et on conclut que (M, A_M, σ_M) est un graphe d'adjacence orienté. □

La proposition précédente montre que toute figure discrète hérite de la structure d'orientation de $(\mathbb{Z}^2, A_4)$. Cependant, les cycles de (M, A_M, σ_M) peuvent différer de ceux de (V, A, σ) où $M \subseteq V$. Soit σ_1 le cycle engendré par l'arc $p \to q$ dans (M, A_M, σ_M) et σ_2 le cycle engendré par l'arc $p \to q$ dans (V, A, σ). Alors, σ_1 est

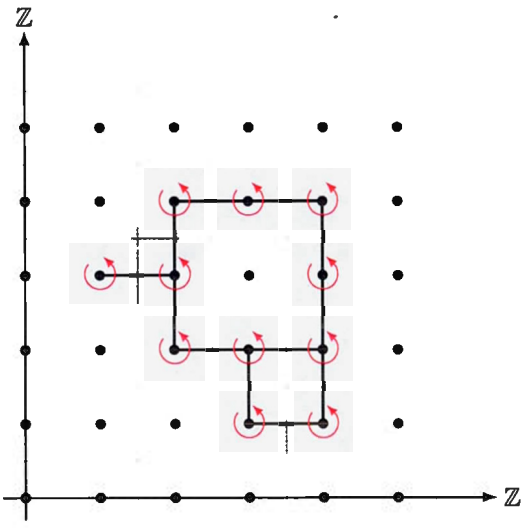
un cycle *atomique* si et seulement si $\sigma_1 = \sigma_2$. Sinon, σ_1 est un cycle *sur le bord*. Par exemple, la figure 4.8 illustre une figure discrète et ses différents cycles.

Finalement, étant donné une figure discrète M, on définit le *bord de* M comme l'ensemble des cycles de bord de M. Intuitivement, on peut imaginer le bord de M comme une courbe située entre les sommets $\delta(M)$ et les sommets de $\overline{M}$ adjacents, un peu à la manière du mortier entre les briques d'une maison (voir figure 4.9).

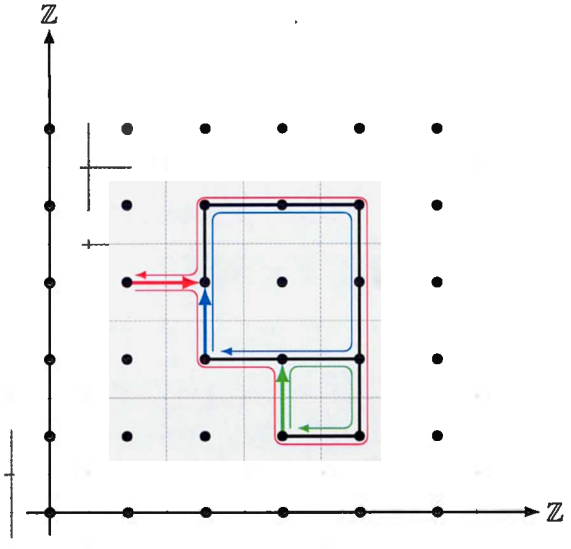
4.2 Structure de données d'arbre quaternaire radix

En 2009, S. Brlek, M. Koskas et X. Provençal ont introduit dans (Brlek *et al.*, 2009a) une structure de données permettant de détecter l'intersection de deux chemins discrets en temps et en espace linéaires. Cette structure est obtenue en superposant le graphe d'adjacence $(\mathbb{Z}^2, A_4)$ avec un arbre quaternaire construit en appliquant l'ordre radix sur les points de $\mathbb{Z}^2$. Ainsi, tout chemin discret de longueur n est parcouru en temps et en espace linéaires. Afin de simplifier les calculs, les auteurs restreignent dans un premier temps les chemins étudiés au premier quadrant $\mathbb{N} \times \mathbb{N}$ de sorte que tous les points traités soient à coordonnées positives. Par la suite, leur structure est étendue à tout le plan $\mathbb{Z} \times \mathbb{Z}$ en fusionnant quatre copies de l'arbre et en étudiant les transitions entre les différents quadrants. Ici, nous redéfinissons la structure d'arbre quaternaire radix afin de l'appliquer directement à tout le plan discret. De plus, nous l'enrichissons dans le but d'étudier diverses opérations géométriques sur les chemins discrets.

Considérons l'ensemble $B = \{\mathbf{0}, \mathbf{1}\}^*$ des mots binaires. Il est alors possible de représenter tous les éléments de B par l'arbre binaire complet T défini de la façon suivante : la racine contient le mot vide ε et tout noeud $w \in B$ admet $w \cdot \mathbf{0}$ et $w \cdot \mathbf{1}$ respectivement comme fils gauche et droit. En effectuant un parcours en largeur de T, on obtient alors l'ordonnancement de B suivant : $\varepsilon, \mathbf{0}, \mathbf{1}, \mathbf{00}, \mathbf{01}, \mathbf{10}, \mathbf{11}, \mathbf{000}, \mathbf{001}, \mathbf{010}, \cdots$, appelé *ordre radix*. Maintenant, on étend



(a)



(b)

Figure 4.8: (a) Une figure discrète (M, A_M, σ_M) induite par $(\mathbb{Z}^2, A_4,$ antihoraire) en tant que graphe d'adjacence orienté. (b) M contient un cycle atomique engendré par $(3,1) \to (3,2)$ (en vert) et deux cycles sur le bord engendrés respectivement par $(2,2) \to (2,3)$ et $(1,3) \to (2,3)$ et dessinés en bleu et rouge respectivement

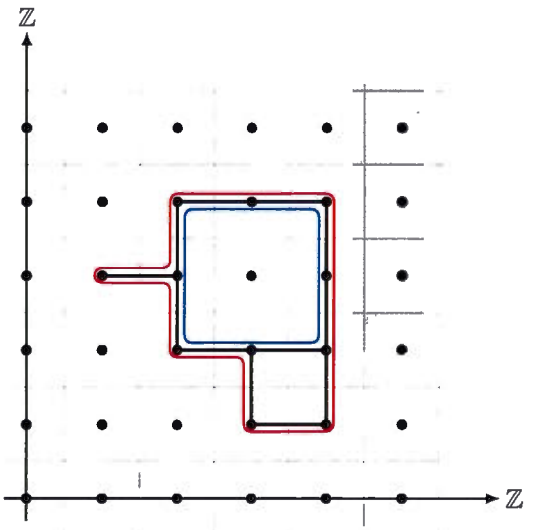


Figure 4.9: Le bord de la figure M est constitué de deux cycles sur le bord

cet arbre à tous les points du premier quadrant comme suit : la racine est le point $(\mathbf{0},\mathbf{0})$ et admet les trois fils $(\mathbf{00},\mathbf{01})$, $(\mathbf{01},\mathbf{00})$ et $(\mathbf{01},\mathbf{01})$. De plus, tout noeud $(u,v) \neq (\mathbf{0},\mathbf{0})$ a quatre fils en allant de gauche à droite : $(u\cdot\mathbf{0}, v\cdot\mathbf{0})$, $(u\cdot\mathbf{0}, v\cdot\mathbf{1})$, $(u\cdot\mathbf{1}, v\cdot\mathbf{0})$ et $(u\cdot\mathbf{1}, v\cdot\mathbf{1})$. Finalement, chaque chaîne binaire est convertie en décimales (e.g. $\mathbf{0101}$ devient 5). On vérifie alors aisément que le point (x',y') est un fils de (x,y) si et seulement si

$$(x',y') = (2x+\alpha, 2y+\beta), \tag{4.1}$$

où $\alpha,\beta \in \{0,1\}$. Par exemple, $(\mathbf{11}\cdot\mathbf{1}, \mathbf{01}\cdot\mathbf{0}) = (7,2)$ est un fils de $(\mathbf{11},\mathbf{01}) = (3,1)$ et on bien que $(7,2) = (2\cdot 3+1, 2\cdot 1+0)$. Cet arbre est illustré à la figure 4.10.

Cette construction est maintenant étendue à tous les points de $\mathbb{Z}^2$. Pour ce faire, les définitions suivantes sont utiles. Une *direction* est l'un des vecteurs unitaires $(1,0)$, $(0,1)$, $(-1,0)$ et $(0,-1)$. De plus, un *chemin discret coloré* est un triplet (p,w,c), où $p \in \mathbb{Z}^2$ est le *point de départ*, $w \in \mathcal{F}^*$ est appelé *mot de direction* et $c \in \{0,1,\ldots,C-1\}$ est la *couleur* du chemin, où C, le nombre de couleurs, est

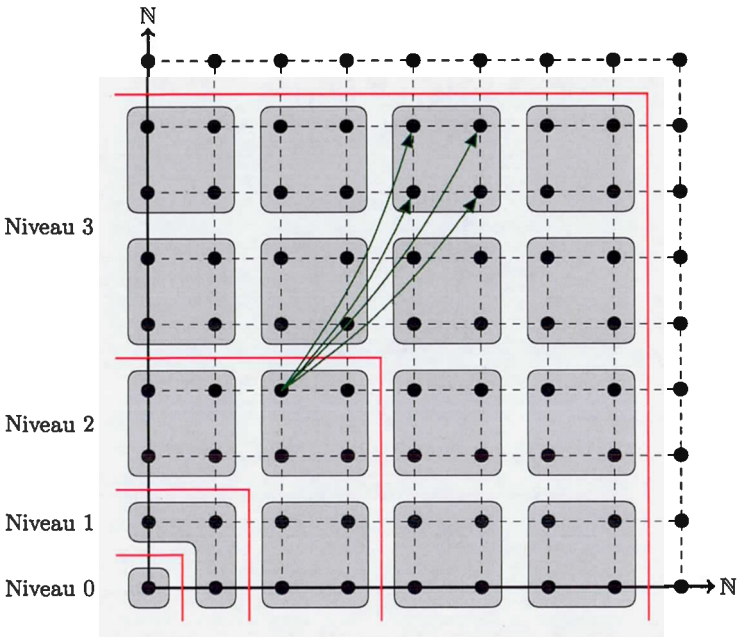


Figure 4.10: Arbre quaternaire radix restreint au premier quadrant. Les lignes pointillés représentent la relation d'adjacence alors que les différents niveaux dénotent l'ordre radix sur les points de $\mathbb{N}^2$

un entier positif.

