

Comportements typiques dans les automates cellulaires

Laurent Boyer

▶ **To cite this version:**

Laurent Boyer. Comportements typiques dans les automates cellulaires. Mathématique discrète [cs.DM]. Université de Grenoble, 2010. Français. NNT : . tel-00862704

HAL Id: tel-00862704
https://theses.hal.science/tel-00862704

Submitted on 17 Sep 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

Docteur de l'Université de Grenoble

Spécialité: Mathématique Informatique

Arrêté ministériel: 7 août 2006

Présentée et soutenue publiquement par

Laurent BOYER

le 7 décembre 2010

Comportements typiques dans les automates cellulaires

Thèse dirigée par **M. Laurent Vuillon** et codirigée par **M. Guillaume Theyssier**

JURY

M. Jean Mairesse	Directeur de Recherche CNRS	Rapporteur
M. Enrico Formenti	Professeur Univeristé de Nice-Sophia Antipolis	Rapporteur
M. Julien Cervelle	Professeur Université Paris-Est Créteil Val-de-Marne	Président
M. Martin Kutrib	Professeur Universität Gießen	Examinateur
M. Laurent Vuillon	Professeur Université de Savoie	Examinateur
M. Guillaume Theyssier	Chargé de Recherche CNRS	Examinateur

Thèse préparée au sein du *Laboratoire de Mathématique de l'Université de Savoie (LAMA)* dans l'École Doctorale *Mathématiques, Sciences et Technologies de l'Information, Informatique*.

Table des matières

Introduction **5**
Quelques éléments historiques . . . 6
Contenu du manuscrit . . . 7

1 Définitions **9**
1 Automates cellulaires . . . 9
1.1 Introduction syntaxique . . . 9
1.2 Point de vue topologique . . . 14
1.3 Propriétés des fonctions globales . . . 15
1.4 Ensembles limites d'AC . . . 16
1.5 Quelques généralités sur les sous-shifts . . . 18
2 Classifications . . . 19
2.1 La classification de Wolfram . . . 19
2.2 Classifications topologiques . . . 20
3 Simulations et universalités . . . 21
3.1 Transformations spatio-temporelles . . . 22
3.2 Sous-automate et simulation intrinsèque . . . 23
3.3 Simulation par facteur . . . 25
4 Sous-familles . . . 26
4.1 Automates additifs . . . 26
4.2 Automates cellulaires captifs . . . 27
4.3 AC totalistiques . . . 27

2 AC multi-ensemblistes **31**
1 Définitions préliminaires . . . 32
1.1 Multi-ensembles . . . 32
1.2 Langages et sous-shifts multi-ensemblistes . . . 34
2 Les automates cellulaires multi-ensemblistes . . . 35
2.1 Définition . . . 35
2.2 AC multi-ensemblistes et simulation . . . 36
2.3 AC multi-ensemblistes et AC totalistiques . . . 38
2.4 La nilpotence des AC multi-ensemblistes . . . 40
3 Changements d'échelle et universalité multi-échelles . . . 40
3.1 Cas unidimensionnel . . . 41
3.2 Cas multi-dimensionnel . . . 44
4 Outer-multi-ensemblistes . . . 47

4.1 Définition . . . 48
4.2 Rééchelonné d'un outer-multi-ensembliste . . . 49

3 Densité de propriétés 51
1 Densité de propriétés . . . 52
1.1 Formalisme . . . 52
1.2 Complexité de Kolmogorov et dénombrement . . . 59
1.3 Quelques objets combinatoires usuels . . . 63
2 Parmi les AC quelconques . . . 65
2.1 Propriétés syntaxiques, propriétés dépendant d'une sous-partie de la table . 65
2.2 Propriétés dynamiques . . . 72
3 Parmi des sous-familles d'AC définies syntaxiquement . . . 80
3.1 Méthode générale . . . 81
3.2 À ensemble d'états croissant . . . 83
3.3 À taille de voisinage croissante . . . 85
3.4 Cas les plus généraux . . . 90

4 Propriétés des ensembles μ-limites 99
1 Définitions . . . 100
1.1 μ-nilpotence . . . 100
1.2 Murs et sensibilité . . . 102
1.3 Langages μ-limites des automates cellulaires non sensibles . . . 103
2 Résultats de calculabilité . . . 104

5 Universalité par facteurs 115
1 Universalité parmi les AC surjectifs 2D . . . 116
2 Universalité parmi les AC persistants 1D . . . 116
2.1 Les AC persistants . . . 116
2.2 Existence d'un AC persistant-universel . . . 117

Perspectives 135

Introduction

La manière la plus simple de se représenter un *automate cellulaire* (ou AC) consiste sans doute à imaginer un réseau infini et régulier de cellules. À chaque instant, chaque cellule est dans un état, imaginons une couleur, choisi parmi un même ensemble fini fixé. Au cours du temps, l'état d'une cellule évolue selon une règle locale (c'est-à-dire uniquement en fonction de son ancien état et de celui d'un certain nombre de ses voisines). Cette règle est toujours la même et elle est appliquée à chaque étape simultanément à toutes les cellules du réseau.

La *description syntaxique* d'un automate, c'est-à-dire la donnée de son ensemble d'états et de sa règle d'évolution, caractérise donc le *comportement global* d'un système infini à la fois par le nombre de cellules et par le nombre potentiel d'étapes de calcul.

Et l'intérêt des AC tient à la vaste palette de comportements complexes qui peuvent émerger malgré la simplicité et la finitude des descriptions syntaxiques. Une grande part des travaux concernant les AC traitent de ce passage de la règle locale au comportement global, illustrant la complexité du lien entre les deux. En ce sens les AC apparaissent comme un bon modèle de l'étude des systèmes complexes. *Bon* dans le sens où les hypothèses de *discrétisation*, de *localité*, d'*uniformité* et de *synchronicité* de l'évolution permettent de simplifier énormément les spécifications des systèmes, mais sans pour autant rendre simpliste le lien entre local et global.

Ceci explique sans doute que de larges classes de systèmes physiques ont été modélisés par des automates cellulaires. Le plus souvent ceux-ci suivent une règle locale déterministe. On parle alors d'AC déterministe, et nous ne considérerons que ce cas dans la suite.

Arrêtons-nous un instant aux principales hypothèses nécessaire pour pouvoir définir un automate cellulaire :

(1) L'espace doit pouvoir être discrétisé, divisé régulièrement en cellules ;

(2) l'ensemble d'états de chaque cellule du système doit être le même, et limité à un nombre fini de possibilités ;

(3) l'évolution du système doit également être discrétisable temporellement c'est à dire divisible en étapes successives ;

(4) cette évolution doit se faire selon une règle locale déterministe.

En ce qui concerne les phénomènes réels, l'hypothèse (4) est à notre échelle largement remplie. Les hypothèses de discrétisations quant à elle sont vérifiées par de larges classes de systèmes, même si la discrétisation est une problématique vaste et qui reste largement incomprise.

Cependant, lister ces hypothèses confirme le fait que de larges classes de systèmes sont simulables, ou plutôt approchables par des AC ou des familles d'AC. Ceci a motivé l'étude des AC comme outil pour la modélisation [CD98] et, en ce qui nous concerne, justifie l'intérêt de l'étude de ce modèle mathématique.

Dans la suite, nous laissons de côté toutes ces questions qui visent à relier les automates cellulaires à des systèmes concrets ou réels, nous concentrant sur l'objet théorique qu'ils constituent et sur ses propriétés.

Quelques éléments historiques

Revenons brièvement sur quelques étapes marquantes de l'étude des automates cellulaires. Ceux-ci apparaissent pour la première fois dans les travaux de J. Von Neuman sur l'auto-reproduction dans les années 1950. Dès cette époque une construction exhibe leur capacité à émuler le comportement des machines de Turing. On parle d'automates Turing-universels.

Ils ont été largement popularisés avec le *Jeu de La vie* de John Conway, présenté en 1966 dans un article de Martin Gardener du Scientific American. Ce *Jeu* constitue sans doute encore aujourd'hui la règle d'AC la plus largement connue. Deux aspects récurrents dans les travaux ultérieurs étaient déjà présents. D'abord les comportements observés (objets se déplaçant périodiquement, interactions entre ces objets) permettaient d'imaginer, même avec cette règle très simple, d'encoder du calcul Turing dans ces configurations [BCG82]. Le second point est lié à la conjoncture : ce qui était introduit en 1966 comme un jeu avec un damier et des pions profita de l'essor des premiers micro-ordinateurs, et nombreux furent ceux qui passèrent des heures à chercher à retrouver les comportements connus ou des nouveaux en partant de configurations aléatoires. Et le Jeu de la Vie présente justement cette capacité à faire très souvent *émerger* d'un chaos apparent des structures particulières non triviales. Ceci n'est pas encore de la recherche scientifique mais peut la stimuler.

Dès cet exemple, on retrouve donc à la fois la possibilité de constructions techniques visant à faire du calcul, ainsi que la problématique de l'évolution à partir de configurations quelconques, ce qui constitue en quelque sorte une première interrogation sur les comportements *typiques* des automates, au sens où ceux-ci apparaissent statistiquement, de manière visible, dans les évolutions de cet automate.

Les années 60 ont vu la généralisation du modèle, sa formalisation et le développement de son étude en tant qu'objet mathématique. Tout ceci s'est fait en lien avec les systèmes dynamiques. Les propriétés des fonctions globales comme la surjectivité, la bijectivité ont été abordées systématiquement, par exemple par Moore et Myhill [Moo62, Myh63]. Et les nombreux travaux de Hedlund et d'autres, synthétisés dans [Hed69], ont véritablement fondés l'étude des AC. En particulier, le théorème essentiel, attribué à Curtis, Lyndon et Hedlund caractérise les comportements des AC en termes topologiques : si on munit, ce qui est naturel, l'ensemble des configurations de la topologie de Cantor (c'est-à-dire de la topologie produit de la topologie discrète de l'ensemble d'états), les fonctions globales des AC sont exactement les fonctions continues et invariantes par décalage spatial. Ceci permet de s'appuyer fermement sur des propriétés classiques issues des systèmes dynamiques pour comprendre des AC.

Ceci ouvre par exemple la voie à l'étude des attracteurs [Hur87]). Le comportement à long terme des AC est ainsi abordé, en utilisant la notion d'*ensembles limites* permettant d'élaguer tout ce qui est transitoire dans le comportement des AC.

Au même moment, les AC sont abordés comme un modèle du calcul massivement parallèle et une algorithmique sur les AC se fait jour ([Moo64]). Tout ceci amène en particulier à formaliser, et étudier précisément les questions liées à l'universalité, au sens du calcul Turing.

Dans les années 80 S. Wolfram, reprenant une approche basée sur l'expérience, propose une classification des comportements des AC 1D ([Wol84]). Malgré l'empirisme de sa démarche, ceci entraîna un grand nombre de propositions de formalisations de cette classification ([CY88]) ou d'autres propositions de classifications ([Gil87, Sut90, Kůrka97]).

Des sous-familles particulières d'AC apparaissent aussi dans les travaux de cette époque, permettant de considérer des ensembles de règles dont le comportement est mieux connu. C'est le cas des additifs qui sont très étudiés depuis longtemps [IÔN83, MR98a, CFMM00] car leur structure algébrique permet une meilleure compréhension du passage du local au global. D'autres familles ont également, en un certain sens, un comportement aussi riche que les AC quelconques. C'est le cas des totalistiques pour lesquels un universel est construit dans [AC87].

Dans le même temps, les problèmes de décidabilités des propriétés des fonctions globales sont abordés. Et des résultats d'indécidabilité, dus entre autre à J. Kari ([Kar92]) confirment la difficulté algorithmique du lien entre règle locale et comportement global du système. Même le fait de toujours converger, et en temps fini, vers une unique configuration uniforme est indécidable !

Enfin plus récemment, une réponse à la problématique de classification est apportée, basée sur des ingrédients issus de la notion de simulations entre AC, sur les possibilités d'encodages du calcul d'un AC dans l'évolution d'un autre ([MR98b, Oll02b, Oll03, Oll02a, The05]). Des notions d'universalité intrinsèques au modèle sont définies, donnant un sens à la capacité de *simuler tous les AC*.

Contenu du manuscrit

Nous allons retrouver tout au long de ce manuscrit les différentes approches citées jusque là. La problématique de la classification, ses liens avec certaines sous-classes, mais aussi les notions de simulations et d'universalités seront récurrentes.

Notre travail de recherche a été motivé principalement par la volonté de comprendre ce que peut être le *comportement typique des automates cellulaires*.

Ceci peut être compris comme dans le cas évoqué pour le Jeu de la Vie : la volonté de comprendre, de quantifier, pour un automate donné, quels sont les motifs, les comportements qui apparaissent *de manière visible*. C'est-à-dire des comportements, non pas liés à une construction particulière, mais au contraire des comportements qui *émergent* naturellement, à partir de configurations initiales *aléatoires*. Ceci peut être appréhendé à l'aide de la notion d'ensembles μ-limites. Nous cherchons à savoir si le fait de considérer le comportement typique, plutôt que la totalité du comportement à long terme comme c'est le cas avec les ensembles limites classiques, permet de simplifier, de rendre calculable le passage de la règle locale au comportement à long terme.

Cette question de comportement typique peut aussi se comprendre comme, la quantification, parmi l'ensemble des automates cellulaires, de certaines propriétés. Nous cherchons alors à savoir quels seraient les propriétés d'un *automate cellulaire typique*. Ceci nécessite de formaliser la notion d'automate aléatoire, afin de donner un sens à la notion de *densité d'une propriété* sur un ensemble d'AC. Une approche également fructueuse est de s'interroger de cette manière sur les propriétés typiques, répandues, parmi certaines familles d'AC. Nous introduisons donc un formalisme permettant d'aborder ces questions et apportons quelques réponses.

Nous nous sommes enfin intéressés à la notion de simulation dite par facteur ou surjective. Celle-ci s'inscrit dans cette même démarche de compréhension des comportements typiques. En effet, à l'opposé de la simulation classique, elle ne repose pas sur un ensemble restreint, très structuré, de

configurations du simulateur. La simulation par facteur implique toutes les évolutions du simulateur en interprétant chacune de celles-ci, quelles qu'elles soient, comme des évolutions du simulé. Nous nous intéressons donc à cette simulation, et en particulier au problème de l'universalité parmi des sous-familles.

Pour aborder ces trois problématiques, on peut pointer trois fils directeurs qui reviendront presque constamment dans notre travail. D'abord les notions d'universalités internes au modèle des AC. Ensuite la notion d'encodage, et les constructions associées à celle-ci : pour montrer des résultats sur l'ensemble total des AC, on utilise des constructions permettant d'exhiber des AC particuliers ou d'introduire des transformations entre automates. Enfin l'intérêt pour des sous-familles d'AC, définies non pas par des propriétés globales, mais par des restrictions syntaxiques sur l'ensemble des règles locales qui seront souvent pertinentes dans notre travail.

Notons enfin que tout ce qui suit, à l'exclusion de quelques résultats, porte sur des automates unidimensionnels, même si une bonne partie des résultats semblent pouvoir être étendus à des dimensions supérieures.

Dans la suite, nous commençons naturellement par formaliser le cadre de notre travail, en rappelant en particulier les notions de base évoquées en introduction, mais aussi quelques classifications, ainsi que les notions de simulations, et les sous-classes d'automates les plus souvent considérées.

Au chapitre 2, nous introduisons une sous-classe particulière, les *AC multi-ensemblistes*, qui capture une notion d'*isotropie* générale indépendante du réseau. Ceux-ci se prêtent particulièrement à une notion de *changement d'échelle* que nous définissons, et notre résultat principal est la construction d'un AC qui conserve à travers toutes les échelles possibles son caractère *intrinsèquement universel*.

Le chapitre 3 est consacré à *l'étude de la densité de certaines propriétés* parmi l'ensemble des automates cellulaires. Après avoir introduit le formalisme, en utilisant une notion de *chemin* parmi l'espace des tailles d'AC, nous lions la densité à la complexité de Kolmogorov. Ceci, entre autre, justifie le lien entre propriétés *denses*, et propriétés des AC *aléatoires*. Nous nous intéressons à la densité de certaines propriétés pour l'ensemble des AC, et montrons en particulier que la *nilpotence* est une propriété de densité nulle. Puis nous nous intéressons à certaines *sous-familles définies syntaxiquement* et en particulier nous montrons à quelle point la propriété d'*universalité intrinsèque*, souvent associée à des constructions techniques, est au contraire fréquente dans ces ensembles. Ce dernier point a amené la publication de [BT09].

Le chapitre 4 traite du comportement typique d'un automate fixé. Nous introduisons la notion de μ-nilpotence qui correspond au fait pour un automate d'évoluer *presque toujours* vers une seule configuration uniforme. On s'intéresse en particulier aux propriétés de décidabilité liés aux ensembles de configurations typiques obtenus à long terme. En particulier, nous montrons que la μ-nilpotence n'est ni énumérable ni co-énumérable. Ceci montre que lorsqu'on se restreint aux comportement typiques, l'indécidabilité reste forte. Ces résultats ont été publiés dans [BPT06].

Enfin le chapitre 5 présente des résultats de [BT10], et traite de la simulation dite *par facteurs*. La question de l'existence d'un automate universel pour cette notion reste ouverte mais nous introduisons deux résultats. D'abord nous montrons l'existence d'un tel automate pour une large sous-famille d'AC, les automates persistants en dimension 1. Puis nous montrons un résultat négatif, montrant la non-existence d'un AC universel dans un autre ensemble important d'AC, les surjectifs à partir de la dimension 2.

