Géométrie pour l'auto-assemblage

Florent Becker

Table des matières

Chapitre 1

Introduction

L'auto-assemblage est un phénomène naturel omniprésent, de manière presque tautologique : de nombreuses structures naturelles émergent parce que leurs composants se sont associés à la faveur d'interactions locales. La nature de ces interactions locales dicte la forme finale de la structure. C'est ainsi que se forment les récifs coralliens, ou que les cristaux croissent, par exemple.

D'un autre côté, les objets fabriqués par l'homme sont en général assemblés par un procédé externe, qu'il s'agisse de l'assemblage de briques par un maçon, ou de l'ajustement de nano-tubes au microscope électronique à balayage. Comme on le voit dans ce deuxième exemple, une telle façon de faire s'accorde mal avec la nanotechnologie, car la quantité d'efforts nécessaires pour réaliser les assemblages devient insurmontable. D'où l'idée de contrôler les phénomènes d'auto-assemblage afin de réaliser des artefacts intéressants sans intervention directe d'un opérateur extérieur.

L'idée est de se donner un certain nombre de briques de base, avec un minimum de capacités –il suffira qu'elles puissent se reconnaître mutuellement, et de les laisser s'assembler pour obtenir la forme voulue. La difficulté est de contrôler des interactions simples entre les briques de bases pour obtenir une structure intéressante à la fin.

Cette approche ne permet pas seulement d'assembler des objets, mais également de faire des calculs chimiquement : il s'agit d'encoder la réponse d'un calcul dans une forme, après quoi on assemble cette forme pour effectuer le calcul.

De l'exploration de l'auto-assemblage par Winfree est né un modèle, celui des pavages auto-assemblants, qui est à la fois proche de l'informatique théorique, puisqu'il s'agit d'une variante des pavages de Wang, et réalisable en pratique. Il semble donc que l'auto-assemblage soit une notion importante, dont l'étude théorique est à entreprendre. En particulier, comprendre quelles

sont les limites inhérentes à ce modèle et les constructions naturelles semble important.

Pour cela, nous nous sommes attachés à comprendre la géométrie de l'auto-assemblage, qui joue un grand rôle dans l'établissement de constructions naturelles. En particulier, la *condition d'ordre* est une condition de régularité assez restrictive, mais qui est indispensable pour pouvoir concevoir des systèmes auto-assemblants. Tous les systèmes de la littérature, ou presque sont ordonnés. Nous allons montrer qu'avec cette condition, l'auto–assemblage forme un modèle cohérent et riche, avec la possibilité de développer une forme de langage de programmation, un début d'universalité propre, et une grande richesse de construction.

1.1 Nanotechnologie, ordinateur à ADN, etc

Donnons maintenant quelques détails sur les origines bio-informatiques du modèle de l'auto-assemblage. Il s'agit d'un modèle de calcul par accrétion de carrés d'ADN.

1.1.1 Les propriétés informatiques de l'ADN

L'ADN est une molécule par nature «informatique», puisqu'elle code de l'information, la réplique, et est à la source de son interprétation par le vivant. Elle possède ainsi des propriétés combinatoires qui la rendent particulièrement intéressante pour la bio-informatique, tout en étant par certains aspects plus simple à manipuler que les protéines.

La propriété essentielle de la molécule d'ADN est sa variabilité, et le fait qu'elle encode une séquence sur un alphabet fini sous forme d'une suite linéaire de bases. Une molécule d'ADN est ainsi, naturellement, un mot, objet informatique par excellence.

On peut aussi utiliser une propriété plus simple de l'ADN pour programmer : la complémentarité entre séquences. L'ADN contient quatre bases, qui forment deux paires de bases *complémentaires*. Entre chacune des paires de bases complémentaires, il existe une force attractive. Cette propriété fait que deux séqueneces complémentaires (dont les bases de mêmes indices sont complémentaires) s'attirent. On obtient ainsi un effet physique —ou plutôt chimique— dépendant d'un objet informatique, un mot. C'est ce dont on a besoin pour calculer. À partir de ce principe, et de nombreux raffinements, on obtient des modèles de calcul biologique par attraction sélective entre séquences.

En particulier, on peut implémenter des remplacements de séquences, ce qui permet d'obtenir des systèmes de réécriture. L'avantage de ce type d'ordinateur biologique sur le silicium et l'électron est le haut niveau de parallélisme : il n'y a pas de contrôle de l'éxécution, donc un haut niveau de parallélisme. On peut aussi s'attaquer à des problèmes NP-complets en temps linéaire... à condition de disposer de suffisamment de ressources en réactifs. Cette démarche pour attaquer les problèmes NP-complets a été initiée par Adleman dans [3].

Paul Rothemund, dans [31] a proposé un système qui généralise l'approche par formation de polymères d'ADN, les origamis d'ADN. Dans ce système, on se donne comme objectif de créer un polymère d'ADN ayant une forme intéressante. Pour cela, on utilise une unique séquence d'ADN dont de nombreux facteurs s'attirent. Grâce à ces facteurs, la séquence va se replier et former un *origami* de la forme voulue. Cet origami est maintenu en place par des séquences *agrafes*, ce qui permet à la forme d'être stable.

1.1.2 Auto-assemblage avec de l'ADN

La technique des origamis a un point faible, c'est qu'on ne peut pas l'utiliser pour résoudre des problèmes au sens informatique : pour chaque instance, il faut revoir totalement la séquence qui sert de base à l'origami pour s'assurer que les interactions sont les bonnes. De même, dans les modèles où une séquence d'ADN correspond à un encodage d'un problème, il faut réencoder la séquence pour chaque instance du problème.

On préférerait un système dans lequel on construit des briques de bases que l'on combine pour exprimer les différentes instances d'un problème. C'est l'idée de l'auto-assemblage d'Erik Winfree. Il s'agit de construire de petites molécules d'ADN, les *tuiles*, qui portent des séquences d'ADN de sorte qu'entre ces molécules, il s'éxerce des forces sélectives : les molécules sont capables de se reconnaître et de s'agréger. De tels molécules sont produites «facilement» par les origamis de Rothemund.

Le programme de l'auto-assemblage est, à partir de cette opération d'agrégation sélective, de réaliser des calculs, de manière non-déterministe.

La donnée des tuiles correspond non plus à une instance, mais à un problème. On dispose de plusieurs manières de coder les instances : soit on utilise le non-déterminisme pour obtenir les solutions correspondant à toutes les instances, soit on encode l'instance dans la configuration de départ de l'auto-assemblage.

Dans les résultats expérimentaux de Winfree, Rothemund, Yang et Seeman [18], [33] les tuiles ont la forme de croix d'ADN avec des séquences sur chaque extrémités. Elles s'attirent quand leurs extrémités portent des

séquences semblables. Les croix sont planaires, et leur assemblage tend à former un treillis carré et planaire. Si l'on s'abstrait de la chimie, on obtient donc un modèle de carrés avec des collés colorés qui se collent les uns aux autres quand les couleurs de leurs bords correspondent. Par un contrôle fin de la chimie, on parvient à faire en sorte que deux interactions soient nécessaires pour faire coller une tuile à un motif. On obtient ainsi un modèle théorique de l'auto-assemblage.

1.2 Approche théorique

Reprenons le cheminement précédent dans l'autre sens, et partons de l'informatique pour arriver au modèle de l'auto-assemblage.

1.2.1 Automates cellulaires

En informatique théorique, parmi les objets calculants les plus simples et les plus réguliers, on trouve les automates cellulaires. Un automate cellulaire consiste en l'application uniforme d'une règle sur une grille régulière de cellules. Si l'on veut une théorie d'un phénomène calculant et simple, il est naturel de chercher à la formaliser sous forme d'automate cellulaire. Nous allons définir un modèle le plus simple possible correspondant à la notion d'«auto-assemblage».

L'auto-assemblage se fait par accrétion d'éléments sur un motif. Notre automate cellulaire aura donc deux types d'états, les états «pleins», pour les cellules qui contiennent un élément, et un état «vide» pour les cellules qui n'en contiennent pas. Des états supplémentaire de calcul représenteraient une complication du modèle. De même, nous n'avons pas besoin d'un mécanisme pour enlever des tuiles. La règle d'évolution de notre automate est donc la suivante : si une cellule et dans l'état «vide», choisir en fonction de ses voisines une tuile convenable, et passer dans l'état «plein» correspondant.

Il reste à définir les règles pour le choix d'une tuile à partir de ses voisines. L'un des modèles de règle les plus simples et les plus étudiés est celui des pavages de Wang : on représente chaque tuile par un carré avec des côtés colorés, et un motif est acceptable si les couleurs des côtés adjacents se correspondent. La règle d'évolution d'une cellule est la suivante : regarder ses voisines, ce qui donne pour chacune soit une couleur, soit «vide» ; choisir une tuile correspondant aux couleurs que l'on voit et la placer.

On voit qu'avec une telle règle, on n'a que très peu de contrôle sur l'assemblage, et que l'on a en fait un algorithme glouton de pavage. On enrichit un peu les règles en disant qu'à chaque ensemble de couleurs, on associe soit

toutes les tuiles correspondantes, soit l'état «vide», ce qui permet de retarder l'ajout d'une tuile.

Il reste deux choix à régler : la grille sous-jacente et le synchronisme. Le choix de la grille est plutôt arbitraire. Par souci de minimalité, on prend une grille carrée avec le plus petit voisinage symétrique, c'est à dire le voisinage de Moore. Pour le synchronisme, supposer une horloge globale ne correspond pas vraiment à la nature de l'auto-assemblage. On prend donc une dynamique totalement asynchrone. Avec ces choix, on retrouve le modèle défini à partir de l'expérience. L'auto-assemblage, dont nous donnerons une définition formelle au chapitre 2, est donc un modèle naturel et important.

1.3 Vision géométrique

Du point de vue calculatoire, Winfree a montré [40] que ce système d'auto-assemblage avait la puissance de calcul Turing. Cependant, à côté de cette puissance de calcul, des contraintes importantes demeurent si l'on veut réaliser des assemblages «naturels». En particulier, la géométrie du plan, dans lequel se passe l'assemblage, joue un rôle majeur, jusque-là négligé, et que cette thèse contribue à explorer.

En analysant la preuve d'universalité de Winfree, il apparaît que la simulation du calcul Turing demande de manière critique de l'espace. En effet, la méthode de simulation consiste à utiliser les tuiles pour dessiner un diagramme espace-temps du calcul à simuler. On a donc besoin pour simuler un calcul d'utiliser un rectangle $t \times s$, où t et s sont respectivement le temps et l'espace du calcul. Cette démarche a deux inconvénients : d'une part, elle impose un facteur d'échelle pour l'assemblage, ce qui réduit le côté «nano» de cette technologie, mais elle fait également complètement fi de la géométrie du plan dans lequel l'assemblage a lieu. On se réserve un rectangle dans lequel la dynamique sera simple, puis on décode le résultat de ce calcul pour poursuivre l'assemblage.

Face à ce constat, le pari de cette thèse était qu'en imaginant des mécanisme propres à l'auto-assemblage, et qui utilisent la géométrie des interactions d'auto-assemblage, on arrive à des constructions plus économiques, plus efficaces, mais aussi plus naturelles. En particulier, la *condition d'ordre* permet dans certaines conditions, de mettre une structure d'ordre partiel sur les productions, et ainsi d'analyser les constructions de manière beaucoup plus naturelle.

Nous avons à la fois montré comment la géométrie peut être une source de constructions élégantes –avec les systèmes de signaux et l'utilisation de l'ordre de construction– et une contrainte importante, dans le cadre des trans-

formations de dynamique.

Enfin, l'étude de l'auto-assemblage s'était jusque-là surtout faite dans un cadre de motifs finis, contrairement à l'étude plus classique des pavages de Wang. Nous avons étudié comment auto-assembler des pavages de tout le plan, et là encore, c'est une méthode géométrique, celle des signaux, qui nous permet d'assembler le pavage de Robinson.

Systèmes de signaux

L'outil technique le plus important que nous introduisons, et qui nous permettra d'obtenir une grande partie des résultats de la thèse est celui des système de signaux. La nature discrète des pavages auto-assemblants rend leur étude abstraite souvent pénible, car elle entraîne de nombreux cas particuliers. Nous avons donc développé un système de programmation qui permet de décrire l'auto-assemblage dans un cadre continu. Grâce à un algorithme de compilation, il est possible de repasser automatiquement de ce cadre abstrait à un jeu de tuiles discret, correspondant au modèle classique de l'auto-assemblage. La description des systèmes auto-assemblants, ainsi que la preuve de leur correction devient ainsi beaucoup plus facile. De plus, la compilation préserve les caractéristiques du système de signaux. Ce travail a été accepté pour publication dans un numéro spécial de la revue *Theoretical Computer Science* sur l'auto-assemblage [8].

Constructions optimales

Si, pour assembler une forme, les résultats de calcul Turing suffisent, pour l'assembler de manière optimale, une étude plus fine s'impose. Nous montrons comment une approche géométrique permet d'obtenir des constructions optimales en nombre de tuiles et en temps de construction. Du point de vue de la faisabilité pratique, le nombre de tuiles correspond à la difficulté pour créer les bonnes séquences d'ADN, et le temps de construction au rendement de la réaction chimique. Ce travail a été réalisé en collaboration avec Éric Rémila et Ivan Rapaport et présenté à la conférence FSTTCS [9], puis en collaboration avec Éric Rémila et Nicolas Schabanal, et présenté à DNA 14 [10].

Nous nous intéressons enfin au cas de la dimension 3, où une étude fine de l'ordre d'assemblage est nécessaire, et où les outils que nous avons introduits prennent toute leur valeur.

Homothéties et préservation de la dynamique

Nous nous penchons ensuite sur la possibilité de transformer géométriquement l'auto-assemblage. Dans le cas des homothéties, nous obtenons à la fois un théorème d'impossibilité dans le cas général, et une construction dans le cas ordonné. Le théorème d'impossibilité montre l'importance de la géométrie de l'assemblage, en particulier de la *condition d'ordre*, tandis que la construction donne un nouvel outil pour une description fonctionnelle de l'auto-assemblage. Ce travail a été présenté à LATA 2008 [7].

Nous donnons également une construction «universelle» pour certains cas, ce qui représente une première séparation de fonctions dans l'auto-assemblage, et un autre moyen d'obtenir une forme de langage de programmation pour l'auto-assemblage.

Assemblage de polygones exacts

Nous présentons un premier résultat de géométrie discrète par auto-assemblage, obtenu par un système de signaux. Nous montrons comment éviter les effets de crénelage inhérents à l'approche purement calculatoire de l'auto-assemblage. Ce travail a aussi été accepté pour publication dans *Theoretical Computer Science*, dans [8].

Pavages de tout le plan

Nous concluons la thèse par un retour aux sources des pavages, avec l'application de l'auto-assemblage pour paver le plan entier. En particulier, nous montrons comment assembler le pavage de Robinson, un exemple classique de pavage quasi-périodique. La quasi-périodicité est une condition forte de régularité, qui force une uniformité du motif *a priori* difficilement compatible avec l'auto-assemblage. En utilisant à la fois l'ordre d'assemblage et une formulation en termes de signaux, nous obtenons la première construction d'un motif quasi-périodique par auto-assemblage.

Chapitre 2

Préliminaires et état de l'art

Après avoir vu pourquoi étudier les pavages auto-assemblants, nous allons maintenant voir leur propriétés de base. Au cours de ce chapitre préliminaire, nous allons donner les définitions formelles de nos objets d'étude, puis nous expliciterons leur cadre d'étude. Nous verrons également la *condition d'ordre*, qui constitue le premier apport de cette thèse, ainsi qu'un outil indispensable à l'étude des plus réguliers des systèmes d'auto-assemblage. Il s'agit d'une extension de la condition *RC* de Winfree.

Nous passerons ensuite en revue les principaux résultats de la littérature du domaine, et verrons comment ils s'articulent avec le présent travail.

2.1 Définition des pavages auto-assemblants

Les considérations du chapitre précédent ont montré que la notion d'auto-assemblage était naturelle, et qu'en l'abordant soit par la pratique, soit par la théorie, on arrivait à des définitions proches. Nous allons maintenant donner une définition formelle, qui deviendra notre cadre d'étude et qui correspond à la convergence des deux approches du chapitre d'introduction. Cette définition est celle de Winfree[39].

Définition 1 (Système d'auto-assemblage)**.** *Un système d'auto-assemblage est un quintuplet composé de :*

- Σ *un alphabet fini, que l'on nommera ensemble des* colles*,*
- t *un entier, que l'on nommera* température*,*
- *une fonction* $f : \Sigma \to \{0, \dots, t\}$ *que l'on appellera la* force *des colles,*
- *un ensemble fini de tuiles* $T \subset \Sigma^4$*,*
- *une tuile* $g \in T$*, la* graine*.*

Dans ce document, on notera $\mathcal{T}$ les systèmes d'auto-assemblage. L'ensemble des tuiles de $\mathcal{T}$ sera alors noté T.

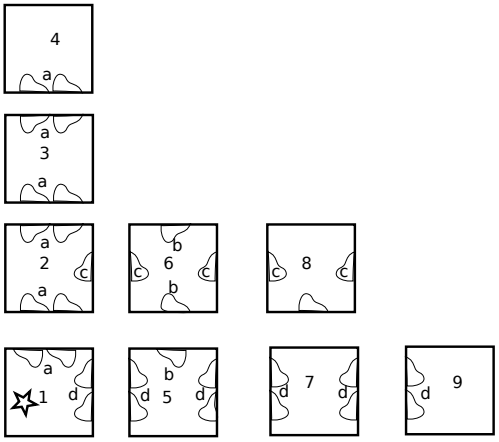


FIG. 2.1 – Un premier exemple de système auto-assemblant à température 2 : les symboles sur les côtés de chaque tuile représentent les différents de l'alphabet Σ, leur nombre de symboles sur un côté représente la valeur de la fonction de force f, la graine porte une étoile. La colle , de force 0 est représentée par une absence de symbole sur les côtés correspondants.

Pour une tuile $t = (n, s, e, w)$, avec n, s, e, w des colles de Σ, on notera $c_N(t) = n$, $c_S(t) = s$, $c_E(t) = e$ et $c_W(t) = w$.

Pour une position $z = (x, y) \in \mathbb{Z}^2$, on notera $N(z) = (x, y+1)$, $S(z) = (x, y-1)$, $E(z) = (x+1, y)$, $W(z) = (x-1, y)$. Ces quatre positions sont les *voisins* de z. On a donc $S = N^{-1}$ et $W = E^{-1}$.

Par exemple, la figure 2.1 représente un système d'auto-assemblage avec

$$\begin{aligned}
\Sigma &= \{a, b, c, d, \text{null}\} \\
T &= \{g = 1 = (a, \text{null}, d, \text{null}), 2 = (a, a, c, \text{null}), 3 = (a, a, \text{null}, \text{null}), \\
&\quad 4 = (\text{null}, a, \text{null}, \text{null}), 5 = (b, \text{null}, d, d), 6 = (b, b, c, c), \\
&\quad 7 = (\text{null}, \text{null}, d, d), 8 = (\text{null}, \text{null}, c, c), 9 = (\text{null}, \text{null}, \text{null}, d)\} \\
f &: f(a) = f(d) = 2 \\
&\quad f(b) = f(c) = 1 \\
&\quad f(\text{null}) = 0
\end{aligned}$$

Maintenant que nous avons donné la définition statique des systèmes d'auto-assemblage, nous allons voir comment ceux-ci agissent en tant que systèmes dynamiques. L'objet sur lequel un système d'auto-assemblage agit est un motif.

Définition 2 (Motif)**.** *Un* motif *sur l'alphabet A est une fonction de $\mathbb{Z}^2$ dans A. Sa* forme *est son domaine, que l'on notera* $\mathrm{Dom}(M)$*. Un motif est fini si sa forme est finie.*

Un sous-motif *d'un motif est un motif induit par une partie de sa forme. On notera $N \subset M$ pour «N est un sous-motif de M».*

Un motif d'un système d'auto-assemblage $\mathcal{T}$ est un motif sur l'alphabet T. Parmi ces motifs, nous distinguerons les configurations, qui sont les motifs pour lesquels les couleurs des tuiles adjacentes se correspondent. Les configurations sont donc les motifs qui respectent une condition à la Wang si l'on regarde les différentes colles comme les couleurs d'un jeu de tuiles de Wang. Ainsi, on parlera parfois de *couleur* d'une colle par analogie avec le cas Wang, il s'agit simplement de son identité.

Définition 3 (Configuration)**.** *Un motif M d'un système d'auto-assemblage est une* configuration *si pour tout $(x, y) \in \mathrm{Dom}(M)$ tel que $N(x, y) \in \mathrm{Dom}(M)$ (respectivement $E(x, y)$), on a : $c_N(M(x, y)) = c_S(N(M(x, y))$. (respectivement $c_N(M(x, y)) = c_S(N(M(x, y)))$*

En particulier, on appelle *configuration initiale* de $\mathcal{T}$ et l'on note $\odot_{\mathcal{T}}$ la configuration $(0, 0) \mapsto g$, c'est à dire la configuration réduite à la graine en $(0, 0)$.

Les colles que nous avons définies ont non seulement une identité, mais aussi une force. Cette force sert à créer des *liens* plus ou moins solides entre tuiles voisines. Ces liens seront le fondement du mécanisme d'auto-assemblage.

Définition 4 (Lien)**.** *Étant donné un système auto-assemblant $\mathcal{T}$ avec pour ensemble de colles Σ et pour fonction de forces $f : \Sigma \to \mathbb{N}$, on définit la fonction $l_{\mathcal{T}}$ par*

$$l_{\mathcal{T}}(c_1, c_2) = \begin{cases} 0 \text{ si } c_1 \neq c_2 \\ f(c_1) \text{ si } c_1 = c_2 \end{cases}$$

Étant donné un motif M et deux sous-motifs disjoints $M_1, M_2 \subset M$, le lien entre M_1 et M_2 est défini par

$$l(M_1, M_2) = \sum_{d \in \{N,S,E,W\}} \sum_{z \in \mathrm{Dom}(M_1), d(z) \in \mathrm{Dom}(M_2)} l_{\mathcal{T}}(d(M(z)), d^{-1}M(d(z))$$

Deux colles égales s'attirent et forment un lien d'intensité égale à la force de cette colle, tandis que deux colles différentes n'établissent pas de lien. Entre deux parties d'un motif, le lien est la somme des liens le long le leur

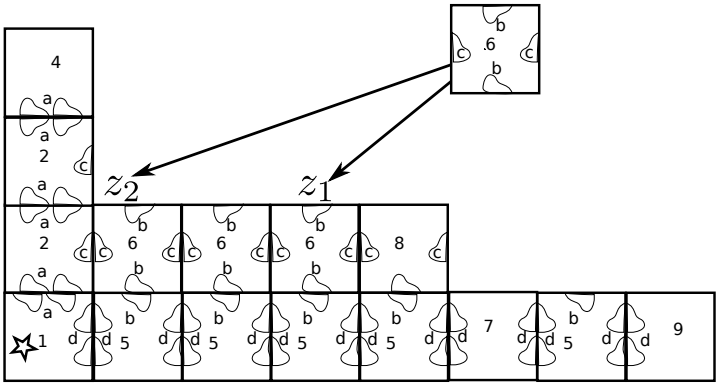


FIG. 2.2 – Dans cette position, le lien entre M et le motif réduit à 6 en z_1 est 1, et le lien entre $\{(z_2 \mapsto 6)\}$ et le motif est 2. À température 2, ajouter 6 à la position z_2 est donc une transition, mais pas l'ajout de 6 en z_1.

frontière. Dans la suite, l'un des deux motifs M_1 ou M_2 sera réduit à une tuile à une certaine position $(x, y) \in \mathbb{Z}^2$.

L'opération de base de l'auto-assemblage consiste à ajouter une tuile. Nous la définissons comme une opération entre *configurations*.

Définition 5 (Transition)**.** *Étant donné deux* configurations c *et* c' *d'un système d'auto-assemblage* $\mathcal{T}$ *de température* τ*, une tuile* t *et une position* (x, y)*, on dit qu'il y a une transition* $t@(x, y)$ *entre* c *et* c'*, et l'on note* $c \xrightarrow[\mathcal{T}]{t@(x,y)} c'$ *si*

$$\begin{cases} (x, y) \notin \mathrm{Dom}(c) \\ \mathrm{Dom}(c') = \mathrm{Dom}(c) \cup \{(x, y)\} \\ c'(x, y) = t \\ l(\{(x, y)\}, c) \geq \tau \end{cases}$$

On note $c \xrightarrow{\mathcal{T}} c'$ *pour* $\exists (t, z) c \xrightarrow[\mathcal{T}]{t@z} c'$.

On dit que la transition $T = t@(x, y)$ *est attachable à* c *si* $c \xrightarrow[\mathcal{T}]{t@z} c'$.
On note $c' = c \cup \{(x, y) \mapsto t\}$

L'idée de l'auto-assemblage est que les transitions se fassent d'elles- mêmes, les unes après les autres.

Définition 6 (Dynamique,Production)**.** *Étant donné un système d'auto-assemblage* $\mathcal{T}$ *de graine* g*, on note* $\xrightarrow{\mathcal{T}}{}^{*}$ *la clôture réflexive et transitive de* $\xrightarrow{\mathcal{T}}$*. On appelle* production *toute configuration* p *telle que* $g \xrightarrow{\mathcal{T}}{}^{*} p$*. On appelle*

dynamique *de $\mathcal{T}$ l'ordre partiel induit par $\xrightarrow{\mathcal{T}}^*$ sur l'ensemble des productions, que l'on note $P_{\mathcal{T}}$.*

Les *productions finales* sont celles à partir desquelles il n'y a plus de transition possibles, c'est à dire les éléments maximaux pour $\xrightarrow{\mathcal{T}}^*$.

2.1.1 propriétés élémentaires

Nous allons voir quelques propriétés des systèmes d'auto-assemblage, qui font partie du folklore du domaine. Elles n'étaient en général pas formalisées sous la forme présentée ci-dessous. Commençons par une observation simple sur la dynamique vue par une position donnée.

Mettons à la place d'une tuile t qui «essaie» de se poser en (x, y) pendant un assemblage. Au début, (x, y) est trop loin du motif, et il n'y a aucune colle qui permette à t de venir se fixer sur le motif. Ensuite, dans le meilleur des cas, t peut se fixer en (x, y). Enfin, au bout de quelques transition, soit une tuile est venue occuper la position (x, y), soit il y a une nouvelle tuile voisine dont la couleur est incompatible avec celle de t. Il est possible que la deuxième étape n'arrive jamais.

Proposition 1. *Soit $\mathcal{T}$ un système auto-assemblant, et une suite de productions $p_0 \xrightarrow{\mathcal{T}} p_1 \xrightarrow{\mathcal{T}} \ldots$. Pour tous $(x, y) \in \mathbb{Z}^2$, $t \in T$, il existe deux entiers i et j, éventuellement infinis tels que :*

$$\begin{aligned} \forall n < i, t@(x, y) &\textit{n'est pas attachable dans } p_n \\ \forall i \leq n < j, t@(x, y) &\textit{est attachable dans } p_n \\ \forall j \leq n, t@(x, y) &\textit{n'est pas attachable dans } p_n \end{aligned}$$

Démonstration. Notons que si $i = j$, la proposition dit que t n'est jamais attachable. De même si $i < j$.

Posons

$$i = \min\{i | l(p_i, \{(x, y) \to t\}) \geq \tau\}$$

et j le plus petit entier tel que $(x, y) \in \mathrm{Dom}(p_j)$ ou tel que $p_j \cup \{(x, y) \to t\}$ ne soit pas une configuration. C'est à dire que p_j a soit une tuile en (x, y), soit un voisin de (x, y) incompatible avec t.

Ces deux entiers satisfont la propriété. □

Proposition 2. *Soit $\mathcal{T}$ un système d'auto-assemblage. Alors $P_{\mathcal{T}}$ muni de $\xrightarrow{\mathcal{T}}^*$ est un demi-treillis inférieur.*

De plus, pour toute production p de $\mathcal{T}$, l'ensemble $\{p' | p \xrightarrow{\mathcal{T}}^* p\}$ muni de $\xrightarrow{\mathcal{T}}^*$ forme un treillis.

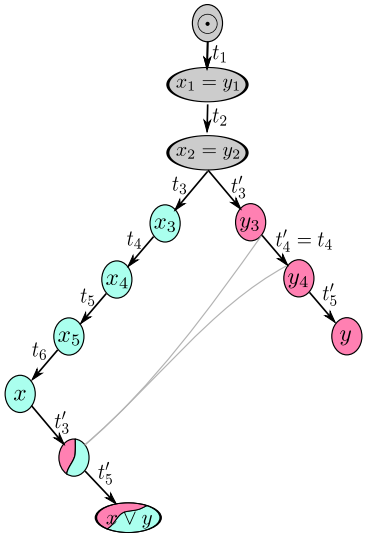


FIG. 2.3 – Transplantation de branches de dynamiques : les transitions qui ont déjà été effectuées dans x sont effacées de la dérivation qui arrive à y.

Démonstration. Commençons par construire $x \vee y$ quand $x \xrightarrow{\mathcal{T}}{}^{*} p$ et $y \xrightarrow{\mathcal{T}}{}^{*} p$: soit p une production, et x, y tels que $x \xrightarrow{\mathcal{T}}{}^{*} p$ et $y \xrightarrow{\mathcal{T}}{}^{*} p$.

Soit d la configuration $d = x \cup y$. Cette définition a un sens car x et y coïncident là où elles sont toutes deux définies. Toute configuration contenant x et y contient d. Montrons que d est elle-même une production et $x \xrightarrow{\mathcal{T}}{}^{*} d$.

Soit $\odot \xrightarrow[\tau_0]{\mathcal{T}} x_1 \xrightarrow[\tau_1]{\mathcal{T}} \dots \xrightarrow[\tau_n]{\mathcal{T}} x_{n+1} = x$ et $\odot \xrightarrow[\tau'_0]{\mathcal{T}} y_1 \xrightarrow[\tau'_1]{\mathcal{T}} \dots \xrightarrow[\tau'_m]{\mathcal{T}} y_{m+1} = y$ deux suites de transitions qui permettent de construire x et y respectivement. Considérons la suite $\tau'' = \text{nub}(\tau_0, \dots, \tau_n, \tau'_0, \dots, \tau'_m)$, où nub est la fonction qui ne garde que la première occurrence de chaque élément dans une liste. On note d_i les productions successives de τ''. Alors $\odot \xrightarrow[\tau'']{\mathcal{T}} *d$. Si ce n'est pas le cas, c'est qu'il y a un i tel que τ''_i n'est pas attachable dans d_i. Dans quelle phase est τ''_i dans d_i (au sens de la propriété 1) ? Pas dans la première puisque toutes les transitions qui permettaient d'attacher τ''_i dans la suite τ' ont été effectuées. Si τ''_i est dans la troisième phase, c'est parce qu'il y a une transition incompatible dans $\tau''_{<i}$. Ce n'est pas possible, car à la fois τ''_i et cette transition doivent être faites pour assembler p. Donc $x \xrightarrow{\mathcal{T}} *d$, et $d = x \vee y$.

Pour définir $x \wedge y$, considérons $S = \{a | a \xrightarrow{\mathcal{T}} * x \text{ et } a \xrightarrow{\mathcal{T}} *y\}$. $S \subset \{x | x < y\}$, donc il a une borne supérieure qui est un maximum. Ce maximum est $x \wedge y$. □

Proposition 3. *Soit* $\odot \xrightarrow[\tau_1]{\mathcal{T}} p_1 \xrightarrow[\tau_2]{\mathcal{T}} p_2 \dots \xrightarrow[\tau_n]{\mathcal{T}} p$ *et* $\odot \xrightarrow[\tau'_1]{\mathcal{T}} q_1 \xrightarrow[\tau'_2]{\mathcal{T}} q_2 \dots \xrightarrow[\tau'_m]{\mathcal{T}} p$ *deux suites de transitions (et de productions) qui arrivent à la même production.*

Alors $n = m$*, et il existe une permutation* σ *telle que* $\forall i, \tau_i = \tau'_{\sigma(i)}$.

Démonstration. En effet, pour chaque transition $\tau_i = t@(x, y)$, $p(t) = (x, y)$, donc il existe i' tel que $\tau'_{i'} = t@(x, y)$. Le raisonnement est symétrique, donc les deux suites sont donc bien égales à permutation près. □

2.1.2 Un premier exemple

Donnons un premier exemple de système auto-assemblant, et examinons sa dynamique. Notre système exemple est celui de la figure 2.1. Il se compose de 9 tuiles, représentées sur la figure ; la graine porte une étoile ; la température est 2.

Examinons sa dynamique : ce système assemble des tableaux. Un tableau est une suite d'entiers décroissante, que l'on représente sous forme d'un empilement de rangées dont la longueur correspond aux éléments de la suite. Il s'agit d'un escalier dont les marches sont irrégulières.

On peut montrer par récurrence que dans chaque production de ce système est un tableau. En effet, comme on le voit sur la figure 2.2, le lien de la nouvelle tuile avec la production ne pourra être 2 que sur la ligne ou la ranguée 0, ou dans les concavités de la production. Pour rendre la démonstration complète, il faut ajouter les invariants suivants, qui expriment que toutes les productions sont de la forme de la figure 2.2 :

- La rangée $y = 0$ est un préfixe d'un mot de la forme $1.5^{a_0}.7^{c_0}.\{5, 7\}^{b_0}.9$. Chacune des rangées suivantes est de la forme $\{2, 3, 4\}.6^{a_i}.8^{b_i}.\{6, 8\}^{c_i}$ avec $a_i + b_i + c_i \leq a_{i-1}$ et $a_i = b_i = c_i = 0$ si la première tuile est dans $\{3, 4\}$.
- On peut ajouter une tuile 7 ou 9 à droite d'une rangée si $a_i + b_i + c_i < a_{i-1}$, et si la rangée n'est pas réduite à une tuiles de $\{3, 4\}$. On peut ajouter une tuile de $\{5, 7, 9\}$ dans la rangée 0 à droite si la dernière tuile n'est pas 9. On peut rajouter une tuile $\{2, 3, 4\}$ au sommet de la colonne 0 si la tuile au sommet n'est pas 4.

Ainsi, la dynamique consiste à construire un L sur la colonne 0 et la rangée 0, et à le remplir dans la direction NE. Les colles de force 0 permettent d'assurer la décroissance des rangées.

2.1.3 Déterminisme local

Dans notre exemple, le système avait plusieurs productions finales. Cependant, nous avons pu exercer un contrôle sur ses productions finales, le projet

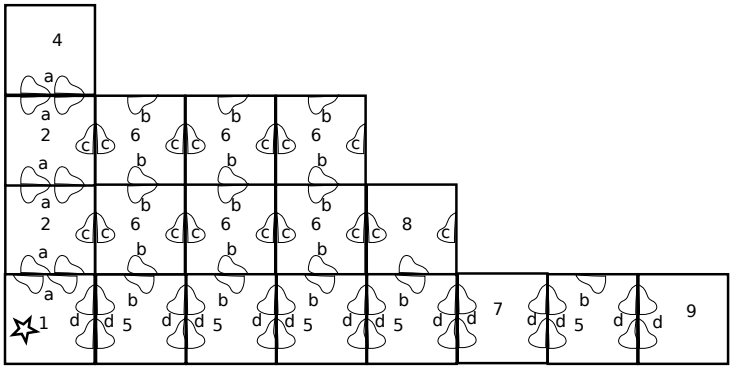


FIG. 2.4 – Une production finale du jeu de tuiles pour les tableaux

de l'auto-assemblage n'est donc pas vain ! Pour prouver ce déterminisme, à partir de la dynamique, qui reste non-déterministe, on utilise le plus souvent la condition de *déterminisme local*.

Quand on cherche à contrôler l'ensemble des productions finales, on est confronté à deux sources de non-déterminisme : le non-déterminisme local et le non-déterminisme par l'ordre des transitions. Le non-déterminisme local intervient lorsqu'à partir d'une production p, deux transitions sont possibles au même endroit. Le non-déterminisme dans l'ordre des transitions est plus subtil : il a lieu quand à partir d'une productions, deux transitions sont possibles à des endroits différents, mais qu'elles s'excluent l'une l'autre. C'est par exemple ce qui se passe dans la figure 2.5. Il est alors impossible de réconcilier les deux branches de la dynamique.

Définition 7. *Une configuration c est* localement déterministe *si pour toute position $(x, y) \in \mathbb{Z}^2$, il y a au plus une tuile t telle que $t@(x, y)$ soit attachable dans t.*

Dans notre exemple, la seule source de non-déterminisme était le choix des tuiles, et non l'ordre des transitions. Ce n'est pas toujours le cas, même avec le déterminisme local. Le déterminisme local ne suffit pas à prouver que le jeu de tuiles est déterministe quand il n'y a qu'un seul choix possible de transition pour chaque position à chaque instant : dans la situation de la figure 2.5, le nombre de tuiles sur la rangée $y = 2$ dépend de l'ordre dans lequel les tuiles ont été fixées dans la rangée $y = 1$.

Ce type de non-déterminisme, lié à l'ordre des tuiles, rend l'analyse des constructions fastidieuse. C'est pourquoi on s'intéresse à la condition d'ordre qui garantit que la seule forme de non-déterminisme est locale, comme le montre le théorème 5.