Étant donné une famille de chemins discrets colorés, nous sommes intéressés à construire une structure d'arbre quaternaire radix enrichi sauvegardant l'information de chaque chemin. Pour ce faire, nous utilisons les opérations suivantes :

— À tout moment, la fonction PointCourant() retourne les coordonnées du point courant, c'est-à-dire l'étiquette du noeud où l'on se trouve dans l'arbre. Initialement, sa valeur est fixée à l'origine $(0, 0)$.

— À partir du point courant, il est possible d'ajouter un arc coloré dans n'importe laquelle des quatre directions en appelant la fonction CreerArc(d, c), où d est une direction et c est la couleur. Ce faisant, nous ajoutons aussi un arc *non coloré* dans la direction opposée, ce qui permet de préserver la nature bidirectionnelle des arêtes de la structure. Cette opération remplace aussi le point courant (x, y) par le point $(x, y) + d$.

— Comme raccourci, nous utilisons la fonction CREERCHEMIN(w, c), où $w = w_1 w_2 \cdots w_n$ est un mot de direction et c est une couleur. Clairement, CREERCHEMIN(w, c) est équivalent à appliquer la suite d'opérations CREERARC(w_i, c) pour $i = 1, 2, \ldots, n$.

— À tout moment, il est possible de réinitialiser le point courant à la valeur de n'importe quel point de $\mathbb{Z}^2$ en appelant la fonction ALLERVERS(x, y), où $(x, y) \in \mathbb{Z}^2$.

Remarquons qu'un arc peut être de plusieurs couleurs différentes, e.g. si deux chemins différents passent par cet arc. Par conséquent, chaque arc possède non pas une couleur mais bien une liste de couleurs.

Le *parent* d'un point $(x, y) \in \mathbb{Z}^2$ est défini par

$$\text{PARENT}(x, y) = \left(\text{sign}(x) \left\lfloor \frac{|x|}{2} \right\rfloor, \text{sign}(y) \left\lfloor \frac{|y|}{2} \right\rfloor\right).$$

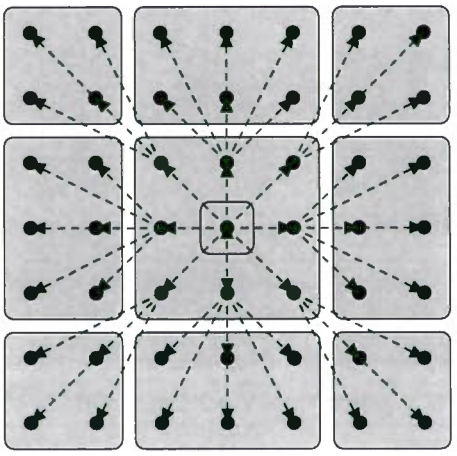
Notons que cette formule découle directement de l'équation (4.1). Inversement, si (x', y') est tel que PARENT$(x, y) = (x', y')$, alors on dit que (x', y') est un *fils de* (x, y). On vérifie aisément que

— $(0, 0)$ a huit fils, notamment $(\pm 1, 0)$, $(0, \pm 1)$, $(\pm 1, \pm 1)$.

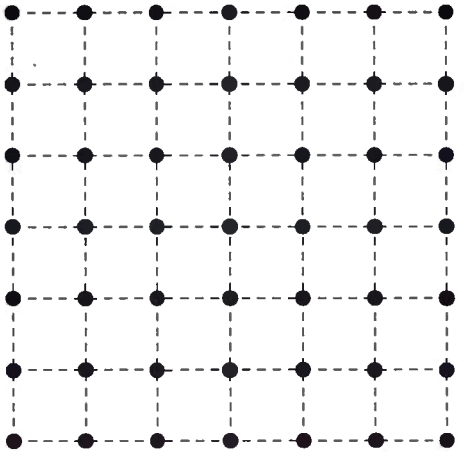
— Tout point $(0, y)$, où $y \in \mathbb{Z}$, a six fils, notamment $(0, 2y)$, $(\pm 1, 2y)$, $(0, 2y+1)$ et $(\pm 1, 2y + 1)$ (On a une situation symétrique pour les points de la forme $(x, 0)$, où $x \in \mathbb{Z}$).

— Tout point (x, y), où $x, y \neq 0$ a exactement quatre fils, notamment $(2x, 2y)$, $(2x + 1, 2y)$, $(2x, 2y + 1)$ et $(2x + 1, 2y + 1)$.

L'arbre enraciné induit par la relation de parenté décrite plus haut est appelé *arbre radix* (voir figure 4.11 (a)). L'efficacité de cette structure repose sur le principe suivant : en se déplaçant le long des quatre directions unitaires dans $\mathbb{Z}^2$, on désire créer le moins de noeuds possible. Pour ce faire, nous insérons seulement les noeuds visités par le chemin discret ainsi que tous leurs ancêtres. En effet, étant donné un



(a) L'arbre enraciné T induit sur $\mathbb{Z}^2$ par l'ordre radix



(b) Le graphe d'adjacence G sur $\mathbb{Z}^2$ avec la relation de 4-adjacence

Figure 4.11: Les deux graphes formant la structure de données. Elle peut être vue comme un sous-graphe de $T \cup G$, à lequel on ajoute des arcs colorés

noeud u, si on désire accéder à l'un de ses voisins v, alors il suffit de trouver leur plus proche ancêtre commun dans l'arbre à partir de u, pour ensuite descendre vers v.

Cependant, cette stratégie entraîne possiblement un coût de navigation élevé. Afin de pallier ce problème, la structure d'arbre radix est enrichie en la superposant avec le graphe de 4-adjacence $(\mathbb{Z}^2, A_4)$ (voir figure 4.11 (b)). Pour ce faire, on

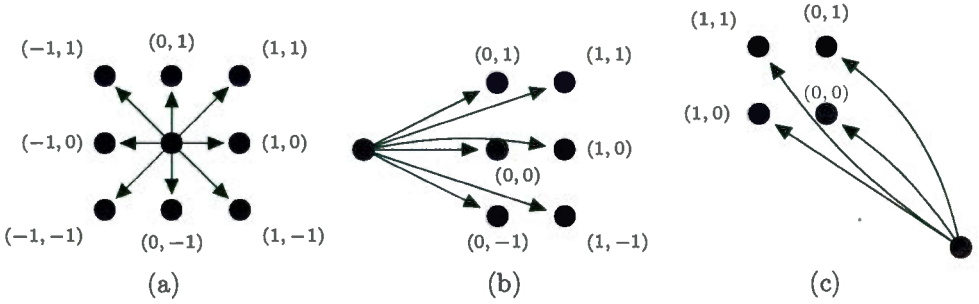


Figure 4.12: Les indices valides pour accéder aux fils d'un noeud. (a) À partir de la racine, (b) à partir d'un point sur un axe et (c) à partir de tout autre point

Algorithme 3 Trouver un fils d'un noeud

Entrée: Un noeud u de l'arbre quaternaire, et les indices (i, j) du fils correspondant

Sortie: Le fils v du noeud correspondant aux indices (i, j)

```
1: Fonction FILS(u : noeud, i, j : indices) : noeud
2:     si u.fils[i, j] n'est pas défini alors
3:         u.fils[i, j] ← NEWNOEUD(u)
4:     retourner u.fils[i, j]
```

ajoute récursivement les liens de 4-voisinage lorsque nécessaire. On obtient alors la structure de données d'*arbre quaternaire radix*. L'algorithme 3 suivant est une fonction simple retournant le fils de tout noeud tout en le créant s'il n'existe pas encore dans la structure. Les fils sont indexés par un couple (i, j) tel que décrit dans la figure 4.12. Nous supposons aussi qu'il existe une fonction NEWNOEUD(u) qui crée un nouveau noeud et ajoute le noeud u comme son parent.

Il est clair que l'algorithme 3 s'effectue en temps constant. À présent, l'algorithme 4 décrit la fonction VOISIN$(u, (x, y), d)$ qui retourne le voisin v de u dans la direction d. Le point représenté par u est (x, y). Remarquons qu'il n'est pas nécessaire de garder en mémoire les valeurs (x, y) dans la structure de données car il est possible de les déduire à partir de la position de u. Il y a cinq cas possibles lorsqu'on accède au voisin d'un noeud :

1. Le lien de voisinage existe déjà et on peut donc l'utiliser directement.
2. Le noeud courant est la racine (lignes 4-5). Alors, le voisin est simplement son fils dans la direction d et on peut donc utiliser le lien directement.
3. Le voisin est la racine (lignes 6-7).
4. Le noeud et son voisin partagent le même parent dans l'arbre radix (lignes 11-12). Alors, on peut accéder directement au voisin et créer le lien.
5. Le noeud et son voisin ne partagent pas le même parent (lignes 13-14). Dans ce cas, on essaie récursivement de lier les parents des deux noeuds. Lorsque

Algorithme 4 Trouver un voisin d'un noeud

Entrée: Un noeud u de l'arbre quaternaire, les coordonnées (x, y) de ce noeud et la direction d pointant vers le voisin
Sortie: Le noeud voisin de u dans la direction d

```
1: Fonction VOISIN(u : noeud, (x, y) : point, d : direction) : noeud
2:     si u.voisin[d] n'est pas défini alors
3:         (x', y') ← (x, y) + d
4:         si (x, y) = (0, 0) alors
5:             v ← FILS(u, d)
6:         sinon si (x', y') = (0, 0) alors
7:             v ← u.parent
8:         sinon
9:             Posons i = −1 si x' = −1, et x' mod 2 sinon
10:            Posons j = −1 si y' = −1, et y' mod 2 sinon
11:            si (x, y) et (x', y') partagent le même parent alors
12:                v ← FILS(u.parent, i, j)
13:            sinon
14:                v ← FILS(VOISIN(u.parent, PARENT(x, y), d), i, j)
15:        u.voisin[d] ← v
16:        v.voisin[−d] ← u
17:    retourner u.voisin[d]
```

c'est fait, on récupère le voisin (qui est le fils du voisin de son parent), avant de créer le lien directement.

Les algorithmes 3 et 4 sont utilisés afin d'implémenter la fonction CREERCHEMIN(w, c) : il suffit de lire le mot w en partant du point courant. À chaque lettre lue, on trouve le voisin du point courant dans la direction de la lettre (e.g. $(1, 0)$ pour la lettre **0**) puis, on ajoute un lien de voisinage coloré. Par exemple, la figure 4.13 illustre l'arbre quaternaire radix pour les chemins discrets colorés

$((0, 0), \mathbf{0001123210122333}, \text{ROUGE})$ et $((0, 0), \mathbf{0^4112^433}, \text{BLEU})$.