Chapitre 1

Définitions

1 Automates cellulaires 9
1.1 Introduction syntaxique 9
1.2 Point de vue topologique 14
1.3 Propriétés des fonctions globales 15
1.4 Ensembles limites d'AC 16
1.5 Quelques généralités sur les sous-shifts 18
2 Classifications 19
2.1 La classification de Wolfram 19
2.2 Classifications topologiques 20
3 Simulations et universalités 21
3.1 Transformations spatio-temporelles 22
3.2 Sous-automate et simulation intrinsèque 23
3.3 Simulation par facteur 25
4 Sous-familles 26
4.1 Automates additifs 26
4.2 Automates cellulaires captifs 27
4.3 AC totalistiques 27

Dans ce chapitre, nous introduisons rapidement les notions usuelles dans l'étude des AC. Après avoir rappelé les définitions et caractérisations les plus fréquentes, ainsi que quelques propriétés classiques qui illustrent la richesse du modèle, nous évoquons quelques propositions de classification, puis les constructions qui définissent la notion de groupage. Enfin nous évoquons quelques sous-familles d'AC, les plus significatives dans la littérature.

1 Automates cellulaires

1.1 Introduction syntaxique

Définition

Un automate cellulaire est donc défini comme la donnée d'un réseau régulier de cellules. À un instant donné, chaque cellule possède une quantité d'information finie, son état. Et cet état évolue à chaque étape, en dépendant seulement de son ancien état, et des états d'un certain nombre de ses voisins. Ceci se formalise par :

Définition 1. *Un automate cellulaire est un quadruplet* $(\mathbb{Z}^d, \Sigma, \mathcal{V}, \delta)$ *avec*
- $\mathbb{Z}^d$ *le réseau de cellules, on appelle* d *la dimension de l'automate,*
- Σ *un ensemble fini d'états, parfois appelé alphabet de l'automate,*
- $\mathcal{V} = \{\vec{\nu_1}, \ldots, \vec{\nu_k}\}$ *un sous-ensemble fini de* $\mathbb{Z}^d$*, le voisinage de l'automate,*
- $\delta : \Sigma^{\mathcal{V}} \to \Sigma$ *la fonction locale de transition.*

Une *configuration* d'un AC $(\mathbb{Z}^d, \Sigma, \mathcal{V}, \delta)$ est un élément de $\Sigma^{\mathbb{Z}^d}$, c'est à dire une affectation d'un état à chaque cellule du réseau. Pour une configuration $c \in \Sigma^{\mathbb{Z}^d}$, et une position $\vec{z} \in \mathbb{Z}^d$, on note $c_{\vec{z}}$ l'état de la cellule en position $\vec{z}$ dans c. La *fonction globale de transition* $F : \Sigma^{\mathbb{Z}^d} \to \Sigma^{\mathbb{Z}^d}$, qui décrit l'évolution des configurations est obtenue en appliquant simultanément la règle locale à toutes les cellules du réseau :

$$\forall c \in \Sigma^{\mathbb{Z}^d}, \forall \vec{z} \in \mathbb{Z}^d, F(c)_{\vec{z}} = \delta(c_{\vec{\nu_1}+\vec{z}}, c_{\vec{\nu_2}+\vec{z}}, \ldots, c_{\vec{\nu_k}+\vec{z}}).$$

où $+$ dénote l'addition coordonnée à coordonnée de $\mathbb{Z}^d$.

Enfin une *orbite*, ou *diagramme espace-temps*, est une suite formée, à partir d'une *configuration initiale*, en considérant ses images successives par la fonction globale de l'automate : il s'agit donc pour un $c \in \Sigma^{\mathbb{Z}}$ initial fixé quelconque de la suite $(F^t(c))_{t\in\mathbb{N}}$. On les verra souvent comme des éléments de $\Sigma^{\mathbb{Z}^d \times \mathbb{N}}$. On utilisera aussi des diagrammes espace-temps partiels d'automates unidimensionnels pour visualiser leur comportement.



FIGURE 1.1: Diagramme espace-temps de l'automate `max`, 0 est représenté par une cellule noire, 1 par une grise.

Exemple 1. La figure 1.1 présente un diagramme espace-temps de l'automate `max` défini par
- le voisinage $\{-1, 0, 1\}$,
- l'alphabet $\{0, 1\}$,
- la règle de transition $\delta(x, y, z) = \max(x, y, z)$ selon l'ordre induit par celui de $\mathbb{N}$.

Exemple 2. La figure 1.2 présente un diagramme espace-temps de l'automate `Just Gliders` défini par
- le voisinage $\{-1, 0, 1\}$,
- l'alphabet $\{0, \leftarrow, \rightarrow\}$,
- la règle de transition δ définie par :
  - les $\leftarrow$ se déplacent à vitesse 1 vers la gauche sur un fond de 0
  - les $\rightarrow$ se déplacent à vitesse 1 vers la droite sur un fond de 0
  - lorsqu'un $\rightarrow$ et un $\leftarrow$ se rencontrent ils disparaissent.

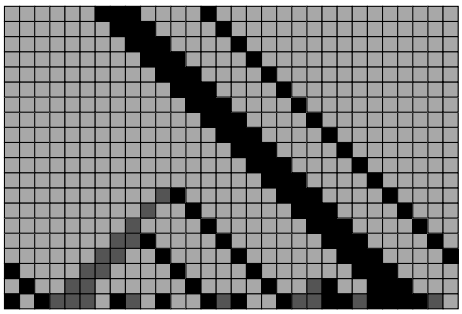


FIGURE 1.2: Diagramme espace-temps partiel de l'automate `Just Gliders`, 0 est représenté par une cellule grise plus claire, $\leftarrow$ par une noire, et $\rightarrow$ par une cellule grise plus foncée.

Exemple 3. Enfin la figure 1.3 représente une évolution de l'AC dit 110 dans la numérotation de Wolfram (qui est évoquée au chapitre 3).

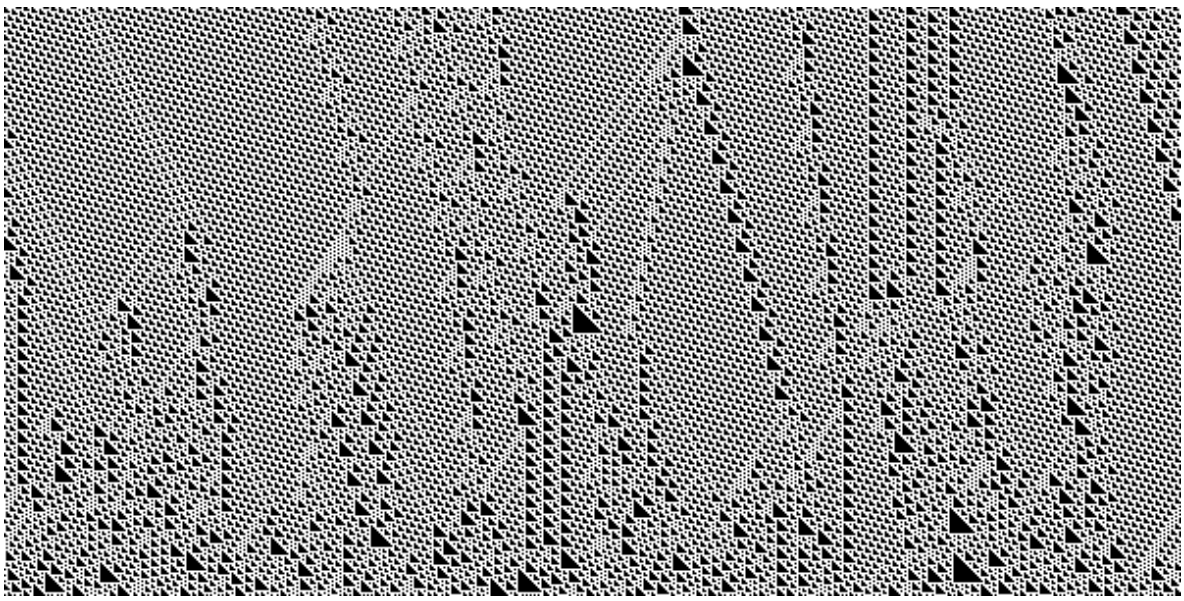


FIGURE 1.3: Diagramme espace-temps partiel de l'automate `110`.

Choix du voisinage et de l'alphabet

On peut immédiatement constater qu'à une fonction globale d'automate ne correspond jamais une description syntaxique unique, c'est à dire un quadruplet unique. Par exemple, on peut toujours étendre le voisinage à un voisinage arbitraire, en ajoutant des variables à la règle locale, et ceci ne change pas le comportement global.

À l'opposé, on utilisera le terme de *voisinage minimal* pour désigner un voisinage ne contenant aucun sous-ensemble strict qui puisse former le voisinage d'une description syntaxique valide de la même règle.

Dans la suite, on considérera le plus souvent des voisinages *connexes et centrés*.

En dimension 1, il s'agit d'un segment, paramétré uniquement par sa *taille* ou *longueur*, notée k, et qui est soit centré sur la cellule s'il est de longueur impaire, soit légèrement décalé sur la droite s'il est de longueur paire. Formellement, le voisinage connexe centré $\mathcal{V}_k$ est donné par :

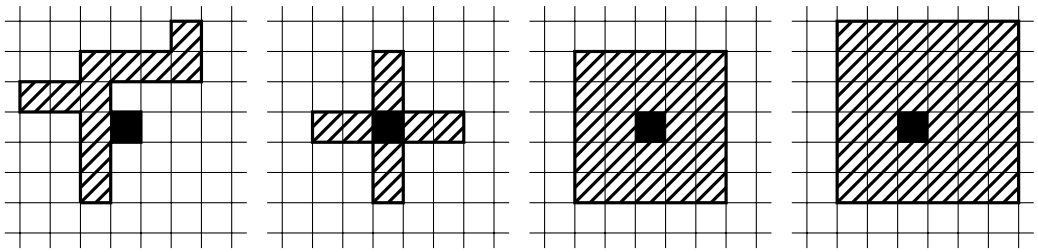
- si k est impaire : $\mathcal{V}_k = \{-\frac{k-1}{2}, \ldots, \frac{k-1}{2}\}$
- si k est paire : $\mathcal{V}_k = \{-\frac{k-2}{2}, \ldots, \frac{k}{2}\}$



(a) Le cas impaire (b) Le cas paire

FIGURE 1.4: Le voisinage connexe centré en dimension 1.

En dimension 2, tout se complique. Le nombre de *formes* possibles pour les voisinages, même connexes, est infini.



(a) quelconque (b) *Von Neuman* taille 5 (c) *Moore* de taille 5 (d) *Moore* de taille 6

FIGURE 1.5: Quelques voisinages en dimension 2.

En réalité, un seul cas particulier nous sera utile, celui du *voisinage de Moore*. Il s'agit de l'hypercube (donc du carré en dimension 2) centré sur la cellule dans le cas impaire, et légèrement décalé dans le sens des coordonnées positives dans le cas paire. Celui-ci est paramétré par son côté l. En dimension d quelconque, l'arité du voisinage est alors $k = l^d$. Formellement, $\mathcal{V}_l^d$ le voisinage de Moore de côté l en dimension d est donné par :

$$\mathcal{V}_l^d = \{(x_1, \ldots, x_d) : \forall i \in [1..d], -\lfloor \frac{l-1}{2} \rfloor \leq x_i \leq \lceil \frac{l-1}{2} \rceil\}$$

Il faut noter que la possibilité d'étendre le voisinage permet toujours d'associer à une fonction globale d'automate une description syntaxique dans laquelle le voisinage qui intervient est un voisinage de Moore.

En ce qui concerne le choix de l'ensemble d'états, la seule information importante est sa cardinalité. En effet, si Σ et Σ' sont des alphabets de même taille, il est toujours possible de considérer un isomorphisme entre l'ensemble des AC d'alphabet Σ et l'ensemble de ceux d'alphabets Σ' : il suffit de le dériver de l'isomorphisme entre alphabets.

Le plus souvent, on considérera pour chaque taille n l'alphabet canonique $\Sigma_n = \{0, 1, \ldots, n-1\}$. Ceci nous permettra aussi souvent d'utiliser l'ordre sur Σ_n induit par les entiers.

Dans la suite, on notera souvent un automate $\mathcal{A}$ par $(\mathbb{Z}^d, \Sigma_\mathcal{A}, \mathcal{V}_\mathcal{A}, \delta_\mathcal{A})$, et on désignera par $k_\mathcal{A}$ la taille de son voisinage si celui-ci est un voisinage de Moore, ainsi que par $n_\mathcal{A}$ la taille de $\Sigma_\mathcal{A}$. Enfin $F_\mathcal{A}$ ou encore $\mathcal{A}$ dénotera sa fonction globale.

De même, dans le cas d'un voisinage connexe, on étendra la notation $\delta_{\mathcal{A}}$ à la fonction de $\Sigma_{\mathcal{A}}^{\geq k_{\mathcal{A}}} \to \Sigma_{\mathcal{A}}^{*}$ obtenue en appliquant la règle locale aux positions successives d'un mot : $\forall u_i \in \Sigma_{\mathcal{A}}$

$$\delta_{\mathcal{A}}(u_1 u_2 \cdots u_l) = \delta_{\mathcal{A}}(u_1, \ldots, u_{k_{\mathcal{A}}}) \cdot \delta_{\mathcal{A}}(u_2, \ldots, u_{k_{\mathcal{A}}+1}) \cdots \delta_{\mathcal{A}}(u_{l-k_{\mathcal{A}}+1}, \ldots, u_l)$$

Remarque 1. Un certains nombre de travaux impliquent des automates cellulaires sur des réseaux différents, depuis le réseau hexagonal, jusqu'à des graphes de Cayley quelconques ([Rók94], [Rók00]). Nous ne traiterons pas de ces cas ici, bien que de nombreux résultats de simulations entre des automates fonctionnant sur différents graphes aient été démontrés, en particulier dans [Rók99].

Les shifts

Une famille particulièrement utile d'automates est celle des *shifts*, parfois appelés *translations*. Étant donné un réseau $\mathbb{Z}^d$, et un alphabet Σ, pour chaque $\vec{x} \in \mathbb{Z}^d$, on note $\sigma_{\vec{x}}$ l'automate de voisinage $\mathcal{V} = \{\vec{x}\}$ et dont la fonction locale est l'identité. En réalité, celui-ci effectue un *décalage* selon le vecteur $\vec{x}$:

$$\forall c \in \Sigma^{\mathbb{Z}^d}, \forall \vec{z} \in \mathbb{Z}^d, \sigma_{\vec{x}}(c)_{\vec{z}} = c_{\vec{z}+\vec{x}}$$

Dans le cas particulier des automates unidimensionnels, le `shift`, noté σ désignera le décalage σ_1. Celui-ci est illustré par la figure 1.6.

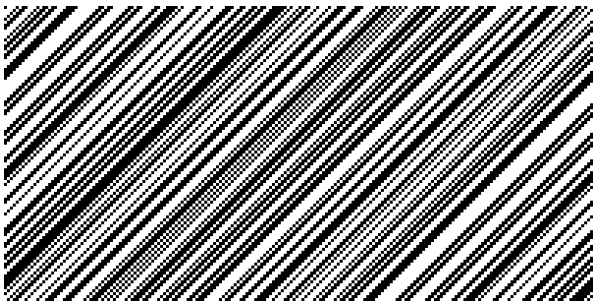


FIGURE 1.6: Un diagramme espace-temps de l'automate `shift` à deux états.

Premières propriétés

Les premières propriétés que nous introduisons sont des propriétés de la règle locale, qui donne des informations sur le comportement d'un état lorsqu'il est présent dans le voisinage.

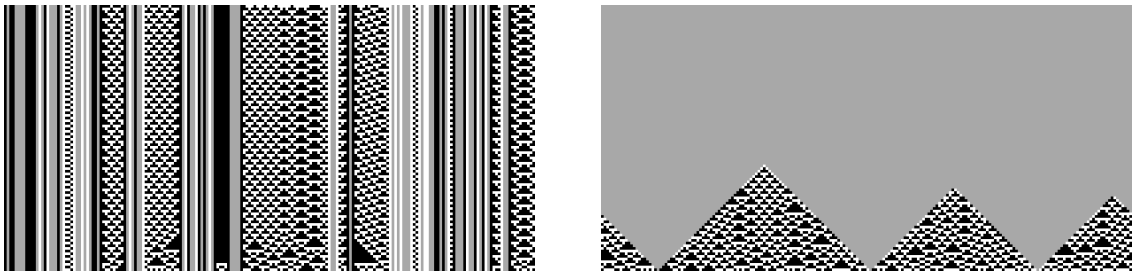
Définition 2. *Soit $\mathcal{A} = (\mathbb{Z}^d, \Sigma_{\mathcal{A}}, \mathcal{V}_{\mathcal{A}}, \delta_{\mathcal{A}})$ un automate cellulaire.*

(1) Un état $q \in \Sigma_{\mathcal{A}}$ est dit quiescent *pour $\mathcal{A}$ s'il vérifie $\delta_{\mathcal{A}}(q^{\mathcal{V}_{\mathcal{A}}}) = q$.*

(2) Un état $q \in \Sigma_{\mathcal{A}}$ est dit persistant *pour $\mathcal{A}$ si $\forall c \in \Sigma^{\mathbb{Z}^d}$, $\forall \vec{z} \in \mathbb{Z}^d$, si $c_{\vec{z}} = q$ alors $\mathcal{A}(c)_{\vec{z}} = q$.*

(3) Un état $q \in \Sigma_{\mathcal{A}}$ est dit envahissant *si $\forall u = (u_1, u_2, \ldots, u_{k_{\mathcal{A}}}) \in \Sigma^{k_{\mathcal{A}}}$, si $\exists i \in [1..k_{\mathcal{A}}]$ tel que $u_i = q$, alors $\delta_{\mathcal{A}}(u) = q$.*

Et on dit qu'un automate est *à état quiescent* (respectivement *persistant*, *envahissant*) s'il possède un état quiescent (resp. persistant, envahissant). La figure 1.7 représente deux exemples d'AC à trois états, l'un avec un état envahissant et l'autre avec un état persistant.