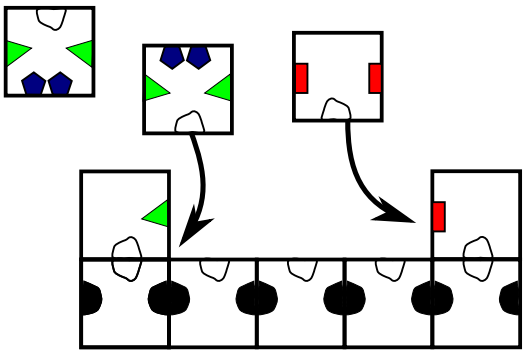


FIG. 2.5 – Le déterminisme local peut ne pas suffire pour obtenir le déterminisme

2.1.4 Dynamique à la Winfree

Dans la définition de transition se cache une hypothèse, celle que les transitions n'ont lieu qu'entre des *configurations*, c'est à dire que les motifs obtenus par auto-assemblage respectent les conditions d'adjacences type Wang.

Dans la dynamique *à la Winfree*, cette condition n'existe pas. On a donc une définition de transition entre motifs.

Définition 8 (Transition à la Winfree)**.** *Étant donné deux* motifs m *et* m' *d'un système d'auto-assemblage* $\mathcal{T}$ *de température* τ*, on dit qu'il y a une transition-Winfree* $t@(x,y)$ *entre* m *et* m'*, et l'on note* $m \xrightarrow[W,\mathcal{T}]{t@(x,y)} m'$ *si*

$$\begin{cases} (x,y) \notin \mathrm{Dom}(c) \\ \mathrm{Dom}(c') = \mathrm{Dom}(c) \cup \{(x,y)\} \\ c'(x,y) = t \\ l(\{(x,y)\}, c) \geq \tau \end{cases}$$

Il s'ensuit une définition de *production à la Winfree* et une définition de *dynamique à la Winfree*.

Avec cette hypothèse, notre dynamique diffère de ce qui se fait dans la littérature, mais elle permet de raisonner un peu plus facilement sur les motifs obtenus par auto-assemblage : si le système auto-assemblant est obtenu à partir d'un jeu de tuiles de Wang, les motifs qu'il assemble seront des configurations du jeu de tuile de Wang. Ceci évite des preuves fastidieuses et mécaniques.

On portera toutefois une attention particulière au fait que les constructions présentées dans cette thèse sont réalisable, généralement telles quelles,

avec une dynamique classique à la Winfree. Le cas échéant, nous indiquerons les petites modifications nécessaires pour faire fonctionner les constructions dans un cadre «à la Winfree»

2.2 Construction de langages de formes

Nous avons présenté les systèmes auto-assemblants comme des systèmes dynamiques discrets. Cependant, comme nous l'avons vu dans l'introduction, ils ne constituent plus un modèle algorithmique que descriptif : on ne connaît pas de phénomène naturel correspondant à ce modèle, mais on sait construire des systèmes artificiels qui y ressemblent. Puisqu'il est question de produire (ou d'assembler) des objets, il est naturel de se focaliser sur les *productions* d'un système auto-assemblant.

Dans la littérature, depuis les premiers travaux de Winfree, le problème est le suivant : on définit une forme-objectif, c'est à dire un sous-ensemble de $\mathbb{Z}^2$ que l'on veut recouvrir de tuiles. On ne s'intéresse pas (du moins dans cette définition) à quelles tuiles recouvrent le sous-ensemble.

Définition 9 (Assemblage déterministe)**.** *Un système d'auto-assemblage $\mathcal{T}$ est* déterministe *et* assemble une forme $f \subset \mathbb{Z}^2$ *si* $\xrightarrow{\mathcal{T}}{}^{*}$ *a un maximum m, dont la forme* $\mathrm{Dom}(m)$ *est f.*

Cette définition, quoiqu'assez naturelle, est un peu restrictive : puisque nous avons un modèle non-déterministe, rien n'interdit d'envisager que le même système puisse donner plusieurs productions finales. On s'intéresse alors à pouvoir assurer une propriété intéressante des productions finales. Par exemple, au lieu d'avoir un système qui assemble uniquement des carrés de taille 19830903, on peut chercher à obtenir un système qui donne des carrés dont le côté est congru à 1 modulo 3. En relâchant ainsi la définition, on se donne plus de latitude dans la conception des systèmes auto-assemblants.

Définition 10 (Assemblage non-déterministe)**.** *On dit qu'un système d'auto-assemblage $\mathcal{T}$* assemble *un ensemble $F \subset 2^{\mathbb{Z}^2}$ si toute production finale p de $\mathcal{T}$ est telle que* $\mathrm{Dom}(p) \in F$.

Dans ce travail, nous nous placerons dans le cadre de l'assemblage non-déterministe Notons que dans la définition de d'assemblage non-déterministe, il n'y a pas de condition d'arrêt de l'assemblage. La raison est simple : il faut choisir entre assurer l'arrêt et pouvoir avoir une infinité de productions finales différentes.

Proposition 4. *Soit $\mathcal{T}$ un système d'auto-assemblage ayant une infinité de productions finales, alors il existe une suite de productions $p_0 \xrightarrow{\mathcal{T}} p_1 \xrightarrow{\mathcal{T}} \dots$ infinie.*

Démonstration. Considérons l'arbre des suites de dérivations de $\mathcal{T}$. C'est un arbre à branchement fini : chaque nœud représente une production, et à partir de chaque production, le nombre de transitions possibles est fini. Les productions finales sont situées sur les feuilles, donc s'il y a une infinité de productions finales, par le lemme de König, il existe une branche infinie dans l'arbre, c'est à dire une suite de productions infinie. □

Cependant, on peut séparer les cas les plus pathologiques avec le critère d'arrêt suivant qui dit que «rien n'est jamais perdu» : quelle que soit la production du système, il est toujours possible qu'il s'arrête.

Définition 11 (Système arrêtable)**.** *On dit qu'un système auto-assemblant $\mathcal{T}$ est* arrêtable *si pour toute production p, il existe une production p' finale telle que $p \xrightarrow{\mathcal{T}}{}^{*} p'$.*

2.3 Modèle stochastique

2.3.1 Chaînes de Markov

Le modèle stochastique que nous allons utiliser est celui des chaînes de Markov en temps continu. Commençons par rappeler ce qu'est une chaîne de Markov en temps discret : certaines propriétés des chaînes de Markov en temps continu s'expriment en termes de chaînes de Markov en temps discret.

Définition 12. *Soit S un ensemble d'*états*, une distribution de probabilités $(X_n)_{n\in N}$ sur S est une* chaîne de Markov *si il existe une fonction f telle que $P(X_{n+1} = x | X_1 = x_1, \dots, X_n = x_n) = f(x_n)$.*

Si S est fini, on le note $\{x_1, \dots, x_n\}$, et la chaîne de Markov M peut être représentée par la matrice $m_{ij} = P(X_{n+1} = x_j | X_n = x_i)$.

Dans l'analyse d'une chaîne de Markov, les variables aléatoires X_i sont appelées «états successifs», et leur indice est assimilé à un temps. L'idée d'une chaîne de Markov en temps continu est d'étendre cette analogie, en se donnant des temps par des réels positifs. Il n'y a plus de probabilité de passage d'un état à un autre, mais un taux de passage d'un état à un autre.

Définition 13 (Chaîne de Markov en temps continu)**.** *Soit S un ensemble d'états, et τ une fonction de S^2 dans R. On définit une famille de variables aléatoires $X(t)$ pour $t \in \mathbb{R}^+$ par : $P(X(t+h) = j | X(t) = i) = \tau(x_i, x_j)h + o(h)$ On appelle $\tau(x_i, x_j)$ le* taux *de transition de x_i à x_j.*

Dans toute chaîne de Markov en temps continu se cache une chaîne de Markov en temps discret : soit X_i la variable aléatoire qui représente la i-ème valeur que prend $X(t)$. Si $X(t)$ est ergodique, alors (X_n) est une chaîne de Markov, et $P(X_{n+1} = x_i | X_n = x_j) = \tau(x_j, x_i) / \sum_{y \in S} \tau(x_j, y)$. C'est en particulier le cas si S est fini.

2.3.2 Modèle stochastique de l'auto-assemblage

Si l'on veut regarder un peu plus en détail comment fonctionnerait une soupe de molécules d'ADN qui serait l'implémentation d'un système auto-assemblant donné, il nous faut un modèle physique de l'auto-assemblage. Nous allons considérer le modèle suivant, dû notamment à Winfree [43] : une configuration c est plongée dans une soupe contenant toutes les tuiles de T à différentes concentrations. Les tuiles arrivent de manière aléatoire en chaque point du plan, avec un taux d'arrivée qui dépend de leur concentration. Quand une tuile t arrive en (x, y), si $t@(x, y)$ est attachable à c, alors on effectue la transition, sinon, la tuile est balayée et rien ne se passe.

Définition 14. *Étant donné un système auto-assemblant $\mathcal{T}$ et une fonction de concentration $\kappa : T \to (0, 1)$ telle que $\sum_{t \in T} \kappa(t) = 1$, on définit la chaîne de Markov en temps continu $M_{\mathcal{T}}$ comme suit. Les états de $M_{\mathcal{T}}$ sont les productions de $\mathcal{T}$, et il y a une transition de c vers c' quand $c \xrightarrow[t@(x,y)]{\mathcal{T}} c'$, avec un taux égal à $\kappa(t)$.*

L'état initial est $\odot$.

On dit qu'un système auto-assemblant muni d'une fonction de concentrations s'arrête presque sûrement si la probabilité d'arriver à une production finale est 1. Cette condition implique que le système est arrêtable. En effet, toute production a une probabilité non-nulle d'être atteinte, donc si l'une d'entre elle empêche l'arrêt, on ne peut avoir $P(\text{l'assemblage s'arrête}) = 1$.

Le principal intérêt de la présentation stochastique des pavages auto-assemblant est qu'elle est assez réaliste vis-à-vis de la chimie [39]. En particulier, elle permet de prédire le temps moyen que va nécessiter l'assemblage d'une production. Cependant, grâce à un théorème d'Adleman que nous avons généralisé 2 pour tenir compte de la *condition d'ordre*, il est possible de donner une évaluation purement combinatoire de ce temps d'assemblage.

2.4 Condition RC et condition d'ordre

Nous l'avons dit plus haut, le déterminisme local ne suffit pas à assurer l'unicité de la production finale. En effet, il est possible que le non-

déterminisme de l'ordre des transition transparaisse dans la production finale. Cet effet est assez gênant, même lorsque l'on cherche à assembler un ensemble de formes. S'il est facile de vérifier qu'un jeu de tuiles est localement déterministe, il n'est est plus difficile de vérifier si les problèmes de synchronisation se manifester dans les productions finales.

Nous allons donc nous intéresser aux systèmes dans lesquels l'ordre dans lequel les tuiles sont placées est le plus indifférent. Winfree [39] avait déjà noté que ces systèmes étaient d'une importance particulière, et en avait caractérisé certains au moyen de la condition RC. La condition d'ordre que je propose est une généralisation de cette condition RC qui dépasse certaines limitations.

Tout d'abord, la condition RC ne perment d'assembler que des productions qui sont HV-convexes, c'est à dire dont l'intersection avec toute ligne horizontale ou verticale de $\mathbb{Z}^2$ est connexe. Cette restriction apparait comme tout à fait artificielle, et n'est pas satisfaisante. En effet, la condition RC n'a pas une formulation très naturelle, et elle est spécifique à la température 2. Ce n'est en fait pas la condition RC que l'on utilise dans les preuves, mais sa conséquence, la condition d'ordre. En isolant cette condition et en la formulant explicitement, nous avons permis une étude plus générale, qui peut s'étendre à d'autres températures que la température 2, à d'autres grilles que $\mathbb{Z}^2$ (par exemple $\mathbb{Z}^3$ au chapitre 5). Cela permet également d'introduire le mécanisme des signaux dans toute sa généralité au chapitre 3.

Définition 15. *Soit $\mathcal{T}$ un système auto-assemblant, et p une production. Pour chaque suite de productions $\odot \xrightarrow{\mathcal{T}} x_1 \xrightarrow{\mathcal{T}} \dots \xrightarrow{\mathcal{T}} p$, on définit l'ordre $<_x$ sur $\mathrm{Dom}(p)$ par $z < z'$ si z et z' sont voisins et s'il existe un x_i tel que $\mathrm{Dom}(x_i)$ contient z mais pas z', c'est à dire si z est recouvert avant z'.*

On dit que p est ordonnée si $<_x$ ne dépend pas de x. On le note alors $<_p$.

On dit que $\mathcal{T}$ est ordonné si toutes ses productions le sont.

Pour les systèmes ordonnés, on peut déduire toute la dynamique de la donnée de l'ordre pour chacune de leurs productions finales. À partir de cet ordre, on peut en particulier déterminer si la construction se fait en parallèle, dans le cas d'un ordre large, ou de manière plus séquentielle, dans le cadre d'un ordre profond. Cela nous permettra, dans la section 2.5 de définir le *temps parallèle* associé à une production. Cette relation sera formalisée par le théorème 2 et est illustrée par la figure 2.7.

Avec la condition d'ordre et le déterminisme local, on obtient le déterminisme des productions finales :

Proposition 5. *Soit $\mathcal{T}$ un système auto-assemblant localement déterministe et ordonné. Si $\mathcal{T}$ a une production finale, elle est unique.*

Démonstration. Montrons que pour toute paire de productions x, y, il existe z tel que $x \xrightarrow{\mathcal{T}}{}^{*} z$ et $y \xrightarrow{\mathcal{T}}{}^{*} z$.

Considérons une paire minimale (x, y) de productions qui ont divergé, c'est à dire pour lesquelles il n'y a pas de tel z. Par minimalité, il existe t tel que $t \xrightarrow{\mathcal{T}} x$ et $t \xrightarrow{\mathcal{T}} y$. On a donc deux transitions possibles à partir de t qui sont incompatibles. Par déterminisme local, elles doivent avoir lieu dans des positions distinctes. Comme ces deux transitions sont incompatibles, elles ont lieu à des positions voisines. Par la condition d'ordre, les deux transitions sont comparables, donc seule l'une des deux peut avoir lieu. Il n'y a donc pas de paire minimale de configurations irréconciliables.

D'où le théorème. □

On aura souvent besoin d'étudier la configuration locale de l'ordre autour d'une tuile. Pour cela, on introduit la *direction* d'une tuile.

Définition 16 (Direction d'une tuile)**.** *Soit p une production ordonnée. On appelle direction de $z \in \mathrm{Dom}(p)$ l'ensemble*

$$\{d^{-1} | d(z) \text{ est un prédecesseur de } z\}$$

À température 2, Winfree [39] a donné une condition suffisante à la condition d'ordre.

Définition 17 (Condition RC (*Row-Column*))**.** *Soit $\mathcal{T}$ un système auto-assemblant à température 2. On dit que $\mathcal{T}$ satisfait la condition RC si pour chaque production de $\mathcal{T}$,*

- *pour chaque ligne au nord de la graine, il y a une unique tuile avec une colle de force 2 sur son côté sud.*
- *pour chaque ligne au sud de la graine, il y a une unique tuile avec une colle de force 2 sur son côté nord.*
- *pour chaque colonne à l'ouest de la graine, il y a une unique tuile avec une colle de force 2 sur son côté est.*
- *pour chaque colonne à l'est de la graine, il y a une unique tuile avec une colle de force 2 sur son côté ouest.*

Ces conditions permettent d'attribuer pour chaque tuile d'un motif une direction avec la sémantique suivante : si une tuile a une direction orthogonale (Nord, Sud, Est ou Ouest), elle est la seule tuile de sa colonne (ou de sa ligne) à porter une colle de force 2 dans la direction opposée. On attribue ainsi les directions dans chaque ligne (*Row*) et dans chaque colonne (*Column*), ce qui donne un ordre pour tout la production.

Ainsi, une tuile N est la seule de sa ligne a porter une colle de force 2 sur son côté Sud, ce qui signifie qu'elle a permis au pavage de s'étendre sur

une nouvelle ligne au Nord. Si elle a une direction diagonale (Nord-est, Sud-est, Sud-ouest ou Nord-ouest), alors elle a étendu le pavage dans la direction correspondante : ainsi, une tuile Nord-est s'est fixée en se collant à ses voisins Sud et Ouest.

Théorème 1 (Winfree)**.** *Si un système auto-assemblant $\mathcal{T}$ vérifie la condition RC, alors il est ordonné.*

En particulier, il est possible étant donné une production p de $\mathcal{T}$ d'associer à chaque position de Dom(p) *une direction dans l'ensemble suivant :*

$$\{N, S, E, W, NE, NW, SE, SW\}$$

Démonstration. Soit $\mathcal{T}$ un système vérifiant la condition RC. Soit p une production de $\mathcal{T}$. On donne d'abord aux position dont la tuile a une colle de force 2 une direction orthogonale (N, S, E, W). Pour chaque colonne (respectivement ligne) située dans une direction $d \in \{N, S, E, W\}$ de la source, on attribue la direction d à la tuile de la colonne (respectivement ligne) qui porte une colle de force 2 dans la direction d^{-1}. Il y a donc une position étiquetée E ou W par colonne, et une position N ou S par ligne.

Ensuite, on attribue la direction NE aux positions qui sont à l'est de la position N sauf si elles sont au sud de la position E de leur colonne. On donne aussi cette direction aux positions situées au nord de la position E de leur colonne, sauf si elles sont à l'ouest de la position N de leur colonne.

On procède de même pour les autres directions, ce qui donne le tableau suivant :

	Nord de E	Sud de E	Nord de W	Sud de W
Est de N	NE	SE*	NE	Impossible
Est de S	NE	SE	Impossible	SE
Ouest de N	NW	Impossible	NW	SW
Ouest de S	Impossible	SW	NW	SW

Le tableau est complet, donc il est bien possible d'attribuer à chaque position une direction.

Il est clair que les entrées du tableau situées sur la diagonale sont correctes.

Les cas marqués comme impossibles le sont réellement : en effet, d'après la condition RC, pour que l'assemblage soit possible, il faudrait que chacune des deux positions portant des directions orthogonales ait été remplies chacune avant l'autre, ce qui est impossible.

Pour voir que l'étiquetage est correct dans les autres cas, examinons le cas marqué *. La tuile ne peut pas avoir étendu le pavage vers le NE, car

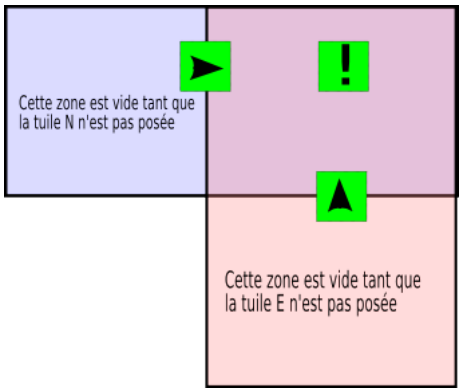


FIG. 2.6 – Placements incompatibles avec la condition RC

pour cela, il faudrait que son voisin sud ait été placé avant elle. Mais alors la tuile marquée E qui est dans sa colonne n'aurait pas été la première placée dans cette colonne, ce qui contredit la condition RC. La tuile a donc étendu le pavage au sud-est : la position a donc succédé à ses voisins N et W □

L'ordre associé à une production ordonnée est bien un ordre de dépendance au sens où une tuile ne peut être posée qu'après toutes celles qui la précèdent. On peut reformuler cette condition en termes d'idéaux.

Proposition 6. *Soit $\mathcal{T}$ un système auto-assemblant et p une production ordonnée. Pour toute production $q \xrightarrow{\mathcal{T}}{}^{*} p$, $\mathrm{Dom}(q)$ est un idéal de $<_p$.*

2.5 Le temps

Lorsque l'on implémente un système auto-assemblant *in vitro*, il est assez naturel de ce demander au bout de combien de temps on verra effectivement apparaître les configurations finales.

Définition 18. *Pour un système auto-assemblant $\mathcal{T}$ et une fonction de concentration κ, on définit le* temps stochastique *pour assembler une production finale p comme l'espérance du temps pour que $M_{\mathcal{T}}$ arrive dans l'état p sachant qu'elle atteint l'état p. On note ce temps $\tau_\kappa(p)$.*

Il existe un moyen de calculer ce temps de manière purement combinatoire, en utilisant la condition d'ordre.

Définition 19 (Temps parallèle)**.** *Soit $\mathcal{T}$ un système auto-assemblant, et p une production ordonnée. On appelle* temps parallèle *de p la longueur de la plus longue chaîne de $<_p$. On note ce temps $\pi(p)$*

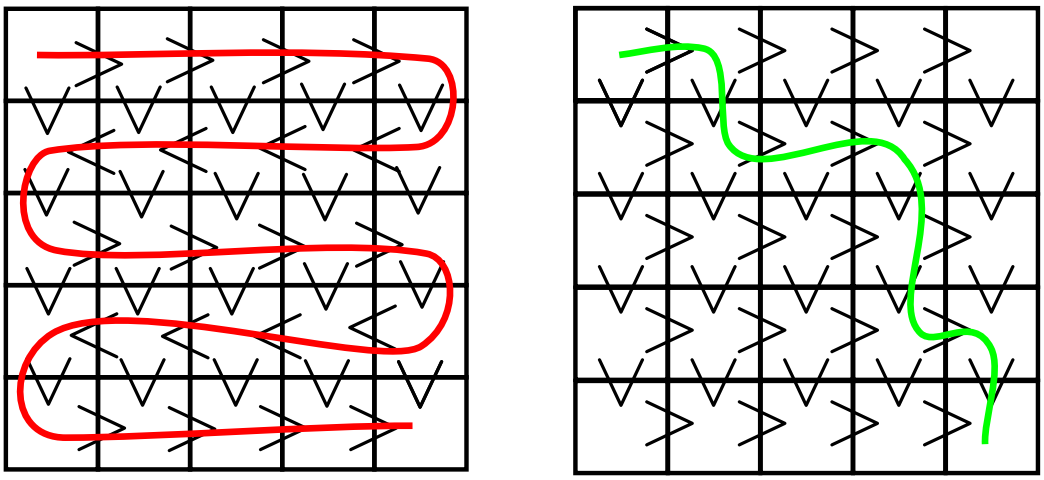


FIG. 2.7 – Un assemblage lent (à gauche), et un plus rapide (à droite). Chacune des deux productions est ordonnée, les flèches représentent l'ordre correspondant. À gauche, on ne peut ajouter qu'une tuile à la fois au plus, à droite, de nombreuses tuiles viennent se fixer simultanément. Sur chacune des productions, on a représenté une chaîne maximale.

Avec cette définition, on voit que ce qui prend longtemps à assembler, c'est bien sûr le fait qu'il y ait de nombreuses tuiles à ajouter, mais aussi le fait qu'elles dépendent beaucoup les unes des autres. Si l'on compare les deux assemblages de la figure 2.7, celui de gauche est plus lent, car à chaque instant on ne peut ajouter qu'une tuile, tandis qu'à droite, de nombreuses tuiles viennent se fixer en parallèle.

Théorème 2 (Adleman)**.** *Soit $\mathcal{T}$ un système auto-assemblant, et p une production ordonnée. Soit κ la fonction qui donne une concentration $\frac{1}{|T|}$ à chaque tuile. On a alors $\tau_\kappa(p) = \Theta(\pi(p))$.*

Ce théorème est donnée dans [4], avec un formalisme un peu différent, et un peu moins général. En effet, l'énoncé donné ici permet de traiter les cas où l'assemblage n'est pas déterministe.

Démonstration. Pour obtenir le Ω, il suffit d'observer que le long d'une chaîne de l'ordre de p, les tuiles ne peuvent être posées en parallèle. On associe à chaque position la variable aléatoire $t(x, y)$ qui correspond au temps pendant lequel la position (x, y) a été attachable. Pour toute chaîne c, le temps nécessaire pour assembler p est au moins $t(c) = \sum_{z \in c} t(z)$.

Réciproquement, dans tout tirage du processus de Markov, il existe une chaîne c de l'ordre de p tel que le temps nécessaire pour assembler p soit $t(c)$.

Comme le nombre de chaînes différentes est exponentiel, par une borne de Chernoff, on obtient le résultat. □

2.6 Simulation d'un AC

Quel est au juste la puissance de calcul de l'auto-assemblage ? Peut-on assembler des formes compliquées ? L'une des motivations pour faire du calcul par ADN était de pouvoir réaliser des calculs non-déterministe avec un haut degré de parallélisme. Pour pouvoir faire cela, il faut pouvoir calculer. Nous allons calculer en simulant un automate cellulaire unidimensionnel avec voisinage $\{0, 1\}$. Ces automates cellulaires unidirectionnels constituent un modèle de calcul Turing-complet.

Définition 20 (Automate cellulaire)**.** *Soit Q un ensemble fini d'états, V un uple fini de Z (le voisinage), δ une fonction $Q^V \mapsto Q$. Le triplet $A = (Q, V, \delta)$ est un automate cellulaire à une dimension. On appellera configurations de A les éléments de Q^Z. On appelle fonction globale de transition de A la fonction $\Delta : Q^Z \mapsto Q^Z$ définie par : $\Delta(\ldots, x_{-1}, x_0, x_1, \ldots) = (\ldots, \delta(V - 1), \delta(V), \delta(V + 1), \ldots)$, où $V + i$ représente V translaté de i.*

Définition 21. *Un automate cellulaire non-déterministe est défini de manière analogue avec une fonction $\delta : Q^V \mapsto \mathcal{P}(Q)$. La fonction de transition globale est alors définie de Q^Z dans $\mathcal{P}(Q^Z)$ par : $\Delta(c) = \prod_{x \in Z} \delta(V - x)$.*

On appelle calcul d'un automate cellulaire non-déterministe une suite de configurations c_i telle que pour tout i, $c_{i+1} \in \Delta(c_i)$. Un automate cellulaire déterministe a un seul calcul par entrée.

Définition 22 (Diagramme espace-temps)**.** *On définit le diagramme espace-temps d'un calcul ainsi : $D_C(x, n) \subset Z \times N$ est l'état de la cellule x au temps n du calcul C.*

On note $T_{(k,i)}(C)$ l'intersection de D_C avec le triangle $T_{(k,i)} = \{(x+i, n) : |x + n| \leq k\}$.

Soit A un automate cellulaire (non) déterministe de voisinage $\{0, 1\}$, il existe un système auto-assemblant $\mathcal{T}_A$ dont les motifs sont les morceaux des diagrammes espace-temps de A (des calculs de A).

Proposition 7. *Il existe une fonction $t : \mathbb{Z}^2 \to \mathbb{Z}^2$ ayant la propriété suivante :*

Pour tout automate cellulaire A de voisinage $\{0, 1\}$, il existe un système auto-assemblant $\mathcal{T}_A$ à température 2 et une fonction σ de l'ensemble T_A des tuiles de $\mathcal{T}_A$ dans l'ensemble Q_A des états de A tels que les productions finales

de $\mathcal{T}_A$ soient en bijection avec l'ensemble des $T_{(k,i)}(C)$ pour tous les calculs de A par $\phi(p) = \sigma \circ p \circ t$.

Démonstration. On considère le jeu de tuiles donné sur la figure ci-dessous (figure 2.8).

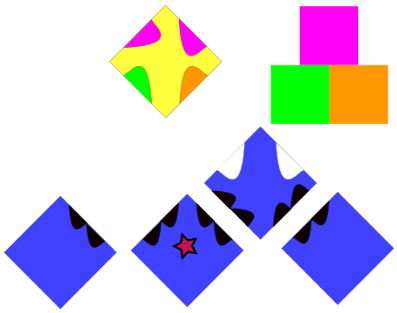


FIG. 2.8 – Simulation d'un automate cellulaire. En haut, les tuiles de règle ; en bas la ligne construisant la ligne de départ.

Chaque colle correspond à un état de l'automate cellulaire, et chaque tuile jaune à une règle de transition.

Les tuiles bleues forment une ligne horizontale dont le haut est couvert de colle blanche, qui correspond à l'état initial (la tuile en 0 a une couleur différente). Ensuite, les tuiles jaunes simulent les règles de A. Ceci définit σ. La fonction t est définie par $t(x, y) = (y - x, x)$. On peut montrer par induction sur les productions que toute production est l'image d'une partie d'un diagramme espace temps de A par ϕ.

Une production finale est un triangle, qui est l'image par t d'un $T_{(k,i)}$.

Il reste à montrer que tout $T_{(k,i)}(C)$ peut être obtenu ainsi. Pour construire $T_{(k,i)}(C)$, on peut choisir l'ordre dans lequel on pose les tuiles. On assemble d'abord la configuration de départ au moyen des tuiles d'initialisation. Ensuite, pour chaque ligne dans $T_{(k,i)}(C)$, il est possible d'ajouter les tuiles correspondant aux états successifs de l'automate.

□

Les modalités de la simulation peut éventuellement être modifiées : seules la présence des tuiles de règles a une importance, on peut changer la façon dont est donnée l'entrée, ainsi que l'interprétation de la fin du calcul (la sortie). Notons aussi que de même qu'un automate cellulaire, on peut simuler une machine de Turing.

Cette simulation nous permet d'obtenir de nombreux résultats d'indécidabilité. En particulier, les conditions de régularité de la construction, la condition d'ordre et la condition RC, sont indécidables.

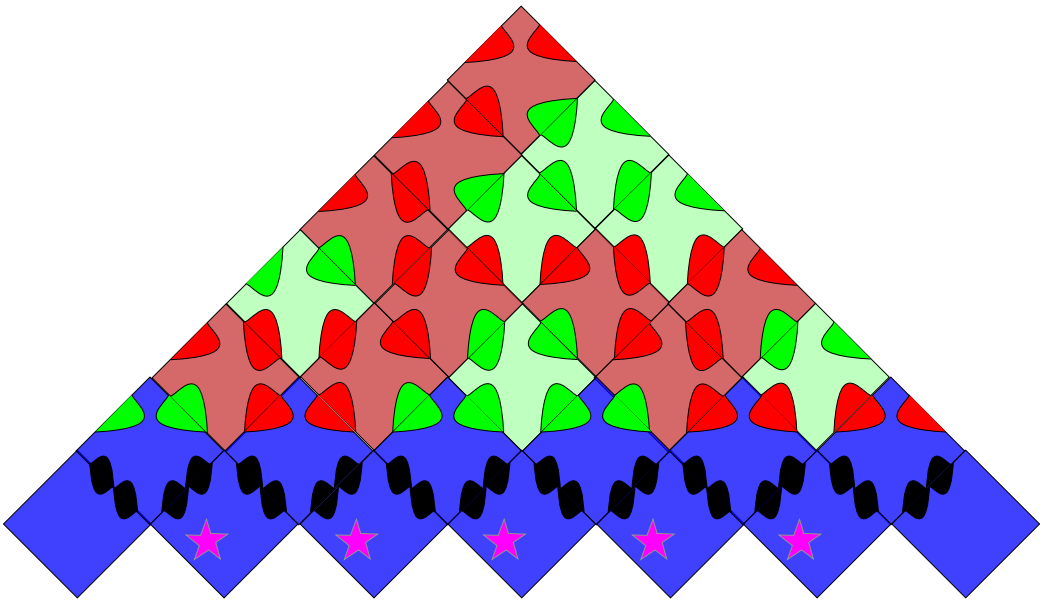


FIG. 2.9 – Une production du jeu de tuiles associé à l'automate cellulaire XOR. Les tuiles bleues représentent l'initialisation, les tuiles brique l'état 1, et les tuiles amande l'état 0. Chaque cellule est représentée par une colonne penchée vers la droite. Adapté de [29]

Proposition 8. *Étant donné un système auto-assemblant $\mathcal{T}$, il est indécidable de savoir si $\mathcal{T}$ est ordonné. Il est aussi indécidable de savoir si $\mathcal{T}$ est RC.*

Démonstration. Reprenons le jeu de tuile que nous avons utilisé pour la preuve de la proposition 7, en simulant un automate cellulaire Turing-universel. Nous encodons sa configuration initiale par un petit nombre de tuiles initiales, et ajoutons pour l'état d'arrêt une tuile dont les deux bords portent une colle «arrêt», de force 2. On se donne une tuile marqueur qui porte sur tous ces côtés la colle «arrêt». Ainsi, si l'état d'arrêt apparaît, la tuile marqueur peut remplir le plan de façon non ordonnée. Sinon, on reste dans le cadre de la simulation précédente qui est ordonnée, et aussi RC.

Il est donc indécidable de savoir si un système auto-assemblant est ordonné. □

2.6.1 Assemblage à l'échelle

Maintenant que nous avons toute la puissance du calcul Turing à notre disposition, on s'attend à des résultats du type : «tout ensemble récursivement

énumérable de formes est assemblable». Cependant, dans la simulation précédente, on voit qu'il faut de la place pour calculer. Cette place, on se la donne grâce à une homothétie : plutôt que d'assembler une forme, on assemble son image agrandie t fois, où t est le temps nécessaire pour la calculer. Chaque point est ainsi remplacé par un carré $t \times t$ dans lequel on a la place de calculer.

Théorème 3 (Assemblage de langages récursivement énumérables)**.** *Soit $\mathcal{E}$ un ensemble récursivement énumérable de formes connexes non vides. Alors il existe un système auto-assemblant $\mathcal{T}$ à température 2 et une fonction $s : \mathcal{E} \to \mathbb{N}$ telle que $\mathcal{T}$ assemble $\{s(f)f | f \in \mathcal{E}\}$.*

Démonstration. L'idée de ce théorème est de simuler la machine de Turing qui énumère $\mathcal{E}$, et de lire son résultat comme une description de chaque forme. On note e_i les éléments de e dans l'ordre où ils sont énumérés.

Plus précisément, on considère une machine de Turing qui sur l'entrée (x, y, n) décide si $(x, y) \in e_n$. Pour chaque n, on note t_n le maximum des temps de calcul de M sur (x, y, n) pour $d((x, y), e_n) \leq$. Autrement dit, t_n est le temps maximal dont M a besoin pour faire la différence entre les éléments de e_n et leurs voisins. La fonction $n \mapsto t_n$ est calculable. Au début de la construction, on choisit n, puis on calcule t_n. On construit ensuite un carré de taille $5t_n$ dont les bords portent la valeur de n, ainsi que les coordonnées des voisins. Pour chacun de ses voisins, s'il est dans e_n, il y a des colles de force 2 sur le côté du carré, sinon il n'y a que des colles de force 0. Ensuite, chaque carré $5t_n \times 5t_n$ lit ses coordonnées à partir de ses voisins, et en déduit lesquels de ses voisins sont présents en simulant M. □

On en déduit en particulier une variante du théorème de Rice pour l'auto-assemblage :

Théorème 4 (Rice)**.** *Étant donné une propriété p des ensembles de formes connexes non vides, soit p est triviale, soit il est indécidable de savoir pour un système $\mathcal{T}$ si il existe une fonction $s : \mathcal{E} \to \mathbb{N}$ et un ensemble de formes $\mathcal{E}$ ayant la propriété p tels que $\mathcal{T}$ assemble $\{s(f)f | f \in \mathcal{E}\}$.*

Le lien entre auto-assemblage et calcul Turing continue, avec un parallèle entre nombre de tuiles et complexité de Kolmogorov, dû à Adleman [4].

Théorème 5. *Soit f une forme de complexité de Kolmogorov k. Alors il existe un entier s et un système auto-assemblant avec $k/\log k$ tuiles qui assemble $\{f\}$.*

Démonstration. On réutilise la même construction que pour le théorème précédent, en observant que le nombre de tuiles pour simuler M est $k/\log k$. □

La réciproque n'est vraie que si l'on considère des formes à échelle 1.

Théorème 6. *Soit f une forme, si $\mathcal{T}$ auto-assemble f, alors f a au moins $\Theta(k/\log k)$ tuiles, où k est la complexité de Kolmogorov de f.*

Démonstration. $\mathcal{T}$ constitue un algorithme pour décrire f, donc il doit être de taille au moins $\Theta(k)$. Il a donc au plus $\Theta(k/\log k)$ tuiles. □

Ces propriétés nous permettent d'établir un parallèle fort entre d'une part le temps dans le cadre du calcul Turing et d'autre part l'échelle d'un système auto-assemblant avec un facteur d'échelle. C'est pourquoi, afin d'échapper à ce parallèle, nous nous sommes efforcés d'obtenir des résultats dans les cadres où il ne s'appliquait pas. Ainsi, nos signaux (chapitre 3) fonctionnent à échelle 1, de même que nos constructions pour les polygones pour lesquelles le pas de discrétisation est arbitrairement petit (chapitre 7), ce qui revient à travailler à échelle 1.

Dans le chapitre 5, la démarche a été duale, puisque nous avons montré comment simuler un assemblage à échelle 1 par un assemblage à échelle quelconque. Tout d'abord, cela permet de définir une notion de simulation pour l'auto-assemblage. Cette notion n'est pas triviale, puisque certaines constructions ne sont pas simulables à une échelle différente. Nous avons isolé avec la condition d'ordre (théorème 19) un cas où le modèle se comporte de manière régulière, c'est à dire qu'il est toujours possible de «perdre du temps» —de changer d'échelle. Il est même possible de le faire de manière *universelle* (théorème 20).

2.7 État de l'art de l'auto-assemblage

2.7.1 Robustesse de l'assemblage

Une des préoccupations majeures du domaine, liée aux limitations du modèle de l'auto-assemblage est la robustesse aux erreurs. En effet, loin de se comporter comme le voudrait la théorie, les modèles expérimentaux de l'auto-assemblage présentent des défauts au cours de l'assemblage. Il arrive au cours de l'assemblage qu'une tuile vienne se fixer sur le motif alors que la somme des liens n'est pas suffisante, ou qu'un lien s'établisse entre deux colles de couleurs différentes. Il peut arriver aussi qu'une tuile se détache spontanément.