Évidemment, l'efficacité de cet algorithme dépend des fonctions ALLERVERS servant à initialiser le point de départ et VOISIN servant à naviguer dans la structure. Le reste de cette section est consacré à la démonstration de l'efficacité de

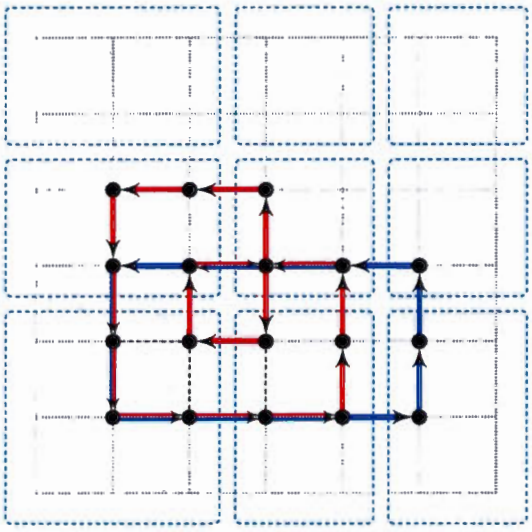


Figure 4.13: Arbre quaternaire radix associé aux chemins discrets colorés $((0,0), \mathbf{0001123210122333}, \textit{rouge})$ et $((0,0), \mathbf{0^4112^433}, \textit{bleu})$

ces fonctions. D'abord, il est immédiat que la fonction VOISIN s'effectue en temps constant, excepté dans le cas où le noeud et son voisin ne partagent pas le même parent. Dans ce cas, plusieurs appels récursifs sont effectués. Cependant, ce cas est plutôt rare. En effet, dans (Brlek *et al.*, 2011), il est démontré que, dans le cas où tout les points appartiennent au premier quadrant $\mathbb{N}^2$, une suite de m appels de la fonction CREERARC s'effectue avec une complexité spatiale et temporelle dans $\mathcal{O}(m)$ ou, de façon équivalente, CREERARC a une complexité amortie dans $\mathcal{O}(1)$. Afin de vérifier que notre généralisation est adéquate, nous démontrons maintenant que cette borne tient toujours lorsqu'on considère l'ensemble des points de $\mathbb{Z}^2$. Nous obtenons alors, comme conséquence immédiate, que l'opération CREERCHEMIN(w, c) a une complexité dans $\Theta(|w|)$. De plus, nous montrons que ALLERVERS(x, y) a une complexité dans $\mathcal{O}(|x| + |y|)$.

Théorème 4.2.1. *En partant de l'origine, l'opération* CREERCHEMIN(w, c) *a une complexité dans* $\Theta(|w|)$ *ou, de façon équivalente, l'opération* CREERARC *a une complexité amortie dans* $\mathcal{O}(1)$.

Démonstration. La preuve est essentiellement la même que dans (Brlek *et al.*, 2011). Elle est incluse ici afin de garder l'exposé autosuffisant et afin de confirmer que notre structure est bien une généralisation à tout le plan discret $\mathbb{Z}^2$.

D'abord, notons que la complexité de CREERCHEMIN(w, c) est proportionnelle au nombre de noeuds dans l'arbre quaternaire. La raison est que toutes les opérations créant les liens entre les noeuds se font en temps constant. Par conséquent, si m est le nombre de noeuds, alors il suffit de démontrer que $m = \mathcal{O}(n)$.

Soit c un chemin discret. Posons POINTS(c) l'ensemble des points de $\mathbb{Z}^2$ visités par c. De plus, PARENT(c) est le chemin discret dont la suite de points est donnée par les parents des points dans c. Alors, l'ensemble X des points de $\mathbb{Z}^2$ apparaissant dans la structure de données est

$$X = \bigcup_{i=0}^{h} \text{POINTS}(\text{PARENT}^i(p)) \tag{4.2}$$

où PARENT^i dénote la fonction PARENT appliquée i fois et h est la hauteur de l'arbre quaternaire.

Or, par le lemme 2 de (Brlek *et al.*, 2011), toute suite de cinq points consécutifs dans un chemin discret admet au plus quatre parents, c'est-à-dire qu'au moins deux points partagent le même parent. En d'autres termes, pour tout chemin discret c,

$$|\text{POINTS}(\text{PARENT}(c))| \leq \frac{4}{5}|\text{POINTS}(c)|,$$

de sorte que

$$|\text{POINTS}(\text{PARENT}^i(c))| \leq \left(\frac{4}{5}\right)^i |\text{POINTS}(c)|.$$

Ceci implique que

$$m = |X| = \sum_{i=0}^{h} |\textsc{Points}(\textsc{Parent}^i(c))| \leq \sum_{i=0}^{\infty} n \left(\frac{4}{5}\right)^i = 5n = \mathcal{O}(n),$$

ce qu'il fallait démontrer. □

La complexité de la fonction ALLERVERS est immédiate.

Corollaire 4.2.2. *L'opération* ALLERVERS(x, y) *a une complexité* $\mathcal{O}(\log(|x| + |y|))$ *dans le pire cas.*

Démonstration. Afin de réinitialiser le point courant à la valeur (x, y), il suffit de construire ses fils. Comme la hauteur de l'arbre est logarithmique en fonction de la longueur du chemin de $(0, 0)$ à (x, y), on a le résultat. Notons qu'il est inutile de construire tout le chemin de $(0, 0)$ à (x, y). □

Mentionnons en terminant que la complexité spatiale de cette structure de données dépend directement du nombre de noeuds créés. On a donc comme conséquence immédiate du théorème 4.2.1 que la structure d'arbre quaternaire radix occupe un espace linéaire.

Finalement, cette structure de données a été implémentée avec le langage *Python* (`https://www.python.org/`). Le code source est disponible à l'adresse `https://bitbucket.org/htremblay/sage-doctorat`.

4.3 Enveloppes externe et convexe

Rappelons qu'en topologie, étant donné un ensemble S, le bord de S, noté ∂S, est l'ensemble des points de la fermeture de S qui n'appartiennent pas à l'intérieur de S. Maintenant, considérons une figure discrète 8-connexe S. Alors, l'*enveloppe*

externe de S, notée $\text{Hull}(S)$, est le bord de l'intersection de tous les ensembles discrets sans trou contenant S. Remarquons que l'enveloppe externe correspond au cycle de bord situé à l'extérieur de l'ensemble S (voir figure 4.9). En d'autres termes, $\text{Hull}(S)$ est le chemin fermé et simple [2] décrivant le contour extérieur de S. La définition 4.3.1 étend la notion d'enveloppe externe à tous les chemins discrets.

Définition 4.3.1. *Soit* γ *un chemin discret. Alors, l'enveloppe externe de* γ, *notée* $\text{Hull}(\gamma)$, *est la face extérieure de la représentation du graphe* $G(\gamma)$.

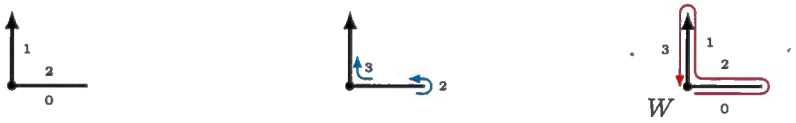
La différence entre la définition 4.3.1 et celle qui la précède est l'utilisation du plongement de γ dans le plan pour décrire l'enveloppe externe, au lieu d'utiliser un ensemble discret. Ce choix n'est pas arbitraire puisqu'il permet de traiter le cas où une partie du chemin discret est un segment de droite dégénéré, c'est-à-dire un ensemble euclidien de mesure nulle. Par exemple, la figure 4.14 illustre l'enveloppe externe du chemin codé par $w = \mathbf{021}$. Remarquons que par la définition 4.3.1, le bord des segments de droites est codé par un mot fermé, e.g. l'enveloppe externe du segment horizontal codé par $\mathbf{0}$ est codée par le mot $\mathbf{02}$.

La définition 4.3.1, constitue une généralisation adéquate de la notion d'enveloppe externe de chemins discrets. En effet, si γ code le contour d'une figure discrete M, alors γ est fermé et simple par définition. Par conséquent, $\gamma = \text{Hull}(\gamma)$ et puisque $\text{Hull}(M)$ est le bord $\partial(M)$ de M, on a par définition

$$\text{Hull}(M) = \partial(M) = \gamma = \text{Hull}(\gamma).$$

Étant donné qu'il existe une bijection entre l'ensemble des chemins discrets sur $\mathbb{Z}^2$ et celui des mots sur $\mathcal{F}$ auxquels on associe un point de départ, nous identifions

2. Ici, le mot *simple* fait référence à un chemin qui peut possiblement se chevaucher mais qui ne se croise pas.



(a) (b) (c)

Figure 4.14: (a) Le chemin discret $w = \mathbf{021}$, (b) son mot des différences finies $\Delta(w) = \mathbf{23}$ et (c) son enveloppe externe $\text{Hull}(w) = \mathbf{0213}$

γ à son mot de contour w et nous écrivons $\text{Hull}(w)$ au lieu de $\text{Hull}(\gamma)$.

Nous effectuons maintenant un court rappel des notions de base concernant la convexité digitale dont des exposés détaillés se trouvent dans (Brlek *et al.*, 2009b; Provençal, 2008). Un résumé des différentes approches utilisées dans cette discipline est aussi disponible dans (Eckhardt, 2001). Soit S un ensemble discret 8-connexe. S est *digitalement convexe* si et seulement si S est la digitalisation de Gauss d'un sous-ensemble convexe R de $\mathbb{R}^2$, i.e. $S = \text{GAUSS}(R) \cap \mathbb{Z}^2$. L'*enveloppe convexe de* S, notée $\text{Conv}(S)$, est l'intersection de tous les ensembles convexes contenant S. Dans le cas d'un chemin fermé et simple w, $\text{Conv}(w)$ est donné par la *factorisation de Spitzer de* w (voir (Brlek *et al.*, 2009b; Spitzer, 1956)). Étant donné un mot $w = w_1 w_2 \cdots w_n \in \mathcal{F}^*$, on effectue d'abord la factorisation standard de w (voir section 1.3). Puis, on calcule la partie NW de cette factorisation comme suit : au départ, considérer la liste $(b_1, b_2, \ldots, b_n) = (w_1, w_2, \ldots, w_n)$. Si la pente $\rho(b_i) = |b_i|_1/|b_i|_0$ de b_i est strictement plus petite que celle de b_{i+1} pour un certain entier positif i, alors

$$(b_1, b_2, \ldots, b_k) = (b_1, \ldots, b_{i-1}, b_i b_{i+1}, b_{i+2}, \ldots, b_k).$$

En répétant ce processus jusqu'à ce qu'il ne soit plus possible de concaténer de

mots, on obtient la factorisation de Spitzer de w. Les parties NE, SE et SW de cette factorisation sont obtenues en permutant les lettres de l'alphabet.