(a) AC avec un état persistant (b) AC avec un état envahissant

FIGURE 1.7: Diagramme espace-temps d'AC à 3 états à partir de configurations aléatoire.

En particulier dans le cas (fréquent) où une cellule est dans son propre voisinage (c'est à dire si $0 \in \mathcal{V}$), on a :

$$\text{envahissant} \Rightarrow \text{persistant} \Rightarrow \text{quiescent}$$

Et c'est le cas aussi bien pour les états que pour les automates.

Un des aspects utiles de l'existence d'un état quiescent est qu'elle permet le maintien d'un fond uniforme. Ceci permet de considérer des configurations, dites *finies* qui sont celles pour lesquelles le nombre de cellules dans un état non quiescent est fini.

Définition 3. *Soit $\mathcal{A}$ un automate possédant un état quiescent $q \in \Sigma_{\mathcal{A}}$, les configurations q-finies sont celles d'état partout q sauf en un nombre fini de cellules.*

Formellement, en notant $|x|_\infty$ la norme infinie usuelle sur $\mathbb{Z}^d$, $c \in \Sigma^{\mathbb{Z}^d}$ est dite finie si $\exists n \in \mathbb{N}$ telle que $\forall x \in \mathbb{Z}^d$, si $|x|_\infty > n$ alors $c_x = q$.

1.2 Point de vue topologique

La définition 1 est syntaxique et finie, elle est particulièrement manipulable, et utilisée en permanence dans l'étude des AC. Cependant, un autre point de vue, *topologique*, a une importance particulière.

L'espace $\Sigma^{\mathbb{Z}^d}$ peut être muni de la topologie dite *de Cantor*. Il s'agit en fait de la topologie produit, sur $\mathbb{Z}^d$, de la topologie séparée de l'alphabet Σ. Muni de celle-ci, $\Sigma^{\mathbb{Z}^d}$ est alors un espace métrique compact. On peut obtenir cette topologie via la distance d, parfois appelée *distance de Cantor* définie par :

$$\forall c_1, c_2 \in \Sigma^{\mathbb{Z}^d},\ d(c_1, c_2) = 2^{-\min\{|\vec{x}|_\infty : c_1(\vec{x}) \neq c_2(\vec{x})\}}.$$

dans laquelle $|\vec{x}|_\infty$ est la norme infinie sur $\mathbb{Z}^d$: pour $\vec{x} = (x_1, x_2, \ldots, x_d) \in \mathbb{Z}^d$,

$$|\vec{x}|_\infty = \max(\{x_1, x_2, \ldots, x_d\}).$$

Intuitivement, deux configurations seront proches si elles coïncident sur une large zone centrale.

Utilisant ce formalisme, l'ensemble des automates cellulaires est alors caractérisé de la manière suivante par le théorème attribué à Curtis, Hedlund et Lyndon ([Hed69]) :

Théorème 1.1. *Soit F une fonction de $\Sigma^{\mathbb{Z}^d}$ dans lui-même. F est une fonction globale d'automate cellulaire si et seulement si F est continue et commute avec le shift.*

Cette caractérisation est essentielle : elle détache largement la notion d'automates cellulaires de la définition syntaxique précédente, et en particulier des paramètres spécifiques (taille du voisinage par exemple) qui apparaissent dans la définition.

Les conséquences de ce théorème sont nombreuses, par exemple il devient évident que la composition de fonctions globales d'AC est encore un AC. Et de même, il devient facile de voir que si la fonction globale d'un AC est bijective, sa fonction inverse est également continue (l'espace étant compact), et est donc aussi un AC.

Ce théorème rattache également l'étude des AC à celle des systèmes dynamiques discrets qu'ils constituent, en considérant un automate $\mathcal{A}$ comme le système dynamique noté $(\Sigma_{\mathcal{A}}^{\mathbb{Z}}, F_{\mathcal{A}})$ d'espace de phase $\Sigma_{\mathcal{A}}^{\mathbb{Z}}$ et de fonction $F_{\mathcal{A}}$.

Enfin ce théorème pointe le rôle particulier que joue le décalage σ dans l'espace des configurations.

1.3 Propriétés des fonctions globales

Le théorème précédent amène naturellement à s'intéresser aux caractéristiques usuelles des fonctions pour les fonctions globales d'automates cellulaires.

Ensembles images, mots d'Eden

Étant donné un automate $\mathcal{A} = (\mathbb{Z}^d, \Sigma_{\mathcal{A}}, \mathcal{V}_{\mathcal{A}}, \delta_{\mathcal{A}})$, on définit son ensemble image comme l'ensemble des configurations atteignables par l'application de sa fonction globale $\mathcal{A}(\mathbb{Z}^d)$.

Toutes les configurations ne sont pas atteignables, par exemple on se convainc assez facilement que pour l'automate `max` introduit plus haut, le mot 010 n'apparait dans aucune configuration de son ensemble image. On parle alors de *mots d'Eden*.

Surjectivité, injectivité, bijectivité

Sans surprise, un automate cellulaire est dit *injectif* (resp. *surjectif*, *bijectif*) si sa fonction globale est injective (resp. surjective, bijective). Et on dit de plus qu'un automate est *injectif sur les configurations finies* si deux configurations finies différentes ont une image finie différente.

Et le théorème connu sous le nom de théorème de Moore-Myhill relie ces différentes notions.

Théorème 1.2. *Un automate cellulaire est surjectif si et seulement s'il est injectif sur les configurations finies.*

Ce théorème s'applique à priori aux automates possédant un état quiescent. Le sens direct dans ce cas a été montré par Moore ([Moo62]), et le sens indirect par Myhill ([Myh63]).

Cependant il s'étend facilement aux AC quelconques : pour tout automate $\mathcal{A}$, il existe en effet t tel que l'automate $\mathcal{A}^t$ possède un état quiescent. Et $\mathcal{A}$ est surjectif si et seulement si $\mathcal{A}^t$ est surjectif.

On peut donc utiliser le théorème de Moore-Myhill pour obtenir le corollaire suivant pour tout automate :

Corollaire 1.3. *Un AC injectif est surjectif, et donc bijectif.*

Une illustration non triviale du lien entre fonctions locales et globales est le résultat suivant :

On dit qu'un automate est l-*équilibré*, pour un certain $l \geq 1$ si les mots de taille l on chacun exactement le même nombre d'antécédents. Et un automate est dit équilibré s'il est l-équilibré pour tout l. En dimension 1, si le diamètre de l'automate est k, et la cardinalité de l'alphabet n, chaque mot de taille l a alors n^{k-1} antécédents.

Théorème 1.4. *Un AC est surjectif si et seulement s'il est équilibré.*

Ce résultat est du à Maruoka et Kimura ([MK76]). Le corollaire 1.3 permet de montrer qu'un AC injectif est également équilibré.

Nous ne les rappellerons pas ici, mais toute une série de résultats relient ces propriétés entre elles, et aux propriétés équivalentes restreintes aux configurations finies ou spatialement périodiques. La thèse de B. Durand ([Dur94]) propose un panorama complet de ces résultats.

Il faut noter que certaines de ces propriétés sont liées à des résultats de décidabilité qui dépendent de la dimension. Par exemple dans le cas de la surjectivité, S. Amoroso et Y.N. Patt ([AP72]) ont montré sa décidabilité en dimension 1 et J. Kari ([Kar94a]) l'indécidabilité en dimension supérieure :

Théorème 1.5. *La surjectivité des AC est*
- *décidable en dimension* 1
- *indécidable en dimension* 2.

La preuve de Kari implique en particulier qu'il n'est pas possible d'énumérer les automates cellulaires surjectifs de dimension 2.

1.4 Ensembles limites d'AC

Une fraction importante de l'étude des automates cellulaires est consacrée à la compréhension de leur comportement à long terme. Dans cette perspective, la notion d'*ensemble limite* qui correspond à l'ensemble des configurations qui peuvent être atteintes après un nombre arbitrairement grand d'étapes a été très étudiée depuis ([CPY89])

Ensembles et langages limites

Nous rappelons d'abord simplement la définition du langage d'une configuration ou d'un ensemble de configuration en dimension d quelconque.

Soit $\mathcal{R}_{z_1,z_2,\ldots,z_d} \subseteq \mathbb{Z}^d$, une *forme rectangulaire*, c'est à dire un ensemble *fini* de positions :

$$\mathcal{R}_{z_1,z_2,\ldots,z_d} = \{x = (x_1, \ldots, x_d) \in \mathbb{Z}^d : \forall i, 0 \leq x_i \leq z_i\}.$$

Un mot ou motif d-dimensionnel rectangulaire de forme $\mathcal{R}_{z_1,z_2,\ldots,z_d}$ sur Σ est formellement une fonction $\mathcal{R}_{z_1,z_2,\ldots,z_d} \to \Sigma$. On note Σ^{*^d}, l'ensemble des mots d-dimensionnels de forme rectangulaire. Et pour un mot $u \in \Sigma^{*^d}$ de forme $\mathcal{R}_{z_1,z_2,\ldots,z_d}$, on notera $u_{\vec{x}}$ l'image de $\vec{x}$ par u, c'est à dire la lettre en position x dans u.

Un langage d-dimensionnel est, comme on l'imagine, un sous-ensemble de Σ^{*^d}.

On dit qu'un mot $u \in \Sigma^{*^d}$ de forme $\mathcal{R}_{z_1,z_2,\ldots,z_d}$ apparaît dans une configuration $c \in \Sigma^{\mathbb{Z}^d}$ en position $\vec{y} = (y_1, \ldots, y_d) \in \mathbb{Z}^d$ si $\forall \vec{x} = (x_1, \ldots, x_d) \in \mathcal{R}_{z_1,z_2,\ldots,z_d}$, $c_{\vec{y}+\vec{x}} = u_{\vec{x}}$.

Étant donnée une configuration $c \in \Sigma^{\mathbb{Z}^d}$, le langage de c est l'ensemble des motifs rectangulaires de toute les formes apparaissant dans c. On note ce langage L_c.

Étant donné un ensemble de configurations $X \subseteq \Sigma^{\mathbb{Z}^d}$, on définit $L_X = \cup_{c \in X} L_c$ le langage de l'ensemble X.

Un langage peut donc a priori correspondre à un grand nombre d'ensembles de configurations différents.

On peut maintenant définir ensemble et langage limite d'un automate :

Définition 4. *L'ensemble limite $\Omega_{\mathcal{A}}$ d'un automate $\mathcal{A}$ est définie par*

$$\Omega_{\mathcal{A}} = \cap_{i \in \mathbb{N}} \mathcal{A}^i(\Sigma_{\mathcal{A}}^{\mathbb{Z}^d}).$$

Le langage limite de $\mathcal{A}$ noté $\Lambda_{\Omega_{\mathcal{A}}}$ est le langage de $\Omega_{\mathcal{A}}$

L'ensemble limite contient toujours au moins une configuration. Le plus simple pour s'en convaincre est de considérer l'évolution des configurations uniformes : l'image d'une configuration uniforme est uniforme, celles-ci sont en nombre fini, il y a donc un cycle de configurations uniforme dans l'ensemble limite.

Par définition un automate $\mathcal{A} = (\mathbb{Z}^d, \Sigma_{\mathcal{A}}, \mathcal{V}_{\mathcal{A}}, \delta_{\mathcal{A}})$ est surjectif si et seulement si $\Omega_{\mathcal{A}} = \Sigma_{\mathcal{A}}^{\mathbb{Z}^d}$.

Par ailleurs ces ensembles limites peuvent être de complexité importante, dans [Hur87] des langages limites non récursifs sont exhibés dans le cas unidimensionnel.

Une propriété importante bien connue est qu'un ensemble limite d'automate cellulaire est soit restreint à une seule configuration soit indénombrable.

Nilpotence

Définition 5. *Un automate cellulaire est dit* nilpotent *si son ensemble limite est réduit à une seule configuration.*

Dans ce cas, par invariance de l'ensemble limite par translation, cette configuration est uniforme, et l'état qui la constitue est nécessairement quiescent.

Le résultat suivant illustre la complexité du lien entre le comportement global d'un automate et sa règle locale

Théorème 1.6. *La nilpotence des automates cellulaires est indécidable.*

C'est donc le cas si on se restreint à ceux ayant un état quiescent.

Ce résultat est particulièrement représentatif de la complexité du passage de la règle locale au comportement de la règle globale. Même la propriété à long terme qui peut être vue comme la plus simple est indécidable. Il est du à K. Čulik, J. Pachl et S. Yu ([CPY89]) pour les dimensions 2 et supérieures et à été étendu par J. Kari ([Kar92]) pour la dimension 1.

J. Kari a aussi étendu ce résultat à un théorème connu sous le nom de *théorème de Rice pour les ensembles limites* d'automates cellulaires ([Kar94b]). Celui-ci a montré que les propriétés non triviales des ensembles limites des AC étaient indécidables en réduisant le problème de décision d'une telle propriété à la nilpotence.

Ceci nécessite d'être un peu plus formel. Une propriété $\mathcal{P}$ de l'ensemble des AC est dite propriété des ensembles limites si $\mathcal{A} \in \mathcal{P}$ et $\Omega_{\mathcal{A}} = \Omega_{\mathcal{B}}$ implique $\mathcal{B} \in \mathcal{P}$. Et une telle propriété est dite non triviale s'il existe $\mathcal{A}$ et $\mathcal{B}$ tels que $\mathcal{A} \in \mathcal{P}$ et $\mathcal{B} \notin \mathcal{P}$. Le problème d'appartenance consiste alors à prendre en entrée un automate $\mathcal{A} = (\mathbb{Z}, \Sigma_{\mathcal{A}}, \mathcal{V}_{\mathcal{A}}, \delta_{\mathcal{A}})$ et à répondre à la question $\mathcal{A} \in \mathcal{P}$ ou $\mathcal{A} \notin \mathcal{P}$.

Le théorème de Kari se formule alors de la manière suivante

Théorème 1.7. *Pour toute propriété non triviale des ensembles limites d'automates cellulaires l'appartenance est indécidable.*

On a vu que en dimension 1, la surjectivité était décidable (il s'agit du théorème 1.5). Or $\mathcal{A} = (\mathbb{Z}^d, \Sigma_{\mathcal{A}}, \mathcal{V}_{\mathcal{A}}, \delta_{\mathcal{A}})$ est surjectif si et seulement si $\Omega_{\mathcal{A}} = \Sigma_{\mathcal{A}}^{\mathbb{Z}^d}$. Cependant ceci n'est pas contradictoire avec le théorème précédent.

En effet, un automate $\mathcal{B}$ peut avoir pour ensemble limite $\Sigma_{\mathcal{A}}^{\mathbb{Z}^d}$ et ne pas être surjectif, si $\Sigma_{\mathcal{A}} \subset \Sigma_{\mathcal{B}}$. La surjectivité n'est donc pas une propriété des ensembles limites.

Ceci soulève alors la question de la décidabilité des propriétés d'ensembles limites à alphabet fixé. P. Guillon et G. Richard on répondu récemment à cette question en montrant ([GR10]) :

Théorème 1.8. *Toutes les propriétés non triviales, hormis la surjectivité, des ensembles limites des AC* sur un alphabet binaire *sont indécidables.*

Non trivial est ici compris dans le sens de l'existence d'un automate sur un alphabet binaire ayant la propriété et d'un autre ne l'ayant pas.

1.5 Quelques généralités sur les sous-shifts

Les sous-shifts forment un type d'ensembles de configurations, largement étudiés en dynamique symbolique.

Définition 6. *Un ensemble $S \subseteq \Sigma^{\mathbb{Z}^d}$ est un sous-shift s'il est invariant par translation σ et fermé pour la topologie de Cantor.*

Et cette classe d'ensembles possède une caractérisation par leur langage.

Pour tout langage d-dimensionnel $L \subseteq \Sigma^{*^d}$, on note S_L l'ensemble des configurations ne comportant aucune occurrence d'aucun mot de L. Ceci revient à dire que $S_L = \{c \in \Sigma^{\mathbb{Z}^d} : L_c \cap L = \emptyset\}$

Et cette définition permet la propriété très utile suivante :

Proposition 1.9. *Un ensemble $S \subseteq \Sigma^{\mathbb{Z}^d}$ est un sous-shift si et seulement s'il existe un langage $L \subseteq \Sigma^{*^d}$ tel que $S = S_L$.*

Parmi l'ensemble des sous-shifts, une famille particulière se dégage qui constitue les sous-shifts que l'on peut caractériser par une liste de mots interdits finie.

Proposition 1.10. *Un sous-shift S est dit* de type fini *(SFT), s'il existe un langage $L \subseteq \Sigma^{*^d}$ fini, tel que $S_L = S$.*

Pour plus de détails sur ces notions de sous-shifts, on pourra consulter [LM95].