Pour modéliser ces situations, on se donne un modèle stochastique comme celui de la section 2.3.2, mais avec des transitions «erronées» qui ne correspondent pas aux règles du modèle, et dont le taux est très faible. On cherche

alors à faire en sorte que la dynamique de l'assemblage reste la même en présence de ces perturbations.

Pour cela, la démarche adoptée par Winfree et Bekbolatov dans [42] consiste à découper chaque tuile en un carré 2×2 pour améliorer la redondance. Cette approche permet de bons résultats. Dans le modèle avec erreurs, on se place à une température $2 - \delta$ au lieu de 2, ce qui provoque plus d'erreurs mais permet d'accélérer la réaction d'assemblage : la constante de réaction est plus favorable. Le système développé dans ce papier permet d'améliorer la vitesse qu'il est possible d'atteindre pour un taux d'erreur donné —le ralentissement est en ϵ (le taux d'erreur) plutôt que ϵ^2. Dans [34], Reich, Sahu et Yin étendent cette technique pour ne pas avoir à multiplier la taille de l'assemblage. Ils s'intéressent en particulier à des assemblages qui sont des encodages de fonctions booléennes.

2.7.2 Cicatrisation

Une autre source d'erreurs dans l'assemblage est l'arrachage d'une partie du motif. Dans ce cadre, on considère un motif dont on enlève une partie, et l'on cherche à reconstruire cette partie manquante à partir de son bord. Par rapport à la situation avec des erreurs, la différence principale est que dans ce cas, le processus de cicatrisation a une dynamique très différente du processus de croissance originel. En particulier, le «futur» de la construction est déjà présent. Dans [41], Winfree montre comment il est possible, en utilisant encore des blocs, de rendre n'importe quelle construction déterministe robuste à l'arrachage, à condition qu'elle se déroule dans un quart de plan. Dans [36], avec Soloveichik et Cook, il montre que cette technique est compatible avec la robustesse aux erreurs.

Un assemblage dans le quart de plan part d'une configuration initiale en forme de L qui est ensuite complétée dans sa concavité, ce qui correspond à une dynamique d'assemblage relativement simple. Chaque tuile est remplacée par un bloc, dont la construction est assez complexe, ce qui permet à la fois d'assurer une bonne redondance et donc de repérer les erreurs, et d'avoir un «chemin inverse» qui permet de reconstruire les blocs quand des tuiles (ou des blocs entiers) ont été arrachés.

2.7.3 Utilisation comme modèle algorithmique

Résolution de problèmes NP-complets

L'auto-assemblage est un modèle algorithmique non-déterministe, et qui peut être implémenté de manière massivement parallèle. Si on le voit comme

un modèle de calcul biologique par ADN, la difficulté principale pour implanter l'auto-assemblage est le choix et la synthèse des séquences d'ADN qui vont servir de colles. Le principal facteur est donc le nombre de tuiles, qui en termes algorithmiques correspond à la complexité de Kolmogorov. Le parallélisme quant à lui, correspond au nombre de tuiles que l'on peut mettre dans le bac de réaction, et qui vont réagir en parallèle suivant toutes les possibilités offertes par le non-déterminisme du jeu de tuiles. Un tel modèle semble donc idéal pour attaquer des problèmes NP-complets par la force brute.

Dans [12], Yuri Brun propose un jeu de 49 tuiles pour résoudre subset-sum. La taille des motifs produits est linéaire en celle de l'instance, donc le temps d'assemblage est linéaire. On résout donc un problème NP-complet avec une bonne probabilité, on prend un bécher exponentiellement grand, et l'on attend un temps linéaire que la réaction se fasse. Ensuite, on filtre les produits obtenus, à la recherche de la tuile «succès». Si on la trouve attachée à un motif, l'instance est positive, sinon elle est négative avec une bonne probabilité. De plus, dans le cas positif, il est possible d'analyser le motif témoin du succès pour en déduire une solution de subset-sum, c'est à dire la composition de l'ensemble ayant la bonne somme.

Réciproquement, la plupart des problèmes sur l'auto-assemblage d'une forme finie sont NP-complets, en particulier celui de savoir si un jeu de tuiles données peut assembler une forme donnée, ou de savoir s'il a le nombre minimal de tuiles parmi ceux qui assemblent cette forme [3].

2.7.4 Dépassement des limites en nombre de tuiles

La limite en nombre de tuiles étant celle qui pose le plus de problèmes en pratique, certains travaux cherchent à la dépasser. Cela n'est bien sûr pas possible dans le cadre du modèle que nous avons présenté. Deux approches sont possibles pour contourner cette impossibilité : soit utiliser une dynamique probabiliste, soit utiliser des changements de température.

Systèmes probabiliste

Dans [9], nous avons introduit, avec Éric Rémila et Ivan Rapaport, l'idée d'assembler un ensemble de formes, et de regarder la probabilité de produire chacune d'entre elles avec le modèle stochastique présenté à la section 2.3.2. En ajustant les concentrations, on peut contrôler un peu la distributions des différentes formes. En particulier, ce système permet d'obtenir un carré de taille choisie avec un nombre constant de tuiles indépendant de la taille en question. Malheureusement, il n'est pas possible de contrôler la forme de la distribution (une hypergéométrique), et on choisit donc plus un ordre de

grandeur de la taille qu'une valeur précise. Nous développerons ces constructions dans la section 4.1.

Dans ce même article, Rémila et Rapaport proposent de réutiliser l'automate cellulaire *firing squad* pour assembler l'ensemble des losanges avec le même genre de distributions de probabilité. Il apparaît possible de reprendre l'idée de cette construction pour enrichir les possibilités des signaux (chapitre 3) et permettre des signaux à «rebrousse-temps».

Ce système a été amélioré par Schweller et Kao dans [23], où l'utilisation d'un compteur probabiliste leur permet d'obtenir une distribution aussi resserrée que nécessaire. Pour les carrés, il devient possible d'assembler un carré $n \times n$ avec probabilité $1 - o(n)$, en jouant sur les concentrations pour choisir la valeur de n. En utilisant ce n comme entrée d'une machine de Turing universelle, il existe un jeu de tuiles universel programmable par la température qui permet d'obtenir n'importe quelle forme à homothétie près, en jouant sur la concentration.

Variation de la température

Un des avantages de l'auto-assemblage est sa simplicité de mise en œuvre : les réactions se font dans un seul bac, que l'on ne touche pas jusqu'à l'obtention du produit final. Dans [22], les auteurs sacrifient un peu de cette simplicité en introduisant la possibilité de changer la température au cours de la réaction. Il est alors possible d'utiliser la suite des températures pour passer n'importe quel mot à l'assemblage, ce qui permet, là encore, d'obtenir un jeu de tuiles universel avec un nombre constant de tuiles.

Chapitre 3

Signaux en auto-assemblage

Décrire un système auto-assemblant devient rapidement une tâche fastidieuse, et prouver qu'il fonctionne comme on l'entend demande un travail d'énumération des détails. Une part importante des détails de la description correspondent à des mécanismes communs à plusieurs constructions, comme faire se déplacer de l'information dans une direction donnée, ou faire interagir plusieurs informations. En tant qu'informaticiens, la réaction naturelle est donc de chercher à faire énumérer les détails pour nous par une machine. On cherchera également à obtenir une ébauche de langage de programmation : des morceaux de constructions réutilisables, chacun avec une fonction précise. L'objet de ce chapitre est de présenter un tel langage de programmation, ainsi que son compilateur associé.

Les systèmes auto-assemblants vivent dans le plan $\mathbb{Z}^2$, notre langage de programmation, les *systèmes de signaux* va donc vivre dans une idéalisation de $\mathbb{Z}^2 : \mathbb{R}^2$. Il s'agira d'un langage de construction de complexe géométriques, que nous transformerons en un système auto-assemblant (à température 2). En associant à chaque tuile de ce système un morceau de figure, les productions finales de ce système auto-assemblant seront des puzzles correspondant aux figures de départ. La qualification d'«algorithme de compilation» pour la démarche de ce chapitre repose sur la définition suivante d'algorithme de compilation : étant donné deux systèmes dynamiques —ici les signaux et les tuiles, et une notion de simulation entre eux —ici la discrétisation, un compilateur est une fonction qui à une instance du premier système dynamique, associe une instance du second qui la simule. En quelque sorte, les signaux constituent un langage de haut niveau, qu'il est simple de programmer mais dont la sémantique n'est pas directement implémentée par la machine, et les tuiles un langage de bas niveau, qui correspond vraiment à la dynamique de la machine.

Le mécanisme de signaux que nous proposons repose sur la planarité des

constructions, ainsi que sur l'orthogonalité de la grille $\mathbb{Z}^2$ pour permettre de décrire de manière compacte l'ordre sous-jacent aux productions, et en déduire un jeu de tuiles. Cette construction est donc intrinsèquement liée à la grille $\mathbb{Z}^2$. La propriété de planarité étant cruciale, s'il est sans doute possible d'étendre la technique à d'autres graphes planaires sous-jacents que $\mathbb{Z}^2$, son extension aux dimensions supérieures semble difficilement envisageable. Un système de programmation en dimension supérieure travaillera probablement plus directement sur l'ordre, comme proposé au chapitre 5.

Le formalisme que nous donnons est inspiré par celui proposé par Durand-Lose [17] dans le cadre des automates cellulaires. La principale différence est l'interprétation du temps : chez Durand-Lose, les objets géométriques sont des diagrammes espace-temps, où le temps est une dimension à part entière. Dans le cadre de l'auto-assemblage, le temps et l'espace ne sont pas aussi indépendants, et il faut en rendre compte, d'où notre modèle qui s'écarte sensiblement de celui de Durand-Lose, même s'il s'inspire de son formalisme.

Pour programmer un système d'auto-assemblage avec des signaux, on part d'un système de signaux, objet que nous allons décrire. Cet objet a une dynamique, qui s'exprime sous forme de *dessins* dans $\mathbb{R}^2$. Nous donnons un algorithme, de *compilation*, qui associe à chaque système de signaux un système auto-assemblant. Le système auto-assemblant assemble des productions qui imitent les dessins du système de signaux. Cette compilation ne réussit en fait que si le système de signaux est bien formé, ou *temps cohérent*. La simulation est en fait plus précise, puisque la dynamique du système auto-assemblant est en fait «la même» que celle du système de signaux.

3.1 Systèmes de signaux

Les systèmes de signaux forment un modèle abstrait de dérivation de figures géométriques. Une figure est donnée par un *complexe* bidimensionnel, c'est à dire un ensemble de points, de segments et de surfaces du plan tels que les bords de chacune des surfaces soient des segments du complexe, et que les extrémités des segments soient des points du complexe. Un système de signaux est un ensemble de modèles pour les éléments de ces complexes (ou *dessins*). On dit qu'un système de signaux *produit* l'ensemble des dessins compatibles avec le système de signaux.

3.1.1 Complexes du plan

Nous allons utiliser une version «de poche» des complexes topologiques afin de définir formellement nos «dessins sur le plan». La notion de base dont

nous allons partir est celle de *complexe* géométrique.

Définition 23. *Un* complexe *c du plan est un triplet* (S_c, A_c, F_c)*, où*
- $S_c \subset \mathbb{R}^2$ *est l'ensemble des sommets*
- $A_c \subset S_c^2$ *est l'ensemble des arêtes —chacune de ces arêtes est vue comme un bipoint*[1] *du plan ;*
- F_c *est l'ensemble des faces ; chaque face est un polygone.*

On pose la condition que l'ensemble des frontières des faces soit A, et que deux arêtes ne se coupent qu'en leurs extrémités.

Un complexe est donc un découpage du plan en régions (les faces) par des segments de droite (les arêtes) dont les extrémités sont les sommets du complexe. Un complexe est *fini* quand S est fini. Un complexe fini a aussi un nombre fini d'arêtes et de faces. De plus, il y a une face e telle que $\bigcup_{f \in F \setminus \{f\})} f$ soit borné. Un complexe est à *coordonnées entières* si tous ses sommets sont à coordonnées entières. Soit c un complexe du plan, et $z \in \mathbb{R}^2$, on note $c(z)$ l'élément de c dans lequel est z.

Nous aurons aussi besoin de définir des complexes *partiels*, où seule une partie du plan nous intéresse. Pour cela, on munit un complexe d'un *domaine*, et on ne regarde que les éléments qui sont inclus dans le domaine.

Définition 24. *Soit* $D \subset \mathbb{R}^2$*, deux complexes c et* c' *sont* équivalents *sur D si* $S_c \cap D = S_{c'} \cap D$*,* $A_c \cap D^2 = A_{c'} \cap D^2$ *et* $\{f \subset D | f \in F_c\} = \{f \subset D | f \in F_c\}$.

Un complexe partiel *de domaine D est une classe d'équivalence pour la relation «être équivalent sur D».*

Si c est un complexe partiel de domaine D et c' *est un complexe partiel de domaine* D'*, on dit que c est inclus dans* c' *(*$c \sqsubset c'$*) si* $D \subset D'$ *et s'ils sont équivalents sur* D'*.*

On appelle complexe étiqueté un complexe dont les éléments (sommets, arêtes, faces) sont munis d'une étiquette. L'équivalence de complexes étiquetés tient compte des étiquettes : les fonctions d'étiquetage doivent coïncider.

3.1.2 Définition des systèmes de signaux

Un *système de signaux* est une «boîte de Meccano» pour construire des complexes. Une telle boîte contient une infinité d'éléments (sommets, arêtes, faces) d'un nombre fini de modèles. Par analogie avec les travaux de J. Durand-Lose, nous appellerons les sommets *collisions*, les arêtes *signaux*, et les faces *régions*. Les modèles de collisions sont les *méta-collisions*, les modèles de signaux sont les *méta-signaux*, et les modèles de régions sont

[1] ou segment orienté

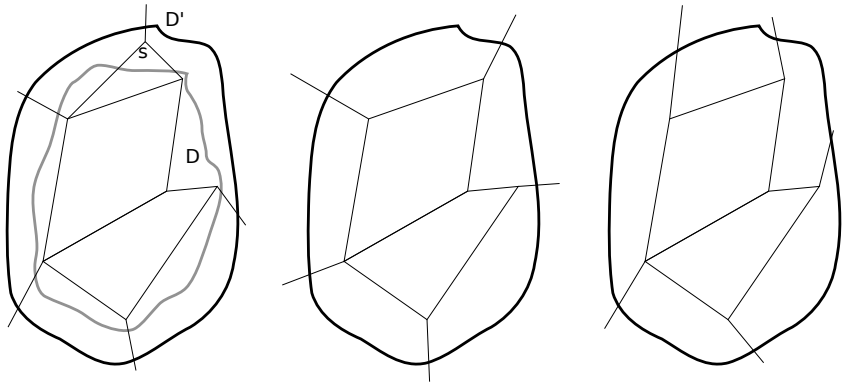


FIG. 3.1 – Le complexe de gauche, de domaine D est inclus dans les deux complexes de droite, de domaine D', qui en fait deux représentations du même complexe partiel de domaine D'. Le complexe partiel de domaine D' de gauche, n'est pas équivalent à celui de droite, car le sommet s n'existe pas dans le complexe de droite.

les *méta-régions*. Formellement, la définition d'un système de signaux est la suivante :

Définition 25. *Un système de signaux est un quintuplet* $(C, S, R, \odot, \text{Ext})$ *où :*

- R *est l'ensemble fini des* méta-régions *;*
- *Une méta-région* extérieure Ext*.*
- S *est l'ensemble des* méta-signaux*. Chaque méta-signal* s *est doté d'un support* $\overrightarrow{s}$*, d'une méta-région gauche* s̀ *et d'une méta-région droite* ś *;*
- C *est l'ensemble des* méta-collisions*. Chaque collisions* c *est définie par deux ensembles de méta-signaux, entrants* (c^-) *et sortants* (c^+) *;*
- *Une méta-collision* source $\odot \in C$ *telle que* $\odot^- = \emptyset$ *;*

À partir de ces éléments, nous allons construire des *dessins*. Ces dessins sont des complexes étiquetés par des éléments d'un système de signaux, respectant certaines contraintes. Comme nous aurons besoin de parler de dessin qui ne sont pas finis, nous introduisons les *dessins partiels* qui sont des complexes partiels, où les contraintes ne sont satisfaites que sur le domaine du dessin.

Définition 26. *Un* dessin partiel d *de domaine* $D \subset \mathbb{R}^2$ *d'un système de signaux* $\mathcal{S}$ *est un* 2*-complexe partiel de domaine* D *étiqueté tel que :*

- *l'étiquette de chaque point de* d *est une méta-collision, l'étiquette de chaque vecteur est un méta-signal, et l'étiquette de chaque surface est une méta-région,*

- *Soit s une arête de d, avec $s = b - a$, avec $a \in D$ ou $b \in D$. Notons l'étiquette de s par σ, l'étiquette de a par α, celle de b par β. On a alors : $\sigma \in \beta^+$, $\sigma \in \alpha^-$ et $b - a = \overrightarrow{\sigma}$.*
- *Si s sépare deux régions $\acute{r}$ à droite et $\grave{r}$ à gauche, alors les étiquettes de $\acute{r}$ et $\grave{r}$ sont respectivement $\acute{\sigma}$ et $\grave{\sigma}$.*

Notez que les contraintes liées au système de signaux ne sont satisfaites que sur le domaine D, c'est à dire pour les éléments qui intersectent d. Dans la suite, les domaines des dessins partiels seront des ensembles très simples, soit des boules centrés sur l'origine, soit des unions de carrés unités centrés sur des coordonnées entières.

Un dessin partiel est *enraciné* s'il contient exactement une occurrence de $\odot$, en $(0, 0)$, et si pour toute collision c de son domaine D, tous ses signaux entrants de c sont aussi contenus dans D.

Définition 27 (Dessin engendré par un système de signaux)**.** *Soit $\mathcal{S}$ un système de signaux. Le* langage *engendré par $\mathcal{S}$ est l'ensemble $L(\mathcal{S})$ des dessins enracinés finis de domaine $\mathbb{R}^2$.*

Parmi les dessins engendrés par un système de signaux, tous ne sont pas finis. Si l'on veut obtenir des dessins finis, il faut imposer une condition de terminaison. Un système de signaux est *terminant* si tout dessin partiel d de domaine $\mathcal{B}(0, n)^2$ enraciné est inclus dans un dessin enraciné fini de domaine $\mathbb{R}^2$.

Théorème 7. *Si $\mathcal{S}$ est terminant, et si tous ses dessins de domaine $\mathbb{R}^2$ sont à coordonnées entières, alors $L(\mathcal{S})$ est récursivement énumérable.*

Démonstration. Nous allons montrer le que pour tout n, l'ensemble D_n des dessins partiels enracinés de domaine $\mathcal{B}(0, n)$ est fini et qu'il existe un algorithme énumérant D_n.

Comme les éléments de $L(\mathcal{S})$ sont finis, un dessin de domaine $\mathbb{R}^2$ est juste un dessin de domaine $\mathcal{B}(0, n)$ où la seule région qui sorte de $\mathcal{B}(0, n)$ est Ext (avec n suffisamment grand). Ainsi, il suffit d'énumérer les D_n pour énumérer $L(\mathcal{S})$.

Comme $\mathcal{S}$ est terminant et que tout ses dessins enracinés de domaine $\mathbb{R}^2$ sont à coordonnées entières, pour tout dessin enraciné de domaine D, tous les points dans D sont à coordonnées entières. Donc si l'on ne considère que la partie qui est dans le domaine de chaque dessin partiel, il n'y a qu'un nombre fini de dessins partiels de domaine $\mathcal{B}(0, n)$.

[2] la boule fermée de rayon n et de centre l'origine

Pour déterminer D_{n+1} à partir de D_n, nous allons utiliser le fait que les dessins sont à coordonnées entières. Soit k la longueur maximale d'un méta-signal dans $\mathcal{S}$. On énumère toutes les attributions possibles d'une collision à un point de $\mathcal{B}(0, n+k+1) \setminus \mathcal{B}(0, n)$. On ajoute ensuite les signaux impliqués par ces collisions, puis les régions adjacentes à ces signaux. Si l'on obtient un complexe valable, alors c'est un dessin de domaine $\mathcal{B}(0, n+1)$, sinon on rejette cette affectation. Il est aisé de vérifier que tout dessin de D_{n+1} s'obtient ainsi à partir d'un dessin de D_n.

On obtient ainsi un algorithme d'énumération de $L(\mathcal{S})$. □

Notons que l'on peut toujours prolonger le domaine d'un dessin partiel jusqu'aux extrémités des signaux sortants.

Lemme 1. *Soit d un dessin de domaine D qui se complète en un dessin de domaine $\mathbb{R}^2$, et soit S l'union des signaux de d qui intersectent à la fois D et $\mathbb{R}^2 \setminus D$. Alors d avec le domaine $D \cup S$ est encore un dessin partiel qui se complète en un dessin de domaine $\mathbb{R}^2$.*

3.1.3 Exemple et conventions graphiques

Nous allons utiliser comme exemple de système de signaux un système relativement simple, représenté sur la figure 3.2.

Ce système de signaux assemble un langage de «rubans repliés», comme on peut voir sur la figure 3.3. Il s'agit d'une bande de largeur constante pliée à angle droit. Dans cet exemple, nous nous intéressons à la forme des dessins de $\mathcal{S}$ plutôt qu'à leur contenu. Étant donné un complexe c de domaine $\mathbb{R}^2$, on notera $\phi(c)$ le complémentaire de la région Ext dans c.

Lemme 2. *Soit $\mathcal{S}$ le système de signaux de la figure 3.2. Alors $\phi(L(\mathcal{S}))$ est l'ensemble des*

$$(0, l) \times (0, h) \cup T((0, h)(l, h + L)(l, h)) \cup (l, l + L) \times (h, h + L)$$

, pour $l, h, L \in \mathbb{Z}^2$. $T(a, b, c)$ étant le triangle de sommets a, b, c.

3.1.4 Vision grammaticale

L'algorithme donné dans la preuve précédente est un peu lourd. Il est plus élégant de le voir comme une grammaire de figures géométriques. Le fonctionnement de l'algorithme est plus clair, mais il est plus difficile de voir pourquoi tous les dessins sont bien des productions de cette grammaire. Une production de cette grammaire est un ensemble de collisions et de signaux

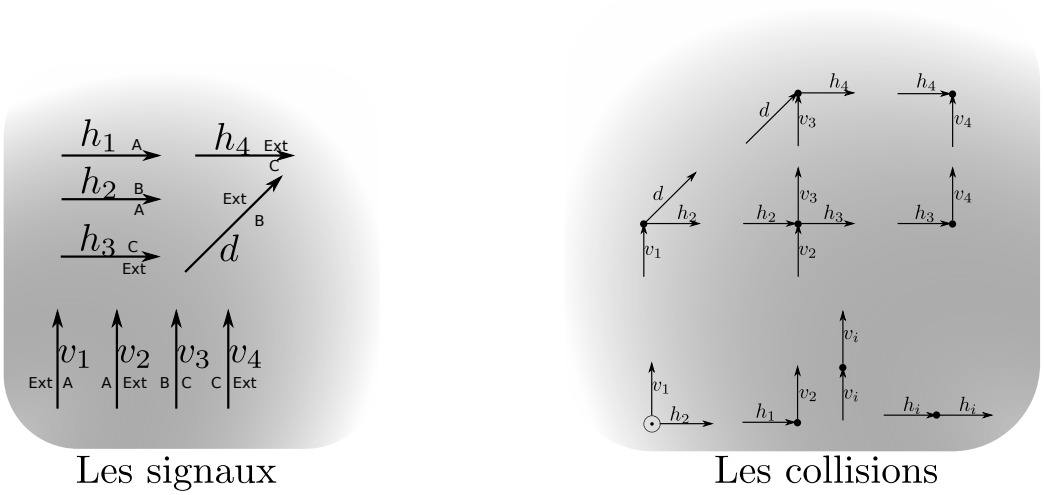


FIG. 3.2 – Un système de signaux avec 9 méta-signaux, 15 méta-collisions et 4 méta-régions. Les méta-régions sont représentées par des lettres. Un méta-signal s est représenté par le vecteur $\vec{s}$, avec à gauche la lettre correspondant à $\grave{s}$, et à droite la lettre correspondant à $\acute{s}$. Chaque méta-signal est identifié par une étiquette. Une méta-collision est représentée par un point vers lequel concourent les signaux de c^- et d'où partent les signaux de c^+.

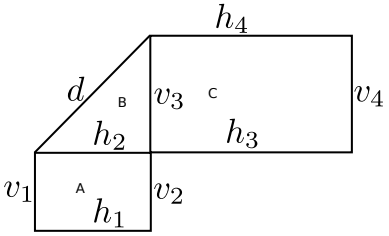


FIG. 3.3 – Un dessin de domaine $\mathbb{R}^2$ du système de signaux de la figure 3.2.

de $\mathcal{S}$. Si elle est finale pour les règles de la grammaire, alors il est possible d'attribuer des régions de façon à obtenir un dessin de $\mathcal{S}$. Réciproquement, tout dessin fini de domaine $\mathbb{R}^2$ s'obtient comme production finale de cette grammaire.

Définition 28. *On appelle esquisse d'un système de signaux $\mathcal{S}$ un ensemble de points et de bipoints étiquetés par les éléments de $\mathcal{S} \times \{\text{actif}, \text{inactif}\}$.*

Les éléments de la grammaire (signaux et collisions) peuvent être actifs ou inactifs. L'idée est qu'un signal actif n'a pas son extrémité, et une collision active n'a pas ces signaux sortants.

On définit la relation $\to_{\mathcal{S}}$ comme suit : $e \to_{\mathcal{S}} e'$ si e' est obtenue à partir de e par une des règles suivantes.

- remplacer une collision active par sa version inactive en ajoutant ses signaux sortants,
- si z est l'extrémité commune des signaux actifs $s_1, \ldots, s_n$ et s'il existe une collision telle que $c^- = \{s_1, \ldots, s_n\}$, remplacer chacun des s_i par sa version inactive, et ajouter la collision c en z.

On note $\to_{\mathcal{S}}^*$ la clôture transitive de $\to_{\mathcal{S}}$.

Une esquisse est une *impasse* si elle est finale et si elle a des éléments actifs.

Proposition 9 (Correction de la grammaire.)**.** *Soit e une esquisse maximale pour $\to_{\mathcal{S}}^*$ telle que $\{(0,0) \xrightarrow{\mathcal{T}} \odot\} \to_{\mathcal{S}}^* e$ et qui n'est pas une impasse. Alors il existe un dessin enraciné d de $\mathcal{S}$ de domaine $\mathbb{R}^2$ dont l'ensemble des collisions est l'ensemble des points de e, et l'ensemble des signaux est l'ensemble des bipoints de e. Réciproquement, tout dessin enraciné de $\mathcal{S}$ s'obtient ainsi à partir d'un élément maximal de $\to_{\mathcal{S}}$.*

Démonstration. Dans le sens direct, il suffit de constater que chaque production non-finale de la grammaire peut être transformée en un dessin partiel dont le domaine est la réunion des régions entourées par des signaux et des collisions inactives. De plus, si une production est finale pour la grammaire, alors sa face externe peut être étiquetée par Ext, et on obtient ainsi un dessin de domaine $\mathbb{R}^2$.

Réciproquement, si d est un dessin fini de domaine $\mathbb{R}^2$, il existe une dérivation de la grammaire qui mène à d. Pour l'obtenir, il suffit de trier les éléments de d (signaux et collisions) par ordre de dépendance : une collision dépend de ses signaux d'entrée, et un signal dépend de son origine. On linéarise cet ordre, et on obtient une suite de dérivations de la grammaire qui arrive à d comme production finale. □

3.1.5 Indécidabilité

Avec ces systèmes de signaux, qui sont une généralisation de ceux de Durand-Lose [17], il est très facile de simuler un calcul, par exemple celui d'un automate cellulaire. Il suffit de dessiner les diagrammes espace-temps de l'automate cellulaire avec le système de signaux.

Théorème 8 (Durand-Lose)**.** *Soit $\mathcal{A}$ un automate cellulaire unidimensionnel de voisinage $\{-1, 0\}$, il existe un système de signaux A avec qui simule $\mathcal{A}$.*

Il existe un sous-ensemble E des méta-collisions de A, et une fonction $\bar{\cdot}$ de E vers les états de $\mathcal{A}$; pour un dessin d, $\bar{d}$ est obtenu en prenant l'image des collisions dans E. De plus, pour chaque état q de $\mathcal{A}$, il y a une collision $\dot{q}$ telle que $\bar{\dot{q}} = q$.

Alors pour toute configuration $w \in \mathcal{A}$, il existe un unique dessin d_w de A contenant $\dot{w_i}$ en $(0, i)$. d_w est alors le diagramme espace temps de $\mathcal{A}$ sur l'entrée w : $d_w(\bar{x}, t) = \mathcal{A}^t(w)_x$, avec $\mathcal{A}^t(w)_x$ l'état de la cellule x au temps t dans le calcul de $\mathcal{A}$ sur w.

Démonstration. L'idée est d'engendrer des dessins qui soient des extraits de diagrammes espace-temps de $\mathcal{A}$ de toutes tailles.

On se donne deux régions, une pour l'intérieur Int, une pour l'extérieur Ext.

Pour cela, on prend se donne trois méta-signaux cadre b, g, d de vecteurs respectifs $(1, 0), (1, 1), (O, 1)$. Ces trois méta-signaux serviront à délimiter une zone de calcul triangulaire. À l'intérieur, on se donne deux signaux q^+, q^0 pour chaque état q.

On se donne une collision pour chaque règle de transition. Soit δ la fonction de transition locale de $\mathcal{A}$. Pour chaque paire d'états q_{-1}, q_0, si $\delta(q_{-1}, q_0) = q'$, alors il y a une collision c dont les signaux d'entrée sont $c^- = \{q^+_{-1}, q^0_0\}$ et dont les signaux de sortie sont $c^+ = \{q'^0, q'^+\}$. On pose $\bar{c} = q'$.

On se donne des collisions de bord : il y a une méta-collision «finale» de signaux entrants g, d et sans signaux sortants. Pour tout signal q'^-, il y a une collision de signaux d'entrée q'^-, g et de signal de sortie g. Ces collisions ne sont pas dans E.

Enfin, l'initialisation, $\odot$ a pour signaux sortants $\odot^+ = \{g, b\}$. Pour chaque état q de $\mathcal{A}$, il y a une collision c avec $c^- = \{b\}$, et $c^+ = \{b, q^-, q^0, q^+\}$. Pour ces collisions, $\bar{c} = q$. Il y a enfin une collision terminant la configuration initiale, avec $c^- = \{b\}$ et $c^+ = \{d\}$, qui n'est pas dans E.

Les dessins assemblés par A sont des triangles contenant des diagrammes espace-temps de $\mathcal{A}$, ce que l'on vérifie aisément. □

De cette construction, on déduit que la plupart des problèmes «intéressants» sur les systèmes de signaux sont indécidables, que ce soit l'occurrence

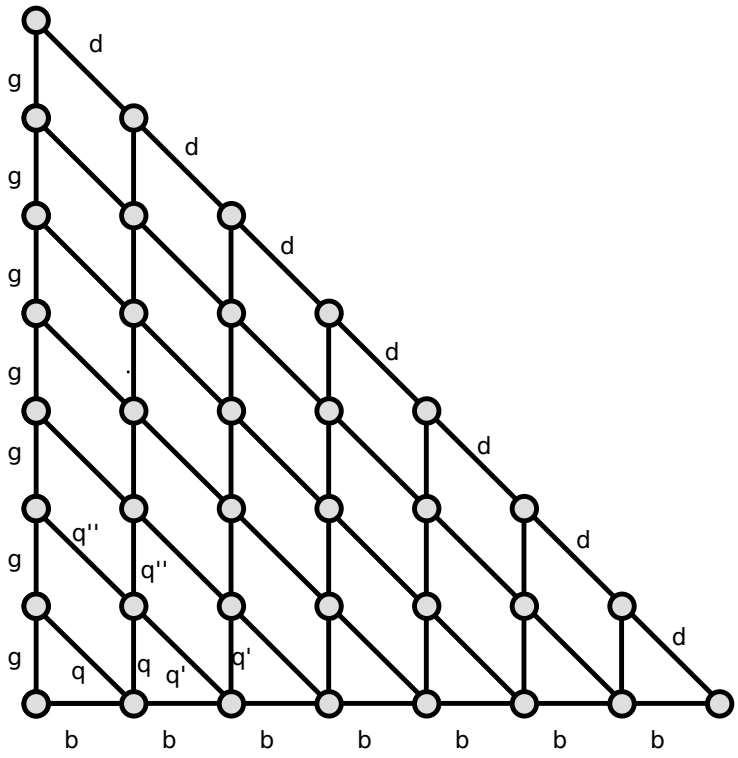


FIG. 3.4 – Un dessin du système de signaux A, avec $q'' = \delta(q, q')$

d'un signal, le fait de couvrir un point, l'occurrence d'une collision, etc.

3.2 La cohérence temporelle

Afin de pouvoir simuler un système de signaux par un système auto-assemblant, il faut que ses dessins puissent être dessinés d'une manière relativement simple, correspondant à la dynamique de l'auto-assemblage. Dans la vision grammaticale, il faut que les dérivations se fassent dans un ordre «raisonnable». Dans la vision purement abstraite des dessins, il faut que le parcours des signaux en suivant leurs orientations ne crée pas de «spirales».

Les conditions que nous avons donnés pour construire une grammaire ne sont pas tout à fait suffisantes pour cela. En effet, la position des collisions et des signaux par rapport à leur dépendances mutuelles peut poser des problèmes pour l'auto-assemblage. Pour éviter ces problèmes, nous allons introduire une condition analogue à la condition d'ordre, la condition de *cohérence temporelle*.

Dans les pavages auto-assemblants, on procède uniquement par ajout de tuiles, et jamais par destruction. Dans les systèmes de signaux, on observe également que l'on n'efface jamais de signaux : les grammaires qui implémentent les systèmes de signaux se contentent d'ajouter des nouveaux signaux au niveau des collisions. Dans les systèmes d'auto-assemblage ordonnés, de plus, il est possible d'associer à chaque partie de la figure une «direction locale du flot du temps». C'est cette condition que nous allons traduire par la cohérence temporelle, et qui nous permettra de simuler les systèmes de signaux par l'auto-assemblage.

En effet, contrairement à ce qui se passe dans les automates cellulaires, dans les systèmes d'auto-assemblage, le médium de propagation de l'information est construit par le système lui-même. Cela implique certaines contraintes, et peut rendre plus difficile la synchronisation du calcul. Prenons la situation de la figure 3.5. Dans cette figure, nous avons représenté une dérivation d'un système de signaux, avec deux signaux qui s'intersectent. L'un vient du bas, l'autre de la gauche. Chacun de ces signaux peut se répéter s'il est seul, mais les deux signaux s'annihilent s'ils s'intersectent.

L'idée pour traduire cette situation en auto-assemblage, est d'avoir une ligne de tuile pour chacun des signaux, une verticale et une horizontale. Puisque le système doit croître en créant son propre médium sous-jacent, quels signaux sont chargés de le créer ? En d'autres termes, et puisque nous allons créer des systèmes de signaux à température 2, où seront situées les colles de force 2.

Pour répondre à cette question, nous introduisons une fonction de force,

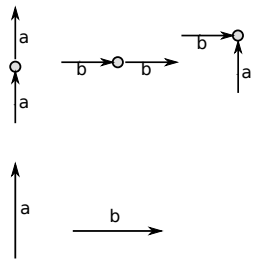


FIG. 3.5 – Un extrait d'un système de signaux avec une intersection de signaux. À chaque collision, chacun des signaux se répète s'il est seul. Les deux signaux s'annihilent lorsqu'ils s'intersectent.

qui va distinguer les signaux qui créent leur médium de propagation, soit dans la direction verticale, soit dans la direction horizontale.

Définition 29 (Système de signaux avec force)**.** *Un système de signaux avec force est un système de signaux muni d'une fonction de force s de l'ensemble de ses méta-signaux dans* $\{H, V, O, H^-, V^-\}$.

À chaque méta-signal est associé une force. La sémantique de ces forces est la suivante : les signaux V sont capables de se propager seuls dans la direction vertical, et les signaux H sont capables de se propager seuls horizontalement. Les signaux O ne peuvent se propager qu'en «prenant appui» sur d'autres signaux, un H et un V, et les signaux H^- et V^- isolent deux parties de la construction, horizontalement ou verticalement. Nous verrons plus loin comment cela s'opère. Les signaux H^- et V^- ne seront utilisés que dans quelques constructions.

Revenons à la figure 3.5, si ces deux lignes progressent de manière autonome (l'une est H et l'autre V), rien n'assure qu'elles vont arriver en même temps au point d'intersection. Si l'une arrive avant l'autre, elle va dépasser ce point d'intersection, et former la figure 3.6. Après la figure 3.6, il n'est plus possible de poser une tuile correspondant à l'intersection des deux signaux, puisque l'endroit où devrait avoir lieu cette intersection est déjà occupé.