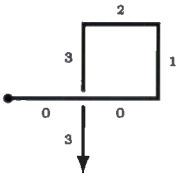
4.3.1 Algorithme pour l'enveloppe externe

Soit $w \in \mathcal{F}^*$ un chemin discret et $G(w)$ son graphe associé. Remarquons que l'application $g : w \mapsto G(w)$ n'est pas bijective puisqu'elle n'est pas injective. Par exemple, $u = \mathbf{0}$ et $v = \mathbf{02}$ admettent la même représentation graphique. Maintenant, considérons le graphe d'adjacence orienté $(G(w), \psi)$ où ψ est l'orientation horaire ou antihoraire. Comme mentionné à la section 4.1, ceci permet de définir des chemins dans $G(w)$. Nous calculons alors aisément l'enveloppe externe de w de la façon suivante : d'abord fixons une orientation ψ sur la surface $\mathbb{R}^2$ et posons $e_i = (u, v)$ un arc reliant le sommet u au sommet v dans $G(w)$ tel que e_i appartient à la face extérieure de $G(w)$ et tel qu'il suive l'orientation ψ. Ensuite, calculons $e_{i+1} = (v, \sigma_v(u))$ où σ_v est la permutation cyclique associée à v dans $G(w)$. En supposant que ψ est l'orientation antihoraire, on peut itérer ce processus afin d'obtenir la face extérieure de $G(w)$. Par exemple, en utilisant le schéma de rotation défini pour $w = \mathbf{001233}$ dans la figure 4.15 et débutant avec l'arc (A, E), on obtient la suite d'arcs

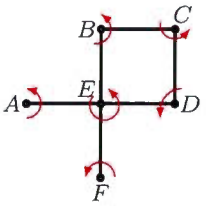
$$(A, E),\ (E, F),\ (F, E),\ (E, D),\ (D, C),\ (C, B),\ (B, E),\ (E, A)$$

qui correspond à l'enveloppe externe de w (voir figure 4.16).

L'exactitude de cette méthode découle de la *règle de la main droite* pour résoudre des labyrinthes. Cette méthode consiste à suivre le mur droit d'un labyrinthe en tournant systématiquement à droite à chaque intersection, de façon à trouver la sortie. Nous adaptons cet algorithme afin de calculer l'enveloppe externe de la façon suivante : étant donné un arc (u, v), l'arc adjacent $(v, \sigma_v(u))$ est obtenu



(a) Le graphe $G(w)$

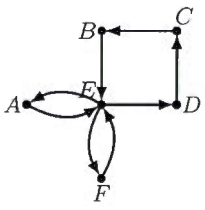


(b) Le graphe d'adjacence $(G(w), \psi)$, où ψ est l'orientation antihoraire

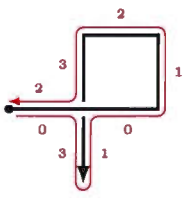
$$
\begin{array}{lcl}
A & : & E \\
B & : & C\ E \\
C & : & B\ D \\
D & : & C\ E \\
E & : & A\ F\ D\ B \\
F & : & E
\end{array}
$$

(c) Liste des permutations cycliques pour chaque sommet de $G(w)$

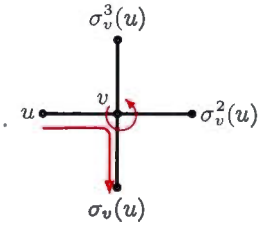
Figure 4.15: Schéma de rotation pour le chemin $w = \mathbf{001233}$



(a)



(b)



(c)

Figure 4.16: (a) La suite d'arcs obtenus en utilisant le schéma de la figure 4.15 ; (b) L'enveloppe externe de $w = \mathbf{001233}$; (c) La suite (u, v), $(v, \sigma_v(u))$ correspond à un virage à droite dans le graphe du chemin, pourvu que l'orientation fixée soit antihoraire

en "tournant à droite" au sommet v (voir la figure 4.16 (c)). Par conséquent, l'idée de notre algorithme est de se déplacer (i.e. marcher) le long du chemin en débutant sur un point appartenant à l'enveloppe externe, de tourner à droite à chaque intersection, jusqu'à revenir au point de départ. La discussion précédente assure que la marche ainsi créée est exactement l'enveloppe externe de w.

Afin d'implémenter efficacement cette procédure, nous devons surmonter plusieurs problèmes. Premièrement, comme mentionné plus haut, la marche doit débuter sur un point appartenant à l'enveloppe externe. Autrement, le chemin résultant ne décrit pas le bon objet. Nous résolvons cette question en choisissant systématiquement le point W de la décomposition standard de w comme point de départ.

Ensuite, à chaque fois qu'un chemin retourne au point W afin de continuer (l'exemple le plus simple est le chemin codé par $w = \mathbf{021}$, voir la figure 4.14), on doit s'assurer que l'algorithme ne termine pas avant que chacun de ces sous-chemins ait été exploré. Une solution simple afin de pallier ce problème est de garder une liste de tous les voisins de W faisant partie du chemin γ associé à w. Cette liste a au plus deux éléments puisqu'aucun sommet de γ n'est situé en-dessous ou à la gauche de W.

Finalement, il est nécessaire de reconnaître les points d'intersection et de décider quel arc constitue le virage le plus à droite. Nous résolvons ce problème en utilisant la structure d'arbre quaternaire radix décrite à la section 4.2. On obtient alors l'algorithme 5 suivant pour calculer l'enveloppe externe d'un chemin discret w.

D'abord, on construit l'arbre quaternaire radix G associé à w et enraciné à W. Ensuite, G est parcouru depuis la racine W en suivant le chemin représenté par w. À chaque intersection c, on doit :

(a) extraire la lettre α associée au vecteur $\overrightarrow{cv}$ pour chaque voisin v de c (e.g. $\textsc{Freeman}((1,0)) = \mathbf{0}$, $\textsc{Freeman}((0,1)) = \mathbf{1}$, $\textsc{Freeman}((-1,0)) = \mathbf{2}$ et

Algorithme 5 Calcul de l'enveloppe externe d'un chemin discret

Entrée: Un mot $w \in \mathcal{F}^*$ codant un chemin discret
Sortie: Un mot simple $w' \in \mathcal{F}^*$ décrivant Hull(w)

```
1: Soit W le point la plus en bas à gauche sur le rectangle englobant de w
2: Construire l'arbre quaternaire G associé à w et enraciné à W
3: Soit N l'ensemble de tous les voisins visités de W
4: c ← W + (1,0) s'il appartient à N ou W + (0,1) sinon
5: w' = FREEMAN(c − W)
6: tant que c ≠ W ou N ≠ ∅
7:     t = 2 mod 4
8:     pour tout voisin v de c
9:         si ([FREEMAN(v − c) − LST(w') + 1] mod 4) ≤ ([t + 1] mod 4) alors
10:            t ← FREEMAN(v − c) − LST(w')
11:            next ← v
12:    fin pour
13:    w' = w' · FREEMAN(next − c)
14:    N.RETIRER(c)
15:    c ← next
16: fin tant que
17: retourner w'
```

FREEMAN$((0,-1)) = \mathbf{3}$) ;

(b) déterminer le virage associé à chaque voisin v, c'est-à-dire $\Delta(w_c \cdot \alpha)$;

(c) choisir le virage le plus à droite, c'est-à-dire le plus près de 3.

Cette procédure termine lorsqu'on revient au point W et que tous ses voisins aient été considérés.

Théorème 4.3.2 (Exactitude de l'algorithme 5). *Pour tout mot $w \in \mathcal{F}^*$, l'algorithme 5 retourne* Hull(w).

Démonstration. Soit $k = |\,\text{Hull}(w)|$. Nous utilisons l'invariant de boucle suivant :

> Au début de la $i^{\text{ième}}$ itération de la boucle **tant que** à la ligne 6, w' est un préfixe de longueur i du mot de contour associé à Hull(w).

L'invariant est vérifié lors de la première exécution de la ligne 6 puisque à ce moment, w' est le premier pas de l'enveloppe externe de w calculé à la ligne 5. Maintenant, supposons que l'invariant est vérifié avant la $i^{\text{ième}}$ itération de la boucle. Alors, les lignes 8 à 12 déterminent le virage le plus à droite pour la coordonnée courante c. Ensuite, à la ligne 13, w' est concaténé avec le pas associé à ce virage. Par la règle de la main droite pour résoudre des labyrinthes, de tels virages à droite mènent toujours à des coordonnées appartenant à l'enveloppe externe de w. Par conséquent, à la fin de l'itération, w' est un préfixe de longueur $i+1$ du mot de contour associé à Hull(w). Finalement, à la fin de la boucle, w' est un préfixe de longueur k du mot de contour associé à Hull(w). En d'autres termes, $w' = \text{Hull}(w)$. Remarquons qu'étant donné que tout voisin de W appartient à Hull(w), la ligne 14 retire clairement tous les éléments de N, assurant ainsi que cet ensemble est vide à la fin de l'algorithme. □

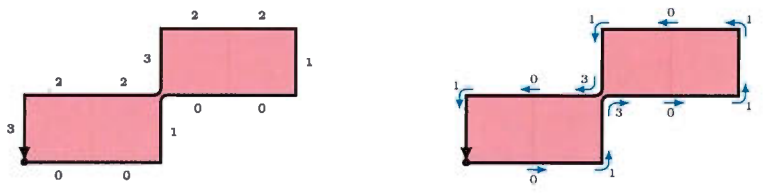
Nous vérifions à présent que l'algorithme 5 a une complexité temporelle et spatiale linéaire.

Théorème 4.3.3. *Soit $w \in \mathcal{F}^*$. Alors, l'algorithme 5 calcule* Hull(w) *en temps et en espace dans* $\Theta(|w|)$.