En ce qui concerne les automates cellulaires, l'ensemble image d'un AC est un sous-shift. Un de ses langages interdits peut être obtenu en considérant le complémentaire des mots atteints comme images de la fonction locale, en commençant par les mots de la taille du voisinage et en augmentant celle-ci progressivement.

$$L_{\mathcal{A}(\Sigma_{\mathcal{A}}^{\mathbb{Z}})} = \cup_{s\in\mathbb{N}}\Sigma_{\mathcal{A}}^{s} \setminus \{\delta_{\mathcal{A}}(u) : u \in \Sigma_{\mathcal{A}}^{k_{\mathcal{A}}+s}\}$$

L'ensemble limite $\Omega_{\mathcal{A}}$ est également un sous-shift, on peut décrire un langage interdit L tel que $\Omega_{\mathcal{A}} = S_L$. C'est par exemple le cas pour

$$L = \cup_{s,t\in\mathbb{N}}\Sigma_{\mathcal{A}}^{s} \setminus \{\delta_{\mathcal{A}}^{t}(u) : u \in \Sigma_{\mathcal{A}}^{k_{\mathcal{A}}+s}\}$$

2 Classifications

On a évoqué en introduction le fait qu'à côté de l'étude des AC comme modèle de calcul, et de l'algorithmique sur les AC, que nous n'évoquerons pas ici, s'est développée toute une connaissance de la structure de l'ensemble des AC.

Une part non négligeable de la littérature concernant les AC est ainsi constituée de propositions de classifications. Loin d'en faire un panorama complet, nous présenterons quelques-unes des plus marquantes.

2.1 La classification de Wolfram

Il s'agit d'une des premières et sans doute de la plus connue des classifications d'automates cellulaires. Elle est basée sur des *expériences*, c'est à dire en fait sur l'observation d'extraits de diagrammes espace-temps d'automates à partir de segments initiaux tirés aléatoirement.

À partir de là, Wolfram a explicité ([Wol84]) une division en quatre classes disjointes :

1. Les automates évoluent vers une même configuration uniforme.
2. Différentes structures, mais stables ou périodiques émergent toujours.
3. Le comportement est plus compliqué, semble aléatoire, mais avec des motifs qui se répètent.
4. Apparition de motifs simples mais ayant des interactions, et donc un comportement global complexe.

Cette classification a des limites évidentes, en premier lieu l'absence d'un formalisme qui permette de se poser réellement pour un automate donné la question de son appartenance. Cette absence fait aussi que même pour les automates les plus petits, bien connus, la classe n'est pas toujours évidente.

Des tentatives ont bien été faites pour formaliser ces classes, en particulier [CY88], mais toutes présentes des limites.

Cependant, cette classification a d'abord le mérite de chercher à distinguer différents types de comportements en fonctions de l'évolution à relativement long terme des AC. Et surtout, elle se base sur l'observation du comportement des automates à partir de configurations aléatoires. On est ici à l'opposé de la construction *à priori* de structure. On cherche à identifier de la structure qui apparaisse de manière visible, c'est à dire *fréquemment*, dans des diagrammes espace-temps partiels des automates.

Enfin elle a motivée toute une série de propositions de classifications qui ont été développées par la suite.

2.2 Classifications topologiques

À l'aide de la distance de Cantor d définie plus haut, et de la topologie induite, il est possible de tenter de classifier les automates cellulaires en s'appuyant sur leur dynamique topologique. C'est ce qu'ont fait d'abord Gilman ([Gil87]) dans une version mesurée, en utilisant en particulier la mesure de l'ensemble des points d'équicontinuité, puis plus tard Kůrka ([Kůrka97]). Nous présentons un peu plus en détail cette dernière classification.

On définit la boule $B_\delta(x)$ de rayon δ centrée en $x \in \Sigma^{\mathbb{Z}^d}$ par

$$B_\delta(x) = \{y \in \Sigma^{\mathbb{Z}^d} : d(x,y) < \delta\}$$

Dans le cas unidimensionnel, une boule est un cylindre, c'est à dire un ensemble de configurations qui coïncident sur un segment central donné. Formellement, pour un mot $u \in \Sigma$ quelconque, le cylindre noté $[u]$ est défini par :

$$[u]_{-\lfloor \frac{|u|}{2} \rfloor} = \{c \in \Sigma^{\mathbb{Z}} : c_{[-\lfloor \frac{|u|}{2} \rfloor .. -\lceil \frac{|u|}{2} \rceil]} = u\}$$

Dans les dimensions supérieures, l'idée est similaire, mais les cylindres sont caractérisés par des motifs rectangulaires d-dimensionnels.

Étant donné un AC $\mathcal{A}$, une configuration $c \in \Sigma^{\mathbb{Z}}$ est un *point d'équicontinuité*, noté $c \in \mathbf{Eq}(\mathcal{A})$, ssi

$$\forall \epsilon > 0, \exists \delta > 0, \forall x \in B_\delta(c), \forall n \geq 0, d(\mathcal{A}^n(x), \mathcal{A}^n(c)) < \epsilon.$$

Intuitivement, il s'agit donc des configurations c pour lesquelles pour tout ensemble fini de cellules, il est possible de délimiter un second ensemble fini de cellules, contenant le premier, et tel que toute configuration qui coïncide avec c sur le second ensemble à l'instant initial coïncidera aussi pour chaque cellule du premier ensemble avec les images successives de c à chaque étape de son évolution.

Un automate est dit *équicontinu* si toutes les configurations sont des points d'équicontinuités pour lui.

On dit ensuite qu'un automate $\mathcal{A}$ est *sensible (aux conditions initiales)* ssi :

$$\exists \epsilon \forall \delta > 0, \forall x \in \Sigma_{\mathcal{A}}^{\mathbb{Z}^d}, \exists y \in B_\delta(x), \exists n, d(\mathcal{A}^n(x), \mathcal{A}^n(y)) \geq \epsilon.$$

Il s'agit donc des automates pour lesquels il existe un ensemble de cellules fixé tel que, pour toute configuration x, il existe une configuration y arbitrairement proche de celle-ci telles que au cours de l'évolution sous l'action de l'AC, les images de x et y vont différer sur l'ensemble de cellules fixé.

Un automate sensible ne possède donc aucun point d'équicontinuité. Et la réciproque est vraie également en dimension 1 : dans ce cas, il est possible de trouver un ϵ qui soit le même pour chaque configuration. En dimension 2 ce n'est pas le cas [ST08].

Enfin un automate $\mathcal{A}$ est dit *positivement expansif* ssi

$$\exists \epsilon, \forall x, y \in \Sigma_{\mathcal{A}}^{\mathbb{Z}^d}, y \neq x, \exists n \geq 0, d(\mathcal{A}^n(x), \mathcal{A}^n(y)) \geq \epsilon.$$

Cette propriété exprime le fait que deux configurations distinctes, aussi proche soit-elles, vont toujours finir par s'éloigner à une certaine distance sous l'action de l'automate.

Kůrka introduit alors la classification topologique suivante, un AC unidimensionnel appartient à une et une seule des classes suivantes :

1. $\mathbf{Eq}(\mathcal{A}) = \Sigma_{\mathcal{A}}^{\mathbb{Z}^d}$: il est équicontinu.
2. $\emptyset \subset \mathbf{Eq}(\mathcal{A}) \subset \Sigma_{\mathcal{A}}^{\mathbb{Z}^d}$: il n'est ni équicontinu ni sensible.
3. $\mathbf{Eq}(\mathcal{A}) = \emptyset$, il est sensible mais il n'est pas positivement expansif.
4. Il est positivement expansif.

L'inconvénient principal de cette classification est que certains automates au comportement très simples, en particulier les shifts, sont sensibles. Plusieurs modifications ont été proposées avec un certain succès, soit en utilisant une topologie différente ([CFMM97]), soit plus récemment en réintroduisant la composante spatiale ([Sab08]). M. Sablik obtient ainsi une classification des AC en fonction des directions selon lesquelles l'AC considéré est équicontinu ou expansif.

3 Simulations et universalités

Au delà de ces tentatives de classifications en un nombre fini de classes, un certain nombre de travaux concernant la théorie *du groupage* ont fourni une autre approche pour appréhender la structure de l'ensemble des automates cellulaires. Il s'agit des travaux de J. Mazoyer, M. Delorme, I. Rapaport, N. Ollinger, et G. Theyssier.

Celle-ci repose sur des notions de *simulation* entre automates cellulaires. De telles relations ont permis d'introduire une structure de pré-ordre sur l'ensemble des AC. Pour une présentation complète, il faut se référer aux thèses écrites ([Rap97, Oll02a, The05]) et à [DMOT10a, DMOT10b] qui font le point sur les connaissances actuelles. Nous nous contentons pour notre part d'introduire les notions qui nous seront nécessaires pour la suite.

Il a été très vite observé que des automates cellulaires sont capables de simuler n'importe quel comportement de machine de Turing, en particulier des machines universelles.

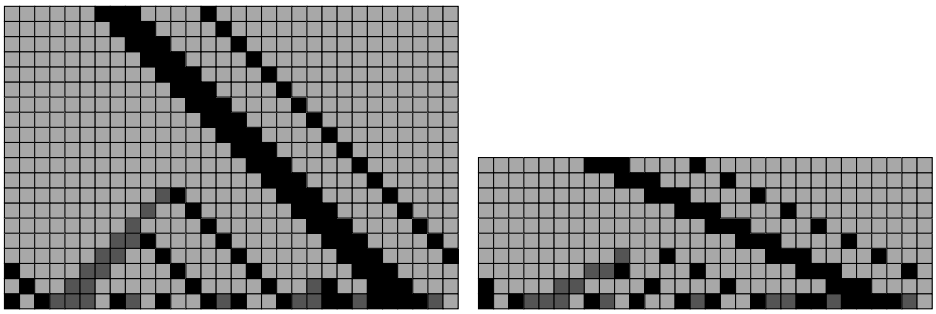
On peut facilement, étant donné une machine de Turing, lui associer un automate cellulaire et une fonction qui associe à un mot d'entrée de la machine une configuration finie de l'automate, telle que l'évolution ultérieure de l'automate corresponde à celle de la machine. On peut définir de cette manière une classe d'automates cellulaires Turing-puissants ; on peut se rapporter à [The05] pour un formalisme précis.

Cependant ceci ne permet pas de rendre compte de toute la puissance de calcul des automates cellulaires.

Comment quantifier la puissance de calcul d'un AC, à l'aune d'une machine de Turing, pour laquelle l'évolution se fait cellule par cellule, alors que la spécificité du modèle des AC tient justement à la mise à jour d'une infinité de cellules *simultanément* par application de la règle locale à tout le réseau. À côté de cela, le nombre de cellules dont l'état est modifié au cours du calcul par une machine de Turing reste borné par le nombre d'étapes du calcul.

Pour dépasser cette limitation, des notions de simulation internes au modèle ont été exhibées. En réalité, les idées de simulation entre AC sont anciennes.

Mais elles n'ont été systématisées que dans les années 1990 et 2000. Sans entrer dans l'historique de ces notions, présente par exemple dans [DMOT10a], nous allons rappeler précisément les deux notions de simulations auxquelles il est fait référence le plus fréquemment et qui sont aussi celles qui nous seront utiles. Nous parlerons d'une part de la simulation dite *intrinsèque*, souvent appelée *simulation injective* dans la littérature, et d'autre part de la simulation que nous appellerons *par facteur*, qui apparaît sous le nom de simulation *surjective*.



(a) AC de départ (b) AC groupé 2 étapes par 2 étapes.

FIGURE 1.8: Exemple de groupage temporel

Remarque 2. La simulation par facteur est aussi une simulation intrinsèque au sens premier du mot, mais pour éviter les confusions, nous n'emploierons le mot *intrinsèque* que pour parler de la première de ces deux simulations.

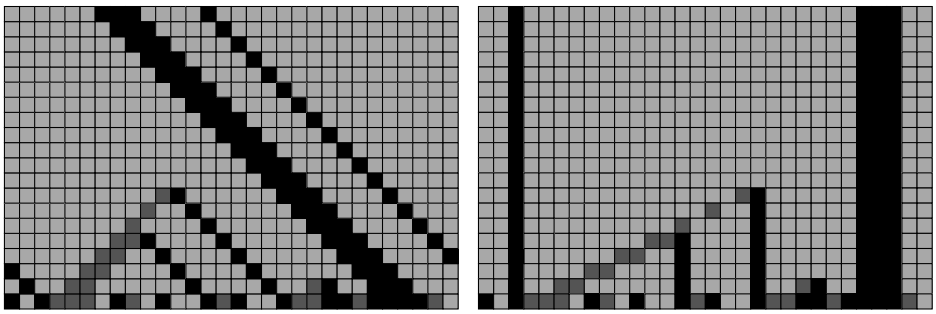
Ces deux simulations peuvent être obtenues en combinant d'une part l'application, à chacun des deux automates, de transformations spatio-temporelles, et d'autre part d'une relation syntaxique entre les deux instances transformées. Et c'est cette dernière qui varie entre les différentes simulations.

3.1 Transformations spatio-temporelles

Ces transformations sont les mêmes dans les deux simulations que nous considérons. Elles permettent de considérer des groupes de cellules au lieu de cellules individuelles, de ne plus considérer l'évolution de l'automate étape par étape mais en appliquant les étapes de calcul *par groupe*, et enfin éventuellement de décaler les configurations selon une translation, la même à chaque étape considérée.

Le *groupage temporel* de paramètre $t \in \mathbb{N}^*$ se fait simplement en itérant t fois l'application de la règle à chaque étape (voir figure 1.8). On associe alors $\mathcal{A}$ à $\mathcal{B}$ s'il existe une constante $t \in \mathbb{N}^+$ tel que $\mathcal{A}^t = \mathcal{B}$.

Le *décalage spatial* de paramètre $\vec{z} \in \mathbb{Z}^d$ se fait en composant la fonction globale de l'automate avec la translation $\sigma_{\vec{z}}$. Ces décalages permettent d'associer un automate $\mathcal{A}$ aux automates $\sigma_{\vec{z}} \circ \mathcal{A}$ obtenus par cette composition. Et il est immédiat que cette transformation est réversible.



(a) AC de départ (b) AC décalé d'1 cellule à chaque étape

FIGURE 1.9: Exemple de décalage spatial.

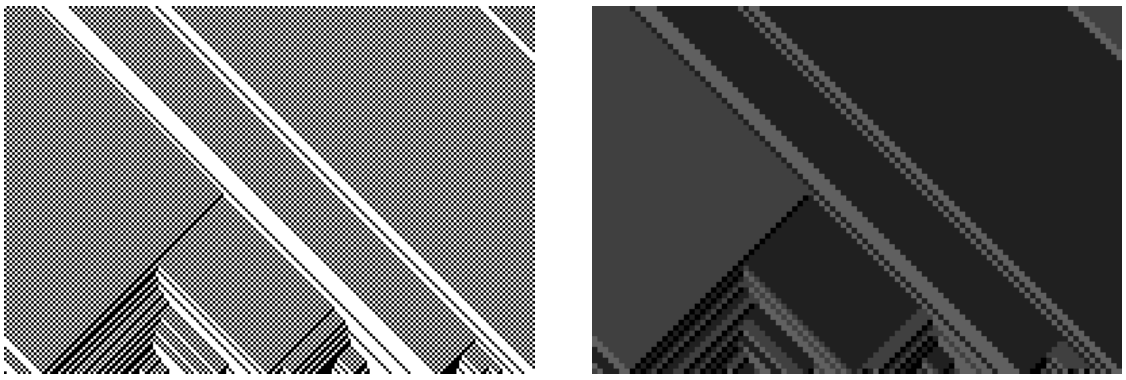
Le *groupage spatial* nécessite lui quelques définitions supplémentaires. Pour les automates unidimensionnels, grouper d'un paramètre $m \in \mathbb{N}^*$ revient à transformer une configuration de $\Sigma^{\mathbb{Z}}$ en une configuration de $(\Sigma^m)^{\mathbb{Z}}$, c'est à dire à considérer non plus une cellule prise individuellement, mais des paquets de cellules, dont on considère l'ensemble d'états. On définit la fonction d'ouverture de ces paquets :

$$o_m : (\Sigma^m)^{\mathbb{Z}} \to \Sigma^{\mathbb{Z}}$$
$$c \mapsto o_m(c) : (o_m(c))_x = (c_{x/m})_{x \bmod m}$$

où x/m est le quotient dans la division entière de x par m.

Il s'agit d'une bijection dont on note l'inverse o_m^{-1}, cette dernière correspond à la création de paquets de cellules.

À partir d'un automate $\mathcal{A}$ qui agit sur l'espace $\Sigma^{\mathbb{Z}}$, on peut définir l'automate $o_m^{-1} \circ \mathcal{A} \circ o_m$ qui agit lui sur des configurations de $(\Sigma m)^{\mathbb{Z}}$.