Nous avons donc besoin d'une condition de régularité de la construction, qui nous assure que la construction est traduisible en auto-assemblage. Comme nous l'avons vu, dans l'auto-assemblage ordonné, il est possible d'attribuer à chaque région de l'espace une «direction locale de l'écoulement du temps». Nous allons de même imposer que les que les dessins soient découpés en *bassins* dans lesquelles l'écoulement du temps est uniforme. Si

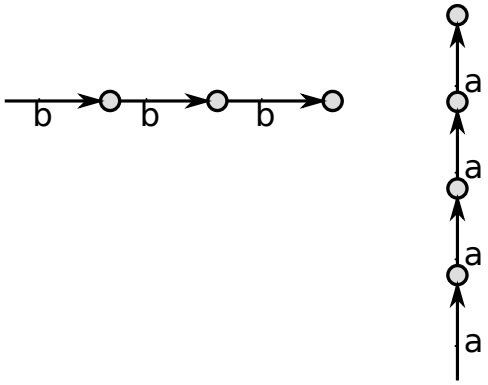


FIG. 3.6 – Synchronisation ratée : le signal vertical a dépassé le point de rencontre

ce découpage est possible, alors dans la traduction en tuiles, la direction de chaque bassin correspond à la direction des dépendances entre tuiles. C'est dans la définition de ces bassins qu'intervient la fonction de force dont nous ne nous sommes pas encore servis.

3.2.1 La condition de cohérence

Les bassins sont définis de la manière suivante. Soit $\mathcal{S}$ un système de signaux. Soit e un dessin de $\mathcal{S}$. On définit les bassins comme suit. Soit $s = ((x, y)(x', y'))$ la position d'un signal de e, de force V. On note $\epsilon^V(z)$ pour $e(z) \in \{\text{Ext}\} \cup \{s | \text{force}(s) = V^-\}$, et $\epsilon^H(z)$ pour $e(z) \in \{\text{Ext}\} \cup \{s | \text{force}(s) = H^-\}$

On définit pour ce segment deux régions du plan :

- G_s est l'ensemble des points à gauche de s, qui ne sont pas séparés de G_s par un signal V^- ni la région Ext. Formellement, $G_s = \{(a, b) | \exists a' < a, (a', b) \in s$ et $\forall a' < a'' < a, \neg\epsilon^V(a'', b)\}$
- G_s est l'ensemble des points à droite de s, qui ne sont pas séparés de G_s par un signal V^-. Formellement, $G_s = \{(a, b) | \exists a' > a, (a', b) \in s$ et $\forall a' > a'' > a, \neg\epsilon^V(a'', b)\}$.

De même, si s a une direction H, nous définissons :

- B_s est l'ensemble des points en dessous de s, qui ne sont pas séparés de G_s par un signal H^-. Formellement, $G_s = \{(a, b) | \exists b' < b, (a, b') \in s$ et $\forall b' < b'' < b, \neg\epsilon^H(a, b'')\}$
- G_s est l'ensemble des points au dessus de s, qui ne sont pas séparés de G_s par un signal H^-. Formellement, $G_s = \{(a, b) | \exists b' > b, (a, b') \in s$ et $\forall b' > b'' > b, \neg\epsilon^H(a, b'')\}$.

Un bassin est l'intersection d'une région définie par un signal H et d'une

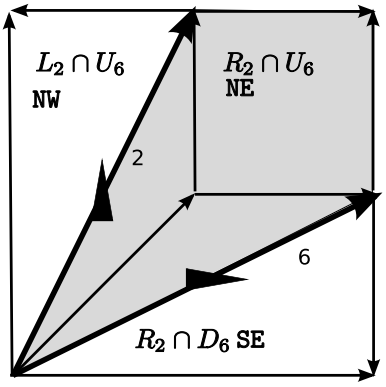


FIG. 3.7 – Un dessin temps-cohérent avec trois bassins. Le signal portant un triangle pointant vers le haut est de force V, celui avec un triangle vers la droite est de force H.

région définie par un signal V. On attribue à chaque bassin la direction prescrite par la table suivante :

$\cap$	H_s	B_s
D_s	NE	SE
G_s	NW	SW

Définition 30. *Un dessin d est* temps-cohérent *(TC) si :*

- *l'ensemble des bassins forme une partition du domaine de d, privé de sa région* Ext
- *Tout signal qui est dans un ensemble H_s a une direction dans le demi-plan nord. Tout signal qui est dans un B_s a une direction dans le demi-plan sud. Tout signal qui est dans un ensemble D_s a une direction dans le demi-plan est. Tout signal qui est dans un ensemble G_s a une direction dans le demi-plan ouest.*

Un système de signaux est temps-cohérent si tous ses dessins de domaine $\mathbb{R}^2$ sont temps-cohérents.

Dans un dessin temps-cohérent, tous les signaux ont la direction prescrite par le bassin auquel ils appartiennent.

3.2.2 Propriétés des systèmes cohérents

La cohérence en temps est une condition forte, qui correspond à la condition d'ordre pour les systèmes auto-assemblants. C'est elle qui va nous permettre de traduire certains systèmes de signaux en systèmes auto-assemblants. Elle permet donc de passer d'une détection des collisions *a priori* globale à des interactions locales.

Elle garantit la constructibilité de la figure sans que les signaux ne soient bloqués en attente les uns des autres.

La possibilité de faire des dérivations en suivant l'écoulement du temps dicté par les bassins n'est pas une obligation, puisqu'elle n'est pas inscrite dans la définition des grammaires. Cette liberté nous permet de décrire bien plus facilement la composition de deux systèmes de signaux que celle de deux système auto-assemblants. Il sera donc beaucoup plus facile de formuler des constructions complexes ou récursives en termes de signaux, puisqu'on pourra analyser des sous-systèmes séparément. En particulier, dans l'analyse des sous-systèmes, on n'aura pas à se préoccuper de leur synchronisation. Il suffira de vérifier que le résultat final respecte la condition de temps-cohérence.

La temps-cohérence est aussi une propriété restrictive. Par exemple, elle favorise les bords de figures qui suivent les directions orthogonales. Il n'y a en effet que peu de constructions de la littérature avec des bords inclinés.

Proposition 10. *Soit $\mathcal{S}$ un système de signaux temps-cohérent, et d un de ses dessins. Soit (a, z) un signal de d qui est un bord (tel que $\acute{d} = \text{Ext}$ ou $\grave{d} = \text{Ext}$). Alors soit $s(a) \in \{H, V\}$, soit z est vertical ou horizontal.*

Cette proposition signifie que si l'on veut construire des figures avec des bords diagonaux à partir de signaux, il faut les construire de l'extérieur vers l'intérieur.

Démonstration. Soit (a, z) un signal de bord, tel que $s(a) \notin \{H, V\}$. Supposons, sans perte de généralité, que $\acute{d} = \text{Ext}$, et que la direction de z soit dans l'intervalle ouvert $(0, \pi/2)$.

Comme e est temps-cohérente, z appartient à un unique bassin. Les signaux qui définissent ce bassin sont à l'intérieur de e, donc en-dessous et à droite de z. Mais alors ce bassin est d'orientation NW. Par temps-cohérence, c'est incompatible avec la direction de z.

D'où le résultat. □

Voici une autre propriété utile, qui recense les frontières possibles entre deux bassins.

Proposition 11. *Si b et b' sont deux bassins adjacents d'un dessin d temps-cohérent. Alors soit b et b' sont séparés par un signal dont la force est dans $\{H^-, V^-\}$, soit leurs directions ne sont pas opposées.*

Démonstration. Si b et b' sont adjacents et ne sont pas séparés par un signal H^- ou V^-, alors l'un des deux signaux qui les définissent leur est commun. Ils partagent donc la direction héritée de ce signal. □

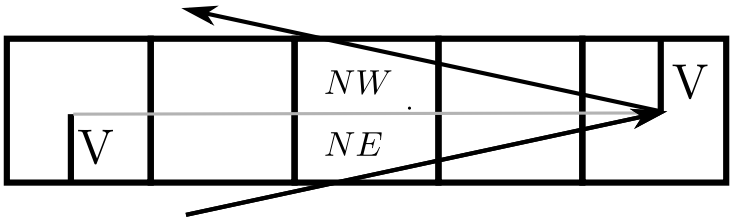


FIG. 3.8 – La discrétisation en tuiles interfère mal avec la condition de cohérence temporelle : la tuile centrale devrait être à la fois de direction NE et NW pour pouvoir transmettre les deux signaux.

3.2.3 Cohérence temporelle et discrétisation

L'objectif de la cohérence temporelle est de pouvoir discrétiser la construction pour la mimer par des tuiles. Cependant, quand on discrétise une construction, certains carrés unités vont être à cheval sur plusieurs bassins. Si ces bassins sont séparés par un signal $\{H, V, H^-, V^-\}$, cette situation ne posera pas de problème, nous la gérerons grâce à ce signal. Dans le cas où il y a une discontinuité entre deux signaux H ou V cependant, certains carrés unités vont contenir deux bassins de direction différentes. On sera amenés à choisir pour chacun de ces carrés unités une seule direction. Il faut donc que dans la direction qui sera écartée, il n'y ait pas de signaux qui se propagent avec une pente incompatible avec la direction retenue. Un système de signaux qui obéit à cette contrainte est dit *aéré*.

Définition 31. *Un système de signaux $\mathcal{S}$ est* aéré *s'il est temps-cohérent, et si dans tout dessin d de domaine $\mathbb{R}^2$, pour pour toute collision c qui est à la frontière entre deux bassins qui ne sont pas séparés par un signal de $\{H, V, H^-, V^-\}$, tous les signaux adjacents à c font avec cette frontière un angle α tel que $|\tan\alpha| > 1$.*

3.3 Compilation : des signaux vers les tuiles

Grâce à la propriété de temps-cohérence, il est possible de passer d'un système de signaux $\mathcal{S}$ à un système auto-assemblant S qui «fait la même chose». Nous définirons cette similitude. Le système auto-assemblant que nous allons utiliser est construit à partir du système de signaux en transformant tous les dessins partiels de domaine $\mathbb{U} = (-1/2, 1/2)^2$ «raisonnables» du système de signaux et en leur attribuant des colles.

À partir d'un motif sur ces tuiles, on peut constituer un «puzzle» par juxtaposition des tuiles. Les puzzles que l'on obtient à partir des productions

finales du système de signaux sont des dessins de domaine $\mathbb{R}^2$ du système de signaux de départ.

Nous allons voir dans la section 3.3.1 que S s'obtient à partir de $\mathcal{S}$ par un algorithme.

3.3.1 Algorithme de compilation

L'algorithme de «compilation» des systèmes de signaux en tuiles consiste essentiellement à énumérer les collisions du système de signaux, et à éliminer celles qui sont impossible pour des raisons locales. Ce sont les conditions de temps-cohérence et de terminaison qui vont nous assurer que cet élagage «local» est suffisant pour obtenir un jeu de tuiles correct.

ensemble de tuiles et interprétation Puisque S est construit spécialement pour obtenir des concrétisations de $\mathcal{S}$, nous allons directement prendre comme ensemble de tuiles une partie de l'ensemble des dessins de $\mathcal{S}$ de domaine $(-1/2; 1/2)^2$. Nous allons noter $T(\mathcal{S})$ de ces dessins.

L'ensemble des colles sera l'ensemble $C = U \times \{H, V\}$, où U est l'ensemble des fonctions du segment $(-1/2, 1/2)$ dans l'ensemble des étiquettes de $\mathcal{S}$. On voit ces fonctions comme les bords possibles des configurations locales. L'élément de $\{H, V\}$ indique si le bord en question est vertical ou horizontal. La fonction de force est définie ainsi :

- les segments contenant la fonction constante Ext sont de force 0
- les segments contentant un point d'un méta-signal de force H^- ou V^- sont de force 0
- les segments horizontaux (respectivement verticaux) traversés par un méta-signal de force V (respectivement H) sont de force 2
- les autres segments sont de force 1.

On note $I(\mathcal{S})$ l'ensemble des dessins de domaine $(-1/2, 1/2)^2$ à coordonnées entières. $I(\mathcal{S})$ est énumérable. On note $\partial I(\mathcal{S})$ l'ensemble des quadruplets dont les éléments sont les bords des éléments de $I(\mathcal{S})$

Notons qu'il n'y a qu'un seul dessin de domaine $(-1/2, 1/2)^2$ qui contient $\odot$ en $(0, 0)$. On le note aussi $\odot$

S est le système d'auto-assemblage $< C, 2, \partial I(\mathcal{S}), \odot >$.

3.3.2 Preuve de l'algorithme

Nous allons montrer que cet algorithme donne un système auto-assemblant qui mime effectivement $\mathcal{S}$. Commençons par définir formellement ce mimétisme.

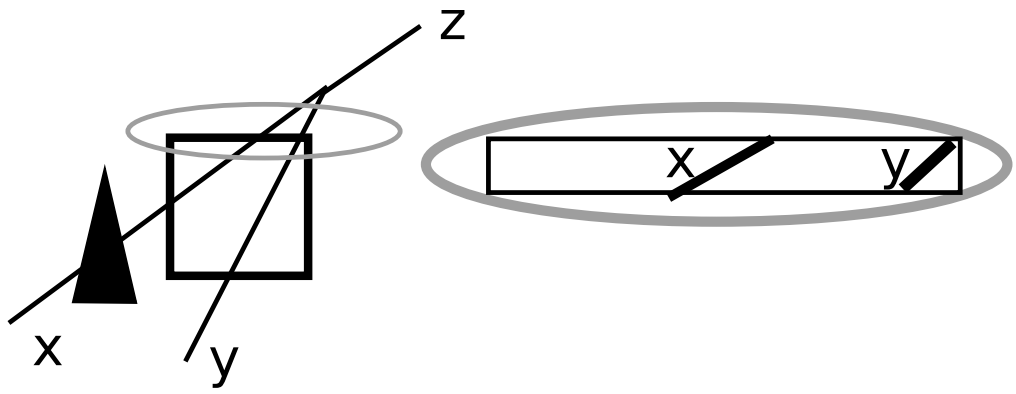


FIG. 3.9 – Une tuile obtenue à partir d'un dessin partiel de domaine $(-1/2, 1/2)^2$. À droite, la colle correspondant à son côté nord.

Définition 32. *On dit qu'un motif m de S est une concrétisation d'un dessin partiel d de domaine D si pour tout $z \in D$,*

- *soit $e(z) = [p(z)](z - [z])$*
- *soit $U(z)$ est contenu dans la région* Ext *de z, et il n'y a pas de tuile en $[z]$.*

$[z]$ est l'arrondi de z à l'entier le plus proche.

Cette définition signifie qu'un motif m est la concrétisation d'un dessin d si en juxtaposant les dessins partiels de chacune des tuiles, on obtient d, avec du vide dans la région Ext.

L'idée est à partir d'un système temps-cohérent et à coordonnées entières, d'utiliser les dessins locaux comme tuiles pour mimer l'action du système de signaux. Idéalement, la condition de temps-cohérence permet d'assurer que ces tuiles transmettent bien l'information des signaux. Si à un endroit donné, tous les signaux se propagent dans un quart de plan, et que l'on ajoute des tuiles dans cette direction, alors on transmet effectivement les signaux simplement en posant les tuiles. Malheureusement, les frontières des bassins ne sont pas sur les frontières entre les tuiles. Quand en plus, ces frontières ne correspondent pas à des signaux, comme sur la figure 3.8, les tuiles doivent transmettre des informations dans deux directions contradictoires à la fois, ce qui pose problème. On a donc besoin de la condition d'aération.

Théorème 9. *Soit $\mathcal{S}$ un système de signaux terminant, à coordonnées entières, temps-cohérent et aéré.*

Il existe un système d'auto-assemblage S et une fonction d'interprétation telle que les productions finales de S sont les exactement les concrétisations des productions finales de $\mathcal{S}$.

On note $U(x,y) = (x-1/2, x+1/2) \times (y-1/2, y+1/2)$. Si p est une production de S, on note $U(p) = \cup_{z \in \mathrm{Dom}(P)} U(z)$.

Nous allons prouver que le système d'auto-assemblage S est bien celui recherché dans le théorème 9.

Montrons que S ne peut pas introduire d'erreurs dans ses productions.

Lemme 3. *Soit p une production de S. Alors p est la concrétisation d'un dessin partiel de $\mathcal{S}$ de domaine $U(p)$.*

Démonstration. On prouve ce lemme par induction sur les productions de S.

Pour la production initiale, consistant en la source de S à l'origine, le lemme est vérifié puisque toutes les tuiles sont des dessins de domaine $(-1/2, 1/2)^2$.

Supposons que la production p soit la concrétisation d'un dessin de $\mathcal{S}$ de domaine $U(p)$, et que $p \xrightarrow[S]{t@(x,y)} p'$.

On considère le complexe obtenu sur $U(p')$, c'est bien un dessin sur $U(p')$, puisque c'est l'union d'un dessin sur $U(p)$ et d'un dessin sur $U(x,y)$ qui coïncident sur leur frontière commune.

□

En effet, il faut prouver que le jeu de tuiles S ne s'arrête pas avant d'arriver à une concrétisation d'un dessin de domaine $\mathbb{R}^2$: il faut montrer que les productions finales sont bien les concrétisations de dessins de domaine $\mathbb{R}^2$. Réciproquement, on donnera aussi la construction d'une production finale de S pour chaque dessin finale de $\mathcal{S}$. Pour cela, il faut étudier de près la dynamique de S. Nous allons montrer qu'elle est ordonnée. De là, nous déduirons d'abord qu'il n'y a jamais de blocage lié aux couleurs des tuiles, puis qu'il n'y a jamais de blocage par manque de colles.

Lemme 4. *S est ordonné. De plus, dans une production p, concrétisation d'un dessin partiel d, l'ordre est donné par les règles suivantes :*

1. *Si une position z a un signal entrant V qui traverse son côté nord (respectivement sud), alors sa direction est* `S` *(respectivement*`N`*).*
2. *Si une position z a un signal entrant H qui traverse son côté est (respectivement ouest), alors sa direction est* `W` *(respectivement*`E`*).*
3. *Pour une position z, si les signaux entrants de $U(z)$ sont O, la direction de z est celle du bassin qui contient les signaux.*
4. *Si $U(z)$ est contenue dans un bassin unique, la direction de z est celle de ce bassin.*
5. *Si $U(z)$ est partagée entre deux bassins séparés par un signal H^- dirigé vers le nord (respectivement sud), sa direction est* `ESW` *(resp.* `ENW`

6. *Si $U(z)$ est partagée entre deux bassins séparés par un signal V^- dirigé vers l'est (respectivement ouest), sa direction est* `NWS` *(resp.* `NES`
7. *Enfin, si $U(z)$ est partagé entre deux bassins qui ne sont pas séparés par un signal, alors ces deux bassins partagent une direction. S'ils partagent la direction* `N` *(respectivement* `S,E,W`*), alors z a la direction de celui qui est le plus au sud (respectivement nord, ouest, est)*

Démonstration. On montre ce lemme par induction sur les productions de S. Soit p une production de S ordonnée, dont l'ordre est donné par le lemme.

Supposons que l'on ajoute une tuile t en $z = (x, y)$ à p. Si t a un signal entrant V qui traverse son côté nord, alors en suivant la droite qui porte ce côté nord, on ne peut trouver un signal V qu'en ayant d'abord traversé un signal V^-. À cause de la règle 5, les voisins de z ne sont donc pas posés. Donc z a bien une direction S.

Si t n'a pas de signaux entrants de force H, V, H^- ou V^-, alors elle a été posée après deux voisins. Ces deux voisins sont ceux qui sont correspondent au bassin qui contient les côtés d'entrée de t, car la frontière entre les $U(z)$, qui est une ligne demi-entière ne peut être une limite entre bassins.

On vérifie de la même manière les cas faisant intervenir des signaux H^- ou V^-. □

Lemme 5. *Soit p une production de S, et $z \in \mathbb{Z}^2$ adjacent à $U(p)$. Soit $U(p) \cap U(z)$ est uniformément* Ext *dans p, soit il existe une tuile t de S telle que $p \cup \{z \mapsto t\}$ soit une configuration de S.*[3]

Démonstration. Pour prouver ce lemme, il suffit de pouvoir appliquer la terminaison à $\mathcal{S}$. En effet, comme $\mathcal{S}$ est terminant, il est possible de compléter tout dessin partiel enraciné en un dessin sur $\mathbb{R}^2$, et donc en particulier, de lui ajouter une tuile. Il faut donc prouver que toute production de p est un dessin enraciné.

On le montre par induction sur les productions. Montrons que quelle que soit la direction dans laquelle une tuile est ajoutée à une production p qui est un dessin enraciné, p reste enraciné.

Si l'on examine le lemme 4, dans tous les cas sauf le dernier, il est clair que le dessin reste enraciné : un signal entrant dans la tuile est nécessairement venu du prédécesseur de la tuile.

Dans le dernier cas, supposons que l'on ait un bassin NE au sud d'un bassin NW. La tuile est donc de direction NE. Pour arriver dans $U(z)$ à partir d'un successeur de $U(z)$, un signal doit venir de la tuile $E(z)$. Mais

[3] pas nécessairement une production. Il n'est pas —encore— exclu qu'il n'y ait pas assez de colles pour ajouter t

alors sa pente est telle que $|\tan\alpha| < 1$, ce qui contredit les hypothèses du théorème 9. □

Lemme 6. *Si p est une production finale de S, alors c'est la concrétisation d'un dessin de domaine $\mathbb{R}^2$.*

Démonstration. Soit p une production finale. Si $\partial U(p)$, la frontière de $U(p)$ ne rencontre que la région Ext, alors le dessin dont p est aussi un dessin sur $\mathbb{R}^2$, en le complétant uniformément par Ext sur $\mathbb{R}^2 \setminus U(p)$.

Supposons que tel ne soit pas le cas. Alors il existe une position (x, y) telle que $\partial U(x, y) \cap \partial U(p)$ ne soit pas uniformément Ext. Nous allons montrer qu'il existe une position adjacente à p dont le lien avec p est de force au moins 2.

Si il existe un segment $\partial U(p)$ qui contient un signal V sur un segment horizontal (ou un signal H sur un segment vertical), alors un ajout de tuile est possible le long de ce côté.

Sinon, alors $\partial U(p)$ contient deux côtés consécutifs formant une concavité et ne contenant pas uniquement Ext. En effet, par la condition de cohérence, si $U(p)$ est un rectangle, alors il y a un signal H ou V qui sort de ce rectangle. Si les bords de cette concavité ne rencontrent pas de signal H^- ni V^-, alors un ajout de tuile est possible à cet endroit, puisqu'il y a deux colles de force 2, et une tuile correspondante dans S.

Dernier cas, il y a un segment de $\partial U(p)$ qui coupe un signal H^- ou V^-. La condition de cohérence temporelle nous assure alors qu'il y a deux concavités comme au paragraphe précédent de part et d'autre (qui peuvent éventuellement se chevaucher). Si ces cavités ne se chevauchent pas, alors au moins une ne contient pas de signal V^- ni H^-, et un ajout de tuile y est possible. Dans le dernier cas, il y a aussi une force 2 au total entre p et la concavité.

Une production finale est donc nécessairement la concrétisation d'un dessin de $\mathcal{S}$ sur $\mathbb{R}^2$. □

Considérons un dessin d, et montrons comment construire une concrétisation de d avec S.

Lemme 7. *Soit d un dessin de $\mathcal{S}$ de domaine $\mathbb{R}^2$, alors il existe une production finale p qui est une concrétisation de d.*

Démonstration. Soit d' un dessin partiel de domaine maximal inclus dans d tel qu'il existe une production p qui soit une concrétisation de d'.

Si p est finale, alors $d' = d$, et le lemme est vérifié.

Sinon, un ajout de tuile est possible, disons en (x, y). Dans la démonstration du lemme 5, on peut choisir la tuile $t = U(x, y) \cap d$, puisqu'elle possède

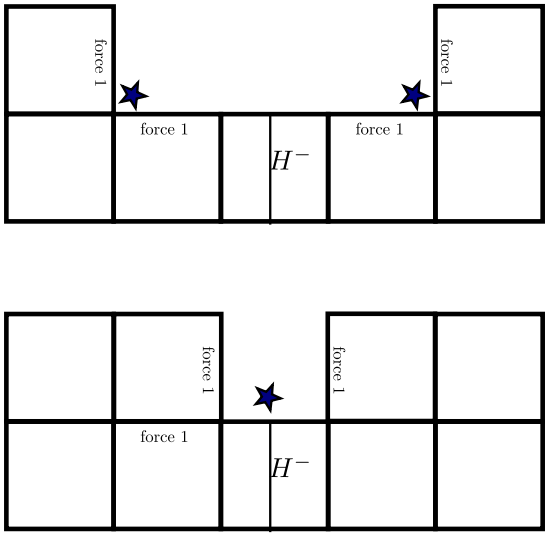


FIG. 3.10 – Le cas d'un signal H^-. Soit l'une des deux concavité a deux côtés avec des colles de force 1, soit les deux concavités sont confondues, et il y a deux colles de force 1 sur deux côtés opposés. Les étoiles représentent les positions où un ajout de tuile est possible.

les bons signaux entrants. Mais alors $p \cup \{(x, y) \mapsto t\}$ contredit la maximalité de d'. Donc le lemme est vérifié. □

Des lemmes 6 et 7, on obtient le théorème 9.

3.3.3 Performance de l'algorithme

On considère comme taille d'un système de signaux $|\mathcal{S}| = |C| + |S| + |R| + \max_{s \in S}(|\vec{s}|)$.

L'algorithme que nous avons présenté est une simple énumération de $I(\mathcal{S})$. Cet ensemble est *a priori* de taille exponentielle en la taille de $\mathcal{S}$.

Lemme 8. *Soit $\mathcal{S}$ un système de signaux à coordonnées entières. Alors $|I(\mathcal{S})| \leq |C||S|^q$, avec $q = \max_{s \in S} |\vec{s}|$.*

De plus, il existe une famille $\mathcal{S}_n$ de systèmes de signaux de tels que $|I(\mathcal{S}_n)| = \Theta(|S_n|^{q_n})$.

Esquisse de preuve. Pour montrer que $|I(\mathcal{S})| \leq |C||S|^q$, il suffit d'observer que le nombre de tuiles dans le système obtenu a au plus autant de tuiles qu'il y a de carré unités différents dans des dessins de $\mathcal{S}$. Dans un carré unités, on peut avoir des signaux de type s dans $|\vec{s}|$ positions différentes, et éventuellement une collision, la présence ou non d'une collision étant

déterminée par les signaux dans le carré unité. On a donc bien$|I(\mathcal{S})| \leq |C||S|^q$.

Il suffit de prendre une famille $\mathcal{S}_n$ de système de signaux ayant n signaux de direction $(1, n)$. Sur un carré unité donné, n de ces signaux peuvent apparaître. Il y a donc au moins n^n tuiles différentes dans $I(\mathcal{S}_n)$, et $q_n = |S_n| = n$. On peut prendre $\mathcal{S}_n$ tel que tous ces carrés unités apparaissent effectivement dans un dessin de $\mathcal{S}_n$ de domaine $\mathbb{R}^2$.

□

Pour pouvoir obtenir un algorithme avec des performances raisonnables, il faut donc imposer une condition supplémentaire, à savoir que deux signaux qui sont proches sont toujours adjacent à une collision commune.

Théorème 10. *Soit $\mathcal{S}$ un système de signaux obéissant aux conditions du théorème 9 et tel que dans tout dessin domaine $\mathbb{R}^2$, et pour tout $z \in \mathbb{Z}^2$, si deux signaux s, s' passent dans $U(z)$, alors il existe une collision à laquelle ils sont tous les deux adjacents.*

Alors il existe un système auto-assemblant S à température 2 tel que les productions finales de S soient exactement les concrétisations des dessins de $\mathcal{S}$ de domaine $\mathbb{R}^2$. De plus, la taille de S est $O(|C||q|^2)$.

Démonstration. Soit $\mathcal{S}$ un tel système de signaux.

Pour chaque méta-collision c de $\mathcal{S}$, on note d_c le dessin contenant c en $(0, 0)$, et uniquement les signaux de c^+ et c^- adjacents. Le domaine de d_c est l'enveloppe convexe de ces signaux. L'aire de cette enveloppe convexe est au plus $|q|^2$

Alors pour tout dessin d enraciné de $\mathcal{S}$ de domaine $\mathbb{R}^2$, si l'on considère le carré unité $U(z)$ autour de $z \in \mathbb{Z}^2$, tous les signaux qui y apparaissent sont adjacent à la même collision $c(z)$. Ce carré unité apparaît donc dans $d_{c(z)}$.

Ainsi, l'ensemble des carré unités inclus dans un c_d donne un ensemble de tuiles qui convient pour fabriquer le système auto-assemblant du théorème 9. Cette ensemble est de taille $O(|C||q|^2)$, et s'énumère en temps $O(|C||q|^2)$. □

3.3.4 Temps de construction et systèmes de signaux

Pour calculer le temps nécessaire à la construction d'un dessin par un système de signaux, nous allons reprendre la même idée que pour les systèmes auto-assemblants. Nous allons appeler *temps* nécessaire pour assembler un dessin la taille de la plus longue chaîne de dépendances dans ce dessin. C'est l'analogue de la définition 19 du temps parallèle pour les systèmes auto-assemblants. Qu'appelons nous une chaîne de dépendance ? Dans un système de signaux, ce sont les signaux qui portent de l'information, dans un sens

déterminé par leur support. Une chaîne de dépendance dans un dessin est donc une suite de signaux s tels que l'extrémité de s_i soit l'origine de s_{i+1}. Pour que sa longueur soit compatible avec ce qui se passe pour les tuiles, on pose $|s| = \sum |s_i|_1$.

Définition 33. *Soit $\mathcal{S}$ un système de signaux, et d un dessin de $\mathcal{S}$. On appelle* chaîne de dépendances *de d une suite de signaux (s_i) telle que pour tout i, l'extrémité de s_i est l'origine de s_{i+1}. La* longueur *de la chaîne s est $l(s) = \sum |s_i|_1$.*

On appelle temps parallèle *de d la quantité :*

$$\tau(d) = \max\{l(s) | s \text{ est une chaîne de dépendances de } d\}$$

Théorème 11. *Soit $\mathcal{S}$ un système de signaux satisfaisant les hypothèses du théorème 9, et soit S le système auto-assemblant qui le simule. Soit d un dessin de $\mathcal{S}$, et p sa concrétisation assemblée par S.*

Alors le temps parallèle de p est égal au temps parallèle de d.

Démonstration. Soit d un dessin, et p la production correspondante. Soit s une chaîne de dépendances de d telle que $l(s) = \tau(d)$. Soit σ la suite des tuiles qui contiennent des signaux de s, orientée suivant les signaux. Cette suite est une chaîne pour l'ordre de p. Donc $\tau(p) \geq \tau(d)$.

Réciproquement, soit σ une chaîne maximale de l'ordre de p. Si les tuiles de σ sont traversées par une chaîne de dépendances de d, alors $\tau(p) \leq \tau(d)$. Sinon, il est possible de construire une chaîne σ' de p de même longueur que σ et qui porte une chaîne de dépendances de d. En effet, si σ passe dans une région, il est possible de trouver σ' de même longueur qui suit les bords de cette région, qui sont des signaux. □

Chapitre 4

Vers l'optimalité

4.1 Optimalité en nombre de tuiles

Dans la littérature, le premier critère que l'on a cherché à optimiser pour les systèmes auto-assemblants est le nombre de tuiles. En effet, si l'on regarde ces systèmes d'un point de vue chimique, le nombre de tuiles dans un système correspond au nombre de composants différents qu'il va être nécessaire de synthétiser. Le plus délicat est la taille de l'alphabet des couleurs : à chaque couleur doit correspondre une séquence d'ADN, et l'on doit maîtriser les interactions entre ces séquences, de façon à ce qu'il n'y ait pas de liaisons entre séquences différentes, et à ce que les forces attractives entre colles de même couleurs soient de la bonne intensité. Plus il y a de tuiles, plus le nombre d'interactions à maîtriser est grand.

D'un point de vue plus informatique, le nombre de tuiles correspond à la complexité de Kolmogorov : un système auto-assemblant est un algorithme particulier pour assembler une forme, et pour l'écrire en binaire, il faut une place qui dépend du nombre de tuiles. Ce rapprochement n'est pas seulement une analogie : d'après [37], si l'on accepte d'assembler des formes *modulo homothétie*, le nombre de tuiles nécessaires pour assembler $\{x\}$ est $\frac{K(x)}{\log\log K(x)}$, où $K(x)$ est la complexité de Kolmogorov de x.

Lorsque l'on cherche à optimiser le nombre de tuiles, on se donne donc un objectif, qui peut être soit une forme donnée, soit un langage de formes. Si l'objectif est une forme, on cherche le système auto-assemblant minimal qui assemble uniquement cette forme. Si c'est un langage, on cherche le système minimal qui assemble exactement les formes de ce langage, et qui soit terminant. Pour examiner l'assemblage de formes géométriques, mieux vaut s'intéresser à un langage, par exemple l'ensemble de tous les carrés qu'à une forme, par exemple un carré de côté 1234. En effet, dans le cas où l'on as-

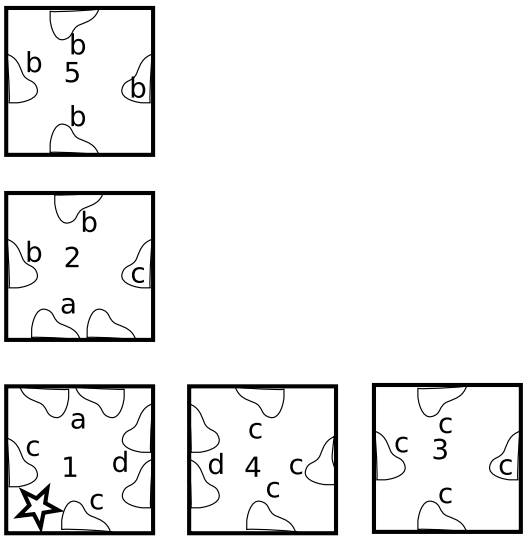


FIG. 4.1 – Le système auto-assemblant optimal pour assembler C.

semble un seul carré, la nécessité de coder sa taille prend le pas sur le nombre de tuiles nécessaire pour assembler des carrés et seulement des carrés.

4.1.1 Carrés

Le premier langage que nous allons assembler est celui des carrés. On définit $C_n = \{0, n\}^2$, c'est à dire que l'on s'intéresse à des carrés avec la source dans le coin inférieur gauche. C_n contient donc $(n+1)^2$ tuiles, mais la distance entre deux sommets consécutifs est n, d'où la notation.

Théorème 12. *Il existe un système auto-assemblant c à température* 2 *avec* 5 *tuiles qui assemble* $C = \{C_n | n > 2\}$.

Le système auto-assemblant est décrit sur la figure 4.1. Pour vérifier qu'il assemble bien des carrés, il suffit de constater que les tuiles d_1 et d_2 ne peuvent apparaître qu'autour de la diagonale, respectivement en $x - y = 1$ et $x - y = -1$. La source apparaît sur la diagonale. La tuile r_1 n'apparaît que dans le demi-plan $x > y + 1$, et r_2 dans son symétrique, comme décrit sur la figure 4.2.

Ce système est optimal.

Théorème 13. *Tout système auto-assemblant à température* 2 *qui produit* C *a au moins* 5 *tuiles.*

Démonstration. Prenons un système auto-assemblant S qui assemble C. Regardons une production finale p de S de domaine $0,1 \times 0,1$, et essayons de

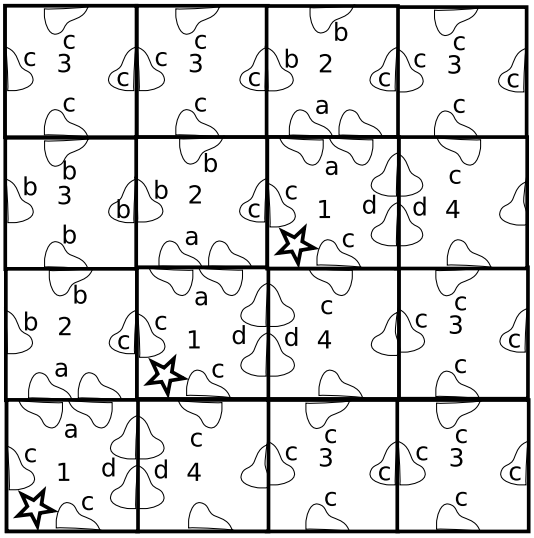


FIG. 4.2 – Une production du système optimal pour les carrés, et l'ordre associé

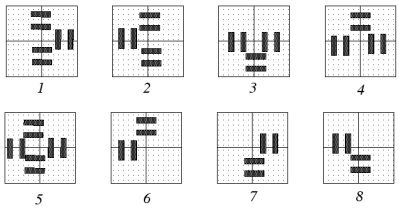


FIG. 4.3 – Les positions possibles des colles de force 2 dans une production 2×2.

voir où sont les colles de force 2. Dans cette production apparaissent 12 colles, qui peuvent être de force 0,1 ou 2. Les colles de force 2 ne peuvent se situer sur un des côtés extérieurs de p, et il doit y en avoir au moins une sur un côté horizontal, et une sur un côté vertical. Cela laisse 8 possibilités, représentées sur la figure 4.3. À symétrie près, il n'y a en fait que 5 cas à considérer.

Dans chacun de ces 5 cas, il y a au moins 3 tuiles différentes, puisqu'il y a au moins 3 positions (sur 4) où les colles de force 2 sont dans des endroits différents.

Si l'on regarde une production de taille 3×3, il apparaît que tout système correspondant aux cas 1 à 7 de la figure 4.3 demande au moins une cinquième tuile pour produire un carré 3×3 : toute attribution des colles de force 2 dans ce carré fait apparaître au moins 5 configurations différentes.