Démonstration. Premièrement, la structure d'arbre quaternaire radix est construite en temps linéaire, comme démontré au théorème 4.2.1. De plus, il est mentionné à la section 1.3 que le point W est calculé en temps linéaire. Par conséquent, les calculs à la ligne 2 sont effectués en temps linéaire. Ensuite, les lignes 1, 4 et 5 sont toutes calculées en temps constant. Par ailleurs, l'ensemble N est construit en temps linéaire en accédant aux voisins de la racine de l'arbre quaternaire, de sorte que la ligne 3 s'effectue en temps linéaire. Maintenant, puisque tout point de la structure a au plus quatre voisins, la boucle **pour** de la ligne 8 est exécutée au plus quatre fois par itération de la boucle **tant que**. La ligne 14 prend un

temps constant. Ceci découle du fait que N contient au plus deux éléments. Étant donné que les instructions aux lignes 7, 9, 10, 11, 13 et 15 se font toutes en temps constant, au plus $\mathcal{O}(k)$ calculs sont effectués lors de l'éxécution de la boucle **tant que**. Par conséquent, l'algorithme 5 est linéaire en temps. Finalement, l'algorithme nécessite un espace linéaire puisque seule la structure d'arbre quaternaire radix est utilisée afin de stocker les informations sur le chemin. □

Exemple 4.3.4. *Considérons le mot* $w = \mathbf{001100322223}$. *Alors, l'algorithme 5 retourne* $w' = \mathbf{001001223223}$ *(voir figure 4.17). On vérifie aisément que* w' *décrit un chemin simple représentant l'enveloppe externe de* w, *de sorte que* $\text{Hull}(w) = w'$.



(a) $w' = \mathbf{001001223223}$ (b) $\Delta(w') = \mathbf{01301101301}$

Figure 4.17: Enveloppe externe de $w = \mathbf{001100322223}$

Notre algorithme a été implémenté à l'aide du langage de programmation C++. Le code source est disponible à l'adresse `https://bitbucket.org/htremblay/sage-doctorat`.

Les notions d'enveloppes externes et d'enveloppes convexes sont intimement liées. En effet, l'algorithme 5 peut être utilisé afin de calculer, en temps et en espace linéaire, l'enveloppe convexe de tout chemin discret. La preuve est basée sur le résultat suivant :

Proposition 4.3.5. *Soit* $w \in \mathcal{F}^*$ *un mot de contour codant un chemin discret.*

Alors,

$$\text{Conv}(w) = \text{Conv}(\text{Hull}(w)).$$

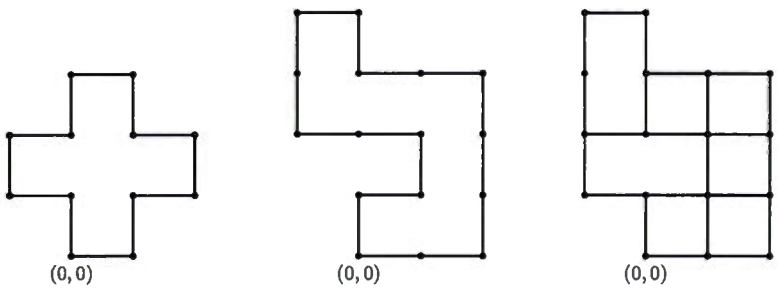
Démonstration. Si w est simple, alors $\text{Hull}(w) = w$ et le résultat est démontré. Sinon, supposons que w n'est pas simple. Alors, par définition, $\text{Hull}(w)$ est le bord de w. Comme $\text{Conv}(w)$ est l'intersection de tous les ensembles convexes contenant w, il doit aussi contenir $\text{Hull}(w)$, d'où $\text{Conv}(w) = \text{Conv}(\text{Hull}(w))$. □

Rappelons que $\text{Hull}(w)$ est un chemin simple (i.e. auto-évitant) pour tout chemin w. La proposition 4.3.5 fournit alors une procédure très simple afin de calculer l'enveloppe convexe d'un chemin discret. Elle utilise l'algorithme de calcul d'enveloppe convexe de chemins simples proposé par S. Brlek et al. dans (Brlek *et al.*, 2009b) :

1. Calculer d'abord $\text{Hull}(w) = w'$;
2. Calculer $\text{Conv}(w')$.

Il est clair que cet algorithme calcule l'enveloppe convexe d'un chemin discret en temps et en espace linéaire. En effet, nous avons montré grâce au théorème 4.3.3 que la première étape est effectuée en temps et en espace linéaires. De plus, il est démontré dans (Brlek *et al.*, 2009b) que la seconde étape est tout aussi efficace. Mentionnons que l'algorithme de Melkman (Melkman, 1987) peut aussi être utilisé à la deuxième étape afin de calculer l'enveloppe convexe en temps linéaire, les points considérés formant un polygone simple.

En terminant, il est intéressant de noter que dans le cas euclidien, le calcul de l'enveloppe convexe se fait de façon optimale en $\mathcal{O}(n \log n)$, où n est le nombre de points de la figure (voir (Chan, 1996; Graham, 1972) et (Chazelle, 1993; Goodman et O'Rourke, 2004; Yao, 1979) pour le cas général). La restriction de ce problème au contexte discret offre donc un avantage indéniable au niveau de l'efficacité algorithmique.



(a) $G(P_1)$ où $P_1 =$ $((0,0);\mathbf{010121232303})$ (b) $G(P_2)$ où $P_2 =$ $((0,0);\mathbf{001^3 22123300323})$ (c) $O(P_1, P_2)$

Figure 4.18: Deux chemins ainsi que leur graphe couvrant

4.4 Graphe couvrant

On nomme *polyomino de Jordan* tout polyomino sans trou tel que son contour est une courbe de Jordan. Par exemple, (p, w) où $p \in \mathbb{Z}^2$ et $w \in \mathcal{F}^*$ est fermé et simple est un polyomino de Jordan. Soit $P_1 = (p_1, w_1)$ et $P_2 = (p_2, w_2)$ deux polyominos de Jordan. Rappelons qu'étant donné le plongement d'un graphe planaire G dans le plan $\mathbb{R}^2$, les faces de G sont les composantes de $\mathbb{R}^2 \setminus G$ (section 1.2). Considérons les graphes $G(P_1)$ et $G(P_2)$ associés aux polyominos de Jordan P_1 et P_2. Il est clair que chaque graphe admet exactement deux faces : son intérieur et son extérieur. De plus, ces faces sont complètement déterminées par une marche sur le contour du polyomino. Notre but est alors de calculer les faces du *graphe couvrant de* P_1 *et* P_2, noté $O(P_1, P_2)$, c'est-à-dire le graphe obtenu en superposant les graphes de P_1 et P_2. Formellement, $O(P_1, P_2)$ est le graphe tel que f est une face de $O(P_1, P_2)$ si et seulement s'il existe des faces f_1 de $G(P_1)$ et f_2 de $G(P_2)$ telles que f est un sous-ensemble connexe maximal de $f_1 \cap f_2$ [3]. Par exemple, la figure 4.18 illustre deux polyominos de Jordan représentés par leur contour ainsi que le graphe couvrant associé.

---

3. Cette définition est tirée de (de Berg *et al.*, 2008) où les auteurs donnent un algorithme en $O(n \log n)$ pour calculer le graphe couvrant de deux régions du plan.

Le résultat suivant est un corollaire du célèbre théorème de Jordan pour les courbes polygonales : toute courbe de Jordan divise le plan en deux régions distinctes, soit l'intérieur et l'extérieur. Le lecteur est invité à consulter (Filippov, 1950) pour une des nombreuses preuves de ce résultat classique en topologie.

Proposition 4.4.1. *Soit γ une courbe polygonale de Jordan de $\mathbb{R}^2$ et $p, q \in \gamma$. Soit ξ un arc polygonal de Jordan allant de p à q tel que ξ est à l'intérieur de γ. Alors, $\xi \cup \gamma$ divise $\mathbb{R}^2$ en exactement trois régions dont les frontières sont des courbes polygonales de Jordan.*

Alors, en appliquant la proposition 4.4.1 au graphe couvrant $O(P_1, P_2)$ de deux polyominos de Jordan tels que leur intersection n'est pas vide, on obtient comme corollaire que le contour de chaque face de $O(P_1, P_2)$ est une courbe de Jordan (en fait, c'est une courbe polygonale de Jordan).

4.4.1 Algorithme de graphe couvrant

Soit $P_1 = (p_1, w_1)$ et $P_2 = (p_2, w_2)$ deux polyominos de Jordan. Nous donnons à présent un algorithme pour calculer le graphe couvrant de P_1 et P_2. Tout comme l'algorithme 5, ce dernier s'inspire de l'algorithme de la main droite pour résoudre des labyrinthes. De plus, la structure d'arbre quaternaire radix développée à la section 4.2 est de nouveau utilisée ici. Par ailleurs, il est dorénavant supposé que w_1 et w_2 codent leur contour respectif dans le sens antihoraire. Ceci est fait sans perte de généralité puisque l'orientation d'un mot se calcule en temps linéaire en utilisant la notion de *nombre d'enroulement*, c'est-à-dire le nombre de fois qu'une courbe s'enroule autour d'un point (voir (Brlek *et al.*, 2005c; Daurat et Nivat, 2005; Massé *et al.*, 2013)).

Remarquons que le graphe couvrant $O(P_1, P_2)$ n'est pas nécessairement connexe. Ce cas survient lorsque les contours de P_1 et P_2 ne s'intersectent pas. Nous traitons

donc ce cas séparément.

Nous sommes maintenant prêts à décrire l'algorithme 6. Comme première étape, on construit l'arbre quaternaire radix enrichi associé à P_1 et P_2 tel que décrit à la section 4.2. Ensuite, on vérifie si les contours de P_1 et P_2 s'intersectent. Dans le cas de frontières ne s'intersectant pas, le graphe couvrant $O(P_1, P_2)$ n'est pas connexe et il y a trois cas à considérer :

(i) P_1 est entièrement inclus dans P_2 ;

(ii) P_2 est entièrement inclus dans P_1 ;

(iii) les polyominos sont disjoints.

On traite ces trois cas aisément en utilisant un algorithme afin de déterminer si un point de $P1$ est dans $P2$ et vice-versa. Par exemple, on peut utiliser le nombre d'enroulement ou l'algorithme de Dobkin et Lipton (Dobkin et Lipton, 1976) afin de réaliser cette tâche en temps linéaire. Sinon, dans le cas de contours s'intersectant, le graphe couvrant $O(P_1, P_2)$ est calculé en utilisant la fonction GRAPHECOUVRANTNONDISJOINT décrite à l'algorithme 7.