(a) AC de départ (b) AC groupé 2×2

FIGURE 1.10: Exemple de groupé

Et on étend naturellement cette définition à un réseau d-dimensionnel en utilisant le groupage selon une forme rectangulaire $\mathcal{R}_{\vec{m}} = \mathcal{R}_{m_1,m_2,\ldots,m_d}$:

Définition 7. *On dit qu'un automate $\mathcal{A} = (d, \Sigma_{\mathcal{A}}, \mathcal{V}_{\mathcal{A}}, \delta_{\mathcal{A}})$ est un groupé d'un automate $\mathcal{B} = (d, \Sigma_{\mathcal{B}}, \mathcal{V}_{\mathcal{B}}, \delta_{\mathcal{B}})$ s'il existe $\vec{m} \in \mathbb{Z}^d$, $\vec{z} \in \mathbb{Z}^d$, et $t \in \mathbb{N}^*$ tel que $\mathcal{A}(c) = o_{\vec{m}} \circ \sigma_{\vec{z}} \circ \mathcal{B}^t \circ o_{\vec{m}}^{-1}(c)$.*

On note $\mathcal{A} = \mathcal{B}^{<\vec{m},t,\vec{z}>}$.

Intuitivement, ceci signifie qu'en *distordant* spatialement et temporellement l'ensemble des diagrammes espace-temps de $\mathcal{B}$, on y retrouve exactement ceux de $\mathcal{A}$.

3.2 Sous-automate et simulation intrinsèque

Les transformations locales que nous considérons sont toutes des relations entre automates de même dimension. C'est en particulier le cas de la relation suivante, qui correspond à la simulation intrinsèque.

Définition 8. *On dit que $\mathcal{A} = (d, \Sigma_{\mathcal{A}}, \mathcal{V}_{\mathcal{A}}, \delta_{\mathcal{A}})$ est un* sous-automate *de $\mathcal{B} = (d, \Sigma_{\mathcal{B}}, \mathcal{V}_{\mathcal{B}}, \delta_{\mathcal{B}})$, et on note $\mathcal{A} \sqsubseteq \mathcal{B}$ si $\mathcal{A}$ est isomorphe à la restriction de $\mathcal{A}$ à un sous-ensemble stable de $\Sigma_{\mathcal{A}}$. C'est à dire s'il existe un morphisme injectif $i : \Sigma_{\mathcal{A}} \to \Sigma_{\mathcal{B}}$, tel que $\mathcal{B} \circ \bar{i} = \bar{i} \circ \mathcal{A}$, où $\bar{i}$ dénote l'extension de i aux configurations, obtenue en appliquant i à chaque cellule.*

On notera parfois $\mathcal{A} \sqsubseteq_i \mathcal{B}$ pour expliciter le morphisme i.

Remarque 3. Si on se limite à cette relation, la structure induite n'est pas très riche. En effet, le nombre de sous-automates d'un automate donné est borné en fonction de la cardinalité de l'ensemble d'états.

Pour obtenir la simulation intrinsèque on combine cette notion et les transformations spatio-temporelles :

Définition 9. *Soit $\mathcal{A} = (d, \Sigma_\mathcal{A}, \mathcal{V}_\mathcal{A}, \delta_\mathcal{A})$ et $\mathcal{B} = (d, \Sigma_\mathcal{B}, \mathcal{V}_\mathcal{B}, \delta_\mathcal{B})$. On dit que $\mathcal{A}$ simule $\mathcal{B}$, noté $\mathcal{B} \preccurlyeq_\sqsubseteq \mathcal{A}$, s'il existe une instance groupée de $\mathcal{B}$ qui est un sous-automate d'une instance groupée de $\mathcal{A}$. Formellement,*

$$\mathcal{B} \preccurlyeq_\sqsubseteq \mathcal{A} \Leftrightarrow \exists t_\mathcal{A}, t_\mathcal{B} \in \mathbb{N}^*, \vec{z_\mathcal{A}}, \vec{z_\mathcal{B}}, \vec{m_\mathcal{A}}, \vec{m_\mathcal{B}} \in \mathbb{Z}^d, i : \Sigma_\mathcal{B}^{\mathcal{R}_{\vec{m}}} \to \Sigma_\mathcal{B}^{\mathcal{R}_{\vec{m}}},$$
$$\mathcal{B}^{<\vec{m_\mathcal{B}}, t_\mathcal{B}, \vec{z_\mathcal{B}}>} \sqsubseteq_i \mathcal{A}^{<\vec{m_\mathcal{A}}, t_\mathcal{A}, \vec{z_\mathcal{A}}>}$$

Cette notion de simulation est la plus étudiée. Intuitivement, elle correspond à la notion informatique d'encodage. Un automate en simule un autre s'il existe un *encodage* des configurations du simulé dans celle du simulateur de manière à ce que les évolutions des deux automates coïncident à travers cette réduction.

La simulation s'effectue alors sur un sous-ensemble des configurations du simulateur. Ce sous-ensemble est définit *par bloc*, il s'agit de l'ensemble $\{i(\Sigma_\mathcal{B}^{\mathcal{R}_{\vec{m}}}\}^{\mathbb{Z}^d}$.

Cette notion de simulation induit naturellement une notion d'universalité intrinsèque au modèle :

Définition 10. *Un automate cellulaire $\mathcal{A}$ est dit intrinsèquement universel s'il simule tout les autres AC, c'est à dire si $\forall \mathcal{B} \in \mathbb{CA}$, $\mathcal{B} \preccurlyeq_\sqsubseteq \mathcal{A}$.*

N. Ollinger a montré l'existence d'un automate cellulaire intrinsèquement universel ([Oll03]).

Et un certain nombre de résultats ont permis de caractériser la structure du pré-ordre induit par cet relation. On ne peut que renvoyer à nouveau vers [DMOT10b] pour un état de l'art.

Il peut également être utile de définir des automates universels pour une sous-famille d'automates. En effet, certaines de ces familles sont stables par la simulation intrinsèque, c'est le cas des AC surjectifs :

$$\mathcal{B} \preccurlyeq_\sqsubseteq \mathcal{A} \text{ et } \mathcal{A} \text{ surjectif} \Rightarrow \mathcal{B} \text{ surjectif}$$

Ceci se montre simplement, en utilisant le corollaire 1.3 du théorème de Moore-Myhill 1.2. En effet, il est évident que l'injectivité sur les configurations finies est conservée par sous-automate. De plus les fonctions $o_{\vec{m}}$ et $\sigma_{\vec{z}}$ sont bijective, donc $\mathcal{A}$ est surjectif, si et seulement si $\mathcal{A}^{<\vec{m},t,\vec{z}>}$.

On en déduit donc qu'aucun AC universel n'est surjectif, mais la question de l'existence d'un AC surjectif capable de simuler tous les surjectifs reste ouverte en 1D.

Ceci se formalise avec la définition suivante :

Définition 11. *Soit $\mathcal{C}$ un ensemble d'automates cellulaires, un automate cellulaire $\mathcal{A}$ est dit $\mathcal{C}$-intrinsèquement universel si :*

(1) $\mathcal{A} \in \mathcal{C}$,

(2) $\forall \mathcal{B} \in \mathcal{C}$, $\mathcal{B} \preccurlyeq_\sqsubseteq \mathcal{A}$.

On dit qu'un automate $\mathcal{A}$ simule fortement $\mathcal{B}$ si, en reprenant les notations de la définition 9,

$$\mathcal{B} \sqsubseteq_i \mathcal{A}^{<\vec{m_\mathcal{A}}, t_\mathcal{A}, \vec{z_\mathcal{A}}>}$$

c'est à dire si la simulation se fait sans groupage du côté du simulé.

Proposition 1.11. *Soit $\mathcal{U}$ un automate cellulaire universel, alors pour tout automate $\mathcal{A}$, $\mathcal{U}$ simule fortement $\mathcal{A}$.*

On trouvera une preuve de ce résultat dans [DMOT10a].

On introduit enfin quelques notations : Étant donné un AC $\mathcal{A}$, on note $\mathcal{S}_\mathcal{A}$ l'ensemble des automates qui simulent $\mathcal{A}$. Et pour un AC $\mathcal{A}$, un alphabet Σ, et un sous-shift par bloc $X \subseteq \Sigma^\mathbb{Z}$ fixés, $\mathcal{S}_{\mathcal{A},X}$ est l'ensemble des AC d'alphabet Σ, qui simulent $\mathcal{A}$ sur un sous-shift de simulation inclus dans X.

3.3 Simulation par facteur

Une autre simulation, introduite par G. Theyssier ([The05]) est basée sur la notion de *coloriage*, ou de *facteur*. On trouvera également l'essentiel de ses propriétés dans [DMOT10b].

Définition 12. *On dit que $\mathcal{A} = (d, \Sigma_\mathcal{A}, \mathcal{V}_\mathcal{A}, \delta_\mathcal{A})$ est un facteur de $\mathcal{B} = (d, \Sigma_\mathcal{B}, \mathcal{V}_\mathcal{B}, \delta_\mathcal{B})$, et on note $\mathcal{A} \trianglelefteq \mathcal{B}$ s'il existe un morphisme surjectif $s : \Sigma_\mathcal{B} \to \Sigma_\mathcal{A}$, tel que $\mathcal{A} \circ \overline{s} = \overline{s} \circ \mathcal{B}$.*

Et on notera parfois $\mathcal{A} \trianglelefteq_s \mathcal{B}$ pour expliciter la fonction s concernée.

Comme dans le cas de la relation de sous-automate, cette notion seule ne permet d'associer à un automate donné qu'un nombre fini de facteurs dont le nombre est borné en fonction de la taille de l'ensemble d'états.

Mais en faisant intervenir les transformations spatio-temporelles, cette définition induit une autre relation de simulation, dite *simulation par facteur*.

Définition 13. *Soit $\mathcal{A} = (d, \Sigma_\mathcal{A}, \mathcal{V}_\mathcal{A}, \delta_\mathcal{A})$ et $\mathcal{B} = (d, \Sigma_\mathcal{B}, \mathcal{V}_\mathcal{B}, \delta_\mathcal{B})$. On dit que $\mathcal{A}$ simule* par facteur *$\mathcal{B}$, si une instance groupée de $\mathcal{B}$ est un facteur d'une instance groupée de $\mathcal{A}$. Formellement :*

$$\mathcal{B} \preccurlyeq_\trianglelefteq \mathcal{A} \Leftrightarrow \exists t_\mathcal{A}, t_\mathcal{B} \in \mathbb{N}^*, \vec{z_\mathcal{A}}, \vec{z_\mathcal{B}}, \vec{m_\mathcal{A}}, \vec{m_\mathcal{B}} \in \mathbb{Z}^d \; s : \Sigma_\mathcal{A}^{m_\mathcal{A}} \to \Sigma_\mathcal{B}^{m_\mathcal{B}},$$
$$\mathcal{B}^{<\vec{m_\mathcal{B}}, t_\mathcal{B}, \vec{z_\mathcal{B}}>} \trianglelefteq \mathcal{A}^{<\vec{m_\mathcal{A}}, t_\mathcal{A}, \vec{z_\mathcal{A}}>}$$

Cette simulation est très différente de la simulation intrinsèque. Celle-ci reposait sur la recherche d'un encodage des orbites de l'automate simulé dans celles du simulateur. À l'opposé ici, chaque orbite du simulateur doit pouvoir être interprétée comme une orbite du simulé. On s'éloigne de la notion usuelle en informatique d'encodage, mais au profit d'une certaine robustesse, au sens où toutes les évolutions du simulateur correspondent à une évolution du simulé. Ceci garantit également une certaine *effectivité* de la simulation. Là ou celle-ci nécessite dans le cas intrinsèque un ensemble de configurations particulier, négligeable, elle a lieu au contraire sur *toute* les configurations dans le cas de la simulation par facteur

On peut comme pour l'universalité intrinsèque définir la notion d'universalité par facteur

Définition 14. *Un automate cellulaire $\mathcal{A}$ est dit* universel par facteur *si pour tout $\mathcal{B}$, $\mathcal{B} \preccurlyeq_{\lhd} \mathcal{A}$.*

Cependant pour cette simulation, la question de l'existence d'un automate cellulaire universel reste ouverte.

Un automate universel serait un automate pour lequel pour tout AC, toutes les orbites de l'universel seraient interprétable comme des orbites de celui-ci. Ceci semble donc plus exigeant que l'universalité intrinsèque qui permettait de se limiter à l'interprétation de certains sous-shifts bien formés du simulateurs.

Cependant on montre simplement ([DMOT10a]), en raisonnant sur les cycles de configuration uniformes, qu'il n'y a pas d'analogue au résultat 1.11 : un universel par facteur ne simulerait pas tous les automates avec un groupage seulement du côté du simulateur.

On peut également introduire comme dans le cas de l'universalité intrinsèque une notion d'universalité interne à certaines sous-familles :

Définition 15. *Soit $\mathcal{C}$ un ensemble d'automate cellulaire, un automate cellulaire $\mathcal{A}$ est dit $\mathcal{C}$-universel par facteur si :*
(1) $\mathcal{A} \in \mathcal{C}$,
(2) $\forall \mathcal{B} \in \mathcal{C}$, $\mathcal{B} \preccurlyeq_{\lhd} \mathcal{A}$.

Nous nous intéresserons particulièrement à cette simulation, et à l'universalité sur des sous-classes au chapitre 5.

4 Sous-familles

Nous finissons cette introduction des notions utiles pour la compréhension de ce travail en évoquant une dernière approche récurrente dans l'étude des automates cellulaires. Il s'agit de l'étude de sous-familles de l'ensemble des AC. En particulier certaines classes très régulières ont connu un vrai succès dans la littérature, nous en présentons quelques unes ici.

Nous nous intéressons donc à des classes définies par des restrictions syntaxiques. C'est à dire en imposant des contraintes, soit à l'ensemble d'états, soit au voisinage, soit à la règle locale. Nous n'évoquerons donc pas les résultats qui concernent d'autre classes étudiées par ailleurs, l'ensemble des AC surjectifs par exemple.

4.1 Automates additifs

La première classe que nous introduisons a été très étudiée. En effet ses propriétés simplifient grandement le passage de la règle locale à la compréhension de son comportement global [IÔN83, MR98a, CFMM00, PY02].

On définit cette classe en munissant l'ensemble d'état d'une structure de groupe, et en requérant la compatibilité de la règle de l'automate avec cette loi. Les AC de cette classe sont parfois appelés *linéaires*.

Définition 16. *Un automate $\mathcal{A}$ est dit* additif *si on peut munir l'ensemble d'état $\Sigma_{\mathcal{A}}$ d'une loi de groupe $+$ telle que pour tous $x_1, y_1, x_2, y_2, \ldots, x_{k_{\mathcal{A}}}, y_{k_{\mathcal{A}}} \in \Sigma_{\mathcal{A}}$,*

$$\delta_{\mathcal{A}}(x_1 + y_1, x_2 + y_2, \ldots, x_{k_{\mathcal{A}}} + y_{k_{\mathcal{A}}}) = \delta_{\mathcal{A}}(x_1, x_2, \ldots, x_{k_{\mathcal{A}}}) + \delta_{\mathcal{A}}(y_1, y_2, \ldots, y_{k_{\mathcal{A}}}).$$

On l'a dit, cette classe a motivé de nombreux travaux. Cependant la restriction syntaxique est très forte. Et en particulier, il est possible de montrer qu'il n'existe pas d'automate additif intrinsèquement universel. Une preuve simple présentée dans [DRT04] consiste à montrer qu'aucun additif ne simule l'automate *Just Gliders* présenté un peu plus haut.

4.2 Automates cellulaires captifs

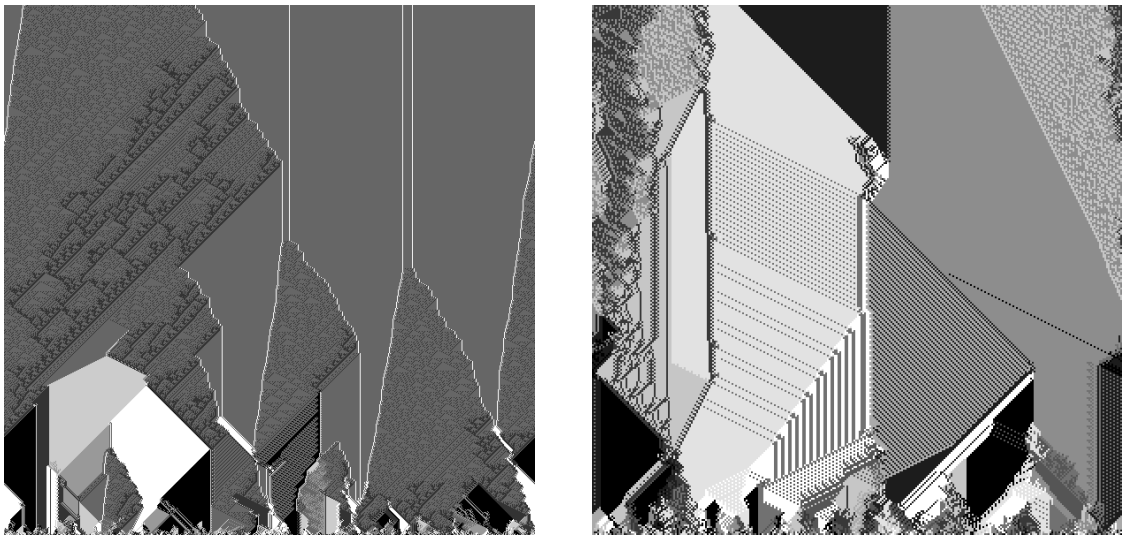
La seconde classe que nous présentons ici a été introduite par G. Theyssier ([The04]).

Elle est fondée sur des contraintes sur la règle locale : on impose que l'image d'un motif par la règle locale soit choisie parmi les états présents dans ce motif.