Enfin, pour qu'un système correspondant au cas 8 et n'ayant que quatre tuiles assemble un carré de taille 4×4, il faut que la tuile t qui n'a pas de

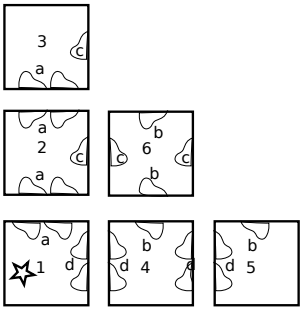


FIG. 4.4 – Le système optimal pour les rectangles

colles de force 2 soit présente dans les coins supérieur-gauche et inférieur-droit. Les deux côtés opposés de cette tuile doivent avoir les mêmes colles, sans quoi il n'est pas possible d'obtenir un carré de taille supérieure à 2. Dans ces conditions, il est possible de remplacer la tuile t dans le coin inférieur-droit par la tuile t'. À ce moment là, il est possible de placer des tuiles en dehors du quart de plan $x > 0, y > 0$, ce qui contredit la définition de C.

Donc tout système auto-assemblant qui assemble C contient au moins 5 tuiles. □

4.1.2 Rectangles

Après avoir assemblé les carrés, nous allons maintenant assembler les rectangles. On définit $R_{n,m} = \{0, \ldots, n\} \times \{0, \ldots, m\}$ et $R = \{R_{n,m} | n, m \geq 2\}$. Pour les rectangles, la taille optimale d'un système auto-assemblant est de 6 tuiles.

Théorème 14. *Le système auto-assemblant r de la figure 4.4 assemble l'ensemble R.*

Démonstration. Remarquons tout d'abord que r est ordonné. L'ordre associé à une production est uniforme : chaque tuile dépend de ses voisins sud et est. Dans la production $R_{n,m}$, la tuile 2 se place sur la colonne $x = 0$, la tuile 4 sur la ligne $y = 0$, la tuile 3 en $(0, m)$, 5 en $(n, 0)$, et la tuile 6 partout ailleurs.

Les tuiles 3 et 5 jouent le rôle d'arrêts de la construction. □

Ce système auto-assemblant, très simple, est en fait le plus petit qui assemble R.

Théorème 15. *Tout système auto-assemblant à température 2 qui assemble R a au moins 6 tuiles.*

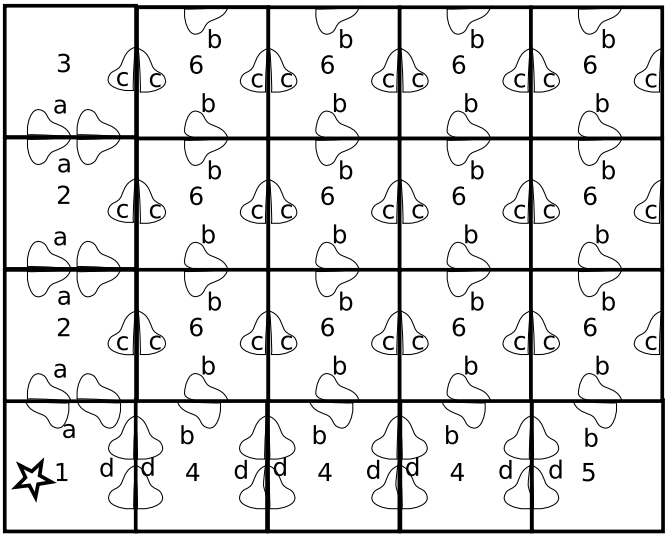


FIG. 4.5 – Une production de r

Démonstration. La preuve de ce théorème est très proche de celle du théorème 13.

Un système qui assemble R doit avoir pour productions finales $C_2 = R_{2,2}$, mais aussi $R_{2,3}$ et $R_{3,2}$. Un système avec n tuiles donne une répartition des colles de force 2 pour chacune de ces productions, avec au plus n tuiles différentes.

Prenons une des configurations de $R_{3,2}$. Dans le cas où la source n'a qu'une colle de force 2 sur son côté nord, il y a 5 configurations possibles avec moins de 5 tuiles, représentées sur la figure 4.6. Il est impossible de paver le rectangle $R_{2,3}$ avec un des jeu de 5 tuiles d'une manière qui soit compatible avec une dynamique d'auto-assemblage.

Un système à 5 tuiles qui assemblerait R avec une seule colle de force 2 sur la source doit donc contenir les 4 tuiles surlignées qui pavent $R_{3,2}$. À ces 4 tuiles, il faut en ajouter une cinquième pour paver $R_{2,3}$. Dans ce cas, les tuiles 1 et 2 ont la même colle sur leur côté nord. Il y a alors des productions qui ne sont pas contenues dans le quart de plan $x \geq 0, y \geq 0$.

On traite symétriquement le cas où la source a une colle de force 2 uniquement sur son côté est.

Les candidats restant ont donc tous deux colles de force 2 sur les côtés nord et est de la source. Parmi ces candidats, examinons la production finale de domaine $R_{3,2}$. Il y a deux cas, suivant la tuile en $(0, 1)$.

– Si cette tuile a deux colles de force 2, au sud et à l'est, il y a 4 systèmes possibles avec 5 tuiles, et un avec 4 tuiles. Aucun de ces systèmes ne permet de construire $R_{2,3}$.

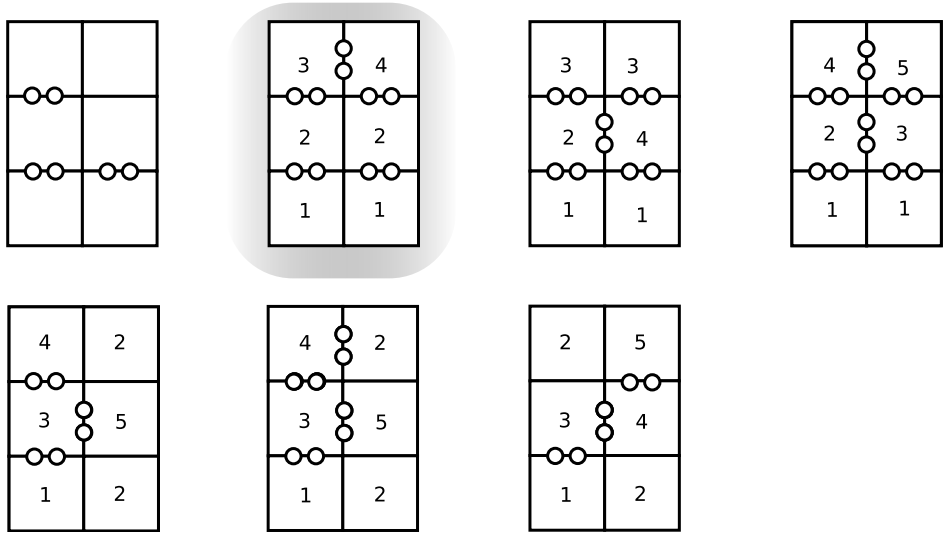


FIG. 4.6 – Positions possibles des colles de force 2 dans $R_{3,2}$ avec une colle de force 2 sur le côté nord de la source.

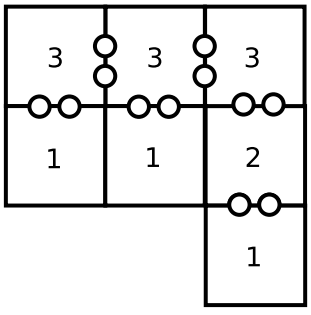


FIG. 4.7 – Une production problématique d'un système issu du système surligné, enrichi avec une tuile 5 pour obtenir $R_{2,3}$.

– De même, si cette tuile n'a qu'une colle de force 2, au sud, il y a 4 systèmes possibles avec 5 tuiles, et un avec 4 tuiles. Ce cas est symétrique du cas où la tuile source a une seule colle de force 2 au nord. Aucun des système à 5 tuiles ne permet donc de construire $R_{2,3}$.

Le seul moyen de construire $R_{2,3}$ et $R_{3,2}$ avec un système a moins de 5 tuiles est donc de prendre la réunion des deux systèmes à 4 tuiles. Ce système a des productions qui ne sont pas dans le quart de plan $x \geq 0, y \geq 0$.

Finalement, après épuisement de tous les candidats, tout jeu de tuiles qui assemble R a au moins 6 tuiles. □

4.2 Optimalité en temps de construction

Après nous être intéressés à minimiser le nombre de tuiles dans nos systèmes auto-assemblants, nous allons nous intéresser à un autre paramètre, le temps nécessaire à réaliser l'assemblage. Ce paramètre a été peu exploré jusqu'à présent, pourtant il est crucial pour toute application « chimique » de l'auto-assemblage.

Notons toutefois qu'en la matière, les marges de progression sont étroites : dans les constructions précédentes du chapitre, nous ne nous sommes pas préoccupés du temps d'assemblage de nos constructions, pourtant il est déjà optimal pour les rectangles, et pour les carrés, il est optimal en ordre de grandeur. Commençons par examiner une borne inférieure commune à toutes les constructions.

Proposition 12. *Soit $f \subset \mathbb{R}^2$ une forme. Alors le temps parallèle nécessaire pour assembler f avec un système ordonné est au moins* $\max\{d_1(0,z) | z \in f\}$, *où $d_1((x,y),(x',y'))$ est la distance $|x-x'|+|y-y'|$ entre (x,y) et (x',y').*

Démonstration. Pour tout $z \in f$, il y a une chaîne qui relie 0 à z. Cette chaîne est de longueur au moins $d_1(0,z)$. □

Cette borne inférieure ressemble aux bornes de *temps réel* utilisées pour les automates cellulaires. Dans les automates cellulaires, on dit qu'un calcul est en temps réel si le résultat du calcul est disponible au premier endroit dans le diagramme espace-temps qui est dans l'intersection des cônes de lumière de tous les points de la donnée []. Par analogie, on dira qu'une production est en *temps réel* si le temps pour l'assembler est égal à son diamètre.

On voit alors que le système que nous avons utilisé pour assembler les rectangles est optimal aussi en temps, puisqu'il assemble $R_{n,m}$ en temps $n+m$. Pour assembler C_n, le temps optimal est $2n$, mais le système que nous avons utilisé assemble C_n en temps $3n$. Est-il possible de faire mieux ? La raison

pour laquelle nous avons eu besoin d'un temps $3n$ pour assembler le carré est que l'on a commencé par construire la diagonale du carré, avant de remplir chacune des moitiés du carré. Est-il possible de faire plus rapide en gardant cette synchronisation ? La réponse est plus aisée à formuler en termes de signaux que directement avec des tuiles.

4.2.1 Assemblage de carrés en temps optimal

Nous allons maintenant présenter un système de signaux dont les dessins sont tous les carrés de coordonnées entières, et de côté pair. Les dessins de ce système sont de temps parallèle $2n$ (où n représente le côté de chaque carré). Nous verrons ensuite une variante pour dessiner tous les carrés de côté entier.

Lemme 9. *Le système de signaux $\mathcal{C}$ assemble l'ensemble des homothétiques de rapport entier de la figure 4.9, et il satisfait les hypothèse du théorème 9.*

Démonstration. Il suffit de constater que tout dessin partiel de $\mathcal{C}$ est contenu dans un homothétique de la figure 4.9. Les autres hypothèses du théorème 9 : coordonnées entières et éloignement des signaux se vérifient sur le dessin. □

Lemme 10. *Le temps parallèle d'un dessin de $\mathcal{C}$ est $2n$, où n est le côté du carré qui contient d.*

Démonstration. Les chaînes maximales arrivent dans un des coins du carré, puisque de n'importe quel point sur un signal dans d, on peut suivre des signaux pour arriver à un coin. Les chaînes de dépendances qui mènent à chacun des coins ont une longueur $2n$, donc le temps parallèle de d est $2n$. □

La variante $\mathcal{C}'$ a les mêmes propriétés que $\mathcal{C}$, mais elle dessine aussi les carrés de côté impair. Pour cela, la collision au centre du dessin diffère suivant que le côté du carré est pair ou impair. Cette information est envoyée au milieu des côtés opposés à la source, où dans le cas impair, un signal vient agrandir le carré d'une unité. En appliquant le théorème 9 à $\mathcal{C}'$, on obtient un jeu de tuiles optimal en temps pour assembler les carrés.

Théorème 16. *Il existe un système auto-assemblant qui assemble l'ensemble des carrés C, et tel que le temps parallèle de la production finale de domaine C_n soit $2n$.*

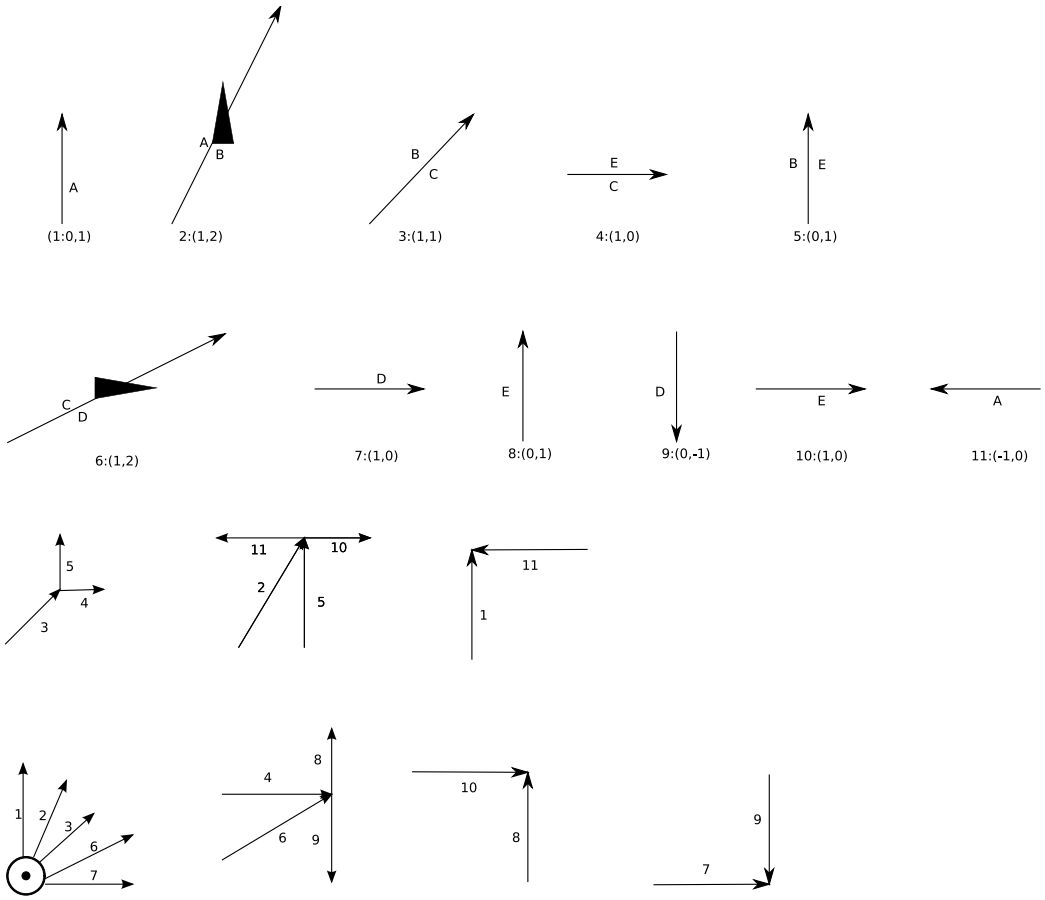


FIG. 4.8 – Le système de signaux $\mathcal{C}$ pour assembler les carrés en temps optimal. Les méta-région sont données par des lettres, les méta-signaux par des entiers. Les méta-collisions ne sont pas nommées. Pour chaque signal, il y a une collision avec $c^+ = c^- = \{s\}$ qui n'est pas représentée et qui permet au signal de se répéter.

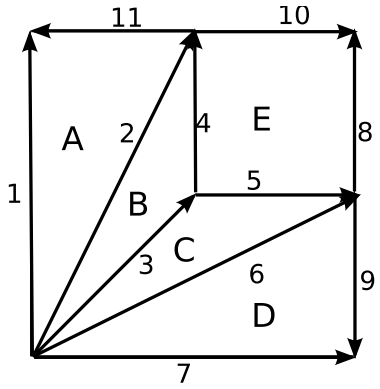


FIG. 4.9 – Les dessins du système de signaux $\mathcal{C}$

Chapitre 5

Auto-assemblage en trois dimensions

5.1 Assemblage en trois dimensions

L'auto-assemblage en trois dimensions est un sujet qui n'avait pas jusqu'à présent été exploré.

La définition de l'auto-assemblage en trois dimensions est la même qu'en deux dimensions. En deux dimensions, nous avons vu que la température 2 jouait un rôle privilégié. En effet, cette température est remarquable à deux titres. D'une part, elle permet le calcul, puisqu'on peut synchroniser de l'information venant de plusieurs positions à la fois, contrairement à la température 1. D'autre part, la température étant égale à la dimension de l'espace, il était possible d'utiliser la température pour vérifier la présence d'un prédécesseur suivant chaque direction, ce qui donnait un rôle particulier aux directions de $\{\mathrm{NE}, \mathrm{SE}, \mathrm{NW}, \mathrm{SW}\}$ dans le cas ordonné. La situation en dimension 3 est moins claire, et l'assemblage à température 2 comme à température 3 méritent d'être examinés.

En dimension 2, pour étudier les systèmes ordonnés, nous avons plusieurs fois utilisé l'argument «la première tuile posée dans une colonne donnée a nécessairement un unique prédécesseur» pour prouver qu'un jeu de tuiles était ordonné. C'est par exemple cet argument qui nous a permis d'obtenir le lemme 4 dans la preuve de la compilation des signaux. On peut reformuler cet argument de la manière suivante : un motif m telle que toute position $z \notin m$ ait au plus un voisin dans m est un rectangle. En dimension 3, on obtient l'équivalent à température 2 : les motifs tels que toute position $z \notin m$ ait au plus un voisin dans m sont les pavés. Pour la température 3, il n'y a pas de caractérisation claire des motifs dont les voisins ont au plus 2 voisins

dans m. De ce point de vue, la température 2 semble donc plus intéressante que la température 3.

Cependant, l'auto-assemblage à température 2 en dimension 3 est difficile à contrôler, et on retrouve les difficultés de l'auto-assemblage à température 1 en dimension 2. Ainsi, supposons un système auto-assemblant dans lequel on a une seule tuile, avec une colle de force 1 sur toutes ses faces. Si l'on a construit 3 faces d'un cube, il est possible de construire n'importe quel plan contenant deux axes du cube. Le remplissage du cube n'est donc pas ordonné, ce qui signifie que la construction est difficile à synchroniser. En particulier, chacun des plans doit pouvoir être construit «en premier», ce qui signifie qu'il est susceptible de bloquer le passage de l'information d'un côté à l'autre. La construction doit donc être extrêmement redondante puisque le flot de l'information est imprédictible au plus haut point. Ce genre de redondance est peut-être possible en utilisant des automates cellulaires réversibles pour tous les calculs, de façon à pouvoir partir de n'importe quel point du calcul.

Dans un premier temps, nous allons donc nous intéresser à la température 3, en gardant à l'esprit la difficulté supplémentaire par rapport à la dimension 2, à savoir qu'il faut vérifier qu'il n'y aura pas de «famine» ou de «dead-lock». Pour cela, nous allons utiliser une analyse plus fine de la condition d'ordre.

De même que la construction «canonique» en dimension 2 était celle du cube, nous allons donner un aperçu des constructions en dimension 3 en construisant un cube en temps optimal.

5.2 Assemblage des cubes en temps réel.

L'idée de la construction des cubes est la même que celle des carrés en temps réel (cf. section 4.2.1) : on envoie à partir de l'origine trois «signaux», tous d'origine $\vec{0}$, vers les centres des faces opposées. Nous n'allons pas développer un système de signaux en dimension 3, car la condition d'ordre (ou de temps-cohérence) devient plus délicate. Au lieu de cela, nous allons étudier l'ordre que l'on veut obtenir sur les tuiles, et en déduire le système auto-assemblant dont nous avons besoin. Nous allons définir cet ordre de manière implicite à l'aide de la fonction qui définit les niveaux de l'ordre. Cette fonction elle-même est en fait définie par trois composantes, qui représentent la distance d'un point à l'origine en passant par chacun des trois signaux. Afin de différencier ces signaux de ceux que nous avons définis plus haut, nous les appellerons *os* ; ensemble, ils constituent le *squelette* sur lequel s'appuie la construction.

5.2.1 Construction d'une fonction de temps optimale

Soit (e_x, e_y, e_z) la base canonique de $\mathbb{R}^3$. L'*os* x est la suite $(a_n)_{n\in\mathbb{N}}$, avec $a_n = (n, \lfloor n/2 \rfloor, \lfloor n/2 \rfloor)$.

Pour chaque cellule $m = (i, j, k)$ de $\mathbb{N}^3$, la fonction f_x induite par l'os x est définie par :

$$f_x(m) = d_1(\vec{0}, a_i) + d_1(a_i, m)$$

.

De manière informelle, considérons qu'une information est transportée par l'os x, puis qu'elle est diffusée dans chaque plan P orthogonal à e_x par la tuile de $P \cap e_x$. La valeur de $f_x(m)$ représente le moment où cette information arrive en m.

La *zone rapide* associée à l'os x est l'ensemble des cellules pour lesquelles le fait que l'information passe par l'os x ne représente pas un détour. On a donc :

$$Z_x = \{m = (i, j, k) \in \mathbb{N}^3, f_x(m) = i + j + k\}$$

Cette zone rapide est caractérisée par :

$$Z_x = \{m \in \mathbb{N}^3, m \geq a_i\} = \{m = (i, j, k) \in \mathbb{N}^3 | 2j + 1 \geq i, 2k + 1 \geq i\}$$

(où $m \leq a_i$ représente l'inégalité composante par composante).

Par symétrie, on définit de même l'os y, correspondant à la suite (b_n), une fonction f_y, un os z, une suite c_n et une fonction f_z. On définit également la zone rapide associée à chacun de ces signaux. Dans la suite, on donnera les propriétés pour un des os, et celles des deux autres se déduisent par symétrie.

De ces fonctions, nous allons déduire l'ordre. Comme cet ordre se manifeste par des relations prédécesseur/successeur locales, on s'intéresse surtout aux variations locales de ces fonctions. Pour chaque cellule, on a :

- $f_x(m) < f_x(m + e_x)$,
- pour $j \geq \lfloor i/2 \rfloor$ (c'est à dire si $2j+1 \geq i$), on a : $f_x(m+e_y) = f_x(m)+1$,
- pour $j \leq \lfloor i/2 \rfloor$ (soit $2j \leq i$) et $j \geq 1$, on a : $f_x(m - e_y) = f_x(m) + 1$,

De la même manière que $f_x(m)$ représente le temps minimal pour que m soit influencée par l'os x, on s'intéresse au temps minimal pour qu'elle soit influencée par tous les os. On introduit donc $f = \max(f_x, f_y, f_z)$.

Pour chaque paire (m, m') de cellules de $\mathbb{N}^3$, on dit que $m <_f m'$ si $f(m) < f(m')$. On dit que m est un prédécesseur de m' si m et m' sont voisins et $m' <_f m'$. On note $<_w$ la clôture transitive de la relation prédécesseur (qui est donc différente de $<_f$, puisqu'on impose la contrainte de voisinage).

Nous allons construire un système auto-assemblant dont l'ordre associé sera exactement $<_w$. Autrement dit, les productions de ce système seront les idéaux de $<_w$. Nous allons étudier en détail la relation prédécesseur pour montrer qu'une telle construction est possible.

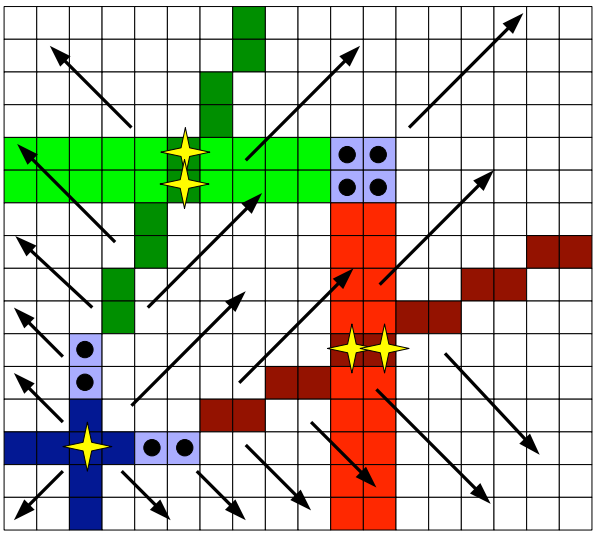


FIG. 5.1 – L'évolution locale de f dans le plan d'équation $z = 5$. Les cellules avec une étoile sont celles qui ont un prédécesseur triple. Celles qui ont un petit disque ont deux prédécesseurs doubles. Les cellules colorées sont celles avec un prédécesseur double. Les flèches indiquent la variation de la fonction f dans le plan $z = 5$. Pour toute cellule m à l'intérieur du carré formé par les cellules colorées, $f(m) < f(m + e_z)$. Pour les cellules à l'extérieur du carré, on a : $f(m) < f(m - e_z)$.

Lemme 11. *Pour toute cellule $m = (i, j, k)$ de $\mathbb{N}^3$,*
- *si $(i, j) \leq (2k + 1, 2k + 1)$, alors $f(m) < f(m + e_z)$,*
- *si $i \geq 2k$ or $j \geq 2k$ (et $k \neq 0$), alors $f(m) < f(m - e_z)$.*

Démonstration. Ce lemme est une conséquence de l'évolution locale de chacune des f_d. Le premier cas vient du fait que $f_x(m+e_z) = f_x(m)+1$ (comme $k \geq \lfloor i/2 \rfloor$), $f_y(m+e_z) = f_y(m)+1$ (comme $k \geq \lfloor j/2 \rfloor$) et $f_z(m+e_z) > f_z(m)$.

Pour le second cas, par symétrie, on peut supposer que $i \geq 2k$, et donc : $f_x(m - e_z) = f_x(m) + 1$. (Puisque $k > \lfloor i/2 \rfloor$). Il faut ensuite procéder à une analyse par cas suivant la valeur de j.

Si $j \geq 2k$, alors $m \geq c_k$ et $m - e_z \geq c_{k-1}$, donc m et $m - e_z$ sont dans F_z.

D'autre part, $f_y(m - e_z) = f_y(m) + 1$.

Donc $f(m - e_z) = \max(f_x(m), f_y(m) + 1)$ et $f(m) = \max(f_x(m), f_y(m))$, ce qui donne : $f(m - e_z) = f(m) + 1$.

Si $\lfloor k/2 \rfloor \leq j < 2k$, alors m et $m - e_z$ sont tous deux des éléments de F_z, et aussi de F_y.

Ainsi, $f(m-e_z) = f_x(m-e_z) = f_x(m)+1$ et $f(m) = f_x(m)$, et le lemme est vérifié.

Si $0 \leq j < \lfloor k/2 \rfloor$ m et $m-e_z$ sont dans F_y. Il faut calculer $f_x(m)$ et $f_z(m)$:

$$f_x(m) = i+\lfloor i/2 \rfloor + \lfloor i/2 \rfloor + \lfloor i/2 \rfloor - j + \lfloor i/2 \rfloor - k = i+(2\lfloor i/2 \rfloor - j)+2\lfloor i/2 \rfloor - k$$

$$f_z(m) = \lfloor k/2 \rfloor + \lfloor k/2 \rfloor + k + i - \lfloor k/2 \rfloor + \lfloor k/2 \rfloor - j + = i + (2\lfloor k/2 \rfloor - j) + hk$$

On sait que $i \geq k$, et $k \leq \lfloor i/2 \rfloor$ (et en particulier $k \leq i$) donc on trouve $f_x(m) \geq f_z(m)$.

De plus, $f_x(m-e_z) = f_x(m)+1$ et $f_z(m-e_z) < f_z(m)$, ce qui donne que $f_x(m-e_z) \geq f_z(m-e_z)$. Finalement, $f(m-e_z) = f_x(m-e_z) = f_x(m)+1 = f(m)+1$.

□

Par symétrie, le lemme 11 contient toute l'information sur les variations locales de f. En particulier, pour toute paire de cellules voisines (m, m'), $f(m) \neq f(m')$. De plus, on n'a jamais à la fois $f(m-e_z) < f(m)$ et $f(m+e_z) < f(m)$.

Un ordre optimal

Nous avons maintenant l'essentiel concernant les propriétés locales de f, et donc de $<_w$. Avant de passer de ces propriétés à un système d'auto-assemblage, nous allons vérifier que ce système sera optimal en temps. Pour cela, il faut montrer que dans le cube $C_n = \{m \in N^3, m \leq (n,n,n)\}$, la fonction f ne dépasse pas $3n$. Comme la fonction f deviendra le temps parallèle au moment de construire le système auto-assemblant, on obtiendra ainsi un système optimal en temps.

Tout d'abord, on vérifie à partir du lemme 11 que pour toute paire (m, m') de cellules voisines telles que $m \in C_n$, mais $m' \notin C_n$, m est un prédécesseur de m'. Les cubes sont donc bien construits de proche en proche. De plus, chaque C_n est un idéal de $<_w$, donc moyennant un processus d'arrêt approprié, notre système construira bien les C_n.

Comme chaque cellule a au moins un successeur dans chaque direction, le maximum de f sur C_n est atteint sur un sommet de C_n. Sa valeur est donnée par le lemme suivant :

Lemme 12.

$$\begin{aligned} t &= max\{f(m), m = (i,j,k) \in N^3, (i,j) \leq (k,k)\} \\ t &= max\{f_z(m), m = (i,j,k) \in N^3, (i,j) \leq (k,k)\} \\ t &= f_z(k,k,k) = 3k \end{aligned}$$

De là, on déduit que $\max\{f(m)|m \in C_n\} = 3n$, ce qui est le résultat attendu.

5.2.2 De la fonction de temps à l'ordre local

Nous allons maintenant montrer que l'on peut transmettre toute l'information sur l'ordre au moyen d'un nombre fini de tuiles. Pour cela, nous allons montrer que la connaissance des prédécesseurs de m suffit presque toujours à déduire ses successeurs. Dans ces cas là, les tuiles sont donc simplement données par les relations prédécesseurs successeurs. Par convention, comme nous allons étudier les configurations en termes de nombre de prédécesseurs, nous allons attribuer des multiplicités aux prédécesseurs des tuiles.

- Chaque cellule a au plus 3 prédécesseurs.
- La cellule $\vec{0}$ est la seule sans prédécesseurs.
- Une cellule a un unique prédécesseur si et seulement si c'est un élément d'un signal. Dans ce cas, on dit que ce prédécesseur est triple.
- Une cellule $m = (i,j,k)$ avec deux prédécesseurs a nécessairement deux voisins opposés qui sont des successeurs. S'il s'agit de $m + e_z$ et $m - e_z$, alors $\lfloor i/2 \rfloor = k$ ou $\lfloor j/2 \rfloor = k$

Si $u = (i,j,k)$ est tel que $\lfloor i/2 \rfloor = k$, le prédécesseur de m d'abcisse i est appelé prédécesseur double de m. Si $u = (i,j,k)$ est tel que $\lfloor j/2 \rfloor = k$, le prédécesseur de m d'ordonnée j est appelé prédécesseur double de m. On définit de même des doubles prédécesseurs pour les cas symétriques.

Les quatre cellules telles que $\lfloor i/2 \rfloor = k$ et $\lfloor j/2 \rfloor = k$ ont deux prédécesseurs doubles.

Lemme 13. *Soit I un idéal non-vide de $<_w$, et $m_0 \notin I$.*

La cellule m_0 a au moins 3 prédécesseurs (en comptant les multiplicités) si et seulement si $I \cup \{m\}$ est un idéal de $<_w$.

Grâce à ce lemme, on voit que $<_w$ peut être implémenté par un système auto-assemblant à température 3 dans lequel les forces des colles correspondent aux multiplicités des relations prédécesseurs-successeurs. Pour cela, il reste bien sûr à vérifier que la transmission d'information peut se faire grâce à un nombre fini de couleurs.

5.2.3 De l'ordre aux tuiles

Étant donné l'ensemble des prédécesseurs d'une cellule $m = (i, j, k) \in \mathbb{N}^3$, est-il possible de déterminer quels sont les multiplicités des successeurs de m en utilisant uniquement la donnée de ces prédécesseurs.

Pour réduire le nombre de cas que nous allons devoir étudier, prenons la convention que si m a une composante nulle, disons x, on dit que $m - e_x$ est un successeur virtuel, dont on choisira la multiplicité pour se ramener à un des cas ci-dessous.

- Une cellule m a un unique prédécesseur triple si et seulement si m est un élément d'un os. Supposons que $m = a_i$. Alors ses quatre voisins dans le plan $x = i$ sont des successeurs doubles. $m + e_x$ peut être soit un simple successeur soit un triple :
  - si i est pair, c'est un successeur triple
  - si i est impair, c'est un successeur simple
- Une cellule m a deux prédécesseurs double si et seulement si $\lfloor i/2 \rfloor = k$ et $\lfloor j/2 \rfloor = k$ (ou un cas symétrique), et ces prédécesseurs sont $m - e_x$ et $m - e_y$. $m - e_z$ et $m + e_z$ sont des successeurs simples.
  - Si i est pair, $m + e_x$ est un successeur double
  - Si i est impair, $m - e_x$ est un successeur simple

  De même pour $m + e_y$.
- Une cellule m a un double prédécesseur si et seulement si $\lfloor i/2 \rfloor = k$, $\lfloor j/2 \rfloor \neq k$ et $\lfloor i/2 \rfloor \neq j$ (à symétrie près). Dans ce cas, m a trois successeurs simples, et un double, qui est l'opposé de son prédécesseur double.
- Dans tous les autres cas, m a trois prédécesseurs simples, il a trois successeurs simples, sauf si un de ses successeurs est dans un os, auquel cas c'est un successeur triple. C'est le cas si $m = a_{2i-1} + e_y + e_z$, et ce successeur est a_{2i}. On a alors $a_{2i-1} <_w a_{2i-1} + e_y <_w m$.

Grâce à cette énumération, on voit que les informations nécessaires pour déterminer les multiplicités des successeurs sont : le nombre et la position des prédécesseurs, la parité des coordonnées, et la proximité d'un os.

À chaque face partagée par deux cellules de $\mathbb{N}^3$, on associe trois étiquettes :

- une direction dans $\{+++, ++, +, -, --, ---\}$ qui indique le sens de la relation prédécesseur successeur et sa multiplicité,
- une parité dans $\{0, 1\}$ –deux faces opposées d'une même cellule sont de parités opposées,
- chaque site pair a_{2i} du l'os x porte une étiquette spéciale, que $a_{2i} + e_y$ transmet à $a_{2i} + e_y + e_z$

Grâce à ces étiquettes, on obtient un système auto-assemblant à température 3 qui remplit le quadrant $\mathbb{N}^3$ en suivant l'ordre $<_w$.

Le processus d'arrêt

À partir de ce système, on peut obtenir un système auto-assemblant qui assemble $\{C_n | n \in \mathbb{N}\}$. Pour cela, comme dans le cas des carrés, on ajoute des étiquettes pour marquer la diagonale principale. On ajoute ensuite une tuile d'arrêt qui vient se fixer sur la diagonale. La position où elle vient se fixer devient $\lfloor n/2 \rfloor$ et donne la valeur n. Comme on a $(\lfloor n/2 \rfloor, \lfloor n/2 \rfloor, \lfloor n/2 \rfloor) >_w (n, \lfloor n/2 \rfloor, \lfloor n/2 \rfloor)$, on peut envoyer un signal stop de cette tuile vers les milieux des faces opposées, où on arrête la croissance des os.

Avec cette construction, on a obtenu un système auto-assemblant pour les cubes en temps optimal.

Théorème 17. *Il existe un système auto-assemblant ordonné à température 3 qui assemble $\{C_n | n \in \mathbb{N}\}$, et tel que le temps parallèle pour assembler C_n soit $3n$.*

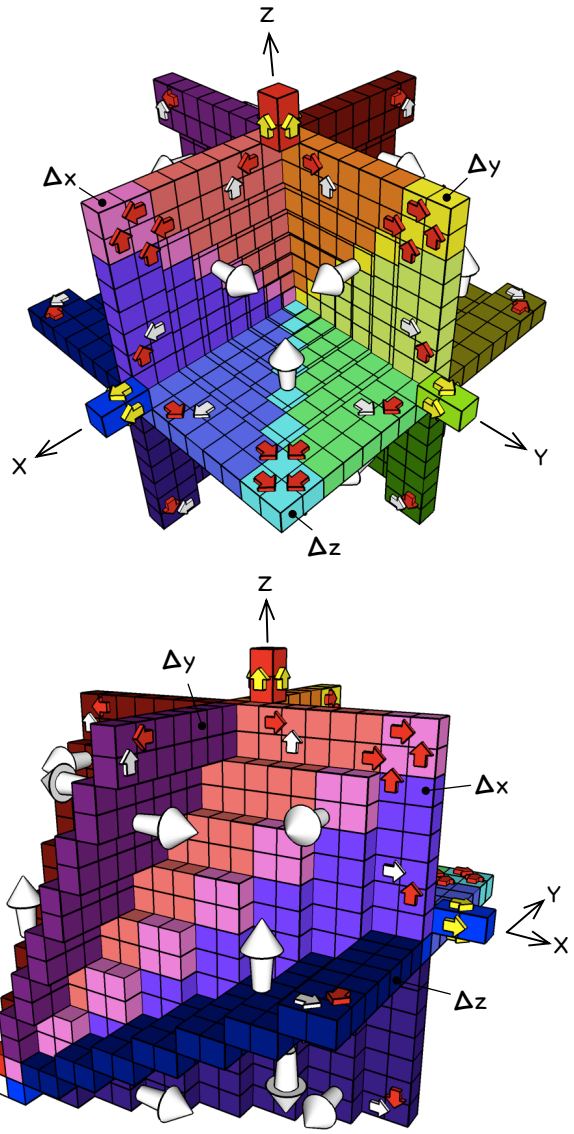


FIG. 5.2 – Les variations locales de f, et l'ordre $<_w$ sur $\mathbb{N}^3$. À gauche, l'ordre vu depuis (n, n, n). À droite, depuis $(n, 0, n)$. On ne représente que les cellules avec au plus 2 prédécesseurs. Les flèches indiquent la croissance de f suivant chacun des axes. Les flèches blanches représentent des prédécesseurs simples, les rouges des prédécesseurs doubles, et les jaunes des prédécesseurs triples. L'ordre est uniforme sur 7 régions séparées par les plans où les cellules ont des prédécesseurs doubles. Les intersections de ces plans sont les 3 signaux, représentés en couleurs primaires.