La fonction GRAPHECOUVRANTNONDISJOINT est implémentée comme suit. Dans un premier temps, on choisit un arc (p, p') de T pour ensuite suivre le chemin antihoraire induit par cet arc et tourner à gauche à chaque point d'intersection, jusqu'à revenir au point de départ p. Lors du parcours du chemin induit, on supprime de T les arcs visités à fur et à mesure qu'ils sont parcourus. Puisque tout arc de voisinage de T appartient à exactement une face du graphe couvrant, on obtient une liste de courbes polygonales de Jordan décrivant le contour de toutes les faces de $O(P_1, P_2)$. Il reste à identifier chaque face à son (ou ses) polyomino(s). Cet objectif est aisément accompli en gardant en mémoire la couleur des arcs sur un chemin donné. Cette procédure est décrite formellement à l'algorithme 7.

Algorithme 6 Calcul de graphe couvrant de deux polyominos de Jordan

Entrée: Deux polyominos de Jordan P_1 et P_2 ayant leur contour respectif orienté dans le sens antihoraire.

Sortie: L'ensemble $\{(w, i) \mid w \in \mathcal{F}^*$ et $i \in 2^{\{0,1\}}\}$ des faces du graphe couvrant $O(P_1, P_2)$ où w est le mot de contour de la face et i dénote à quel polyomino elle appartient.

1: **Fonction** GRAPHECOUVRANT($P_1 = (p_1, w_1)$ et $P_2 = (p_2, w_2)$)
2: ▷ On insère P_1 et P_2 dans la structure de données
3: $T \leftarrow$ ARBREQUATERNAIRERADIX()
4: T.CREERCHEMIN($w_1, 0$)
5: T.ALLERVERS($p_2 - p_1$)
6: T.CREERCHEMIN($w_2, 1$)
7: **si** aucun point de T n'est visité au moins deux fois **alors**
8: **si** $P_1 \subset P_2$ **alors**
9: **retourner** $\{(P_1, \{0,1\}), (P_2, \widehat{P_1}, \{1\}), (\widehat{P_2}, \emptyset)\}$
10: **sinon si** $P_2 \subset P_1$ **alors**
11: **retourner** $\{(P_2, \{0,1\}), (P_1, \widehat{P_2}, \{0\}), (\widehat{P_1}, \emptyset)\}$
12: **sinon**
13: **retourner** $\{(P_1, \{0\}), (P_2, \{1\}), (\widehat{P_1}, \widehat{P_2}, \emptyset)\}$
14: **sinon**
15: **retourner** GRAPHECOUVRANTNONDISJOINT(T)

Elle utilise une file de type *first-in-first-out* (littéralement *premier entré, premier sorti*) afin de traiter l'information sur les points visités[4]. Étant donné un arc e, la fonction *e.couleur()* retourne la couleur de e, c'est-à-dire rouge, bleu, bicolore ou sans couleur. Notons que la possibilité d'avoir des arc sans couleur permet de considérer les faces extéreures des polyominos. Finalement, conformément à la règle de la main droite, la fonction PLUSAGAUCHE considère tous les voisins d'un arc sans égard à la couleur. De cette façon, on s'assure que tous les arcs sont traités par l'algorithme.

Comme mentionné à la section 4.2, les fichiers nécessaires à l'implémentation de

4. Un algorithme similaire est présenté dans (de Berg *et al.*, 2008) où une liste doublement chaînée et une méthode basée sur un balayage du plan sont utilisées afin d'obtenir une borne temporelle dans $\mathcal{O}(n \log n)$ pour le cas euclidien.

Algorithme 7 Graphe couvrant de deux polyominos de Jordan s'intersectant

Entrée: Une structure d'arbre quaternaire radix contenant deux polyominos de Jordan.

Sortie: L'ensemble des faces du graphe couvrant $O(P_1, P_2)$.

```
1:  Fonction GrapheCouvrantNonDisjoint(T : arbre quaternaire radix)
2:      O ← ∅
3:      Soit Q une file contenant tous les points de T ;
4:      tant que Q n'est pas vide
5:          s ← le premier élément de Q ;
6:          si s a au moins un arc sortant (s, s') alors
7:              F ← NewFace(s, ε, ∅)
8:              e ← (p, p') ← (s, s')
9:              répéter
10:                 F.Couleurs.Ajouter(e.couleur())
11:                 F.Mot ← F.Mot · Freeman(p' − p)
12:                 p'' ← PlusAGauche(e)
13:                 T.Supprimer(e)
14:                 e ← (p, p') ← (p', p'')
15:             jusqu'à ce que p = s
16:             O.Ajouter(F)
17:         sinon
18:             Q.Defiler(s) ;
19:     fin tant que
20:     retourner O
```

cet algorithme sont disponibles à l'adresse `https://bitbucket.org/htremblay/sage-doctorat`.

Théorème 4.4.2 (Exactitude de l'algorithme 7). *Pour toute paire de polyominos de Jordan P_1 et P_2 s'intersectant sur leur contour, l'algorithme 7 retourne toutes les faces de $O(P_1, P_2)$.*

Démonstration. La première fois que la ligne 4 est exécutée, aucune face du graphe couvrant n'a encore été calculée. Les lignes 7 et 8 initialisent une nouvelle face en choisissant un arc sortant du point courant p. Ensuite, il y a deux cas à considérer, selon que p ait un arc sortant ou pas.

Si p a un arc sortant, alors on suit le chemin induit par cet arc en tournant à gauche à chaque point d'intersection, tout en gardant en mémoire les couleurs de l'arête (lignes 10 à 14). Par l'algorithme de la main droite, ceci définit le contour d'une face du graphe couvrant $O(P_1, P_2)$. Notons que cette face n'a pas encore été calculée puisque chaque arc appartient au contour d'exactement une face de $O(P_1, P_2)$ et sont supprimés à fur et à mesure qu'ils sont parcourus (ligne 13).

Sinon, p est tout simplement défilé de la file Q et aucune nouvelle face n'est calculée. Finalement, à la fin de la boucle, tous les arcs sortant pour tous les points de la structure de données ont été considérés exactement une fois. Comme tout arc de la structure appartient au contour d'exactement une face du graphe couvrant (les arêtes sont représentées par deux arcs de même direction mais de sens opposé), toutes les faces de $O(P_1, P_2)$ ont été calculées. □

Corollaire 4.4.3 (Exactitude de l'algorithme 6). *Pour toute paire de polyominos de Jordan P_1 et P_2, l'algorithme 6 retourne toutes les faces de $O(P_1, P_2)$.*

Démonstration. Il y a deux cas possibles selon que $O(P_1, P_2)$ est connexe ou non. On a immédiatement que $O(P_1, P_2)$ est connexe si et seulement si P_1 et P_2 s'intersectent sur leur contour. Si ce n'est pas le cas, alors il reste à déterminer la position relative des deux polyominos. Comme on obtient facilement cette position en étudiant l'enroulement du contour de $P1$ par rapport à celui de $P2$ et vice-versa, on a le résultat. □

Exemple 4.4.4. *Soit*

$$\begin{aligned} P_1 &= ((0,0), \mathbf{0^3 1^5 23^4 21123^3}) \\ P_2 &= ((0,2), \mathbf{010^3 12^3 123^3}). \end{aligned}$$

Le graphe couvrant $O(P_1, P_2)$ est représenté à la figure 4.19. Puisque les contours

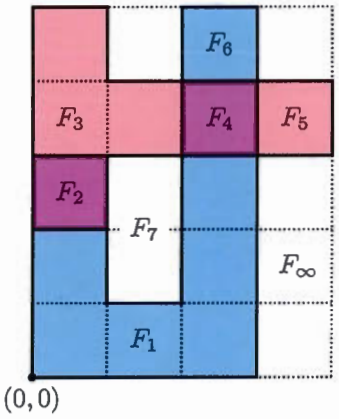


Figure 4.19: Les faces du graphe couvrant de $P_1 = ((0,0), \mathbf{0^3 1^5 23^4 21123^3})$ et $P_2 = ((0,2), \mathbf{010^3 12^3 123^3})$ calculées par l'algorithme 7

des polyominos s'intersectent, l'algorithme 7 s'applique. Après calculs, on obtient les faces suivantes pour le graphe couvrant :

$$\begin{aligned}
\text{FACE}_\infty &= \{(0,0), \mathbf{1^5 0301030323^3 2^3}, \emptyset\} \\
\text{FACE}_1 &= \{(0,0), \mathbf{0^3 1^3 23321233}, \{\text{BLEU}\}\} \\
\text{FACE}_2 &= \{(0,2), \mathbf{0123}, \{\text{BLEU, ROUGE}\}\} \\
\text{FACE}_3 &= \{(0,3), \mathbf{00121233}, \{\text{ROUGE}\}\} \\
\text{FACE}_4 &= \{(2,3), \mathbf{0123}, \{\text{BLEU, ROUGE}\}\} \\
\text{FACE}_5 &= \{(3,3), \mathbf{0123}, \{\text{ROUGE}\}\} \\
\text{FACE}_6 &= \{(2,4), \mathbf{0123}, \{\text{BLEU}\}\} \\
\text{FACE}_7 &= \{(1,1), \mathbf{011233}, \emptyset\}.
\end{aligned}$$

Tout comme l'algorithme 5 pour calculer l'enveloppe externe d'un chemin discret, l'algorithme 7 a une complexité temporelle et spatiale linéaire.

Théorème 4.4.5. *Soit $P_1 = (p_1, w_1)$ et $P_2 = (p_2, w_2)$ deux chemins discrets*

fermés et simples. Alors, le graphe couvrant de G_{P_1} et G_{P_2} est calculé en temps et en espace dans $\mathcal{O}(|w_1| + |w_2|)$.

Démonstration. Toutes les conditions aux lignes 12, 7, 8 et 10 sont vérifiées en temps linéaire puisque la structure de données est construite en temps linéaire (ligne 3). De plus, la ligne 5 a un coût linéaire par le corollaire 4.2.2. Toutes les autres lignes s'effectuent en temps constant.