Définition 17. *Un automate $\mathcal{A}$ est dit captif si pour tout état $x_1, x_2, \dots x_{k_\mathcal{A}} \in \Sigma_\mathcal{A}$, sa règle locale vérifie :*

$$\delta_\mathcal{A}(x_1, x_2, \dots, x_{k_\mathcal{A}}) \in \{x_1, x_2, \dots, x_{k_\mathcal{A}}\}.$$

La figure 1.11 montre des diagrammes espace-temps d'AC captifs.



(a) Voisinage de taille 5, ensemble d'état de taille 6

(b) Voisinage de taille 5, ensemble d'état de taille 10

FIGURE 1.11: Deux diagrammes espace-temps d'AC captifs tirés aléatoirement

Ainsi pour un AC captif, tout sous-ensemble d'états est stable, et permet donc de définir un sous-automate.

En particulier, il a montré qu'il existe des automates cellulaires captifs universels. Un autre de ses résultats importants concerne le travail effectué autour de la quantification de l'universalité intrinsèque parmi cette famille. Ce type d'approche, qui lui a permis de montrer que les AC captifs sont *presque tous* universels sera discutée en détail au chapitre 3.

4.3 AC totalistiques

Enfin, nous présentons rapidement ici une dernière classe définie par contrainte sur la règle locale : les AC totalistiques. Pour ceux-ci, le nouvel état d'une cellule ne dépend que de la somme des états de ses voisins. Ici, on considère forcément des ensembles d'états $\Sigma \subseteq \mathbb{N}$.

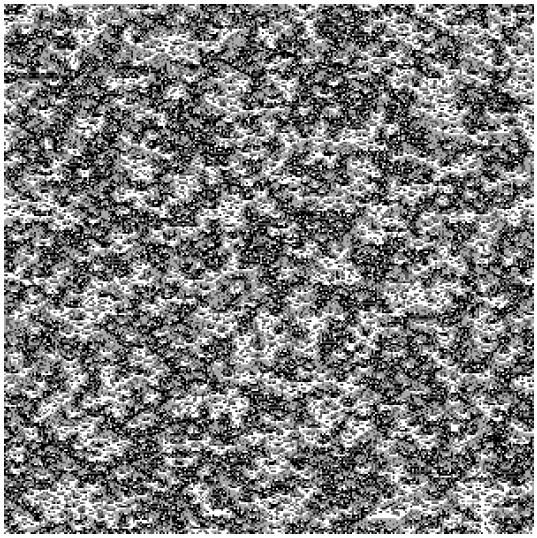
Ces automates ont été largement étudiés dans la littérature ([AC87, Gor87] par exemple). Un des aspects qui a motivé leur étude est certainement leur caractère isotrope. C'est la première famille dont la restriction vise à supprimer la dépendance en la provenance spatiale de l'information. C'est l'étude de ce genre de restrictions qui motivera notre travail au chapitre 2.

Définition

Définition 18. *Un automate cellulaire $\mathcal{A}$ est dit* totalistique, *s'il existe une fonction $\gamma : \mathbb{N} \to \Sigma_{\mathcal{A}}$ telle que sa règle locale vérifie pour tout $x_1, x_2, \ldots, x_{k_{\mathcal{A}}} \in \Sigma_{\mathcal{A}}$*

$$\delta_{\mathcal{A}}(x_1, x_2, \ldots, x_{k_{\mathcal{A}}}) = \gamma(x_1 + x_2 + \cdots + x_{k_{\mathcal{A}}})$$

Le diagramme espace-temps 1.12 illustre le comportement d'un AC totalistique 1D tiré au hasard.



(a) Voisinage de taille 3, ensemble d'état de taille 4

FIGURE 1.12: Un diagramme espace-temps d'AC totalistique tiré aléatoirement

Discussion sur l'ensemble d'états

Parmi les AC totalistiques, on peut distinguer ceux dont l'ensemble d'états est connexe, de la forme $\Sigma = \{0, \ldots, n-1\}$.

Par exemple il est montré dans [AC87] que pour tout automate il existe un AC totalistique qui le simule en temps réel et qui utilise au plus quatre fois plus d'états. Cependant ce résultat suppose que l'ensemble d'états *utilisés* n'est pas une sous-partie connexe de $\mathbb{N}$, il implique l'utilisation d'états très grands les uns par rapport aux autres.

On peut toujours compléter cet automate pour obtenir un automate universel avec un ensemble d'états connexe, mais dans ce cas le nombre d'états explose.

Problèmes de non clôture

Dans tous les cas, cette classe présente le problème de ne pas être stable par permutation de l'ensemble d'état : selon le choix de l'ordre des états dans la définition syntaxique d'un automate, celui-ci peut être totalistique ou non, c'est ce qu'illustre l'exemple suivant.

Exemple 4. On décrit un automate à trois états, $\{0,1,2\}$, de voisinage $\{0,1\}$ de manière totalistique, mais tel que la description obtenue en permuttant les états 1 et 2 ne le soit pas :
Soit $\mathcal{A}$ de règle locale $\delta_{\mathcal{A}}$ définie par $\delta_{\mathcal{A}}(x,y) = \gamma(x+y)$ avec $\gamma(z) = 2$ si $z = 4$ et $\gamma(z) = 0$ sinon. Par définition cette règle est totalistique. Mais si on échange 0 et 1, ce n'est plus le cas. En effet la règle locale nécessite alors de distinguer la somme de 1 et 1, de celle de 2 et 0, ce qui n'est pas possible pour un AC totalistique.

Comme le montre l'exemple précédent, même la définition moins restrictive, où l'on n'impose pas que l'ensemble d'états soit un segment $\{0,1,\ldots,n-1\}$ n'est pas stable par permutation des états. On s'intéressera au chapitre suivant à la clôture par la relation de renommage des états de cette famille.

Et si on se restreint à la classe des automates totalistiques à ensemble d'états connexe, cette famille n'est même plus stable par sous-automate.

Exemple 5. Soit $\mathcal{A}$ l'automate totalistique sur un alphabet $\{0,1,2,3\}$ de voisinage $\{-1,0,1\}$ défini à l'aide de la fonction γ telle que $\gamma(4) = 1$ et $\forall x \neq 4$, $\gamma(x) = 0$. Alors cet automate admet un sous-automate sur $\{0,1,3\}$. La fonction locale de celui-ci vérifie $\{0,0,0\} \mapsto 0$, $\{1,1,1\} \mapsto 0$, $\{3,3,3\} \mapsto 0$, mais $\{0,1,3\} \mapsto 1$. Or pour une règle totalistique sur $\{0,1,2\}$, la valeur du motif à trois états distincts est forcément identique à celle d'un des motifs à trois états identiques, cet automate ne peut donc pas être défini sur $\{0,1,2\}$.

Il est par contre simple de voir que la clôture de la classe des AC totalistiques à ensemble d'états connexe, par l'opération de sous-automate est l'ensemble des AC totalistiques. Cette dernière classe est en effet stable par cette opération.

C'est, entre autre, le fait que le comportement de cette classe ne soit pas satisfaisant par rapport aux propriétés usuelles qui motive l'étude conduite au chapitre suivant.

Chapitre 2

AC multi-ensemblistes

1 Définitions préliminaires 32

1.1 Multi-ensembles 32

1.2 Langages et sous-shifts multi-ensemblistes 34

2 Les automates cellulaires multi-ensemblistes 35

2.1 Définition 35

2.2 AC multi-ensemblistes et simulation 36

2.3 AC multi-ensemblistes et AC totalistiques 38

2.4 La nilpotence des AC multi-ensemblistes 40

3 Changements d'échelle et universalité multi-échelles 40

3.1 Cas unidimensionnel 41

3.2 Cas multi-dimensionnel 44

4 Outer-multi-ensemblistes 47

4.1 Définition 48

4.2 Rééchelonné d'un outer-multi-ensembliste 49

Ce chapitre est consacré à l'introduction et à l'étude de quelques propriétés spécifiques d'une sous-classe d'automates cellulaires définie syntaxiquement : les AC multi-ensemblistes. Il s'agit des AC dont la règle locale est stable par permutation des positions des voisins. Intuitivement les règles locales ne peuvent déterminer de quelles cellules viennent les informations qu'ils reçoivent.

Nous pensons que cette classe est celle qui permet de rendre compte le plus simplement, et surtout en toute généralité, des comportements isotropes des automates cellulaires.

Dans la littérature, elle n'apparait pas en tant que telle bien que entre autres, tous les automates totalistiques à 2 états soient aussi multi-ensemblistes. Après quelques définitions préliminaires, nous expliquons en quoi elle constitue une généralisation intéressante de la classe des AC totalistiques et nous traitons ensuite la question de la nilpotence parmi ces AC. Ensuite, nous nous intéressons surtout à une propriété particulière de cette classe, sa capacité à se prêter facilement à des *changements d'échelles*, notion que nous introduisons pour cette famille.

1 Définitions préliminaires

1.1 Multi-ensembles

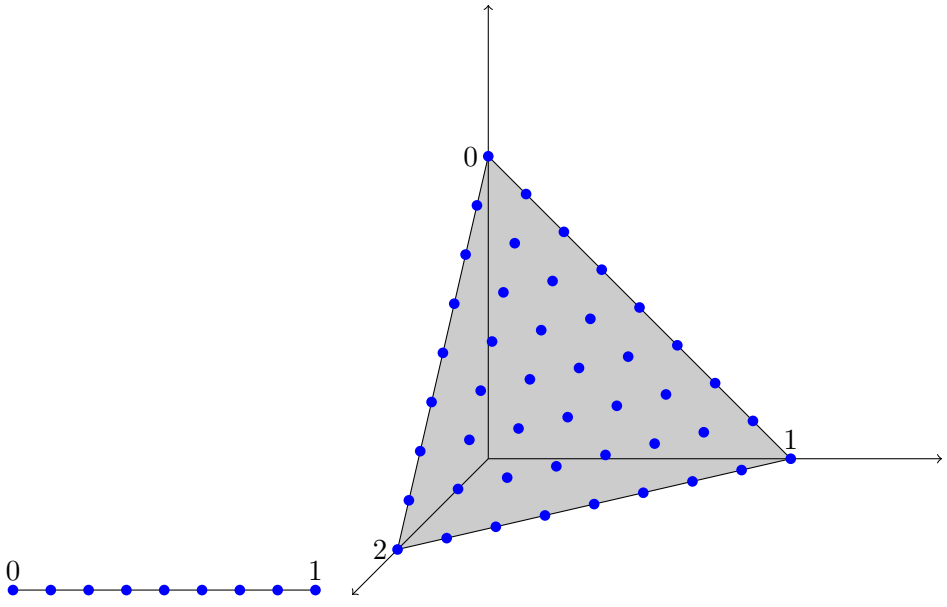
Définition

Soit Σ un alphabet de taille $\#(\Sigma) = n$. On considère que Σ est ordonné.

Formellement, un multi-ensemble de taille k sur Σ est une fonction $u : \Sigma \to \mathbb{N}$ telle que $\sum_{\alpha\in\Sigma} u(\alpha) = k$. Pour tout $q \in \Sigma$, $u(q)$ est la multiplicité de q dans u. On note $\Sigma^{(k)}$ l'ensemble des multi-ensembles à k éléments sur Σ, et $\Sigma^{(*)}$ l'ensemble des multi-ensembles de taille quelconque sur Σ.

Étant donné un mot $u \in \Sigma^*$, on lui associe le multi-ensemble $\{\{u\}\}$ défini par $\forall q \in \Sigma$, $\{\{u\}\}(q) = |u|_q$.

On peut écrire un multi-ensemble $u \in \Sigma^{(k)}$ sous la forme d'un vecteur d'entiers à n coordonnées $u = (u_1, u_2, \cdots, u_n)$, avec $u_i \in \mathbb{N}$ la multiplicité de la i-ème lettre de Σ dans u, et $k = \sum_i u_i$. Cette dernière notation permet de représenter les multi-ensembles sous forme de points à coordonnées entières, dans un espace à n dimensions. Les multi-ensembles d'une taille k donnée sont alors exactement représentés par les points de coordonnées entières $(x_1, x_2, \cdots, x_n)$, tels que $\sum_i x_i = k$ et $0 \leq x_i \leq k$. Ils appartiennent donc au même hyperplan, et plus précisément à la zone délimitée par les plans orthogonaux aux axes de coordonnées. Ceci se visualise bien dans le cas à deux ou trois états, c'est ce qu'exprime les schémas 2.1.



(a) Le cas 2 états (b) Le cas 3 états

FIGURE 2.1: Les multi-ensembles représentés comme des points à coordonnées entières.

Projeté et approximation

On utilisera dans la suite les liens entre multi-ensembles de tailles différentes. On formalise le passage entre deux tailles différentes en définissant le projeté d'un multi-ensemble d'une taille

donnée sur une autre taille.

Définition 19. *Soit $x = (x_1, x_2, \dots x_n)$ un point de $\mathbb{R}^n$ tel que pour tout $i \in \{1..n\}$ $x_i \geq 0$ et $\sum_{i\in\{1..n\}} x_i = k$ pour un entier $k \in \mathbb{N}$ quelconque.*

*Pour un entier k' quelconque, le k'-*projeté *de x est le point*

$$\pi_{k'}(x) = \left(\frac{k'x_1}{k}, \frac{k'x_2}{k}, \cdots, \frac{k'x_n}{k}\right).$$

Son projeté, *noté $\pi(x)$ est le 1-projeté : $\pi(x) = \pi_1(x)$.*

Par identification des multi-ensembles avec les points de $\mathbb{N}^d$, on a défini le k'-projeté d'un multi-ensemble de taille k. Cependant, il faut noter que le projeté d'un multi-ensemble n'est pas toujours un multi-ensemble.

La projection est une fonction bijective, et donc en particulier réversible : Pour un multi-ensemble $u \in \Sigma^{(k)}$, pour tout $k' > 0$, $\pi_k(\pi_{k'}(u)) = u$.

On définit une distance d_2 sur les multi-ensembles de $\Sigma^{(*)}$ à l'aide de la distance euclidienne dans l'espace associé $\mathbb{R}^d$: étant donné $u = (u_1, u_2, \cdots, u_n) \in \Sigma^{(k)}$ et $v = (v_1, v_2, \cdots, v_n) \in \Sigma^{(k')}$, $d_2(u, v)$ est la distance entre les 1-projetés de u et v.

$$d_2(u, v) = \sum_{i=1}^{n} \left(\frac{u_i}{k} - \frac{v_i}{k'}\right)^2.$$

Il ne s'agit pas d'une distance au sens propre car deux multi-ensembles multiples l'un de l'autre (c'est-à-dire $x = (x_1, x_2 \dots x_n)$ et $y = (y_1, y_2, \dots y_n)$ tels que $\exists p \neq 1, \forall i\ x_i = p.y_i$) sont différents mais à une distance nulle. Cependant d_2 vérifie l'inégalité triangulaire et la symétrie, il s'agit donc d'une pseudo-distance.

d_2 permet de donner un sens à l'idée de proximité entre multi-ensembles de mêmes alphabets mais de tailles différentes. Intuitivement, on juge que deux multi-ensembles sont proches si les proportions de chacun des éléments sont proches.

Définition 20. *On définit la k-approximation (ou approximation de taille k) d'un multi-ensemble u comme le multi-ensemble $r_k(u)$, de taille k, le plus proche de u selon d_2.*

S'il y a plusieurs multi-ensembles à même distance, on choisit celui dont l'écriture sous la forme de liste de coefficients est la plus petite dans l'ordre lexicographique.

Contrairement à la projection, l'approximation n'est pas forcément réversible, mais son intérêt est qu'il s'agit d'une transformation entre multi-ensembles.

Le lemme suivant nous fournit un résultat technique sur les relations entre les coordonnées d'un point et celles de son approximation.

Lemme 2.1. *Pour tout multi-ensemble u et pour tout k, si $\pi(u) = (x_1, x_2, \cdots, x_n)$ et $\pi(r_k(u)) = (y_1, y_2, \cdots, y_n)$ alors pour tout $i \in \{1, \cdots, n\}$:*

$$x_i - \frac{1}{k} < y_i < x_i + \frac{1}{k}$$

Preuve : Par l'absurde, supposons qu'il existe u et i_0 tels que $y_{i_0} \geq x_{i_0} + \frac{1}{k}$ (le cas symétrique est analogue).

Par définition $\sum_i x_i = 1 = \sum_i y_i$. Et comme $y_{i_0} > x_{i_0}$, il existe $i_1 \in \{1, \cdots, n\}$ tel que $y_{i_1} < x_{i_1}$.

Considérons maintenant le multi-ensemble $v' = (y'_1, y'_2, \cdots, y'_n)$ avec $y'_{i_0} = y_{i_0} - \frac{1}{k}$, $y'_{i_1} = y_{i_1} + \frac{1}{k}$ et $y'_i = y_i$ pour $i \notin \{i_0, i_1\}$.

On va obtenir une contradiction en comparant $d_2(u, r_k(u))$ et $d_2(u, v')$.

$$\begin{aligned} d_2(u, v') - d_2(u, \pi(r_k(u))) &= (y'_{i_1} - x_{i_1})^2 + (y'_{i_0} - x_{i_0})^2 \\ &\quad - (y_{i_1} - x_{i_1})^2 - (y_{i_0} - x_{i_0})^2 \\ &= \left(y_{i_1} - x_{i_1} + \frac{1}{k}\right)^2 - (y_{i_1} - x_{i_1})^2 \\ &\quad + \left(y_{i_0} - x_{i_0} - \frac{1}{k}\right)^2 - (y_{i_0} - x_{i_0})^2 \\ &= \frac{2}{k}\left(\frac{1}{k} + \underbrace{(y_{i_1} - x_{i_1})}_{<0} - \underbrace{(y_{i_0} - x_{i_0})}_{\geq \frac{1}{k}}\right) \\ &< 0 \end{aligned}$$

Ce qui contredit la définition de $\pi(r_k(u))$.