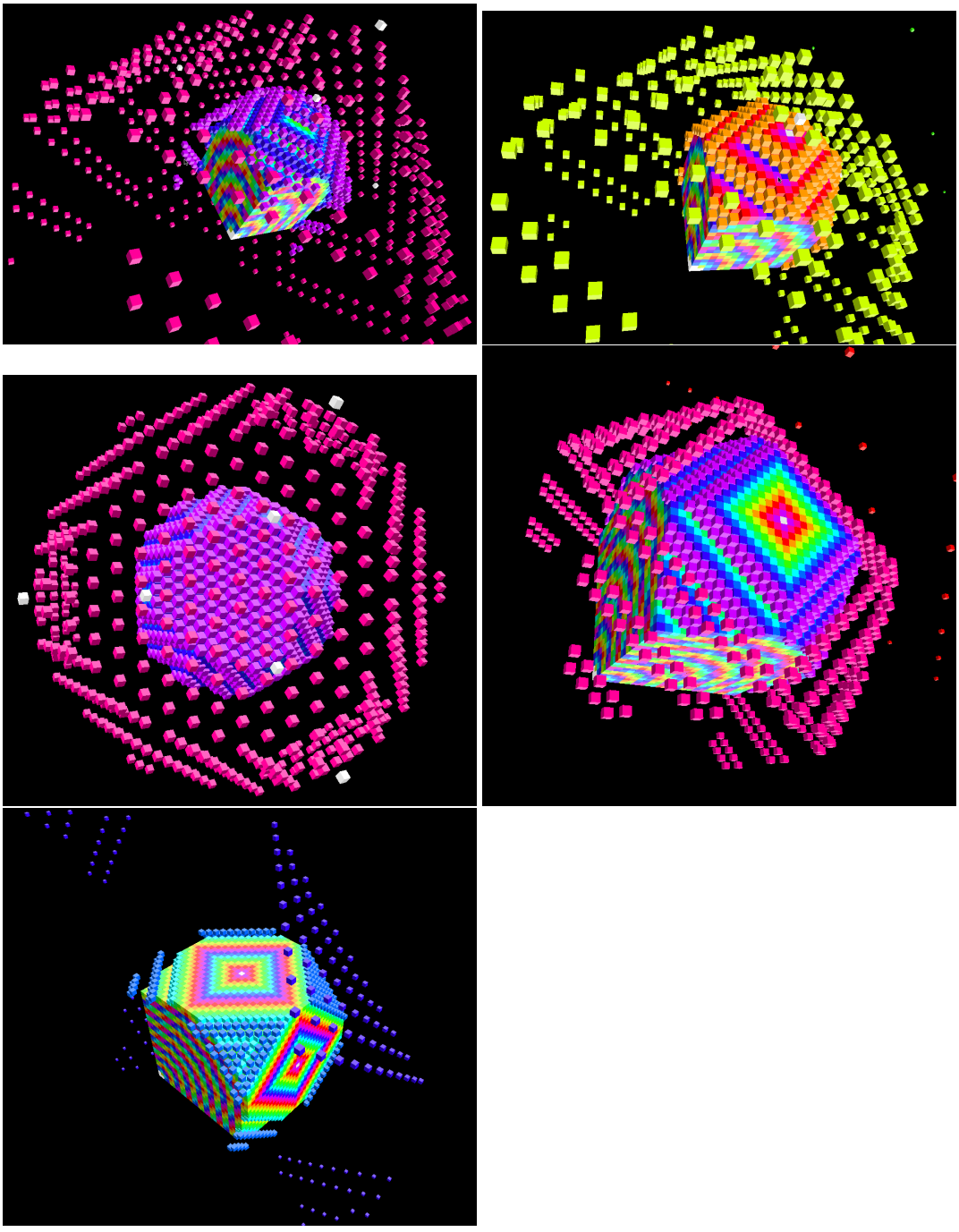


FIG. 5.3 – Quelques étapes de l'assemblage d'un cube en temps optimal suivant la dynamique parallèle : chaque image représente les cellules où $f < \rho_0$ pour des valeurs croissantes de ρ_0.

Chapitre 6

Homothéties

Après les signaux, nous allons présenter une autre manière d'étendre les possibilités de l'auto-assemblage. Il s'agit d'implémenter des transformations géomètriques. Implémenter une transformation géomètrique peut se faire de plusieurs manières.

On peut tout d'abord considérer qu'on implémente une transformation géométrique si on donne un algorithme qui transforme un système assemblant un langage L en un système qui assemble $f(L)$, l'ensemble des transformées des élements de L.

On peut aller plus loin et vouloir que cet algorithme transforme non seulement les productions finales, mais aussi la dynamique. On veut alors que l'assemblage du système transformé ressemble à celui du système de départ. Dans ce cas, on retrouve sur le système transformé des propriétés, par exemple de temps ou de robustesse aux erreurs, que l'on avait dans le système de départ.

Enfin, on peut vouloir un ensemble de tuiles qui réalisent une transformation de manière universelle. On ajoute alors ces tuiles à un système auto-assemblant, et la réunion des deux systèmes est la transformée du système de départ.

Il existe une dernière possibilité de transformation, «en deux bains». Il s'agit d'auto-assembler des *macro-tuiles* dans un premier «bain», puis de les faire réagir entre elles dans un second bain. Pour cela, on se donne un premier système T, dont on interprète les productions comme les tuiles d'un système auto-assemblant S, que l'on appelle *macro-tuiles*. On peut alors avoir par exemple $|T| << |S|$, ce qui correspond à une économie importante en nombre deu tuiles. Transformer la dynamique, c'est à dire montrer que la dynamique de T est l'image de celle de S revient à faire mieux, et à assembler les macro-tuiles en même temps qu'elles interagissent. On revient ainsi dans le modèle à un bain.

Nous allons présenter ces concepts sur les plus simples des transformations géométriques, les homothéties. Dans ce cadre, les macro-tuiles sont donc des carrés.

6.1 Transformation de la dynamique

6.1.1 Transformation de la dynamique et transformation du langage

Commençons par illustrer la différence entre transformation du langage et transformation de la dynamique par un exemple. Considérons trois systèmes d'auto-assemblage à température 2, S, F et D. S est un système qui assemble l'ensemble C des carrés, de manière assez simple. F assemble l'ensemble $2C$, de même que D. Cependant, la dynamique de D ressemble à celle de S, pas celle de F.

La dynamique de S S est le jeu de tuiles que nous avons vu au chapitre précédent pour assembler les carrés avec un nombre minimal de tuiles. Rappelons que l'assemblage se fait en partant de la diagonale : les tuiles de la diagonale avancent, en même temps que le reste du carré se remplit à partir de la diagonale. C'est un jeu de tuile ordonné, cette dynamique est donc contenue dans la donnée de l'ordre pour chaque production finale.

La dynamique de F F assemble l'ensemble $2C$ des carrés de taille paire, mais avec une dynamique qui n'a pas grand-chose à voir avec celle de S. Pour assembler un carré de taille $2n$, F assemble un premier carré de taille n. Puis il assemble son symétrique par rapport à un axe vertical. Ensuite, il assemble le symètrique par rapport au centre. Enfin, il complète le dernier quart du carré.

La dynamique de D La dynamique de D, quant à elle, correspond bien à celle de S. Si l'on regarde les carrés de taille 2 dont les coordonnées sont paires, on peut associer à chaque motif qui apparait dans ces carrés une tuile de S. L'ordre et la position dans lesquels ces motifs apparaissent correspondent à la dynamique de S. Ainsi, si l'on prend une production non-terminale de S, et que l'on remplace chaque tuile de S par le motif 2×2 correspondant de D, là où un ajout de tuiles est possible dans S, il est possible d'ajouter les 4 tuiles correspondantes dans D.

On dit alors que la dynamique de D est l'image de celle de S par une homothétie de facteur 2. Plus simplement, D est une 2-expansion de S.

6.1.2 Définition formelle

Définissons plus formellement cette notion d'expansion. Dans cette section, S et T sont deux systèmes d'auto-assemblage, et l'on veut savoir si S est une s-expansion de T. Pour cela, il nous faut d'abord définir une notation pour les homothéties sur les formes.

Définition 34. *Soit ϕ une forme, on note $h_s(\phi) = \{(x,y)|(\lfloor x \rfloor, \lfloor y \rfloor) \in \phi\}$.*

Pour pouvoir étendre cette définition aux motifs, il faut pouvoir compacter l'information contenue sur s^2 tuiles de S en une tuile de T.

Définition 35. *Une* s-macrotuile *de T est une fonction de $\{0,\dots,s-1\}^2$ dans T. On note T_s l'ensemble des s-macrotuiles de T. Pour un motif m, la s-macrotuile en (x,y) de m est définie par $(i,j) \mapsto m(sx+i, sy+j)$.*

Une fonction d'interprétation *est une fonction de S dans T_s.*

Avec cette fonction d'interprétation, il est possible d'interpréter un motif de S comme une dilatation d'un motif de T_s. Il suffit d'appliquer la fonction d'interprétation à tous les carrés de la grille de pas s.

Définition 36 (réduction)**.** *Une motif m est réductible si il existe ϕ telle que* $\mathrm{Dom}(m) = h_s(\phi)$.

Étant donné une fonction d'interprétation i et un motif réductible m, on note $i_s(m)$ le motif défini par :

- $\mathrm{Dom}(i_s(m)) = \phi$
- $\forall (x,y) \in \phi$, *si t est la s-macrotuile en (sx,sy), alors* $(i_s(p))(x,y) = i(m)$.

On note $D_{s,S}$ le sous-treillis des productions s-réductibles de S.

À partir de cette définition sur les productions de S, nous pouvons comparer $D_{s,S}$ et la dynamique D_T de T.

Définition 37. *Soit D_T le treillis des productions de T, and D_S celui de S. S est une s-expansion de T si*

1. *Il y a une interprétatino i qui envoie les s-macrotuiles de S dans T, telle que i_s soit un isomorphisme entre $D_{s,U}$ et D_T.*
2. *pour toute production p of S, il y a $p' \geq p$ qui est s réductible.*

Cette définition formalise la similitude entre D et S dans l'exemple. Il était possible d'attribuer à chaque 2-macrotuile de D une tuile de S de façon à ce que les deux dynamiques soient isomorphes.

Comme on s'y attend, cette condition est plus forte que le fait que S assemble l'image du langage de T.

Proposition 13. *Soit S et T deux jeux de tuiles, qui assemblent L_s et L_T respectivement. Si S est une s-expansion de T, alors $L_S = h_s(L_T)$.*

Démonstration. Soit p une production finale de T. $I_s^{-1}(p)$ est bien défini. Comme i_s est un isomorphisme, c'est une production finale de S, et sa forme vérifie $\mathrm{Dom}(i_s^{-1}(p)) = h_s(\mathrm{Dom}(p))$.

Réciproquement, soit p une production finale de S. Alors, p est s-réductible et $i_s(p)$ est une production finale de T dont la forme vérifie $\mathrm{Dom}(p) = h_s(\mathrm{Dom}(i_s(p)))$. □

L'existence d'un ismomorphisme entre D_T et $D_{s,S}$ ne signifie pas que l'on peut grouper les transitions de S par paquets de s^2 et les traduire en transitions de T. On a seulement une correspondance entre les chemins. En effet, dans S, plusieurs macrotuiles peuvent se construire en parallèle. Quand un tel cas se produit, dans $D_{s,S}$, on saute des étapes qui avaient lieu dans D_T.

Il est cependant possible de retrouver les transitions de D_T dans $D_{s,S}$. En effet, à tout chemin c dans la dynamique de S, il est possible d'associer un chemin c' contenant les mêmes transitions dont on peut extraire une suite $(m*)$ de motifs réductible tels que la suite $(i_s(m*))$ est un chemin valable dans la dynamique de T.

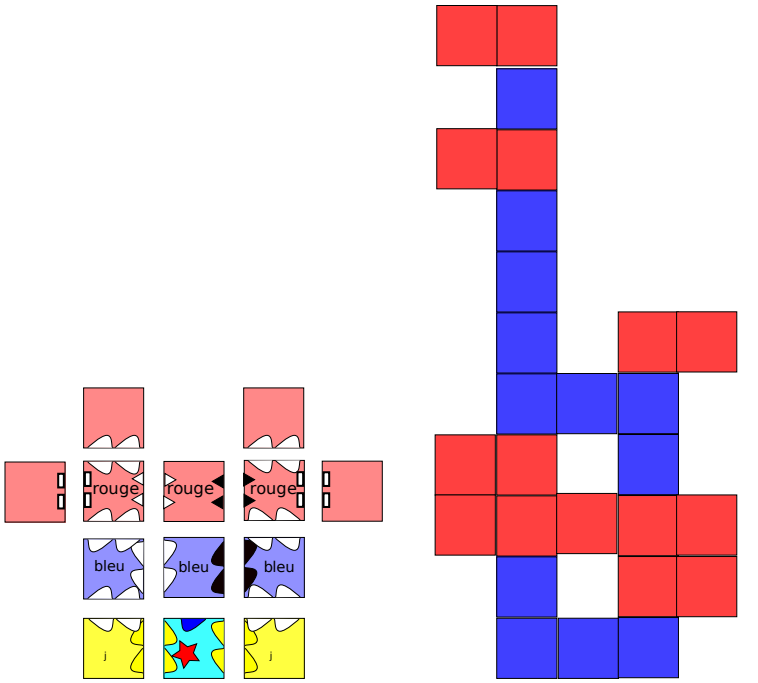
Réciproquement, si $m = m(0) \xrightarrow{\mathcal{T}} \ldots \xrightarrow{\mathcal{T}} m(n)$ est un chemin de D_T, alors il existe un chemin m' dans la D_S qui passe par les $i_s^{-1}(m(k))$ pour chaque k. Ce chemin n'est pas unique, et il y a d'autres chemins qui vont de $i_s^{-1}(m(0))$ à $i_s^{-1}(m(n))$ sans passer par tous les $i_s^{-1}(m(k))$. Cependant, comme ces chemins ont les mêmes extrémités que c', ils ont les mêmes transitions, mais dans un ordre différent.

6.2 Transformabilité

Maintenant que nous avons défini ce que signifie transformer la dynamique, il est temps de vérifier que cette notion n'est pas oiseuse. Nous allons voir que dans le cas des systèmes ordonnés, il est possible, pour tout s, de trouver une s-expansion. Dans le cas général, cependant, il n'est pas toujours possible de trouver une s-expansion.

Pour montrer qu'il n'existe pas toujours d's-expansion d'un système auto-assemblant, nous allons utiliser le fait que l'auto-assemblage est asynchrone, et donc que si la localité est détruite, des problèmes de synchronisation apparaissent. Cette absence de synchronisation va «casser» la dynamique au moment de passer à l'échelle.

Théorème 18. *Il existe un système auto-assemblant S qui n'a pas d's-expansion pour $s \geq 2$.*



(a) Les tuiles

(b) Un exemple de production

FIG. 6.1 – Le système S, de température 2, qui constitue notre contre-exemple

Démonstration. Soit S le système de la figure 6.1(a). S produit le langage L défini comme suit : pour chaque production finale p, il existe deux entier l_1 et l_{-1} contiennent des tuiles aux endroits suivants :

- en $\{-1\} \times \{0, \ldots, l_{-1}\}$; ces tuiles sont soit rouges soit bleues ;
- en $\{1\} \times \{0, \ldots, l_{-1}\}$, rouges ou bleues aussi ;
- dans le sous ensemble de $\{0\} \times \{0, \ldots, h_0\}$ défini par $\{(0, y) | p(-1, y) = p(1, y)\}$ (avec $l_0 = \min(l_1, l_{-1})$) ;
- en $(-2, y)$, il y a une tuile quand la tuile en $(-1, y)$ est rouge.
- en $(2, y)$, il y a une tuile quand la tuile en $(1, y)$ est rouge.

Dans les productions finales, il y a une tuile en $(y, 0)$ quand il y a des tuiles à la fois en $(y, 2)$ et $(y, -2)$, ou quand il n'y en a dans aucune des deux positions.

Nous allons utiliser cette caractéristique de L pour montrer que S n'a pas de s expansion. Supposons que S_s soit une s-expansion de S, à température τ avec $s \geq 2$. Considérons la partie de la dynamique de S représentée sur

FIG. 6.2 – La partie de la dynamique qu'il n'est pas possible de reproduire à une autre échelle

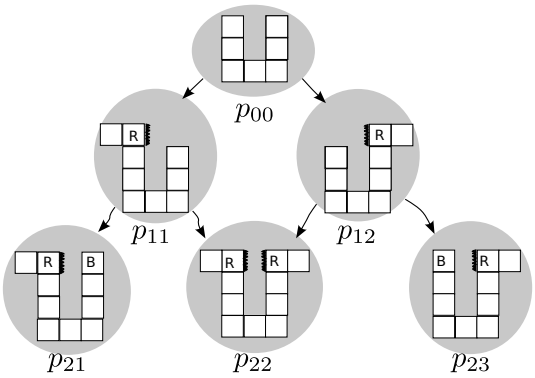


FIG. 6.3 – Un extrait de la dynamique, qui n'est pas réalisable à plus grande échelle. Chaque flèche représente plusieurs transitions, mais ces transitions ne concernent qu'un voisin de $(0, 3)$ à la fois.

la figure 6.3, dans laquelle aucune tuile ne peut être ajoutée en $(0, 1)$ ni en $(0, 2)$.

Soit p'_{ij} l'antécédent de p_{ij} par i_s. Les p_{ij} et les p'_{ij} sont dans le même ordre. Donc à partir de p'_{21} et p'_{23}, aucune tuile ne peut être ajoutée dans la colonne centrale. Il n'y a donc pas de colle de force τ sur les côtés marqués de p'_{21} et p'_{23} (en gras).

De même, il ne peut pas y avoir de colle de force τ sur les côtés marqués de p'_{22}. Comme il n'y a pas de colles de force τ dans les endroits marqués, aucune tuile ne peut être ajoutée dans la colonne centrale. Ceci contredit l'hypothèse que S_s soit une s expansion de S. □

6.3 Homothéties dans le cas ordonné.

Pour éviter ce problème, nous allons utiliser la condition d'ordre. En effet, la condition d'ordre nous préserve de ces problèmes de localité quand chaque tuile n'a que deux prédecesseurs, sur des côtés adjacents.

Théorème 19. *Soit $\mathcal{S}$ un système auto-assemblant, et $s \geq 2$. Si $\mathcal{S}$ est ordonné et si dans toute production p, toute position $z \in \mathrm{Dom}(p)$ a une direction dans $\{N, S, E, W, NE, SE, SW, NW\}$, alors il existe un système $\mathcal{S}_s$ qui est une s-expansion de $\mathcal{S}$.*

Dans les sections suivantes, nous allons décrire $\mathcal{S}_s$, et donner la preuve qu'il est bien une expansion de $\mathcal{S}$.

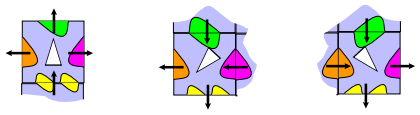


FIG. 6.4 – On ajoute une flèche sur chaque côté des tuiles pour marquer l'ordre.

6.3.1 Le système utilisé

Pour construire $\mathcal{S}_s$, nous allons commencer par construire une version «ordonnée» $\mathcal{S}'$ de $\mathcal{S}$ dans laquelle chaque tuile ne peut apparaître que dans une direction donnée. Ensuite, c'est cette version ordonnée que nous allons dilater. Comme $\mathcal{S}$ est ordonné, sa dynamique est la même que celle de $\mathcal{S}'$. Donc $\mathcal{S}_s$, la version dilatée de $\mathcal{S}'$ sera bien une s-expansion de $\mathcal{S}$.

Construction de $\mathcal{S}'$ Soit t une tuile de $\mathcal{S}$. Pour chaque direction $d \in \{N, S, E, W, NE, SE, NW, SW\}$, on se donne une tuile t^d si t peut étendre un motif dans la direction d. Par exemple, si $d = NE$, on se donne une tuile t^{NE} si la somme des colles sur les côtés sud et ouest de t est au moins τ, la température.

Les colles du $\mathcal{S}'$ sont le produit d'une colle de $\mathcal{S}$ par une direction $d \in N, S, E, W$. Les forces des colles sont héritées de $\mathcal{S}$. Une tuile t^d a sur son côté c la colle $(\sigma_c(t), c)$ si $c \in d$, et une colle $(\sigma_c(t), c^{-1})$ sinon. Ainsi, on peut voir les directions des colles de $\mathcal{S}'$ comme des flèches qui pointent sur chaque côté vers l'intérieur ou l'extérieur des tuiles et qui indiquent l'ordre de chaque production.

Pour la source, on ajoute la tuile $\odot$, c'est à dire une tuile dont toutes les flèches pointent vers l'extérieur.

Montrons que cette transformation ne change pas la dynamique.

Lemme 14. *Soit U la fonction qui à t^d associe t. On étend naturellement U aux motifs, et à la dynamique. Alors $U(D_{\mathcal{S}'}) = D_{\mathcal{S}}$.*

Démonstration. Il suffit de remarquer que pour chaque ajout d'une tuile t dans la direction d dans $\mathcal{S}$, il est possible d'ajouter t^d dans $\mathcal{S}$, et réciproquement. □

Les flèches que l'on a ajoutées aux tuiles de $\mathcal{S}'$ correspondent bien à leur direction, ce qui va nous permettre de les découper. En effet, le découpage va dépendre de la direction de chaque touche.

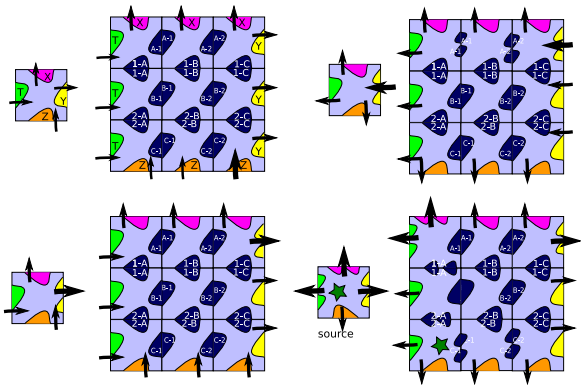


FIG. 6.5 – Découpage d'une tuile de $\mathcal{S}'$ en s^2 tuiles de $\mathcal{S}_s$. À gauche, deux tuiles NE, avec ou sans colle sortante de force 2, à droite une tuile N et la source. Les autres directions sont identiques à ces deux cas à une rotation près.

Lemme 15. *Dans une production de $\mathcal{S}'$, la direction d'une tuile t^d est d.*

Démonstration. On montre par induction que dans toute production de $\mathcal{S}'$, les flèches qui sont à la frontière de la production pointent vers l'extérieur de celle-ci. Donc tout ajout se fait par les côtés où les directions des colles pointent vers l'intérieur de la tuile. Donc toute tuile t^d est ajoutée dans la direction d. On appellera donc d la direction de t^d. □

Passage de $\mathcal{S}'$ à $\mathcal{S}_s$. Pour chaque tuile de $\mathcal{S}'$, nous allons donner une tuile de $\mathcal{S}_s$, comme montré sur la figure 6.5. Le découpage que l'on utilise dépend de la direction de la tuile. Chaque tuile de $\mathcal{S}'$ est découpée en s^2 morceaux, qui vont former une s-macrotuile. Chaque arête interne à la macrotuile a une colle de couleur unique. Les forces de ces colles sont détaillées sur la figure 6.5. Pour les arêtes externes, les colles sont des copies des colles de $\mathcal{S}'$ correspondante. On se donne deux copies de chaque colle de force 2, une de force 2, et une de force 1. On utilise la copie de force 2 une seule fois par côté de chaque macrotuile, pour la tuile la plus à gauche ou la plus haute du côté. Le reste du côté est recouvert de colles de force 1

Pour la source, on se donne un ensemble de tuiles qui assemblent un carré de côté s, avec sur ses côtés, l'encodage des colles de la source de $\mathcal{S}'$.

L'interprétation i que l'on utilise pour ce système auto-assemblant est simple : les macrotuiles qui correspondent au découpage d'une tuile de $\mathcal{S}'$ sont envoyées sur cette tuile, et la fonction i n'est pas définie sur les autres.

6.3.2 Preuve de la transformation

Observons maintenant la dynamique de $\mathcal{S}_s$, et vérifions qu'il s'agit bien d'une s-extension de $\mathcal{S}'$.

Nous allons utiliser la notion de «production locale». Une *production locale* est une configuration qui apparaît au cours de la construction d'une macrotuile m. Une production locale de m est une configuration inclue dans le carré $[0, s)^2$ qui peut être atteinte à partir d'un carré vide avec les couleurs d'entrée de m sur les arêtes adjacentes aux côtés d'entrée de m.

Pour montrer que si $i_s(c) \leq i_s(c')$ dans T', alors $c \leq c'$, il suffit de montrer que s'il y a une transition entre $i_s(c)$ et $i_s(c')$, alors $c \leq c'$. Cela s'établit facilement, il suffit de reproduire les constructions de la figure 6.5. Il est toujours possible de le faire, grâce au lemme 15.

Pour la suite, on ajoute un élément $\perp$ à $D_{\mathcal{S}'}$ tel que $\perp \leq c$ pour tout $c \in D_{\mathcal{S}'}$. On ajoute de même un élément $\perp$ à $D_{\mathcal{S}_s}$. On pose $i_s(\perp) = \perp$. On obtient alors le lemme suivant :

Lemme 16. *Pour tout production réductible d et pour tout $c \geq d$, il y a une configuration e de $\mathcal{S}$ telle que :*

- $\mathrm{Dom}(e)$ *est le plus petit domaine $\sigma \subset \mathbb{Z}^2$ tel que $h_s(\sigma) \supset \mathrm{Dom}(c)$*
- $e \geq i_s(d)$
- *Pour tout carré C de taille $s \times s$ en xs, ys, c restreint à C est une production locale de la macrotuile correspondant à $e(x, y)$.*
- *et c est ordonnée.*

Démonstration. Pour un d donné, prouvons ce lemme par induction sur c. Quand $c = d$, $e = i_s(d)$ convient.

Soit $c \geq d$, et $c \xrightarrow[\lfloor]{\mathcal{T}} t@(sx + i, sy + j)]c'$ (avec $0 \leq i, j < s$). Soit e donné par l'hypothèse d'induction pour c.

Si $(x, y) \in \mathrm{Dom}(e)$, alors e vérifie aussi le lemme pour c'. La condition sur les formes est clairement vérifiée, et on a bien $e \geq i_s(d)$. Puisque c est fait de production locales et qu'il satisfait la condition d'ordre, t ne peut être placée que s'il a des voisins sur ses côtés d'entrée (à cause du lemme 15). Elle ne peut donc être ajoutée qu'en accord avec la production locale, et suivant la condition d'ordre.

Si $(x, y) \notin \mathrm{Dom}(e)$, alors soit $t' \in \mathcal{S}'$ la tuile correspondant à la macrotuile dans laquelle t apparaît. Soit t_1 et (éventuellement) t_2 les voisins de t sur ses côtés d'entrée, et t'_1 et t'_2 les analogues dans $\mathcal{S}'$. Comme t a pu être ajoutée à côté de t_1 et t_2, t' peut être ajoutée entre t'_1 et t'_2. Soit $e' = e \cup \{(x, y) \to t'\}$. e' vérifie les conditions du lemme.

□

Avec ce lemme, si $c \leq c'$ et que c et c' sont réductibles, alors $i_s(c) \leq i_s(c')$.

Soit c une production de $\mathcal{S}_s$. Il existe une production correspondante e de $\mathcal{S}'$ telle que c est composée de productions locales de e. Soit $(x_0, y_0) \dots (x_k, y_k)$ une chaîne de transitions de $\perp$ à e, et i le plus petit k tel que $\mathrm{Dom}(c) \cap ([sx_k, s(x_k+1)[\times[sy_k, s(y_k+1)[) \neq [sx_k, s(x_k+1)[\times[sy_k, s(y_k+1)[$, s'il existe. Alors, comme $\mathrm{Dom}(c) \cap ([sx_k, s(x_k+1)[\times[sy_k, s(y_k+1)[)$ est une production locale et que ses prédécesseurs sont complets, il est possible d'ajouter une tuile à c dans $[sx_k, s(x_k+1)[\times[sy_k, s(y_k+1)[$. Donc si c n'est pas réductible, il existe $c' > c$ qui est réductible.

Donc $\mathcal{S}_s$ est bien une s-expansion de $\mathcal{S}'$, et donc aussi de $\mathcal{S}$, en utilisant une interprétation qui oublie les directions.

6.3.3 Un système un peu plus économique

Le système que nous avons donné dans la section 6.3.1 permet de prouver le théorème 19, mais sa taille est relativement importante : chaque tuile est remplacée par $4s^2$ tuiles. Il y a une grande redondance : en effet, chaque macrotuile code le paramètre s. On peut économiser beaucoup de tuiles en ne codant ce paramètre que dans la macrotuile représentant la source, et en le transmettant de proche en proche. On a alors 9 types de tuiles par macrotuile autre que la source.

Pour la source, il faut coder le paramètre s. On peut utiliser pour cela un compteur binaire à $O(\log s)$ tuiles, comme présenté dans [32]. On obtient alors un ensemble de taille $O(\log s)$ tuiles qui permettent d'assembler un carré $s \times s$ portant sur ces côtés les couleurs correspondant au découpage de la graine.

Il est aisé de vérifier que ce jeu de tuiles est aussi une s-expansion de $\mathcal{S}'$.

6.4 Système universel pour les homothéties

Dans le cas ordonné, nous avons donc construit une s-expansion. Cela signifie qu'à partir d'un système auto-assemblant, nous sommes parvenus à construire un système ayant la même dynamique, mais à une échelle différent. Nous allons aller un peu plus loin, en introduisant un système *universel* pour les homothéties. On peut considérer qu'il n'est pas très satisfaisant d'avoir à créer un système auto-assemblant homothétique pour chaque système de départ en partant de zéro. Nous allons exhiber un ensemble de tuiles T_s universel tel qu'en ajoutant T_s à n'importe quel système S, on obtient une s-expansion de S.

Il y a biens sûr des restrictions à cela. Tout d'abord, nous aurons toujours besoin de la condition d'ordre, et même d'une condition plus forte, la condition RC. De plus, notre ensemble $G(s)$ doit dépendre de l'ensemble des couleurs de S, car sinon, aucune interaction ne sera possible entre les tuiles de $G(s)$ et celles de S. De même, S doit avoir une température 2. La construction de la graine est à part, donc $G(s)$ va dépendre de la graine de S. Enfin, cette construction n'est pas exacte (et il ne peut pas y avoir de construction exacte) : la dynamique est la même, mais elle va «un peu plus loin» que les productions finales qu'elle devrait assembler.

Commençons par définir cette notion d'approximation. Pour cela, nous avons besoin d'une distance entre objets.

Définition 38. *Soit $(x, y), (x', y') \subset \mathbb{Z}^2$ deux formes, la distance entre p et p' est donnée par : $d(p, p') = \max(\max\{d(x, p) | x \in p'\}, \max\{d(x, p') | x \in p\})$.*

Avec cette distance, nous définissons une approximation de la dynamique. Un treillis de productions est une s-expansion approchée si ses productions finales ne s'éloignent pas de plus d'une distance s de ce qu'elles devraient être.

Définition 39. *Soit D_U la dynamique d'un système U, et D_T celle de T. On dit que U est une s-expansion approchée de T à l'échelle s quand :*

1. *Il existe une fonction d'interprétation i qui envoie les s-macrotuiles de U sur T, telle que i_s soit un isomorphisme entre $D_{s,U}$ et D_T*
2. *Pour toute production c de U, il y a un c' qui est réductible tel que $\mathrm{Dom}(c') \subset \mathrm{Dom}(c)$ et $d(c, c') \leq s$.*

Nous pouvons maintenant construire nos tuiles universelles pour les homothéties.

Théorème 20. *Soit $T = (C, g, T, 2, \odot)$ un jeu de tuiles satisfaisant la condition RC tel qu'aucune tuile autre que $\odot$ n'ait deux côtés avec des colles de force 2, et soit s un entier.*

Il y a alors
- *un ensemble de couleurs $C' \supset C$,*
- *une fonction de force $g' : C' \to \{0, 1, 2\}$ compatible avec g sur C*
- *un ensemble $G(C, s) \subset C^4$ de tuiles, ne dépendant que de C*
- *un ensemble $G_\odot(C, s)$ de tuiles, ne dépendant que de C et $\odot$*
- *une tuile $\odot' \in G_\odot(C, s)$*

tels que :

$T_s = (C', g', T \cup G(C, s) \cup G_\odot(C, s), 2, seed')$ est une s-expansion approchée de T.

L'idée de la construction, que nous allons détailler, est que chaque tuile $t \in T$ sera représentée par une macrotuile qui contient t dans un coin. Les autres tuiles de la macrotuile permettre de transmettre les couleurs entre les tuiles de t qui sont distantes pour leur permettre d'interagir.

6.4.1 La construction

Chacune des macrotuiles que nous allons utiliser sera soit un *berceau* soit une *tombe*. Chacune de ces deux constructions contient exactement une tuile de T, qui définit la valeur de la fonction d'interprétation sur cette macrotuile.

Commençons par énoncer une série d'invariants sur les productions de T_s. Soit p une telle production

1. Toute s-macrotuile qui apparaît dans p est soit un berceau soit une tombe
2. La première tuile ajoutée dans une tombe est une tuile de T ;
3. chaque tombe partiellement construite peut être terminée ;
4. les tuiles dans un berceau ont toute la direction du berceau[1] ;
5. la dernière tuile ajoutée dans un berceau est une tuile de T
6. soit p' la partie de p formée de macrotuiles complètes. Alors $i_s(p')$ est une configuration de T.

Les interactions entre macrotuiles se font le long de leurs frontières. Nous allons donc commencer par définir le langage de ces frontières. Le long d'une frontière, il n'y a que quatre types de colles qui peuvent apparaître : soit des colles de T, soit l'une des trois colles de C' suivantes : $Coin$, $Presque$, $Cote$, toutes de force 1.

La colle $Coin$, et les colles de C apparaissent dans les coins des macrotuiles, la colle $Presque$ sur les deuxième et avant-dernière tuile de chaque côté, la colle $Cote$ sur le reste du côté. Sur chaque côté de chaque macrotuile, il y a une colle de C et une colle $Coin$.

Construction d'un berceau La construction d'un berceau reprend l'idée de la construction des macrotuiles de la section 6.3.3 : chaque berceau construit un carré de la même taille que ses voisins, avec une tuile de T dans un de ses coins. Les colles internes, au lieu d'être toutes différentes, servent à transmettre les couleurs des côtés d'entrée à la tuile de T qui va être ajoutée. Un berceau se présente comme indiqué sur la figure

[1] la direction des berceaux sera définie plus bas

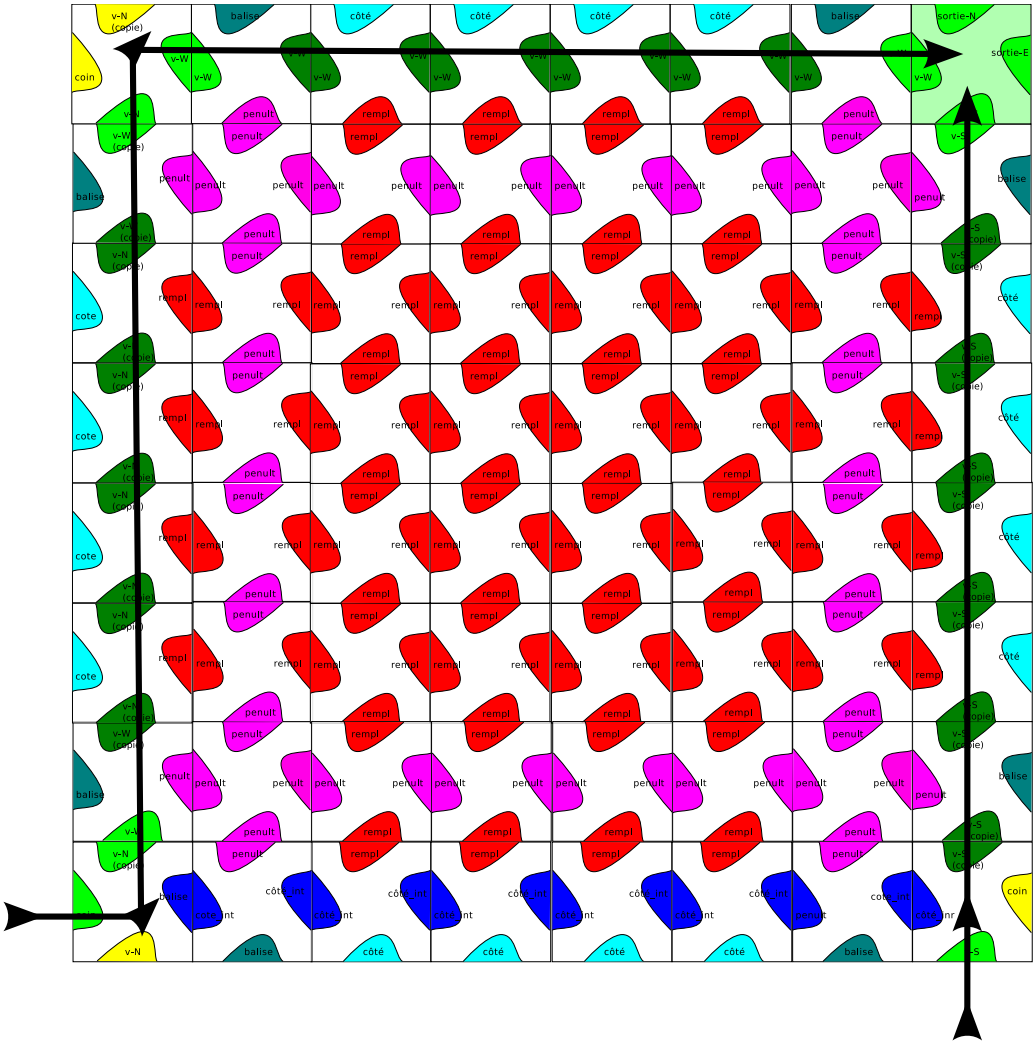


FIG. 6.6 – Un berceau (avec la tuile de T au nord-est). Les flèches représentent la transmission de l'information « couleur de T ».