Il reste à compter le nombre de répétitions des boucles **tant que** et **répéter** dans l'algorithme. D'abord, remarquons qu'aucune opération ENFILER n'est effectuée. De plus, les éléments de Q sont défilés à chaque fois qu'ils n'ont pas d'arc sortant. Or, ces arcs sont retirés à la ligne 13. Par conséquent, la boucle **tant que** est répétée $\mathcal{O}(|w_1| + |w_2|)$ fois. Par ailleurs, les instructions de la boucle **répéter** sont exécutées une fois par arc de la structure de données. Puisque le nombre d'arcs est au plus $2(|w_1| + |w_2|)$, on a que le nombre de répétitions de cette boucle est de l'ordre de $\mathcal{O}(|w_1| + |w_2|)$ fois.

Finalement, comme l'algorithme 6 repose presque entièrement sur la structure de données d'arbre quaternaire radix pour stocker l'information, sa complexité spatiale est linéaire. □

En terminant, il est intéressant de noter qu'il existe une relation simple entre le nombre de faces du graphe couvrant et le nombre de points d'intersection. Soit e, e_1 et e_2 le nombre d'arêtes de $O(P_1, P_2)$, G_{P_1} et G_{P_2} respectivement. Comme P_1 et P_2 sont des polyominos de Jordan, on a que

$$v_1 = e_1 = |w_1| \text{ et } v_2 = e_2 = |w_2|.$$

Posons i le nombre de points d'intersection du graphe couvrant, c'est-à-dire le

nombre de points ayant strictement plus de deux voisins. Alors,

$$v = v_1 + v_2 - i \text{ et } e \leq e_1 + e_2.$$

Par la formule d'Euler pour les graphes planaires[5], on sait que le nombre f de faces d'un graphe planaire donné s'exprime par l'équation

$$f = 2 - v + e.$$

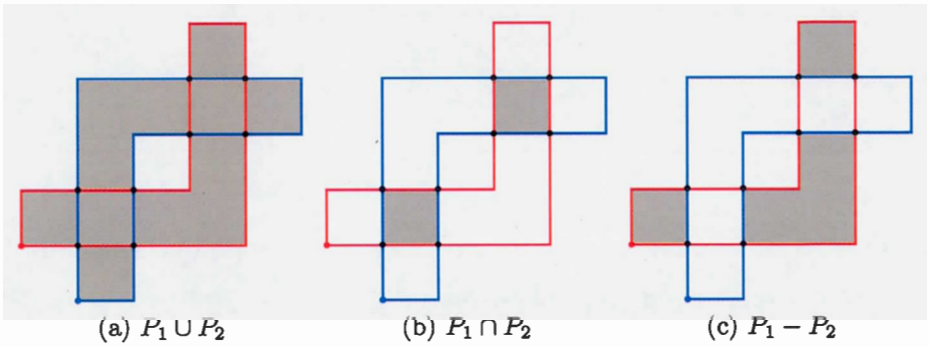
Par conséquent, le nombre de faces de $O(P_1, P_2)$ respecte l'inégalité suivante :

$$\begin{aligned} f &= 2 - v + e \\ &\leq 2 - (v_1 + v_2 - i) + e_1 + e_2 \\ &= 2 + i. \end{aligned}$$

4.4.2 Union, intersection et différence de polyominos

Si les polyominos sont représentés par des matrices booléennes, il est très facile de calculer leur intersection, leur union et leur différence : il suffit d'appliquer l'opération logique appropriée sur chaque bit. Ici, nous nous intéressons au calcul de ces opérations en étudiant uniquement le contour des figures. Des algorithmes similaires ont été proposés afin de calculer l'aire de l'intersection, de l'union et de la différence de deux polyominos. Ceux-ci se basent sur une version discrète du théorème de Green (Brlek *et al.*, 2005a). Cependant, aucun algorithme linéaire

5. Il existe bon nombre de preuves de ce résultat classique en théorie des graphes. Par exemple, (Gross et Tucker, 1987; Diestel, 2010).



(a) $P_1 \cup P_2$ (b) $P_1 \cap P_2$ (c) $P_1 - P_2$

Figure 4.20: Diverses opérations booléennes sur deux polyominos de Jordan P_1 (en rouge) et P_2 (en bleu)

pour ces opérations n'a encore été développé dans la littérature.

À partir de l'algorithme 6, il est assez simple de construire de tels algorithmes, du moins dans le cas où les deux polyominos P_1 et P_2 sont des polyominos de Jordan. Par contre, le résultat n'est pas nécessairement un polyomino de Jordan. En général, le résultat de ces opérations est une famille de polyominos avec possiblement des trous. Par exemple, la figure 4.20 illustre de tels cas.

Afin d'obtenir le résultat de ces opérations booléennes, on commence par calculer le graphe couvrant $O(P_1, P_2)$. Puis, on choisit les régions du graphe couvrant correspondant à l'opération effectuée : $P_1 \cap P_2$ est l'ensemble des faces dont l'ensemble de couleurs est $\{0, 1\}$ (i.e. appartenant à la fois à P_1 et à P_2), $P_1 \cup P_2$ est l'ensemble des faces dont l'ensemble de couleurs est non vide (i.e. appartenant à P_1, à P_2 ou au deux) et $P_1 \setminus P_2$ est l'ensemble des faces dont l'ensemble de couleurs est $\{0\}$ (i.e. appartenant à P_1 mais pas à P_2). Il découle alors directement de la section 4.4.1 que l'intersection, l'union et la différence de polyominos de Jordan sont calculées en temps et en espace linéaires.

CONCLUSION

Dans cette thèse, nous avons étudié les figures discrètes à l'aide de la théorie des graphes et de la combinatoire des mots en analysant leur contour. Ce faisant, nous avons développé plusieurs algorithmes efficaces pour effectuer bon nombre d'opérations géométriques fondamentales. En plus de fournir un modèle pertinent à l'étude de l'imagerie numérique, nous contribuons à l'avancement des connaissances en géométrie discrète, en théorie des codes, en combinatoire des mots et en algorithmique. De plus, l'exactitude et l'originalité de cette recherche ont été validées par des pairs par le biais de publications avec comité de lecture, tant dans des conférences que dans des revues reconnues en mathématiques et en informatique.

Après un court rappel des notions de bases en géométrie discrète, en théorie des graphes et en combinatoire des mots, nous avons introduit le concept de réseau parallélogramme. Ceci a permis la résolution d'une conjecture concernant les codes circulaires. Ensuite, nous avons étudié les notions de polyominos premiers et composés. Finalement, nous avons fourni un cadre d'algorithmes optimaux pour effectuer plusieurs opérations géométriques fondamentales sur les figures discrètes, notamment l'enveloppe externe, l'enveloppe convexe, l'union, l'intersection et la différence.

Évidemment, plusieurs questions restent en suspens et plusieurs nouvelles avenues méritent d'être explorées. D'abord, de récents calculs informatiques nous poussent à supposer que la décomposition de polyominos composés en polyominos premiers est unique pour certaines classes de morphismes parallélogrammes. Il serait donc intéressant, dans un premier temps, de démontrer l'unicité de cette décomposition

et, dans un deuxième temps, de qualifier et quantifier la famille des polyominos premiers. Il est intéressant de noter que le théorème 2.3.2 constitue, à notre avis, un premier pas capital vers la démonstration de la conjecture 3.2.5. Par ailleurs, comme l'ensemble des polyominos parallélogrammes $\mathcal{P}$ est un monoïde, l'opération de composition induit une action de monoïde $\mathcal{P} \times X \rightarrow X$ où X est l'ensemble de tous les polyominos. Ceci suggère la pertinence d'une approche algébrique afin d'étudier ce problème.

Ensuite, les algorithmes et opérations géométriques étudiés au chapitre 4 méritent eux aussi d'être investigués davantage. Comme première étape, il serait intéressant d'étendre les algorithmes présentés à tout type de polyominos, notamment aux polyominos avec trous. Ceci pourrait être accompli, par exemple, en considérant chaque polyomino comme une famille de mots sur $\mathcal{F}$. Une autre avenue pertinente serait d'étendre les définitions des algorithmes pour les appliquer à un ensemble quelconque de polyominos, au lieu de considérer seulement deux polyominos. Il serait alors intéressant d'étudier les complexités temporelle et spatiale ainsi obtenues. Finalement, la notion de chemin discret se généralise à trois dimensions. Un projet de recherche futur pourrait alors s'attarder à la généralisation de nos algorithmes à l'espace discret $\mathbb{Z}^3$.

RÉFÉRENCES

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

Beauquier, D., Nivat, M., Remilia, E. et Robson, M. (1995). Tiling figures of the plane with two bars. *Computational Geometry*, *5*(1), 1–25.

Bern, M. W., Flaherty, J. E. et Luskin, M. (1999). *Grid Generation and Adaptive Algorithms*. Springer-Verlag.

Berstel, J., Lauve, A., Reutenauer, C. et Saliola, F. (2008). *Combinatorics on Words : Christoffel Words and Repetition in Words*, volume 27 de *CRM monograph series*. American Mathematical Society. 147 pages.

Berstel, J., Perrin, D. et Reutenauer, C. (2009). *Codes and Automata*, volume 129 de *Encyclopedia of Mathematics and its Applications*. Cambridge University Press.

Blondin Massé, A., Brlek, S., Garon, A. et Labbé, S. (2011). Two infinite families of polyominoes that tile the plane by translation in two distinct ways. *Theoretical Computer Science*, *412*(36), 4778 – 4786. `http://dx.doi.org/10.1016/j.tcs.2010.12.034`. Récupéré de `http://www.sciencedirect.com/science/article/pii/S0304397510007358`

Blondin Massé, A., Brlek, S. et Tremblay, H. (2015). Efficient operations on discrete paths. *Theoretical Computer Science*. `http://dx.doi.org/http://dx.doi.org/10.1016/j.tcs.2015.07.033`. Récupéré de `http://www.sciencedirect.com/science/article/pii/S0304397515006672`

Blondin Massé, A., Garon, A. et Labbé, S. (2012). Combinatorial properties of double square tiles. *Theoretical Computer Science*. `http://dx.doi.org/10.1016/j.tcs.2012.10.040`. Récupéré de `http://www.sciencedirect.com/science/article/pii/S0304397512009723`

Blondin Massé, A. (2012). *À l'intersection de la combinatoire des mots et de la géométrie discrète : Palindromes, symétries et pavages*. (Thèse de doctorat). Université du Québec à Montréal.

Blondin Massé, A., Lapointe, M. et Tremblay, H. (2016). Parallelogram morphisms and circular codes. Dans *Language and Automata Theory and Applications, 10th International Conference, LATA 2016*. LNCS, Elsevier.