□

1.2 Langages et sous-shifts multi-ensemblistes

Étant donné un mot u, on dit qu'un multi-ensemble v est un *facteur multi-ensembliste* de u si v est le multi-ensemble associé à un facteur de u.

On définit de même les facteurs multi-ensemblistes d'une configuration $c \in \Sigma^{\mathbb{Z}}$.

Nous ne nous intéressons pas ici aux sous-shifts définis à l'aide de langages multi-ensemblistes, c'est-à-dire stables par permutations de l'ordre des lettres de ses mots.

Nous aurons simplement besoin dans la suite du lemme technique suivant qui caractérise une certaine famille de ces sous-shifts.

Dans celui-ci la fonction $\overline{h}$, désigne la configuration obtenue en appliquant la fonction locale $h : w_i \to i$ en chaque position d'une configuration : $\overline{h}(c)_z = h(c_z, \ldots, c_{z+k_{\mathcal{A}}-1})$.

Lemme 2.2. *Soit $w_i \in \{0, 1, \ldots, k_{\mathcal{A}} + 1\}^{(k_{\mathcal{A}})}$ le multi-ensemble formé de $k_{\mathcal{A}}$ entiers consécutifs (modulo $k_{\mathcal{A}} + 2$) commençant par i. Soit $L = \{w_0, w_1, \ldots, w_{k_{\mathcal{A}}+1}\}$ l'ensemble de ces $k_{\mathcal{A}} + 2$ multi-ensembles possibles.*

Soit S un sous-shift tel que :

(a) les facteurs multi-ensemblistes de longueur $k_{\mathcal{A}} + 2$ de S sont inclus dans L,

(b) S est stable sous l'action de $\overline{h}$.

Alors S est le sous-shift constitué soit des configurations ${}^{\omega}(0 \cdot 1 \cdots (k_{\mathcal{A}} + 1))^{\omega}$ soit des configurations ${}^{\omega}((k_{\mathcal{A}} + 1) \cdots 1 \cdot 0)^{\omega}$, soit de la réunion de ces deux ensembles.

Preuve : Nous allons montrer que dans une configuration appartenant à un sous-shift satisfaisant nos hypothèses, un mot de multi-ensemble w_i ne peut être suivi que par un mot de multi-ensemble soit $w_{i+1 \bmod k_{\mathcal{A}}+2}$, soit $w_{i-1 \bmod k_{\mathcal{A}}+2}$.

Soit c appartenant à S satisfaisant nos hypothèses, et tel que le multi-ensemble du facteur de taille $k_{\mathcal{A}}+2$ en position 0, c'est-à-dire $c_{[0..k_{\mathcal{A}}+1]}$, est w_i. On vérifie que le multi-ensemble du facteur en position 1, c'est-à-dire de $u = c_{[1..k_{\mathcal{A}}+2]}$ est

- soit le même : $\{\{u\}\} = w_i$.
- soit un multi-ensemble obtenu en changeant une seule lettre de w_i. Et les seuls multi-ensembles de L qui peuvent être obtenus ainsi sont $w_{i+1 \bmod k_{\mathcal{A}}+2}$ et $w_{i-1 \bmod k_{\mathcal{A}}+2}$.

N'importe quel autre multi-ensemble de L contient au moins deux lettres qui ne sont pas dans w_i.

Cependant, dans le premier cas, l'application de la fonction locale h donne un mot image v ayant une même lettre en deux positions consécutives. D'où $\{\{v\}\} \notin L$, ce qui contredit la condition (b).

En itérant ce raisonnement, on en déduit que chaque mot de multi-ensemble w_i est prolongé par un mot de multi-ensemble $w_{i+ \bmod k_{\mathcal{A}}+2}$ ou $w_{i- \bmod k_{\mathcal{A}}+2}$.

Il reste à vérifier un dernier point. En effet, par exemple, un facteur de multi-ensemble w_1 pourrait se prolonger en w_2, et ce dernier se prolonger à nouveau en w_1. Il faut montrer qu'aucun "demi-tour" ne se produit. Formellement, supposons qu'il existe une position, $z \in \mathbb{Z}$ telle que $\{\{c_{[z..z+k_{\mathcal{A}}+1]}\}\} = w_i$, $\{\{c_{[z+1..z+k_{\mathcal{A}}+2]}\}\} = w_{i+1 \bmod k_{\mathcal{A}}+2}$ et $\{\{c_{[z+2..z+k_{\mathcal{A}}+3]}\}\} = w_i$. Alors l'image par h de c contient en position z un mot v contenant le facteur $i \cdot (i+1) \cdot i$, ce qui contredit (b).

Ceci suffit à conclure la preuve.

□

2 Les automates cellulaires multi-ensemblistes

2.1 Définition

Intuitivement, les automates cellulaires multi-ensemblistes sont ceux dont la règle locale ne tient pas compte de la position de chacun des états dans le motif de voisinage. Il s'agit donc de ceux dont la règle locale est invariante par permutation des voisins.

Définition 21. *Un automate cellulaire $\mathcal{A}$ est dit multi-ensembliste, noté $\mathcal{A} \in \mathcal{MS}$ si sa règle locale vérifie*

$$\forall u, v \in \Sigma_{\mathcal{A}}^{k_{\mathcal{A}}}, \text{ si } \{\{u\}\} = \{\{v\}\} \text{ alors } \delta_{\mathcal{A}}(u) = \delta_{\mathcal{A}}(v).$$

Les figures 2.2 illustrent deux évolutions d'AC captifs à partir d'une configuration cyclique.

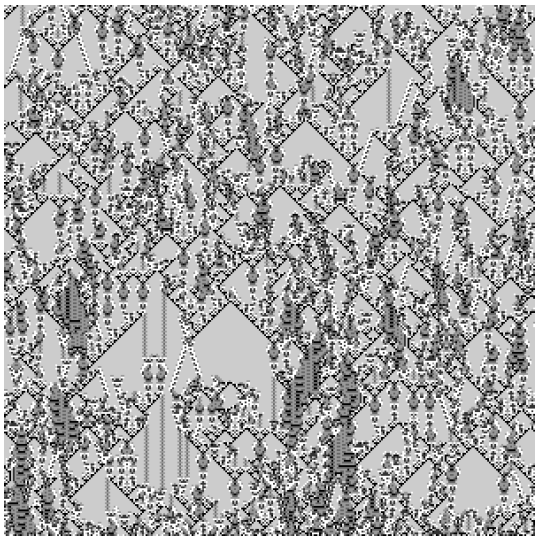
Souvent lorqu'il n'y aura pas d'ambiguïté, si $\mathcal{A} \in \mathcal{MS}$, on notera $\delta_{\mathcal{A}}(x)$ avec x un multi-ensemble.

Le nombre de tels automates d'une taille n, k donnée est $\#(\mathcal{MS}_{n,k}) = \binom{n+k-1}{n-1}$.

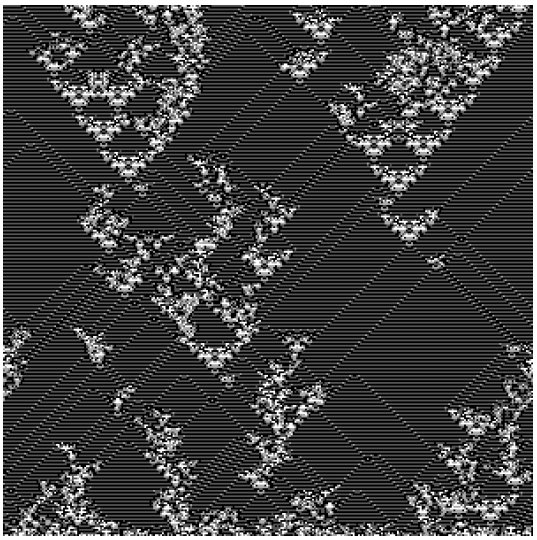
Proposition 2.3. *La classe des AC multi-ensembliste est stable par sous-automate, par facteur, par isomorphisme de l'ensemble d'états.*

Cette propriété découle directement de la définition, et ne nécessite réellement aucune preuve.

Les automates cellulaires multi-ensemblistes ne sont jamais surjectifs, on le montre facilement par non injectivité sur les configurations finies.



(a) Voisinage de taille 3, ensemble d'état de taille 6



(b) Voisinage de taille 5, ensemble d'état de taille 4

FIGURE 2.2: Deux diagrammes espace-temps d'AC multi-ensemblistes tirés aléatoirement à partir d'un segment initial aléatoire.

2.2 AC multi-ensemblistes et simulation

Il est possible de simuler un automate cellulaire $\mathcal{A}$ quelconque par un automate cellulaire multi-ensembliste.

Pour cela, il suffit d'ajouter à l'alphabet $\Sigma_\mathcal{A}$ de l'automate de départ une couche de marqueurs positionnels $\{1, 2, \ldots, k_\mathcal{A} + 1\}$: ceci revient à travailler sur l'alphabet $\Sigma_\mathcal{A} \times \{1, 2, \ldots, k_\mathcal{A} + 1\}$. On construit alors un automate multi-ensembliste sur ce nouvel alphabet et de même voisinage $k_\mathcal{A}$ que $\mathcal{A}$. Sur les configurations de seconde couche périodique de période $1 \cdot 2 \cdots (k_\mathcal{A} + 1)$ celui-ci peut détecter quelle est sa position (à l'aide du marqueur manquant), et sait à quelle marqueur est associé un état donné de son voisinage. Il peut ainsi maintenir la seconde couche inchangée, et appliquer la règle $\delta_\mathcal{A}$ à la première couche en ordonnant les états présents. On vérifie facilement que tout ceci est possible avec une règle multi-ensembliste.

Cette transformation, explicitée convenablement, nous permettrait déjà d'obtenir le résultat important suivant :

Théorème 2.4. *Il existe un automate cellulaire multi-ensembliste intrinsèquement universel.*

Cependant cette transformation n'est pas entièrement satisfaisante, on aimerait décrire une transformation plus fidèle, c'est-à-dire qui associe à un automate quelconque un automate cellulaire multi-ensembliste, *de puissance de simulation comparable*.

Pour aller dans cette direction, nous introduisons une transformation légèrement différente, mais qui préserve l'universalité intrinsèque (théorème 2.6).

Cette transformation Ψ utilise l'idée évoquée plus haut de marqueurs positionnels.

Soit $\mathcal{A}$ un automate cellulaire quelconque d'ensemble d'état $\Sigma_\mathcal{A}$, de voisinage de taille $k_\mathcal{A}$, et de fonction locale $\delta_\mathcal{A}$.

Pour construire $\Psi(\mathcal{A})$, on va lui rajouter une couche de marqueurs positionnels qui permettent à la règle multi-ensembliste de reconstituer l'information positionelle (sur certains sous-shifts), et

donc d'émuler les transitions de $\mathcal{A}$. Pour montrer que $\Psi(\mathcal{A})$ simule $\mathcal{A}$, il suffit de supposer ce marquage positionnel statique, mais dans notre construction il subira un décalage au cours du temps, ce qui sera utile pour garantir que $\Psi(\mathcal{A})$ n'est universel que si $\mathcal{A}$ l'est.

Formellement, le voisinage de $\Psi(\mathcal{A})$ est le même que celui de $\mathcal{A}$ et son ensemble d'état $\Sigma_{\Psi(\mathcal{A})}$ est donné par $\Sigma_{\Psi(\mathcal{A})} = (\Sigma_{\mathcal{A}} \times \{0, 1, \cdots, k_{\mathcal{A}} + 1\}) \bigcup \{\#\}$.

$\Psi(\mathcal{A})$ émule le comportement de $\mathcal{A}$ sur la première couche des configurations dont la seconde couche est périodique de motif $0 \cdot 1 \cdot 2 \cdots (k_{\mathcal{A}} + 1)$. Et simultanément, il décale cette seconde couche de $\lfloor \frac{k_{\mathcal{A}}+1}{2} \rfloor$.

Pour cela la règle locale de $\Psi(\mathcal{A})$ est définie de la manière suivante :
s'il existe $a_1, \cdots, a_{k_{\mathcal{A}}} \in \Sigma_{\mathcal{A}}$ et $i \in \{0..k_{\mathcal{A}} + 1\}$ tels que

$$\{\{u\}\} = \{\{(a_1, i), (a_2, i + 1 \text{ mod } k_{\mathcal{A}} + 2) \cdots (a_{k_{\mathcal{A}}}, i + k_{\mathcal{A}} - 1 \text{ mod } k_{\mathcal{A}} + 2)\}\},$$

alors on dit que u est un motif *légal*, et

$$\delta_{\Psi(\mathcal{A})}(u) = (\delta_{\mathcal{A}}(a_1, \cdots, a_{k_{\mathcal{A}}}), i + \lfloor (k_{\mathcal{A}} + 1)/2 \rfloor \text{ mod } k_{\mathcal{A}} + 2)$$

sinon

$$\delta_{\Psi(\mathcal{A})}(u) = \#.$$

Avec cette définition, on peut montrer le résultat suivant, qui sera utile pour montrer le théorème 2.6.

Lemme 2.5. *La transformation $\Psi : \mathbb{CA} \to \mathcal{MS}$ vérifie : Pour tout AC $\mathcal{B}$ à deux états, sans état envahissant, alors $\Psi(\mathcal{A})$ simule fortement $\mathcal{B}$ si et seulement si $\mathcal{A}$ simule fortement $\mathcal{B}$.*

Pour démontrer ce résultat, on se repose largement sur le lemme suivant qui caractérise les sous-shifts ayant comme multi-ensemble de leurs facteurs des éléments d'un type donné bien précis :

On peut maintenant prouver le lemme 2.5.

Preuve : [du lemme 2.5]. Soit $\mathcal{B}$ un automate à deux états et sans état envahissant. Et on suppose que $\mathcal{B}$ est simulé fortement par $\Psi(\mathcal{A})$. On va montrer que $\mathcal{A}$ simule aussi $\mathcal{B}$. Soit i : $\Sigma_{\mathcal{B}} \to \Sigma^m_{\Psi(\mathcal{A})}$ pour un certain m, une fonction injective, et t et z tels que $\mathcal{B} \sqsubseteq_i \Psi(\mathcal{A})^{<m,t,z>}$.

Soit $b_0 = i(0)$ et $b_1 = i(1)$ les mots de $\Sigma^m_{\Psi(\mathcal{A})}$ utilisés dans la simulation, et $S = \bar{i}(\Sigma^{\mathbb{Z}}_{\mathcal{A}})$.

Rappelons qu'un motif de longueur $k_{\mathcal{A}}$ est dit légal si son multi-ensemble est l'un de ceux apparaissant dans la simulation de $\mathcal{A}$ par $\Psi(\mathcal{A})$.

On considère maintenant les orbites du sous-shift de simulation de $\mathcal{B}$ par $\Psi(\mathcal{A})$, $\bigcup_{s\in\mathbb{N}} \Psi(\mathcal{A})^s(S)$.

De deux choses l'une : soit ces orbites n'impliquent que des motifs légaux, soit des motifs illégaux apparaissent au cours de la simulation.

Supposons qu'un motif illégal apparaisse. Par définition, son image par $\Psi(\mathcal{A})$ est $\#$. Donc il existe une configuration $c \in \bigcup_{s\in\mathbb{N}} \Psi(\mathcal{A})^s(S)$ et une position x telle que $c_x = \#$. Comme $\#$ est un état envahissant, pour tout $i \in \mathbb{N}$ $\#^i$ apparait dans $\bigcup_{s\in\mathbb{N}} \Psi(\mathcal{A})^s(S)$. On en déduit que soit b_0 soit b_1 vaut $\#^m$, et donc que l'état correspondant (0 ou 1) de $\mathcal{B}$ est envahissant, ce qui contre-dit le choix initial de $\mathcal{B}$.

Par conséquent, la simulation de $\mathcal{B}$ par $\Psi(\mathcal{A})$ n'implique que des configurations dont les facteurs de longueur k sont légaux. En particulier, la seconde couche de chacun de ces mots, celle des marqueurs positionnels, contient exactement k états consécutifs (modulo $k_{\mathcal{A}} + 2$). Comme de plus le sous-shift de simulation est stable, on peut utiliser le lemme 2.2. Celui-ci nous garantit que les seules configurations dont chaque facteur est légal sont exactement celles de seconde couche

périodique de période ${}^{\omega}0 \cdot 1 \cdots (k_{\mathcal{A}}+1)^{\omega}$ ou bien son renversé ${}^{\omega}(k_{\mathcal{A}}+1)\ldots 1 \cdot 0^{\omega}$. Dans la suite on supposera, sans perte de généralité, que les configurations du sous-shift de simulation sont du premier type.

En particulier les configurations ${}^{\omega}b_0^{\omega}$, ${}^{\omega}b_1^{\omega}$, ${}^{\omega}(b_0b_1)^{\omega}$ ont cette même seconde couche périodique. Donc les mots b_1 et b_2, ont une longueur m multiple de $k_{\mathcal{A}}+2$ et une seconde couche de la forme

$$d \cdot (d+1) \cdots (k_{\mathcal{A}}+1) \left(0 \cdot 1 \cdots (k_{\mathcal{A}}+1)\right)^{p} 0 \cdot 1 \cdots (d-1)$$

avec $p = \frac{m}{k_{\mathcal{A}}} - 1$.