Construction d'une tombe La construction d'une tombe est un peu plus difficile que celle d'un berceau, car elle ne s'appuie que sur un voisin. Supposons que l'on construise une tombe à partir du côté sud, et que le voisin sud ait sa colle de C dans son coin nord-ouest (notre coin sud-ouest). La tuile de T correspondant à la tombe que l'on construit doit se coller dans le coin sud-ouest. C'est donc sa voisine de droite qui commence la construction de la tombe à proprement parler. On envoie un signal en diagonale pour donner une forme carrée à la tombe, et des signaux messagers qui vont transmettre les couleurs de sortie de la tuile de T vers les autres côtés.

6.4.2 Preuve de la transformation

Les figures 6.6 et 6.7 nous indiquent quelles sont les productions locales pour chaque macrotuile. On en déduit facilement que toute transition de T peut être remplacée par s^2 transitions de T_s, qui correspondent à l'ajout des tuiles de la macrotuile. Il suffit pour cela de vérifier que quand un coin est vide entre deux macrotuiles correspondant à deux prédécesseurs de $t \in T$, on peut construire le berceau correspondant. De même, si t a un unique prédécesseur, alors on peut construire la tombe de t à partir de la macrotuile de ce prédécesseur.

Réciproquement, nous devons montrer que toute suite de transitions de T_s entre productions s-réductibles correspond à une suite de transitions de T, puis que toute production est au plus à une distance s d'une production réductible.

On considère pour cela une suite de transitions $a_i = t_i@(x_i, y_i)$ de D_s. Définissons trois fonction $prem(k)$, $dern(k)$ et $\chi(k)$ qui sont les numéros des transitions «clés» de a_i. Pour cela, on définit la fonction auxiliaire α comme suit :

$$\begin{aligned} \alpha(1) &= 1 \\ \alpha(i+1) &= \alpha(i') \text{ s'il existe } i' \leq \text{ tel que } h_s(x_{i+1}, y_{i+1}) = h_s(x_{i'}, y_{i'}) \\ \alpha(i+1) &= \max\{\alpha(j) | j \leq i\} + 1 \text{ sinon} \end{aligned}$$

On définit $prem(k)$ comme la suite des temps où une nouvelle macrotuile a été commencée. $first(k) = \min \alpha^{-1}(k)$. De même, $dern(k)$ est la suite des temps où une macrotuile a été achevée : $dern(k) = \max\{\alpha^{-1}(k)\}$. $\chi(k)$ est le plus petit i tel que $\alpha(i) = k$ et $t_i \in T$.

On définit de même $prem(x, y)$, le plus petit i tel que $h_s(x_i, y_i) = (x, y)$, $dern(x, y)$ comme le plus grand, et $\chi(x, y)$ celui tel que $t_i \in T$. On s'en sert pour trouver les tuiles «caractéristiques» d'une macrotuile donnée par sa position.

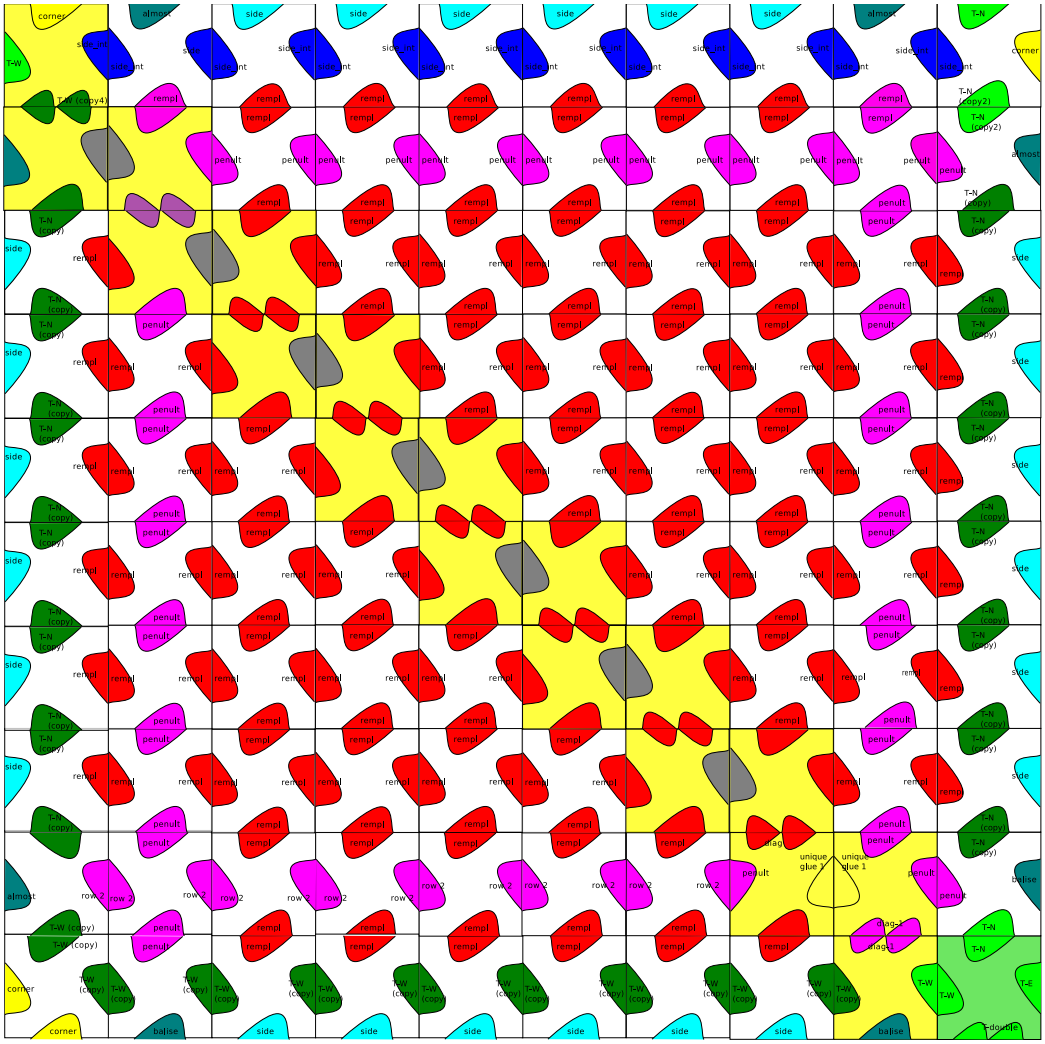


FIG. 6.7 – Une tombe (avec la tuile de T au sud-est)

Nous allons montrer par induction que les propriétés 6.4.1 sont vérifiées à chaque étape de a.[2]

Il convient d'abord de vérifier ces propriétés sur les productions locales de la macrotuile source.

Ensuite, soit $i > 1$. S'il n'y a pas de k tel que $i = last(k)$ ou $i = last(k)$, alors soit $(x, y) = h_s(x_i, y_i)$. À cause des invariants, $t_i \notin T$. On vérifie alors que tous les invariants sont préservés. Que $h_s^-(x, y)$ soit un berceau ou une tombe, a_i continue sa construction. Comme $t_i \notin T$, il ne peut être posé que s'il a un voisin dans $T_s \setminus T$ au moins (car il n'y a pas de tuiles de $T_s \setminus T$ avec une colle de force 2 de C ou deux colles de force 1 de C). Ce voisin permet d'assurer que t_i est correcte. On vérifie simplement que tous les ajouts de tuiles hors de T dans une macrotuile sont déterministes.

Si $i = first(k)$, soit $t_i \in K$, soit $t_i \notin T$. Si a_i a une direction dans $\{N, S, E, W\}$, alors $t_i \in T$, et elle peut être recouverte par une tombe. Les invariants sont toujours vérifiés après son ajout. Si a_i à une direction diagonale et que l'un de ses voisins au moins a une colle $Coin$, alors $t_i \notin T$, et a_i commence la construction d'un berceau. Dans ce cas encore, les invariants sont préservés.

Le dernier cas, celui où $first(k)$ est l'ajout d'une tuile de T dans une direction diagonale, est impossible. Supposons que l'ajout a lieu en (x, y), dans la direction SE. Regardons les orientations des tuiles en $h_s(x, y + 1)$ et $h_s(x - 1, y)$. La tuile en $h_s(x, y + 1)$ ne peut pas avoir une orientation dans $\{W, NW, SW\}$, puisqu'elle n'a pas encore de voisin ouest. De même, la tuile en $h_s(x + 1, y)$ ne peut pas être N, NE ni NW.

Si l'une des deux voisines est SE, alors la macrotuile correspondante présente la colle $Coin$. Il ne nous reste donc plus qu'a traiter les cas résumés dans la table 6.1 pour les orientations de $h_s(x, y + 1)$ et $h_s(x - 1, y)$.

	S	E	W	SW
N	2	1	3	3
S	1	4	1	1
E	3	1	2	4
NE	3	1	3	3

TAB. 6.1 – Argument utilisé en fonction de la direction des tuiles en $h_s(x, y+1)$ (horizontalement) et en $h_s(x - 1, y)$ (verticalement)

1. Les cas marqués 1 peuvent être être éliminés en regardant les orientations possibles de $h_s(x - 1, y - 1)$. Cette position est soit d'orientation

[2] On en déduit que $\chi(k)$ est l'unique i tel que $\alpha(i) = k$ et $t_i \in T$.

NE soit d'orientation SW. Dans ces deux cas, la construction des voisins garantit que la tuile de T de l'un d'entre eux est dans la dernière colonne, ce qui signifie qu'il y a bien une colle $Coin$ en $\sigma_E(x-1,y)$ ou $\sigma(x,y+1)$.

2. Le cas marqué 2 est impossible car il ne peut pas y avoir deux tuiles marquées N et S (ou E et W) dans deux lignes adjacentes (respectivement deux colonnes adjacentes).
3. les cas marqués 3 sont exclus car ils ne sont pas compatibles avec la condition RC.
4. dans le cas 4, la tuile en $h_s(x-1,y+1)$ devrait avoir deux colles de force 2, ce qui contredit les hypothèses du théorème.

La tuile en $prem(i)$ initie toujours la construction d'une tombe ou d'un berceau, ce qui garantit les invariants.

Grâce à cela, on obtient que la suite des $\chi(i)$ est un chemin valable dans la dynamique de T.

Il est facile de voir que quand la construction d'un berceau (par exemple de direction NE ne peut pas être terminée, parce qu'aucune tuile (de T) ne peut être ajoutée en (x_0,y_0), le quart de plan $x > x_0, y > y_0$ reste vide.

Chapitre 7

Assemblage de polygones exacts

Pour aller plus avant dans la géométrie discrète, nous allons tenter l'assemblage de polygones convexes à coordonnées entières. Nous allons assembler ces polygones avec une précision arbitraire. Autrement dit, étant donné un polygone *modèle*, dont les sommets ont des coordonnées entières, nous avons un système auto-assemblant qui assemble tous les homothétiques de ce polygone. Plus l'on assemble une version agrandie du polygone, plus la discrétisation que l'on opère en passant à des tuiles est fine. Ce n'est donc pas le même type d'homothétie discrètes que celles que l'on avait décrites au chapitre précédent.

Étant donné une suite de sommets $p = (p_i) \in Z^*$ représentant les sommets d'un polygone convexe, on dit qu'un système auto-assemblant T assemble le polygone p si T assemble $\{z \in \mathbb{Z}^2 | U(z) \cap H(p) \neq \emptyset\}$, où $H(p)$ représente l'enveloppe convexe de p, et $U(z)$ le carré unité centré en z. Cette discrétisation des polygones correspond bien à la discrétisation que l'on a utilisée quand on est passés des systèmes de signaux aux tuiles dans le théorème 9.

Dans le système auto-assemblant que nous allons décrire, le choix de l'échelle est non-déterministe. Cela peut sembler peu pratique, mais il est possible d'intégrer à la construction un compteur binaire qui permet de fixer l'échelle de la construction en ajoutant quelques tuiles. En contrepartie, la présentation du système est plus claire ainsi, et peut se faire simplement en terme de signaux.

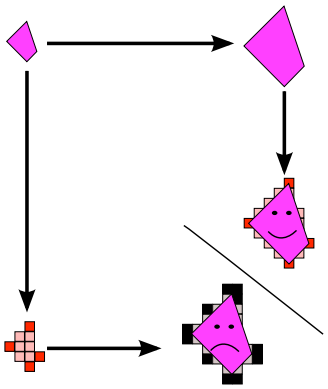


FIG. 7.1 – Discrétisation et changement d'échelle ne commutent pas.

7.1 Une construction pour les cas simples

À cause des difficultés inhérentes à la géométrie discrète, nous allons d'abord présenter une construction qui ne fonctionne que pour les cas où la discrétisation du polygone est «raisonnable».

On dit qu'un polygone est *connectable* si pour tout sommet s de p, il existe une demi-droite verticale ou horizontale issue de v dont l'intersection avec p n'est pas réduite à $\{v\}$.

Soit $p = (s_0, \ldots, s_k) \in (\mathbb{Z}^2)^k$ un polygone avec s_N sont sommet le plus haut —celui dont la coordonnée y est maximale, s_S le plus bas, s_W le plus à gauche, et s_E le plus à droite. On appelle *quartier* NE les sommets entre s_N et s_E, et de même pour SE, NW, SW.

Pour deux sommets successifs $z_0 = (x_0, y_0)$ et $z_1 = (x_1, y_1)$, $z_0 \wedge z_1$ le point de $\{(x_0, y_1), (x_1, y_0)\}$ qui est du même côté de la ligne $z_0 z_1$ que p. En général, $z_0 \wedge z_1$ est dans p, mais il peut arriver qu'il soit «au-delà» de p si p est très «fin».

On appelle *triangles externes* les éléments de

$$O = \{z_0, z_0 \wedge z_1, z_1 | z_O, z_1 \text{ deux sommets successifs du même quartier de } p\}$$

on appelle *triangles internes* les éléments de

$$I = \{z_0 \wedge z_1, z_1, z_1 \wedge z_2 | z_0, z_1, z_2 \text{ trois sommets successifs du même quartier de } p\}$$

Un polygone est *joli* si aucun de ces triangles ne s'intersectent, et si tous leurs côtés sont de longueur au moins 3.

Dans un premier temps, nous allons montrer comment construire les jolis polygones. Nous montrerons comment s'adapter au cas moins joli dans la section 7.2.

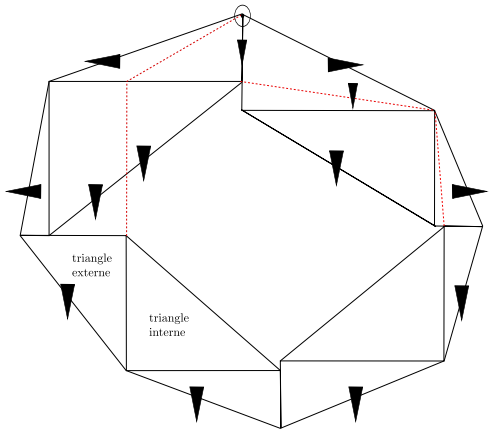


FIG. 7.2 – Un premier système de signaux pour construire un polygone

7.1.1 Solution naïve

Le schéma général de la solution que nous allons utiliser est représenté sur la figure 7.2.

Théorème 21. *Soit p un joli polygone. Il existe un système de signaux dont les formes des dessins de domaine $\mathbb{R}^2$ sont les $\{np\}$ pour $n \in \mathbb{N}$. Le nombre de signaux de ce système est $O(|p|)$, le nombre de côtés de p.*

Dans la figure 7.2, tous les points de construction sont à coordonnées entières, car ce sont des intersections de lignes verticales et horizontales passant par des points de coordonnées entières.

On se donne une méta-collision pour chaque sommet de chaque triangle, interne ou externe, de p. La méta-collision associée à s_N est la source. Pour chacune de ces méta-collisions, c^- et c^+ sont définis par la figure 7.2, et par la description des signaux qui suit.

Les deux triangles extérieurs adjacents à $\odot$ sont des cas particuliers, nous les traiteront à la fin. Dans le reste de la description, on écrit «tous les triangles», ou «tous les triangles externes» pour «tous les triangles (externes) sauf ces deux là».

À chaque côté de chaque triangle interne ou externe, on associe un méta-signal. Ces signaux sont orientés vers le bas, et quand ils sont horizontaux, ils sont orientés vers l'extérieur de p dans les quartiers NE et NW, et vers l'intérieur dans les quartiers SE et SW. Les forces des hypoténuses sont données dans la table 7.1, les autres côtés sont de force O. Chacun de ces signaux est répétitif : il y a une collision c_s avec $c_s^+ = c_s^- = \{s\}$.

quartier	NE	SE	NW	SW
triangle interne	V	O	V	O
triangle externe	H	V	H	V

TAB. 7.1 – Les forces des hypoténuses de chaque triangle.

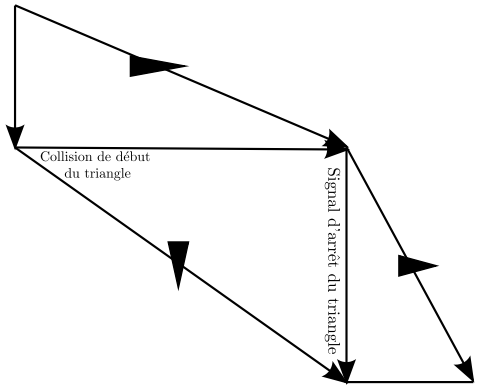


FIG. 7.3 – Le détail de trois triangles dans le quartier SE

Dans chaque quartier, la taille des triangles est transmise de proche en proche par leur côté commun. Ce côté commun agit aussi comme un signal d'arrêt pour le deuxième triangle dans l'ordre de la construction.

Entre deux quartiers successifs, il n'est pas possible de transmettre ainsi la taille de la construction, car le côté commun n'est qu'une *partie* du côté de l'un des deux triangles. Considérons la frontière entre les quartiers NE et SE. Le cas des quartiers NW et SW est symétrique. Il y a deux possibilités : soit le triangleNEest plus large que le triangle SE, soit c'est le triangle SE qui est le plus large. On appelle t_N le dernier triangle externe du quartier NE, et t_S le premier triangle externe du quartier SE.

Soit (x_0, y_0) le sommet sud-ouest de t_N, (x_1, y_0) le sommet nord-ouest de t_S, et $(x_2, y_0) = s_E$, le sommet du polygone. Dans le premier cas, qui est effectivement représenté à l'est sur la figure 7.2, et repris sur la figue 7.4, on ajoute dans t_N un signal O qui coupe la ligne $y = y_0$ en (x_1, y_0). Ce signal part de (x_N, y_0), le sommet le plus haut de t_N. Ce signal a bien une pente rationnelle, donc on peut l'implémenter dans nu système de signaux. On utilise un signal qui se répète, de façon à pouvoir gérer les différentes échelles. On peut ainsi avoir une collision en (x_1, y_0) qui arrête la construction de t_N et continue la construction du reste du polygone.

Dans le second cas (représenté à l'ouest sur la figure 7.2, et en détail sur

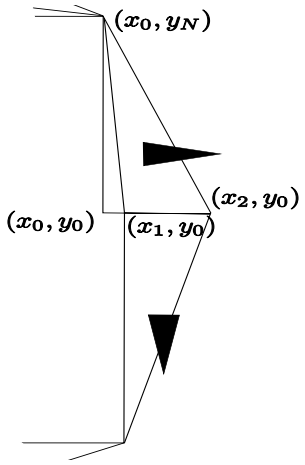


FIG. 7.4 – Vue de détail de la transmission entre quartiers dans le premier cas

la figure 7.5, il y a un triangle externe du quartier NW ou NE qui intersecte la ligne $x = x_1$. Soit t'_N ce triangle, et y_N l'ordonnée de son côté inférieur. On peut lancer un signal O vertical à partir de (x_1, y_N). Ce signal devra croiser l'hypoténuse d'un triangle interne. On remplace cette hypoténuse par deux segments de pente légèrement différente pour que l'intersection soit de coordonnées entières. On envoie donc un signal O pour repérer le point (x_1, y_N), puis un signal vertical vers (x_1, y_0). On envoie également un signal horizontal de (x_0, y_0) vers (x_1, y_0). L'intersection de ces deux signaux donne le point (x_1, y_0) d'où l'on peut continuer la construction. Si l'angle formé par l'hypoténuse de t_N et l'horizontale est inférieur à $\pi/4$, alors le système de signaux n'est plus aéré. On remédie à ce problème en courbant cette hypoténuse au besoin une deuxième fois pour qu'elle ait le bon angle.

Supposons pour le moment que l'on ait défini correctement les éléments qui permettent de construire les deux premiers triangles à toute échelle.

Ce système de signaux est construit *ad-hoc*, et il est facile de vérifier que l'ensemble de ses dessins de domaine $\mathbb{R}^2$ est l'ensemble des homothétiques de la figure 7.5. Cependant, aucun de ces dessins n'est temps-cohérent, car les signaux V à gauche et à droite de p interfèrent.

7.1.2 Le mur de Berlin

On ajoute donc au centre du carré un signal V^- qui empêche ces interférences, comme représenté sur la figure 7.6

Avec ce signal V^-, les dessins de $\mathcal{S}$ sont bien temps-cohérents.

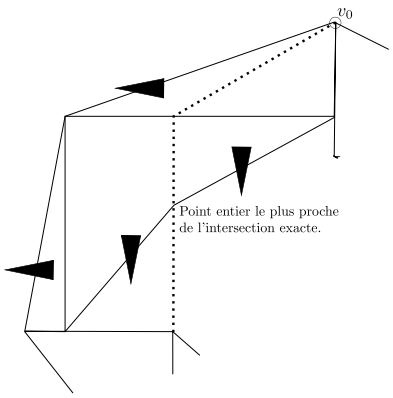


FIG. 7.5 – Transmission de la construction entre les deux quartiers ouest. La ligne en pointillés est le nouveau signal, qui "tord" légèrement une hypoténuse pour que l'intersection soit un point entier.

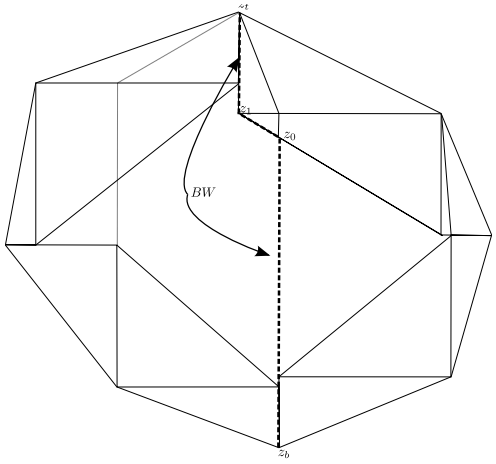


FIG. 7.6 – Le signal V^- qui sépare les deux moitiés du polygone

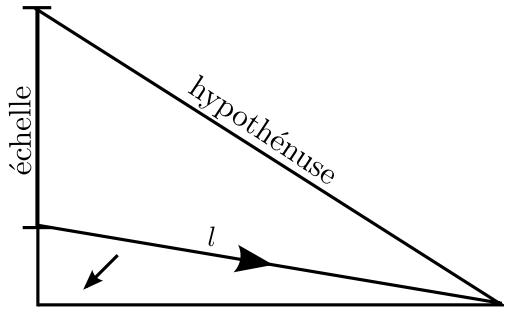


FIG. 7.7 – La construction du plus grand des deux triangles initiaux

Initialisation —les deux premiers triangles

Il nous reste à construire les deux triangles adjacents à s_N, qui seront utilisés comme «étalon» des mesures pour la suite de la construction. Plus précisément, on appelle *étalon* le segment que ces deux triangles partage. Sa longueur code la valeur de n. On considère celui dont le côté sud est le plus élevé, disons $((x_N, y_N)(x_N, y_N - h)(x_N + w, y_N - h))$. On envoie un signal de direction $(0, -h)$ pour construire la ligne $(nx_t, ny_t)(nx_t, ny_t - nh)$, qui nous sert d'étalon. (Avec une collision qui répète ce signal, et qui sert à choisir la valeur de n. Ensuite, la construction du triangle $((x_t, y_t)(x_t, y_t - h)(x_t + w, y_t - h))$ a lieu comme celle des autres triangles. On transforme le plus grand des deux triangles en un triangle non rectangle, de façon à partager la longueur de l'étalon entre les deux triangles.

Ceci achève, dans le cas des jolis polygones, la construction du système de signaux.

On vérifie que les hypothèses du théorème 9 sont vérifiées, et l'on obtient ainsi un jeu de tuiles qui assemble p à toute résolution.

7.2 Le cas général

7.2.1 Cas des triangles dégénérés

Dans la construction précédente, on a supposé que p n'avait pas de côtés verticaux ni horizontaux. S'il y en avait, alors un des triangles externes aurait été réduit à un segment, et donc p n'aurait pas été joli. Ce n'est pas un problème : si le triangle dégénéré n'est pas l'un des deux premier, on peut le retirer de la construction sans en changer le fonctionnement. Si c'est l'un des deux premiers triangles, on peut le remplacer par un rectangle.

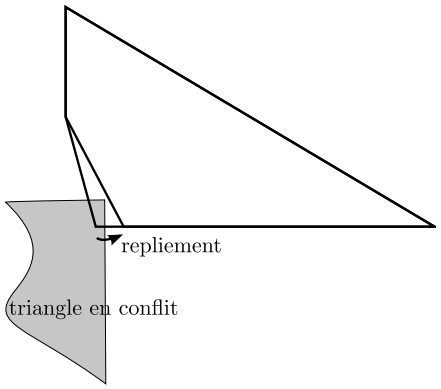


FIG. 7.8 – Une étape de repliement

Intersections non vides entre triangles

On résout le problème des intersections entre triangles en remplaçant les triangles en conflit par des quadrilatères. En particulier, quand un des triangles n'est pas inclus dans p, il intersecte nécessairement d'autres triangles, donc ce cas sera résolu en même temps.

On dit qu'un triangle externe t_1 est *impliqué dans un conflit* avec un autre triangle externe t_2 si t_1 intersecte un triangle interne adjacent à t_2 (ou inversement).

Nous allons modifier nos triangles de façon à éliminer les conflits par le processus suivant : on remplace chacun des triangles externes par des quadrilatères, et l'on résout les conflits en *repliant* ces quadrilatères vers les côtés de p. Pour le reste de la description, on considère un triangle comme un quadrilatère dont deux sommets sont confondus.

Soit $t = (z_0, z_1, z_2)$ un triangle externe, et z_1 son angle droit. On suppose que t est dans le quartier NE. On remplace t par le quadrilatère $q = (z_0, z_1^N, z_1^E, z_2)$, où z_1^N et z_1^E sont deux instances de z_1, que l'on appellera les sommets *mobiles* de q. On dit que z_1^d est de direction d. Le triangle interne au sud de t devient (z_1^E, z_2, z_3), et le triangle interne à l'ouest devient (z_1^N, z_0, z_{-1}), ce qui signifie qu'ils seront affectés par les déplacements à venir de z_1^N et z_1^E pendant le repliage de q. On définit les conflits entre quadrilatères de même que les conflits entre triangle.

Le repliage consiste à répéter le processus suivant tant qu'il reste des conflits dans le polygone : prendre un quadrilatère q qui est impliqué dans un conflit, choisir un de ses sommets mobiles et le déplacer d'une unité dans sa direction, sauf si cela rend q dégénéré. Ce processus est illustré sur la figure 7.8.

Lemme 17. *Pour tout polygone p, $5p$ est rattrapable.*

Dans $5p$, une fois que tous les triangles ont été repliés de sorte que leurs côtés orthogonaux sont de longueur au plus 1, il ne peut plus y avoir de conflits. Il est possible de replier les triangles de la sorte, car $5p$ contient suffisamment de points entiers.

On en déduit le théorème suivant :

Théorème 22. *Pour tout polygone convexe p, il existe un système de signaux dont les dessins de domaine $\mathbb{R}^2$ sont les homothétiques de $5p$. Ce système de signaux satisfait les hypothèse du théorème 9.*

On obtient ainsi les deux corollaires suivants :

Théorème 23. *Soit p un joli polygone, alors il existe un jeu de tuiles qui assemble le langage $\{np | n \in \mathbb{N}\}$*

Théorème 24. *Soit p un polygone convexe, alors il existe un jeu de tuiles qui assemble le langage $\{5np | n \in \mathbb{N}\}$.*

Chapitre 8

Pavages de tout le plan

8.1 Pavages de tout le plan

Dans ce chapitre, nous allons nous rapprocher de thématiques plus classiques dans le monde des pavages de Wang, en ne regardant plus l'assemblage de motifs finis, mais le problème de pavages de tout le plan.

La démarche générale de ce chapitre est la suivante : prenons un jeu de tuiles de Wang. Il est en général difficile, voire impossible, de donner un algorithme *infaillible* —c'est à dire sans backtrack— pour recouvrir le plan avec ces tuiles.

Cependant, pour de nombreux jeux de tuiles ou familles de jeux de tuiles, on connaît de tels algorithmes de pavages. Cependant, ils ne sont en général pas formulés de manière locale : pour décider quelle tuile ajouter à un motif déjà construit, ils examinent ce motif en entier.

Nous allons chercher à formuler des algorithmes de pavages *locaux*, *asynchrones* et *infaillibles*, et qui soient «portés par les tuiles elles-mêmes».

Cette démarche est une extension des travaux visant à exprimer les contraintes de pavages par des règles locales, comme ceux de Mozès [28], ou Goodman-Strauss [20] pour les pavages par substitutions. Avec des règles locales, on obtient un algorithme local qui permet d'engendrer tous les pavages d'un certain type, mais cet algorithme est *faillible* : il est possible, en ajoutant des tuiles suivant les règles locales, d'arriver à une configuration dans laquelle il y a une position où aucune tuile ne peut être ajoutée sans violer les contraintes locales. La situation est d'autant plus grave que pour certains jeux de tuiles [] pourtant munis de règles locales, il n'existe pas d'algorithme (même non-local) pour échapper à cette situation. Nous allons chercher à caractériser les jeux de tuiles qui sont suffisamment simples pour qu'un algorithme infaillible existe, mais qu'il soit aussi local. Un algorithme sera local

s'il consiste à itérer le procéder suivant : choisir une position, examiner ses voisins, et éventuellement y ajouter une tuile.

Les pavages auto-assemblants semblent être un bon outil pour étudier cette localité. En comparant la température 1 et les températures supérieures, nous verrons que le contrôle de l'ordre de construction est une propriété fondamentale qui permet des constructions non triviales. En effet, à température 1, il n'est pas possible d'assembler un ensemble de motifs non-périodiques (théorème 26). Autrement dit, si l'on applique les règles d'adjacence sans s'autoriser à dire «je ne sais pas quelle tuile je dois poser», on ne peut obtenir de manière infaillible que des motifs triviaux.

En revanche, à température 2, et au-dessus, il est possible d'assembler des motifs plus complexes. Comme exemple de production avec une structure complexe, nous expliquerons comment assembler le motif de base du pavage de Robinson.

8.2 Définitions

Dans toute la suite, on considérera un jeu de tuiles de Wang $\mathcal{W}$ qui assemble un ensemble de pavages $\mathcal{M}$. On dit qu'un motif fini m est correct s'il existe un pavage $p \in \mathcal{M}$ tel que $m = p_{|dom(m)}$. Un motif correct est ainsi un motif fini que l'on peut étendre sur tout $\mathbb{Z}^2$. Décider si un motif est correct est évidemment indécidable dans le cas général, mais nous nous intéresserons à *préserver* la correction des motifs.

Commençons par nous donner un cadre d'algorithmes de pavages.

Définition 40. *On appellera* algorithme de pavage *un algorithme $\mathcal{A}$ non-déterministe qui étant donné un motif m correct, et renvoie un couple $(t, z) \in (\mathcal{W}, \mathbb{Z}^2)$ tel que $m \cup (t, z)$ soit encore un motif correct. On notera par abus $\mathcal{A}(m)$ pour $m \cup (t, z)$, ce qui permet d'itérer $\mathcal{A}$. $\mathcal{A}(m)$ est une extension possible de m par $\mathcal{A}$.*

Un tel algorithme $\mathcal{A}$ est couvrant *si pour tout motif m et tout $z \in \mathbb{Z}^2$, il existe une tuile t et un entier n tels que $(t, z) \in \mathcal{A}^n(m)$.*

Cette définition est un peu moins forte que la définition plus classique, où l'algorithme de pavage décide si un motif est correct. Elle est ainsi adaptée à des algorithmes locaux, qui ne peuvent pas examiner tout le motif pour décider s'il est correct. En itérant un algorithme couvrant à partir du motif vide, on obtient un pavage de tout le plan, à condition de choisir les emplacements qui seront couverts de manière équitable. Le motif que l'on obtient alors est une *limite* de $\mathcal{A}$. On dit qu'un algorithme de pavage assemble l'ensemble de ses limites.

Définition 41. *Soit $\mathcal{A}$ un algorithme de pavages, et m_i une suite de motifs tels que $m_{i+1} \in \mathcal{A}(m_i)$. Si $\bigcup_{n\in\mathbb{N}} \mathrm{Dom}(m_n) = \mathbb{Z}^2$, on dit que le motif $m = \bigcup_{n\in\mathbb{N}} m_n$ est une* limite *de $\mathcal{A}$.*

On dira qu'un algorithme $\mathcal{A}$ assemble un ensemble de motifs $\mathcal{P}$ si $\mathcal{A}$ est un algorithme de pavage couvrant et si l'ensemble de ses limites est $\mathcal{P}$.

Si l'on prend une vision «automates cellulaires», un algorithme de pavage est un automate cellulaire dont chaque cellule converge vers une tuile de $\mathcal{W}$. De plus, une fois qu'une cellule est dans l'état $\mathcal{W}$, elle ne change plus d'état. On a donc à la fois la convergence de chaque cellule vers une tuile d'un motif m, mais aussi un affichage du fait que les cellules ont convergé.

Si l'on se donne un système auto-assemblant qui est aussi un algorithme de pavage, la contrainte est encore plus forte, puisque chaque étape de calcul —chaque ajout de tuile— doit correspondre à une tuile ajoutée au motif m. Il n'est donc pas possible d'effectuer des calculs intermédiaires. Cette contrainte est extrêmement forte. Nous allons donc la relâcher légèrement en autorisant une projection. On prend un système auto-assemblant $\mathcal{T}$ et un ensemble de tuiles de Wang $\mathcal{W}$, et on se donne une fonction i des tuiles de $\mathcal{T}$ dans $\mathcal{W}$. On étend i aux motifs de $\mathcal{T}$ en l'appliquant tuile par tuile. On dira donc que $\mathcal{T}$ assemble un ensemble de motifs M sur $\mathcal{W}$ si M est l'image de l'ensemble des limites de $\mathcal{T}$. On a ainsi une sur chaque tuile une couche visible (l'image par i) et une couche cachée de calcul, qui est détruite par i.

Notons que pour qu'un système d'auto-assemblage soit un algorithme de pavage couvrant, il faut que pour toute production p, et tout $z \in \mathbb{Z}^2$, il y ait une production $p \xrightarrow{\mathcal{T}}{}^{*} p'$ telle que $z \in \mathrm{Dom}(p')$. On dit alors que le système est *infaillible*.

8.3 Cas de la température 1

Nous allons considérer des systèmes auto-assemblants à température 1 sans colles de force 0. Dans ce cas, le contrôle sur l'assemblage est très limité, et la condition d'infaillibilité limite beaucoup les possibilités. L'assemblage peut avoir lieu dans n'importe quel ordre, tant que le motif reste connexe. Comme on peut s'y attendre, les seules contraintes que l'on peut assurer dans ces conditions sont périodiques.

Pour toute fonction i sur un ensemble de tuiles, si m est un motif périodique, $i(m)$ est périodique aussi. Dans cette section, nous n'allons donc pas utiliser de projections.