Blondin Massé, A., Makhtar Tall, A. et Tremblay, H. (2014). On the arithmetics of discrete figures. Dans *Language and Automata Theory and Applications, 8th International Conference, LATA 2014*, Lecture Notes in Computer Science 8370, 198–209. Springer.

Bose, P., Dujmović, V., Hurtado, F. et Morin, P. (2009). Connectivity-preserving transformations of binary images. *Computer Vision and Image Understanding, 113*, 1027–1038.

Brlek, S., Koskas, M. et Provençal, X. (2009a). A linear time and space algorithm for detecting path intersection. Dans *Discrete Geometry for Computer Imagery, 15th IAPR International Conference, DGCI 2009*, Lecture Notes in Computer Science 5810, 398–409. Springer.

Brlek, S., Koskas, M. et Provençal, X. (2011). A linear time and space algorithm for detecting path intersection. *Theoretical Computer Science, 412*, 4841–4850.

Brlek, S., Labelle, G. et Lacasse, A. (2005a). Algorithms for polyominoes based on the discrete green theorem. *Discrete Applied Mathematics, 147*(2-3), 187–205. `http://dx.doi.org/10.1016/j.dam.2004.09.011`. Récupéré de `http://dx.doi.org/10.1016/j.dam.2004.09.011`

Brlek, S., Labelle, G. et Lacasse, A. (2005b). The discrete green theorem and some applications in discrete geometry. *Theoretical Computer Science, 346*, 200–225.

Brlek, S., Labelle, G. et Lacasse, A. (2005c). A note on a result of daurat and nivat. Dans *Developments in Language Theory, 9th International Conference, DLT 2005, Palermo, Italy, July 4-8, 2005, Proceedings*, 189–198. `http://dx.doi.org/10.1007/11505877_17`. Récupéré de `http://dx.doi.org/10.1007/11505877_17`

Brlek, S., Lachaud, J.-O., Provençal, X. et Reutenauer, C. (2009b). Lyndon + Christoffel = digitally convex. *Pattern Recognition, 42*, 2239–2246.

Brlek, S., Tremblay, H., Tremblay, J. et Weber, R. (2014). Efficient computation of the outer hull of a discrete path. Dans *Discrete Geometry for Computer Imagery, 18th IAPR International Conference, DGCI 2014*, Lecture Notes in Computer Science 8668, 122–133. Springer.

Chan, T. M. (1996). Optimal output-sensitive convex hull algorithms in two and three dimensions. *Discrete & Computational Geometry*, *16*, 361–368.

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

Chazelle, B. (1993). An optimal convex hull algorithm in any fixed dimension. *Discrete & Computational Geometry*, *10*, 377–409.

Cormen, T. H., Leiserson, C. E., Rivest, R. L. et Stein, C. (1990). *Introduction to algorithms* (troisième éd.). MIT Press et McGraw-Hill.

Daurat, A. et Nivat, M. (2005). Salient and reentrant points of discrete sets. *Discrete Applied Mathematics*, *151*, 106–121. `http://dx.doi.org/10.1016/j.dam.2005.02.024`

de Berg, M., Cheong, O., van Kreveld, M. et Overmars, M. (2008). *Computational geometry : Algorithms and applications* (troisième éd.). Springer.

Diestel, R. (2010). *Graph theory* (quatrième éd.). Springer-Verlag.

Dobkin, D. et Lipton, R. J. (1976). Multidimensional searching problems. *Society for Industrial and Applied Mathematics Journal on Computing*, *5*(2), 181–186.

Du, Z.-M., Ye, F.-Y., Shi, H. et Zhu, G.-P. (2015). A fast recovery method of 2d geometric compressed sensing signal. *Circuits, Systems, and Signal Processing*, *34*(5), 1711–1724. `http://dx.doi.org/10.1007/s00034-014-9913-3`. Récupéré de `http://dx.doi.org/10.1007/s00034-014-9913-3`

Eckhardt, U. (2001). In G. Bertrand, A. Imiya, et R. Klette (dir.), *Digital and Image Geometry* chapitre Digital Lines and Digital Convexity, 209–228. New York, NY, USA : Springer-Verlag New York, Inc.

Filippov, A. F. (1950). An elementary proof of Jordan's theorem. *Uspekhi Mat. Nauk*, *5*(39), 173–176.

Freeman, H. (1961). On the encoding of arbitrary geometric configurations. *IRE Transactions on Electronic Computers*, *EC-10*(2), 260–268.

Fulton, W. (1998). *Intersection Theory* (seconde éd.). Springer.

Gamelin, T. W. et Greene, R. E. (1999). *Introduction to Topology* (seconde éd.). Dover.

Golomb, S. W. (1965). *Polyominoes.* Charles Scribners' Sons.

Goodman, J. E. et O'Rourke, J. (2004). *Handbook of discrete and computational geometry* (seconde éd.). CRC Press.

Goppa, V. D. (1977). Codes associated with divisors. *Problems of Information Transmission*, *12*(1), 22–27.

Graham, R. A. (1972). An efficient algorithm for determining the convex hull of a finite planar set. *Information Processing Letters*, *1*(4), 132–133.

Gross, J. L. et Tucker, T. W. (1987). *Topological graph theory.* Wiley & Sons.

Gross, J. L. et Yellen, J. (2004). *Handbook of graph theory.* CRC Press.

Gruber, P. M. (2007). *Convex and discrete geometry.* Springer-Verlag.

Grünbaum, B. et Shephard, G. C. (1987). *Tilings and patterns.* W. H. Freeman and co.

Høholdt, T., van Lint, J. H. et Pellikaan, R. (1998). Algebraic geometry codes. In W. H. V.S. Pless et R. Brualdi (dir.), *Handbook of Coding Theory*, volume 1. Amsterdam : Elsevier.

Jensen, I. et Guttmann, A. J. (2000). Statistics of lattice animals (polyominoes) and polygons. *Journal of Physics A*, *33*, L257–L263.

Kelley, J. L. (1955). *General topology* (seconde éd.). Springer-Verlag.

Kenyon, C. et Kenyon, R. (1992). Tiling a polygon with rectangles. Dans *Proceedings of the 33rd Annual Symposium on Fondations of Computer Science*, 610–619. IEEE.

Kim, M.-A., Lee, E.-J., Cho, H.-G. et Park, K.-J. (1997). A visualization technique for DNA walk plot using k-convex hull. Dans *Proceedings of the fifth international conference in Central Europe in computer graphics and visualization*, 212–221., Plzeň , Czech Republic. Západočeská univerzita.

Kim, Y. S. (1992). Recognition of form features using convex decomposition. *Computer-Aided Design*, *24*(9), 461–476.

Kobilarov, M., Desbrun, M., Marsden, J. et Sukhatme, G. (2007). A discrete geometric optimal control framework for systems with symmetries. Dans *Proceedings of Robotics : Science and Systems*, Atlanta, GA, USA.

Labelle, G. et Lacasse, A. (2009). Discrete versions of Stokes' theorem based on families of weights on hypercubes. Dans *Discrete Geometry for Computer Imagery, 15th IAPR International Conference, DGCI 2009*, Lecture Notes in Computer Science 5810, 229–239. Springer.

Linial, N., London, E. et Rabinovich, Y. (1995). The geometry of graphs and some of its algorithmic applications. *Combinatorica*, *15*(2), 215–245. http://dx.doi.org/10.1007/BF01200757. Récupéré de http://dx.doi.org/10.1007/BF01200757

Lothaire, M. (1997). *Combinatorics on words.* Cambridge University Press.

Massé, A. B., Garon, A. et Labbé, S. (2013). Combinatorial properties of double square tiles. *Theor. Comput. Sci.*, *502*, 98–117. http://dx.doi.org/10.1016/j.tcs.2012.10.040. Récupéré de http://dx.doi.org/10.1016/j.tcs.2012.10.040

McCallum, D. et Avis, D. (1979). A linear algorithm for finding the convex hull of a simple polygon. *Information Processing Letters*, *9*(5), 201–206.

McEliece, R. J. (1978). A public-key cryptosystem based on algebraic coding theory. *Jet Propulsion Laboratory Deep Space Network Progress Report*, *4244*, 114–116.

Melkman, A. (1987). On-line construction of the convex hull of a simple polyline. *Information Processing Letters*, *25*, 11–12.

Okabe, A., Boots, B. et Sugihara, K. (1992). *Spacial tesselations : Concepts and applications of Voronoi diagrams.* Wiley.

Pavlidis, T. (1977). *Structural Pattern Recognition.* Springer-Verlag.

Provençal, X. (2008). *Combinatoire des mots, géométrie discrète et pavages.* (Thèse de doctorat). Université du Québec à Montréal.

Redelmeier, D. H. (1981). Counting polyominoes : yet another attack. *Discrete Mathematics*, *36*, 191–203.

Rosenfeld, A. et Klette, R. (2004). *Digital geometry : Geometric methods for digital image analysis.* Morgan Kaufmann.

Sherali, H. et Adams, W. (1990). A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM Journal on Discrete Mathematics*, *3*(3), 411–430.

Spitzer, F. (1956). A combinatorial lemma and its application to probability theory. *Transactions of the American Mathematical Society*, *82*, 323–339.

Stein, W. *et al.* (2012). *Sage Mathematics Software (Version 4.8).* The Sage Development Team. http://www.sagemath.org.

Stewart, J. (2005). *Analyse, Concepts et Contextes Volume 2. Fonctions de plusieurs variables* (seconde éd.). de Boeck.

Sunada, T. (2013). *Topological Crystallography.* Springer.

Tarjan, R. (1972). Depth-First Search and Linear Graph Algorithms. *Society for Industrial and Applied Mathematics Journal on Computing, 1*(2), 146–160.

Veblen, O. (1905). Theory on plane curves in non-metrical analysis situs. *Trans. Amer. Math. Soc., 6*, 83–98.

Vella, A. (2005). *A fundamentally topological perspective on graph theory.* (Thèse de doctorat). University of Waterloo.

Wijshoff, H. et van Leeuwen, J. (1984). Arbitrary versus periodic storage schemes and tessellations of the plane using one type of polyomino. *Information and Control, 62*(1), 1 – 25.

Yao, A. C.-C. (1979). *A lower bound to finding the convex hulls.* (Thèse de doctorat). Stanford University.

INDEX