Commençons par montrer que si le système auto-assemblant est localement déterministe, alors l'unique motif qu'il assemble est périodique.

8.3.1 Le cas déterministe

On rappelle qu'un système à température 1 est localement déterministe si pour chaque couleur c et chaque direction d, il existe une unique tuile t telle que $\sigma_d(t) = c$.

Théorème 25. *Soit $\mathcal{T}$ un système auto-assemblant à température 1, avec uniquement des colles de force 1, qui est localement déterministe.*

Si $\mathcal{T}$ est infaillible, il a une unique limite qui est un motif périodique.

Démonstration. Nous allons construire un automate fini a à partir de $\mathcal{T}$, qui reconnaît la limite l de $\mathcal{T}$. Comme a est fini, l sera périodique.

Les états de l'automate a sont les tuiles de W, et ses transitions ont comme étiquettes les directions dans $\{N, S, E, W\}$. Il y a une transition d'une tuile t à une tuile t' avec la direction d si $\sigma_d(t) = \sigma_{d^{-1}}(t')$. L'état initial de a est $\odot$, la source de $\mathcal{T}$. Pour tout mot $w \in \{N, S, E, W\}^*$, on note $a(w)$ l'état de a après qu'il a lu w.

S'il y a une transition de t à t' avec l'étiquette d, alors il est possible d'avoir t' du côté d de t dans une production de $\mathcal{T}$. Plus fort, si la cellule du côté d d'une tuile t est libre dans une production p, alors il est possible d'y ajouter t'. Si tel n'était pas le cas, le système ne serait pas infaillible.

Soit $y \in N$. La tuile qui apparaît en $(0, y)$ dans l est $a(N^y)$. Donc il existe (p, q) tels que $\forall y > q, l(0, y) = l(0, y + p)$. On a la même propriété pour les trois autres demi-axes. Soit $(x, y) \in \mathbb{N}^2$. La tuile en (x, y) est $a(N^y E^x)$. On a donc $\forall (x, y) > (q, q), t(x, y) = t(x, y + p) = t(x + p, y)$.

En allant suffisamment loin dans chaque quartier, on a donc un motif périodique. Il reste à prouver que ces quatre «quarts de périodicité» se recollent au centre. Soit $(0, 0) \leq (x, y) < (q, q)$. Soit $r \geq \lceil p/q \rceil$. On pose $w_1 = N^{rp} E^{rp} N^y E^x S^{rp} W^{rp}$, et $w_2 = N^{rp} E^{rp} N^y E^x S^{rp} W^{rp}$. Ces deux mots amènent a dans le même état, donc $l(x, y) = l(x, y + p)$. Par symétrie, l est bien périodique. □

8.3.2 Le cas général

Dans le cas général, il est possible d'utiliser le non-déterminisme pour assembler des motifs qui ne soient pas périodiques. On peut par exemple se donner un ensemble de colles C, et prendre comme ensemble de tuiles C^4 tout entier. On obtient clairement un système infaillible qui a des limites non-périodiques, puisque tous les motifs de C^4 respectant les contraintes de voisinages sont des limites.

Nous allons munir les tuiles de $\mathcal{T}$ d'une relation d'équivalence qui nous permettra d'obtenir des motifs périodiques. Cette relation d'équivalence entre

tuiles dit qu'à une distance 2, deux tuiles équivalentes sont impossibles à distinguer. Le résultat que l'on obtient peut donc se lire : «les motifs assemblés par un système à température 1 sont des coloriages arbitraires d'un motif périodique».

Définition 42. *Soit $\mathcal{M}$ un ensemble de motifs sur l'alphabet Σ, deux tuiles t_1, t_2 sont équivalentes ($t_1 \equiv_{\mathcal{M}} t_2$) si pour tout $m \in \mathcal{M}$ tel que $m(0,0) = t_1$, il existe un $m' \in \mathcal{M}$ tel que $m'(0,0) = t_2$ et pour $|x,y|_1 > 1$, $m'(x,y) = m(x,y)$.*

On étend $\equiv$ aux motifs, en disant que deux motifs sont équivalents s'ils sont équivalents tuile à tuile. On peut ainsi associer à tout motif $m \in \mathcal{M}$ un pavage sur $\Sigma/\equiv_{\mathcal{M}}$ défini par $\tilde{m}(z)$ est la classe d'équivalence modulo $\equiv$ de $m(z)$. On peut aussi définir $\tilde{\cdot}$ sur les classes de $/\equiv$.

Théorème 26. *Soit $\mathcal{T}$ un système auto-assemblant à température 1 avec uniquement des colles de force 1. L'ensemble $\mathcal{L}$ de ses limites est tel que $\mathcal{L}/\equiv$ est réduit à une unique classe d'équivalence. Pour tout $l \in \mathcal{L}$, $\tilde{l}$ est un pavage périodique.*

Démonstration. On peut reprendre la construction de l'automate a, mais sans l'hypothèse déterministe, on obtient un automate non-déterministe. Nous allons montrer qu'en déterminisant a, on obtient un automate dont les états sont les classes d'équivalence de $\equiv$.

On définit $\bar{a}$, la version déterminisée de a en posant que $\bar{a}(w)$ est l'ensemble des états de $a(w)$.

Soit s un état de $\bar{a}$, et $t_1, t_2 \in s$. Soit p un pavage tel que $p(u,v) = t_1$. Comme $t_1, t_2 \in s$, il existe deux chemins (c^1, c^2) de $(0,0)$ à (u,v), tels que $t_i \in a(c^i)$. Supposons, sans perte de généralité que ces deux chemins ne s'intersectent pas. En itérant a sur c^1 et c^2, on obtient deux suites de transitions τ_1 et τ_2 qui aboutissent respectivement à poser t_1 et t_2 en (x,y).

On considère une suite de production qui procède dans l'ordre suivant : d'abord elles suit les suites de transitions τ_i, puis couvre le reste du carré de taille $(|u| + 2 \times |v| + 2)$ centré en $(0,0)$ mais laissent les positions voisines de (u,v) vides.

À partir de cette production, il est possible de poser soit t_1 soit t_2 en (x,y), et dans les deux cas, il est possible de continuer à paver le plan. Donc $t_1 \equiv t_2$.

Réciproquement, si $t_1 \equiv t_2$, on veut prouver que les deux tuiles apparaissent dans le même état de $\bar{a}$, c'est à dire qu'elles peuvent apparaître au même endroits dans deux limites de $\mathcal{T}$. C'est possible par définition de $\equiv$.

Les états de $\bar{a}$ sont donc les classes d'équivalence de $\equiv$, d'où le théorème. □

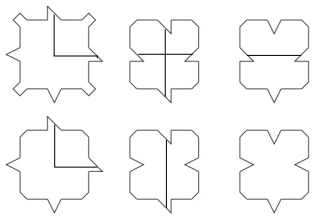


FIG. 8.1 – Les six tuiles de Robinson

La réciproque de ce théorème s'obtient facilement : étant donné un motif m qui est (p, q)-périodique, il suffit de pq tuiles pour assembler m à température 1.

8.4 Assemblage du pavage de Robinson

À température 2, il devient possible d'utiliser l'ordre de pose des tuiles pour faire du calcul. On obtient ainsi des motifs bien plus riches qu'à température 1. Ainsi, Rothemund, Papadakis et Winfree [29] ont construit un système à température 2 qui assemble le triangle de Sierpinski. Nous n'avons pas de caractérisation des motifs auto-assemblable à température 2. Nous allons cependant montrer qu'il est possible d'obtenir un ensemble de motifs quasi-périodiques stricts, l'ensemble des pavages de Robinson.

Définition 43 (Quasi-périodicité)**.** *Un motif $m \in \mathcal{W}^{Z^2}$ est quasi-périodique s'il existe une fonction $q : \mathbb{N} \mapsto \mathbb{N}$ telle que pour tout motif de taille $n \times n$ qui apparaît dans m apparaît dans tout carré de taille $q(n) \times q(n)$ extrait de m.*

8.4.1 Le pavage de Robinson

Le jeu de tuiles $\mathcal{R}$ de Robinson se compose des 6 tuiles de la figure 8.1, ainsi que de leurs images par rotation de $k\pi/2$ et par symétrie. Nous allons décrire rapidement les motifs qu'il assemble. On trouvera dans [] ou [] une étude plus détaillée de ce pavage.

Si l'on regarde un pavage par le jeu de tuiles $\mathcal{R}$ en ne considérant que les lignes dessinées sur les tuiles et non les tuiles elles-mêmes, on obtient un motif comme celui de la figure 8.2.

Structure récursive des motifs On appelle carré une ligne fermée, comme indiqué sur la figure 8.2. Le motif consiste en des carrés entrelacés, de différentes tailles. Le centre d'un carré de rang n est le sommet d'un carré de

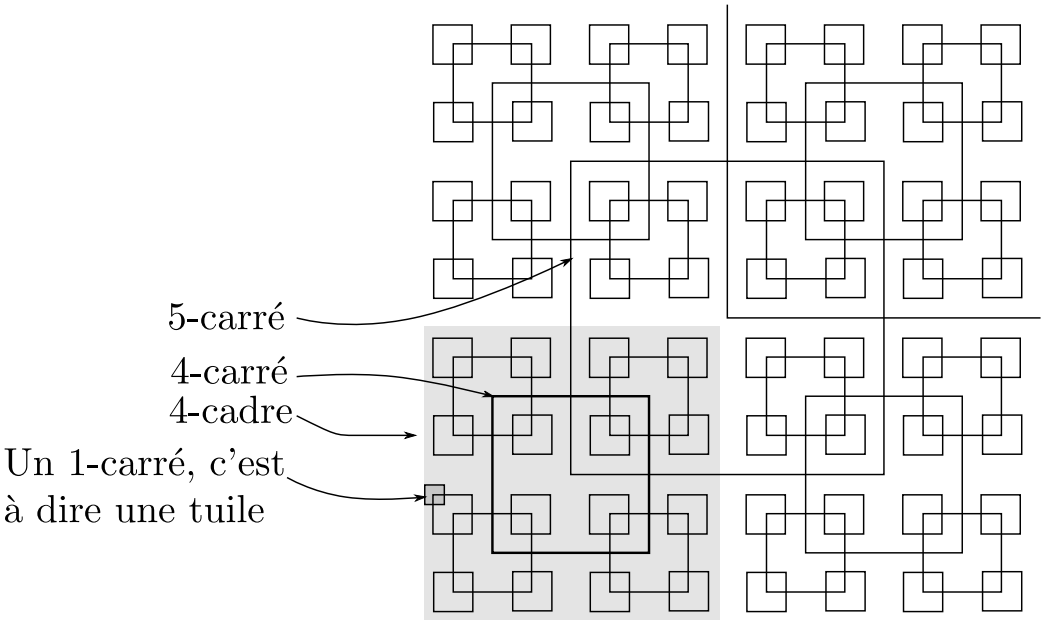


FIG. 8.2 – Un motif de $\mathcal{R}$.

rang $n+1$. Chaque n-carré est contenu dans un n-cadre, qui contient l'ensemble des tuiles qui «dépendent» de ce carré. Chaque n-cadre contient 4 $n-1$-cadres, dont les centres sont les sommets du n-carré correspondant au cadre. Chaque cadre contient un L en son centre. On attribue à chaque cadre une *direction*, qui est celle du L en son centre. Ainsi, un n-cadre NE occupe le quart Nord-est du $n+1$-cadre qui le contient.

Ces motifs ont une structure récursive : on obtient un $n+1$-cadre à partir d'un n-cadre en choisissant la direction du $n+1$-cadre, c'est à dire la position du L en son centre. La position relative des deux cadres dépend de l'orientation du n-cadre. Connaissant cette position, il suffit de remplir les trois autres quartiers, les 4 segments de la croix centrale et le centre du $n+1$-cadre. Les trois quartiers sont obtenus par symétrie, et le reste ne dépend que de la direction du $n+1$-cadre.

Dans ce procédé récursif, si l'on choisit la même direction pour tous les cadres, on ne couvre qu'un quart de plan. Dans ce cas, le motif peut être complété par un L infini autour de ce quart de plan, et trois motifs similaires. De même dans le cas où l'on ne couvre qu'un demi-plan.

Le Nord-ouest déterminisme Le pavage de Robinson a ceci de particulier qu'il peut s'exprimer sous la forme d'un jeu de tuiles *nord-ouest-déterministe*. On peut ajouter des marques aux tuiles de telle façon que

connaissant les tuiles en $(1, 0)$ et $(-1, 0)$, il y a au plus une tuile qui puisse convenir en $(0, 0)$. La construction du jeu de tuiles ayant cette propriété est assez délicate, et on se reportera à [24] pour plus de détails. On obtient en fait une propriété plus forte, le 4-déterminisme, mais dans cette construction, nous n'utiliserons que le fait que ce jeu de tuiles est à la fois nord-ouest, nord-est, sud-ouest et sud-est-déterministe.

On peut grâce à cette propriété obtenir une bonne partie de notre algorithme de pavage : à partir du moment où l'on connaît une tuile dans la colonne x et une tuile dans la ligne y, il suffit de propager les contraintes locales pour déterminer la taille en (x, y). Le vrai problème est donc de poser la première tuile de chaque colonne ou ligne.

8.4.2 Auto-assemblage du pavage de Robinson

Théorème 27. *Il existe un système auto-assemblant $\mathcal{T}$ à température 2 qui assemble l'ensemble des pavages de Robinson sans ligne infinie.*

Nous obtenons par ce théorème un algorithme local et sans backtrack pour assembler des motifs strictement quasi-périodiques. L'information globale qui permet de s'orienter dans un motif quasi-périodique est en fait présente non pas dans les tuiles, mais dans leur ordre de pose.

Notons que l'on obtient aussi les pavages avec une ligne infinie, mais comme motifs couvrant un quart de plan ou un demi-plan : ce ne sont pas des limites de l'assemblage comme défini plus haut.

Avant de passer à la preuve, on notera que le pavage de Robinson est un exemple de pavage de Wang obtenu à partir d'une règle de substitution. Le fonctionnement de $\mathcal{T}$ fait appel au fonctionnement précis des règles, et en particulier à la possibilité de les rendre 4-*déterministes* [24]. Cependant, l'idée de la preuve, qui est d'utiliser une charpente en signaux exploitant les symétries issues de la substitution reste valide pour d'autres pavages par substitution. Cependant, un gros travail technique est nécessaire pour l'appliquer, ou pour obtenir une généralisation du type de [28] pour tous les pavages par substitution sur $\mathbb{Z}^2$.

Le système $\mathcal{T}$ que nous allons construire est un système en deux couches : une couche $\mathcal{R}$ contient la version déterministe du jeu de tuiles de Robinson, et une deuxième couche, $\mathcal{F}$, la *fondation*, qui va guider la construction de façon à synchroniser les décisions. L'ensemble des tuiles de $\mathcal{T}$ sera un sous-ensemble de $\mathcal{F} \times \mathcal{R}$. Comme $\mathcal{R}$ est un simple jeu de tuiles de Wang, les forces colles seront données par la composante $\mathcal{F}$. Nous allons décrire $\mathcal{F}$ par un système de signaux.

Nous allons utiliser $\mathcal{F}$ pour construire une hiérarchie de carrés qui correspondent à des cadres dans le pavage de Robinson. On couvrira ainsi tout le plan. Dans un motif de Robinson, on appelle cadres *principaux* les cadres qui contiennent $(0, 0)$, et cadres *secondaires* les autres. La partie *propre* d'un cadre principal est celle qui n'est pas comprise dans des cadres principaux de rang inférieur.

Comme $\mathcal{R}$ est déterministe dans toutes les directions diagonales, quand une tuile est ajoutée au motif dans une direction diagonale, elle ne peut pas introduire d'erreurs dans le motif. Ainsi, pour toute tuile $t \in \mathcal{F}$ qui n'a pas de colles de force 2 on met toutes les tuiles de $\{t\} \times \mathcal{R}$ sont dans $\mathcal{T}$.

Le problème est donc de deviner la valeur de la composante $\mathcal{R}$ quand il y a des colles de force 2. Cependant, notons que sur la diagonale d'un cadre, les tuiles sont symétriques par rapport à cette diagonale. Elles ne dépendent donc que d'un voisin et de l'information «cette tuile est sur la diagonale d d'un cadre de direction d'».

Enfin, notons que la direction d'un cadre est indiquée par le type de la tuile au milieu d'un des côtés du cadre. La première tuile que nous poserons dans un cadre sera donc le milieu d'un de ses côtés.

Lemme 18. *Il existe un système de signaux terminant et temps-cohérent $\mathcal{S}$ et une bijection ϕ entre l'ensemble des motifs de Robinson sans ligne infinie et l'ensemble des dessins enracinés de domaine $\mathbb{R}^2$ de $\mathcal{S}$ telle que pour tout dessin d de domaine $\mathbb{R}^2$,*

- *les signaux de type V ou H sont localisés*
  - *sur les diagonales des cadres secondaires de $\phi(d)$,*
  - *ou sont des signaux de longueur 1, deux par cadre principal, au milieu des côtés de ce cadre principal*
- *Le type de chacun des éléments dans la partie propre d'un cadre principal reflète l'orientation de ce cadre*
- *le centre des cadres secondaires contient une collision dont le type dépend de l'orientation de ce cadre secondaire*
- *il y a des signaux sur la croix centrale de chaque cadre principal.*

Ce lemme, qui donne la structure de $\mathcal{F}$, est illustré sur la figure 8.3.

Avant de détailler la preuve de ce lemme, voyons en quoi il permet d'obtenir le théorème 27. Supposons ce lemme acquis, ce qui nous permet de définir le système auto-assemblant $\mathcal{F}$. Pour obtenir $\mathcal{T}$ à partir de $\mathcal{F}$, il faut choisir le bon sous-ensemble de $\mathcal{F} \times \mathcal{R}$. Il y a quatre types de tuiles dans $\mathcal{F}$:

- Les tuiles qui seront attachées par deux colles de force 1, pour lesquelles $\{t\} \times \mathcal{R} \subset \mathcal{T}$,
- les tuiles sur les diagonales de cadres secondaires. Pour ces tuiles, il suffit de connaître un voisin, et le type de signal qui les porte, ce qui

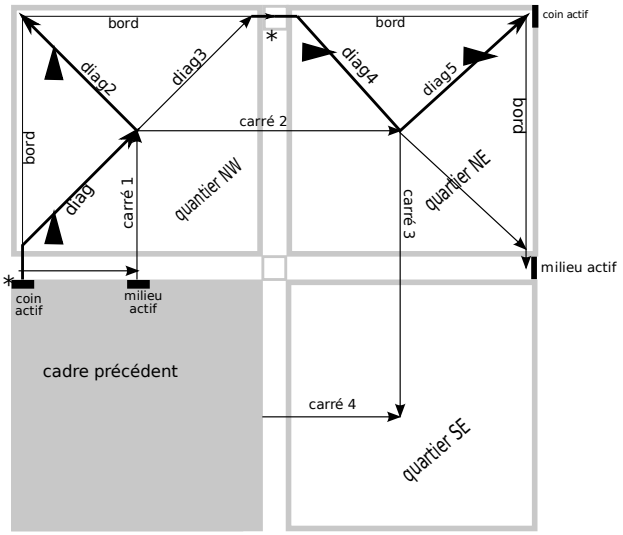


FIG. 8.3 – La construction d'un cadre principal par $\mathcal{F}$. Les signaux qui ne sont pas marqués d'une étoile se répètent : ils ont une collision telle que $c^+ = c^- = \{s\}$

indique la direction du cadre, et celle de la diagonale. Ainsi, étant donné une tuile $f \in \mathcal{F}$, le signal auquel elle appartient et un voisin sur le côté où elle a une colle de force 2, on peut choisir un unique $r \in \mathcal{R}$ et ajouter (f, r) à $\mathcal{T}$,

- les tuiles au centres des cadres secondaires sont un cas particulier du cas précédent : leurs voisins ne sont pas symétriques, mais on peut deviner un des voisins à partir du type du cadre principal qui les contient. Il y a donc aussi une unique tuile qui convient et pour laquelle $(f, r) \in \mathcal{T}$;
- le dernier cas est celui des milieux des cadres principaux, c'est là qu'a lieu le choix de l'orientation de ces cadres. On ajoute donc à $\mathcal{T}$ toutes les tuiles compatible avec l'unique voisin du côté de la colle de force 2.

Preuve du lemme 18. Le fonctionnement de $\mathcal{T}$ est résumé sur la figure 8.3. On construit récursivement un cadre de chaque taille, chacun inclus dans le suivant. Ces cadres seront les cadres principaux du pavage complet. Pour pouvoir faire cette construction récursive, il faut donc pouvoir construire un $n+1$-cadre à partir d'un n-cadre.

On se donne une famille de signaux, chacun ayant comme paramètre soit une direction soit une paire de directions, et éventuellement quelques données supplémentaires. La première direction est celle du $n+1$-cadre dont le signal permet la construction.

`border` marque la limite de chaque $n+1$-cadre.

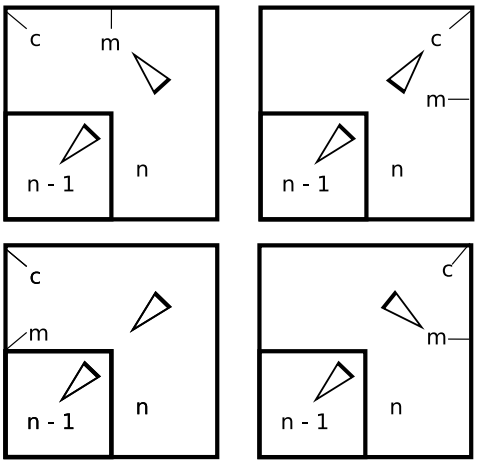


FIG. 8.4 – La position des signaux active corner (c) et active middle (m) en fonction de l'orientation des deux derniers cadres. Le choix de ces positions est arbitraire, à condition de placer m et c sur un côté pointé par le cadre, et d'éviter le cadre précédent.

`square` permet de trouver les centres des n-cadres secondaires, de même que `diag`, `diag2`, `diag3` et `diag4`. Les signaux `diag*` se trouvent sur les diagonales des n-cadres secondaires, et ils ont comme paramètre l'orientation du n-cadre, ainsi que celle du $n+1$-cadre. Ce sont ces signaux qui seront H ou V, conformément au lemme 18. La figure 8.3 indique lesquels sont effectivement H ou V.

Le signal `active corner` lance la construction du $n+1$-cadre. La position de la collision qui a ce signal sortant dépend du type du n-cadre précédent. Il est sur le coin $d+\pi/2$ du cadre précédent, dont d est la direction. Dans le cas de deux cadres successifs de directions opposées, un tel placement n'est pas possible, et on place `active corner` à l'autre extrémité du même côté. Ce signal a comme paramètre la direction du nouveau $n+1$-cadre, et passe ce paramètre aux autres signaux. Le choix de la direction par ce signal est la seule source de non-déterminisme de la construction. De ce signal dépend le type des autres signaux, mais aussi la position de `active corner` et `active middle` pour le *prochain* cadre. Les différents cas sont indiqués sur la figure

On vérifie que ce système de signaux est aéré et temps-cohérent par induction : si un cadre principal vérifie ces conditions, le $n+1$-cadre que l'on construit au-dessus les vérifie aussi. □

À partir de ce lemme, on obtient par le système $\mathcal{F}$. Par les remarques précédentes, on peut extraire de $\mathcal{F} \times \mathcal{R}$ un sous-ensemble qui convient. On obtient ainsi $\mathcal{T}$.

Chapitre 9

Conclusion et perspectives

Les résultats que nous avons présentés montrent qu'une vision géométrique de l'auto-assemblage permet une analyse différente de l'auto-assemblage, qui complète les outils développés jusqu'à présent. En s'affranchissant de la contrainte d'avoir des systèmes déterministes, l'étude de la dynamique de l'assemblage devient primordiale. La condition d'ordre, qui n'était qu'un outil dans la preuve du déterminisme des constructions devient un objet d'étude en soi et une source de constructions propres au modèle d'auto-assemblage.

S'éloigner de la simulation du calcul classique pour saisir ce que l'auto-assemblage a de spécifique nous a mené aux systèmes de signaux, qui nous ont rapproché d'un langage de programmation géométrique. Nous obtenons ainsi des constructions dont le comportement est plus facile à prouver. Ce langage de signaux nous permet de nous attaquer à deux problèmes restés en friche dans l'étude de l'auto-assemblage : la possibilité de faire de la géométrie discrète, et la possibilité d'assembler des motifs quasi-périodiques.

Nous obtenons une autre manière de réduire la complexité de l'auto-assemblage en définissant une notion de transformation qui respecte la dynamique. Cette notion est importante pour trois raisons. Le résultat d'impossibilité tout d'abord, confirme l'importance de la condition d'ordre pour avoir des constructions qui se comportent bien. Une construction qui n'est pas ordonnée est en effet dépendante de phénomènes de synchronisation qui sont dépendants de l'échelle. Ensuite, dans le cas ordonné, il est possible d'effectuer une homothétie sur la dynamique. Or c'est justement par des homothéties sur la dynamique que les constructions déterministes ont pu être rendues robustes. Cette possibilité de faire des homothéties confirme donc que le cadre non-déterministe est viable. Enfin, la possibilité d'effectuer une transformation géométrique indépendamment de l'assemblage sous-jacent signifie qu'il est possible d'avoir un langage de programmation avec des fonctions pour l'auto-assemblage.

C'est encore en étudiant la géométrie de l'assemblage que nous avons pu créer des systèmes auto-assemblants optimaux pour certains langages simples comme les carrés, les rectangles ou les losanges. Cette démarche nous a également permis d'étudier le cas tridimensionnel, où la possibilité de contourner des morceaux d'assemblage rend une étude non fondée sur l'ordre de construction délicate.

Pour rapprocher les *pavages* auto-assemblants de leur racines que sont les pavages de Wang, nous avons donné quelques éléments sur les pavages du plan obtenus par auto-assemblage. Dans le cadre des pavages du plan, il n'est plus possible de changer d'échelle pour se donner de la place pour calculer : on n'a plus besoin d'une aire de calcul, mais d'une densité de calcul. Cependant, quand il y a peu de calcul, comme dans le cas du motif de base du pavage de Robinson, l'auto-assemblage permet d'obtenir des résultats dont la géométrie est intéressante, ici un motif quasi-périodique. On peut essayer de revisiter ce résultat à la lumière de ceux de Goodman-Strauss [20] sur les pavages par substitution. En effet, nous sommes partis d'un pavage par substitution, le pavage de Robinson, et avons obtenu des règles locales qui constituent un algorithme de pavage. Dans [20], on part d'une substitution pour obtenir des règles locales qui constituent un algorithme de vérification du pavage. La tentation est forte d'essayer d'affiner ces règles locales obtenues à partir d'une substitution —presque— quelconque pour obtenir un système d'auto-assemblage.

Cette étude est bien sûr loin d'avoir épuisé le sujet, et il reste beaucoup à faire. Le problème le plus évident est celui de l'influence de la grille sous-jacente. Que se passe-t-il pour l'auto-assemblage sur la grille hexagonale ? Quelles sont les techniques qui se transfèrent sur d'autres grilles, et celles qui sont liées à $\mathbb{Z}^2$, voire $\mathbb{Z}^2$ avec le voisinage de Von Neumann. En particulier, dans $\mathbb{Z}^2$ à température 2, dans une production ordonnée, l'ordre est toujours très simple, ce qui facilite la preuve qu'un système est ordonné.

Quant aux signaux, ils pourront probablement servir de base à un langage de programmation pour l'auto-assemblage, mais ils restent assez bas-niveau. Peut-on trouver une formulation plus fonctionnelle qui permette de passer d'une description simple d'un langage de formes à un système auto-assemblant pour l'obtenir ? D'autre part, l'utilisation des signaux implique de se restreindre aux techniques qu'ils permettent d'exprimer. En particulier, la simulation d'automates cellulaires par les signaux ne se fait pas du tout de manière abstraite. C'est un manque qu'il faut combler pour concilier méthodes géométriques et calcul.

C'est sur les pavages de tout le plan que les questions les plus profondes restent ouvertes. Quand on regarde le plan tout entier, la puissance de l'auto-assemblage n'est plus celle de Turing. Il faut donc établir les limites de l'auto-

assemblage dans ce cadre. On peut supposer l'existence de deux types de limites : des limites en termes de complexité et en termes de géométrie et de synchronisation.

Les limites en termes de complexité viennent du constat qu'en auto-assemblage, pour faire progresser le calcul, on est tenus de poser des tuiles, donc de s'engager sur une partie des résultats du calcul. Ainsi, si l'on considère un jeu de tuiles dont l'ensemble des carrés qui apparaissent dans un pavage de tout le plan a une complexité importante (par exemple `EXPTIME`-complet), alors il faut poser 2^n tuiles avant de finir le premier carré $n \times n$ complet. Cela signifie que les 2^n tuiles forment un graphe de treewidth au plus n, ce qui semble limiter le calcul qui a pu être effectué par ces 2^n tuiles, et compromettre l'assemblage.

Les limites en terme de géométrie viennent du constat que l'on ne peut pas croiser d'informations. Cela semble limiter les interdépendances possibles entre les choix qui se font à des endroits distincts. Il doit donc exister des motifs simples d'un point de vue de théorie de la complexité qui ne sont pas auto-assemblable. Pour les trouver, il peut être utile de regarder un motif auto-assemblé comme la trace d'un protocole de synchronisation, et chercher comment exprimer dans ce formalisme des résultats d'algorithmique concurrente.

Le lien entre ces deux types de limitations semble indiquer que la puissance de calcul dépend de la géométrie. Le choix de la grille sous-jacente n'est donc pas anodin. À l'extrême, je conjecture que sur une grille hyperbolique non-planaire, il n'y a pas de différence entre calcul Turing et auto-assemblage : la croissance de la grille assure que l'on a la place pour calculer, et surtout pour transmettre les résultats des calculs, et la non-planarité permet de croiser les informations librement.

Bibliographie

[1] *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2006, Miami, Florida, USA, January 22-26, 2006*. ACM Press, 2006.

[2] Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors. *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part I : Tack A : Algorithms, Automata, Complexity, and Games*, volume 5125 of *Lecture Notes in Computer Science*. Springer, 2008.

[3] Len Adleman, Qi Cheng, Ashish Goel, Ming-Deh Huang, David Kempe, Pablo Moisset de Espanès, and Paul Wilhelm Karl Rothemund. Combinatorial optimization problems in self-assembly. In *STOC '02 : Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 23–32, 2002.

[4] Leonard M. Adleman, Qi Cheng, Ashish Goel, and Ming-Deh A. Huang. Running time and program size for self-assembled squares. In *STOC*, pages 740–748, 2001.

[5] Cyril Allauzen and Bruno Durand. *Classical decision problems*, chapter Tiling Problems. Springer-Verlag, 1997.

[6] S. Arun-Kumar and Naveen Garg, editors. *FSTTCS 2006 : Foundations of Software Technology and Theoretical Computer Science, 26th International Conference, Kolkata, India, December 13-15, 2006, Proceedings*, volume 4337 of *Lecture Notes in Computer Science*. Springer, 2006.

[7] Florent Becker. Transformations and preservation of self-assembly dynamics through homotheties. In Martín-Vide et al. [26], pages 101–112.

[8] Florent Becker. Picture worth a thousand tiles, a geometric programming language for self-assembly. *Theoretical Computer Science*, À paraître, 2009.

[9] Florent Becker, Ivan Rapaport, and Eric Rémila. Self-assemblying classes of shapes with a minimum number of tiles, and in optimal time.

In *LNCS proceedings of Found. of Software Tech and Theo Comp Sci*, pages 45–56, 2006.

[10] Florent Becker, Éric Rémila, and Nicolas Schabanel. Time optimal self-assembly of 2d and 3d shapes : the case of squares and cubes. In *proceedings DNA14, 14th International Meeting on DNA Computing*. LNCS, to appear in 2008.

[11] R. Berger. The undecidability of the domino problem. *Memoirs of the american mathematical society*, 1966.

[12] Yuriy Brun. Solving np-complete problems in the tile assembly model. *Theor. Comput. Sci.*, 395(1) :31–46, 2008.

[13] Junghuei Chen, Natasha Jonoska, and Grzegorz Rozenberg, editors. Natural Computing. Springer, 2006.

[14] Junghuei Chen and John H. Reif, editors. *DNA Computing, 9th International Workshop on DNA Based Computers, DNA9, Madison, WI, USA, June 1-3, 2003, revised Papers*, volume 2943 of *Lecture Notes in Computer Science*. Springer, 2004.

[15] S. Barry Cooper, Benedikt Löwe, and Leen Torenvliet, editors. *New Computational Paradigms, First Conference on Computability in Europe, CiE 2005, Amsterdam, The Netherlands, June 8-12, 2005, Proceedings*, volume 3526 of *Lecture Notes in Computer Science*. Springer, 2005.

[16] B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 2002.

[17] Jérôme Durand-Lose. Abstract geometrical computation : Turing-computing ability and undecidability. In Cooper et al. [15], pages 106–116.

[18] Nadrian C. Seeman. Erik Winfree, Xiaoping Yang. Universal computation via self-assembly of dna : Some theory and experiments. *DNA Based Computers II*, 27 of DIMACS :191–213, 1998.

[19] Claudio Ferretti, Giancarlo Mauri, and Claudio Zandron, editors. *DNA Computing, 10th International Workshop on DNA Computing, DNA 10, Milan, Italy, June 7-10, 2004, Revised Selected Papers*, volume 3384 of *Lecture Notes in Computer Science*. Springer, 2005.

[20] Chaim Goodman-Strauss. Matching rules and substitution tilings. *Annals of Mathematics*, 147 :181–223, 1998.

[21] Y. Gurevich and E. Gradel. *Classical decision problems*. Springer-Verlag, 1997.

[22] Ming-Yang Kao and Robert T. Schweller. Reducing tile complexity for self-assembly through temperature programming. In *SODA* [1], pages 571–580.

[23] Ming-Yang Kao and Robert T. Schweller. Randomized self-assembly for approximate shapes. In Aceto et al. [2], pages 370–384.

[24] Jarkko Kari and P Papasoglu. Deterministic aperiodic tile sets. *Geom. Func. Anal.*, 9(2) :353–369, 1999.

[25] Chengde Mao and Takashi Yokomori, editors. *DNA Computing, 12th International Meeting on DNA Computing, DNA12, Seoul, Korea, June 5-9, 2006, Revised Selected Papers*, volume 4287 of *Lecture Notes in Computer Science*. Springer, 2006.

[26] Carlos Martín-Vide, Friedrich Otto, and Henning Fernau, editors. *Language and Automata Theory and Applications, Second International Conference, LATA 2008, Tarragona, Spain, March 13-19, 2008. Revised Papers*, volume 5196 of *Lecture Notes in Computer Science*. Springer, 2008.

[27] Jacques Mazoyer and Veronique Terrier. Signals in one-dimensional cellular automata. *Theor. Comput. Sci.*, 217(1) :53–80, 1999.

[28] Shahar Mozes. substitution systems and dynamical systems generated by them. *J. Analyse Math.*, (53) :139–186, 1989.

[29] Erik Winfree. Paul W.K. Rothemund, Nick Papadakis. Algorithmic self-assembly of dna sierpinski triangles. *PLoS Biology*, 2004.

[30] Raphael M. Robinson. Undecidability and nonperiodicity for tilings on the plane. *Inventiones Mathematicae*, 12(3) :177–209, september 1971.

[31] Paul W. K. Rothemund. Design of dna origami. In *ICCAD*, pages 471–478. IEEE Computer Society, 2005.

[32] Paul W. K. Rothemund and Erik Winfree. The program-size complexity of self-assembled squares (extended abstract). In *STOC*, pages 459–468, 2000.

[33] Paul W.K. Rothemund. *Theory and Experiments in Algorithmic Self-Assembly*. PhD thesis, University of Southern California, 2001.

[34] Sudheer Sahu and John H. Reif. Capabilities and limits of compact error resilience methods for algorithmic self-assembly in two and three dimensions. In Mao and Yokomori [25], pages 223–238.

[35] Rebecca Schulman and Erik Winfree. Programmable control of nucleation for algorithmic self-assembly. In Ferretti et al. [19], pages 319–328.

[36] David Soloveichik, Matthew Cook, and Erik Winfree. Combining self-healing and proofreading in self-assembly. *Natural Computing*, 7(2) :203–218, 2008.

[37] David Soloveichik and Erik Winfree. Complexity of self-assembled shapes. In Ferretti et al. [19], pages 344–354.

[38] Erik Winfree. On the computational power of DNA annealing and ligation. In *DNA Based Computers*, pages 199–210, 1995.

[39] Erik Winfree. *Algorithmic Self-Assembly of DNA*. PhD thesis, Caltech, 1998.

[40] Erik Winfree. Simulations of computing by self-assembly. In *DNA Based Computers IV*, 1998.

[41] Erik Winfree. Nanotechnology : Science and computation. In Chen et al. [13], chapter Self-healing tilesets.

[42] Erik Winfree and Renat Bekbolatov. Proofreading tile sets : Error correction for algorithmic self-assembly. In Chen and Reif [14], pages 126–144.

[43] Erik Winfree, Furong Liu, Lisa Wenzler, and Nadrian C. Seeman. Design and self-assembly of two-dimensional dna crystals. *Nature*, 1998.

[44] Éric Rémila et Ivan Rapaport. Self-assemblying (classes of) shapes with a constant number of tile. Technical report, LIP, ÉNS Lyon., 2004.