Et comme la fonction $i : \Sigma_{\mathcal{B}} \to \Sigma_{\Psi(\mathcal{A})}^{m}$ est injective, les premières composantes de b_0 et b_1 sont différentes. Donc la fonction $j : \Sigma_{\mathcal{B}} \to \Sigma_{\mathcal{A}}^{m}$ définie par $j = \pi_1 \circ i$, qui à un motif de $\Sigma_{\mathcal{B}}$ associe la première composante de son image par i, est injective.

En notant $\overline{j}$ l'extension continue de j, et $\overline{\pi_1}$ celle de π_1, on a donc montré qu'il existe une fonction injective j telle que

$$\overline{j} \circ \mathcal{B} = \overline{\pi_1} \circ \overline{i} \circ \mathcal{B} = \overline{\pi_1} \circ \Psi(\mathcal{A})^{<m,t,z>} \circ \overline{i} = \mathcal{A}^{<m,t,z>} \circ \overline{j}$$

On a montré que $\mathcal{A}$ simule $\mathcal{B}$.

□

On peut maintenant démontrer le résultat suivant qui justifie l'intérêt de cette transformation, et introduit un lien (un tout petit peu) plus fin entre l'ensemble des AC en général et l'ensemble des AC multi-ensemblistes.

Théorème 2.6. *La transformation $\Psi : \mathbb{CA} \to \mathcal{MS}$ vérifie :*

(1) $\Psi(\mathcal{A})$ simule $\mathcal{A}$,

(2) $\Psi(\mathcal{A})$ est universel si et seulement si $\mathcal{A}$ l'est.

Preuve : On note $f : \Sigma_{\mathcal{A}}^{k_{\mathcal{A}}+2} \to \Sigma_{\Psi(\mathcal{A})}^{k_{\mathcal{A}}+2}$ la fonction qui a un mot $u = u_0u_1 \cdots u_{k_{\mathcal{A}}+1}$ associe le mot $f(u) = (u_0, 0)(u_1, 1) \cdots (u_{k_{\mathcal{A}}+1}, k_{\mathcal{A}}+1)$.

Et on note $\overline{F} : \Sigma_{\mathcal{A}}^{\mathbb{Z}} \to \Sigma_{\Psi(\mathcal{A})}^{\mathbb{Z}}$ l'extension uniforme de f obtenue en l'appliquant bloc à bloc.

L'ensemble $\overline{F}(\Sigma_{\mathcal{A}}^{\mathbb{Z}})$ est exactement le sous-shift formé par les configurations dont la seconde composante est périodique de période $0 \cdot 1 \cdot 2 \cdots k_{\mathcal{A}}+1$.

Et on vérifie facilement que $\forall c \in \Sigma_{\mathcal{A}}^{\mathbb{Z}}$, $\overline{F} \circ \mathcal{A}(c) = \Psi(\mathcal{A}) \circ \overline{F}(c)$. La bijection de $\Sigma_{\Psi(\mathcal{A})}^{\mathbb{Z}}$ vers $\Sigma_{\mathcal{A}}^{\mathbb{Z}}$ définie par l'oubli de la seconde composante est compatible avec les règles de transitions : à travers cette bijection, $\Psi(\mathcal{A})$ simule $\mathcal{A}$.

La deuxième partie est une conséquence du lemme 2.5. En effet,il existe un automate cellulaire $\mathcal{B}$ universel, à deux états, et sans état envahissant. Par conséquent, d'après la propriété 1.11, un automate cellulaire est universel si et seulement s'il simule fortement $\mathcal{B}$. Le lemme permet de conclure.

□

2.3 AC multi-ensemblistes et AC totalistiques

Les AC multi-ensemblistes, tout comme les AC totalistiques forment une famille définie par restriction de la puissance de la règle locale. On s'intéresse ici au lien entre ces deux familles.

Rappelons que par définition un AC $\mathcal{A}$ est totalistique s'il existe une fonction $\gamma : \mathbb{N} \to \Sigma_{\mathcal{A}}$ telle que $\forall x_1, \ldots, x_{k_{\mathcal{A}}} \in \Sigma_{\mathcal{A}} \subset \mathbb{N}$, $\delta_{\mathcal{A}}(x_1, \ldots, x_{k_{\mathcal{A}}}) = \gamma(x_1 + \cdots + x_{k_{\mathcal{A}}})$. Un AC totalistique est donc toujours multi-ensembliste.

On peut alors se demander dans quels cas la réciproque est vraie. Pour répondre à cette question, nous explicitons d'abord l'isomorphe d'un automate par changement de l'alphabet.

Étant donné un automate $\mathcal{A} = (\mathbb{Z}^d, \Sigma_\mathcal{A}, \mathcal{V}_\mathcal{A}, \delta_\mathcal{A})$, et un alphabet $\Sigma_\mathcal{B}$ tel que $|\Sigma_\mathcal{A}| = |\Sigma_\mathcal{B}|$, pour chaque isomorphisme $i : \Sigma_\mathcal{A} \to \Sigma_\mathcal{B}$, on définit l'automate isomorphe à $\mathcal{A}$ sur $\Sigma_\mathcal{B}$ par $i : \mathcal{I}_i(\mathcal{A}) = (\mathbb{Z}^d, \Sigma_\mathcal{B}, \mathcal{V}_\mathcal{A}, \delta_\mathcal{B})$ avec $\forall x_1, \ldots, x_{k_\mathcal{A}} \in \Sigma_\mathcal{B}$, $\delta_\mathcal{B}(x_1, \ldots, x_{k_\mathcal{A}}) = i(\delta_\mathcal{A}(i^{-1}(x_1), \ldots, i^{-1}(x_{k_\mathcal{A}})))$.

Proposition 2.7. *Un AC $\mathcal{A} = (\mathbb{Z}^d, \Sigma_\mathcal{A}, \mathcal{V}_\mathcal{A}, \delta)$ est multi-ensembliste ssi il existe un ensemble d'états $\Sigma_\mathcal{B}$ tel que $|\Sigma_\mathcal{A}| = |\Sigma_\mathcal{B}|$ et un isomorphisme $i : \Sigma_\mathcal{A} \to \Sigma_\mathcal{B}$ tel que $\mathcal{I}_i(\mathcal{A})$ est totalistique.*

Preuve : Commençons par le sens indirect. Soit $\mathcal{A} = (\mathbb{Z}^d, \Sigma_\mathcal{A}, \mathcal{V}, \delta)$, $\Sigma_\mathcal{B}$ et $i : \Sigma_\mathcal{A} \to \Sigma_\mathcal{B}$ tel que $\mathcal{I}_i(\mathcal{A})$ soit totalistique.

Alors par définition il existe $\gamma : \mathbb{N} \to \Sigma_\mathcal{B}$ telle que $\forall x_1, \ldots, x_{k_\mathcal{A}} \in \Sigma_\mathcal{A}$, $\delta_\mathcal{B}(x_1, \ldots, x_{k_\mathcal{A}}) = \gamma(x_1 + \cdots + x_{k_\mathcal{A}})$. Donc $\forall y_1, \ldots, y_{k_\mathcal{A}}$, on a :

$$\delta_\mathcal{A}(y_1, \ldots, y_{k_\mathcal{A}}) = i^{-1}(\delta_\mathcal{B}(i(y_1), \ldots, i(y_{k_\mathcal{A}}))) = i^{-1} \circ \gamma(i(y_1), \ldots, i(y_{k_\mathcal{A}})).$$

Par conséquent, $\delta_\mathcal{A}$ est invariant par permutation des positions des cellules du voisinage, $\mathcal{A}$ est multi-ensembliste.

Réciproquement, soit $\mathcal{A} = (\mathbb{Z}^d, \Sigma_\mathcal{A}, \mathcal{V}, \delta)$ un AC multi-ensembliste. Notons $\Sigma_\mathcal{A} = \{a_1, \ldots, a_{n_\mathcal{A}}\}$. Considérons maintenant l'alphabet $\Sigma_\mathcal{B} = \{b_1, b_2, \ldots, b_{n_\mathcal{A}}\}$ défini par $\forall i$, $b_i = (k_\mathcal{A} + 1)^{i-1}$, et $\phi : \Sigma_\mathcal{A} \to \Sigma_\mathcal{B}$ définie par $\phi(a_i) = b_i$.

Pour un entier x, on introduit son écriture en base $k_\mathcal{A} + 1$:

$$x = \sum_{i \geq 0} \mathbf{x}^i_{k_\mathcal{A}+1}(x)(k_\mathcal{A} + 1)^i.$$

Soit maintenant $\gamma : \mathbb{N} \to \Sigma_\mathcal{B}$ la fonction définie par

$$\gamma(x) = \phi(\delta_\mathcal{A}(\mathbf{x}^1_{k_\mathcal{A}+1}(x), \mathbf{x}^2_{k_\mathcal{A}+1}(x), \ldots, \mathbf{x}^{k_\mathcal{A}}_{k_\mathcal{A}+1}(x))).$$

On vérifie alors que la règle locale de l'automate $\mathcal{B} = \mathcal{I}_i(\mathcal{A})$ vérifie

$$\forall x_1, \ldots, x_{k_\mathcal{A}} \in \Sigma_\mathcal{A}, \delta_\mathcal{B}(x_1, \ldots, x_{k_\mathcal{A}}) = \gamma(x_1 + \cdots + x_{k_\mathcal{A}})$$

$\mathcal{B}$ est bien totalistique ce qui conclut la preuve.

□

Cette proposition signifie que la classe des automates cellulaires multi-ensemblistes est la clôture par l'opération de renommage de l'ensemble d'états de la classe des automates cellulaires totalistiques.

Si on s'intéresse aux AC totalistiques ayant un ensemble d'états connexe, l'équivalence ne fonctionne plus dès que les AC considérés ont strictement plus de deux états et un voisinage de taille au moins deux. En effet deux multi-ensembles d'états $\{x, x+2\}$ et $\{x+1, x+1\}$ ne sont pas discernables pour un totalistique , alors qu'ils le sont pour un multi-ensembliste.

Mais, comme la classe des AC totalistiques est exactement la clôture de la classe des AC totalistiques à ensemble d'états connexe, par l'opération de sous-automates, *les AC multi-ensemblistes forment la clôture par permutation des états puis sous-automates de la famille des AC totalistiques à ensemble d'états connexe*.

2.4 La nilpotence des AC multi-ensemblistes

Lorsqu'on se restreint à une sous-classe d'AC, une question qui vient rapidement est la décidabilité de la nilpotence dans cette classe restreinte. Le résultat suivant, qui repose sur une variation de la transformation Ψ introduite au 2.2 montre que l'indécidabilité de cette question est maintenue lorsqu'on se restreint aux multi-ensemblistes. Ceci nous assure que la dynamique des AC multi-ensembliste maintient une certaine complexité.

Proposition 2.8. *La nilpotence des automates multi-ensemblistes est indécidable.*

Preuve : On le montre par réduction du problème de la nilpotence des automates multi-ensemblistes avec un état envahissant à celui des automates généraux ayant un état envahissant.

On s'inspire de la fonction Ψ qui nous a permis de prouver l'existence d'un automate cellulaire multi-ensembliste intrinsèquement universel.

En particulier, on conserve le codage positionnel introduit plus haut en associant à un automate $\mathcal{A}$ de taille $n_{\mathcal{A}}$,$k_{\mathcal{A}}$ et d'alphabet $\Sigma_{\mathcal{A}}$ un automate $\Psi(\mathcal{A})$ d'alphabet $\Sigma_{\Psi(\mathcal{A})} = \Sigma_{\mathcal{A}} \times \{0, 1, \cdots, k_{\mathcal{A}}\}$.

Plus précisément, dorénavant, on se limite au cas des automates $\mathcal{A}$ ayant un état envahissant, qu'on notera $\#$, et on modifie Ψ comme suit :

- $\Sigma_{\Psi(\mathcal{A})} = \Sigma_{\mathcal{A}} \setminus \{\#\} \times \{0, 1, \cdots, k_{\mathcal{A}}\} \cup \{\#\}$. En fait on a fusionné les états ayant l'état $\#$ sur la première composante.
- L'état $\#$ est envahissant aussi dans $\Psi(\mathcal{A})$.
- On conserve le fonctionnement conforme à $\mathcal{A}$ et qui maintient le codage positionnel correct.
- Les autres transitions donnent toutes l'état $\#$.

Avec cette nouvelle version de Ψ, on va maintenant montrer que $\mathcal{A}$ est nilpotent si et seulement si $\Psi(\mathcal{A})$ l'est.

Notons $i : \Sigma_{\mathcal{A}}^{k_{\mathcal{A}}} \to \Sigma_{\Psi(\mathcal{A})}^{k_{\mathcal{A}}}$ la fonction injective définie par $i(a_1 \cdots a_{k_{\mathcal{A}}}) = p_1(a_1) \cdots p_{k_{\mathcal{A}}}(a_{k_{\mathcal{A}}})$ avec $p_i : \Sigma_{\mathcal{A}} \to \Sigma_{\Psi(\mathcal{A})}$ tel que si $x = \#$, $p_i(x) = \#$ et sinon $p_i(x) = (x, i)$.

Si $\bar{i}$ est l'extension continue de i, on a bien $\forall c \in \Sigma_{\mathcal{A}}^{\mathbb{Z}}$, $\bar{i} \circ \mathcal{A}(c) = \Psi(\mathcal{A}) \circ \bar{i}(c)$. Et on sait également que $\bar{i}(c) = {}^{\omega}\#^{\omega}$ si et seulement si $c = {}^{\omega}\#^{\omega}$.

Supposons que $\Psi(\mathcal{A})$ est nilpotent. Alors il existe un entier s tel que $\forall c \in \Sigma_{\Psi(\mathcal{A})}^{\mathbb{Z}}$, $\Psi(\mathcal{A})^s(c) = {}^{\omega}\#^{\omega}$. Pour toute configuration $c \in \Sigma_{\mathcal{A}}^{\mathbb{Z}}$, on a $\bar{i} \circ \mathcal{A}^s(c) = \Psi(\mathcal{A})^s \circ \bar{i}(c) = \Psi(\mathcal{A})^s(\bar{i}(c)) = {}^{\omega}\#^{\omega}$. Par conséquent $c = {}^{\omega}\#^{\omega}$. Et donc $\mathcal{A}$ est nilpotent.

Supposons maintenant que $\mathcal{A}$ est nilpotent. Considérons un mot $u \in \Sigma_{\Psi(\mathcal{A})}^*$, fini, de longueur $k_{\mathcal{A}} + (k_{\mathcal{A}} - 1)(s - 1)$ et son image après s étape $\Psi(\mathcal{A})^s(u) \in \Sigma_{\Psi(\mathcal{A})}$. Si u est l'image d'un mot de $\Sigma_{\mathcal{A}}$ par i, alors par nilpotence de $\mathcal{A}$ et compatibilité de i avec les règles de $\mathcal{A}$ et $\Psi(\mathcal{A})$, $\Psi(\mathcal{A})^s(u) = \#$. Sinon, si ce n'est pas le cas, en une ou deux étapes (d'après le lemme 2.2) un état $\#$ apparait, et comme il est envahissant, alors $\Psi(\mathcal{A})^s(u) = \#$.

Par conséquent, $\forall u \in \Sigma_{\Psi(\mathcal{A})}^*$, $\Psi(\mathcal{A})^s(u) \in \{\#\}^*$, $\Psi(\mathcal{A})$ est nilpotent.

□

3 Changements d'échelle et universalité multi-échelles

Nous avons introduit une classe d'AC qui correspond à l'invariance par permutation sur le voisinage. Nous abordons maintenant un aspect lié à cette définition, qui est intéressant dans la mesure où il repose sur des propriétés spécifiques à cette classe, et qu'on ne voit pas de manière évidente comment l'adapter à l'ensemble des AC.

Il s'agit de la capacité des AC multi-ensemblistes à s'adapter à différentes échelles. En effet, étant donné une règle locale, il peut être utile de l'adapter pour d'autres tailles de voisinage avec l'idée d'obtenir un AC au comportement similaire. Dans le cas général ceci est compliqué, mais pour les multi-ensemblistes, nous allons voir que ceci peut se formaliser très facilement. Cependant le comportement obtenu n'est pas forcément proche de l'AC de départ.

Cette capacité est sans doute un des éléments qui explique que les règles multi-ensemblistes ou des règles proches (par exemple les *outer-multi-ensemblistes* évoqués au 4) apparaissent dans les études de modélisations. On imagine en effet que le fait de pouvoir raffiner plus ou moins les pas de discrétisation spatiale est un atout indéniable. Ceci donne un argument supplémentaire dans l'étude de ces règles.

D'un point de vue théorique, les travaux de M. Pivato et K. M. Evans ([Eva01, Piv07]) ont ouvert cette voie. Ils ont étudié le comportement de la règle du Jeu de la vie en considérant des automates à différentes échelles. Ils ont définis ainsi une famille d'automates, les *Larger Than Life*, qui utilisent des voisinages plus grands, et dont la règle d'évolution est une adaptation de la règle du Jeu de la Vie, à un plus grand nombre de voisins, en conservant les proportions déterminant de la *survie* ou de l'apparition d'un état. Ceci leur a permis d'observer que le même type de particules ou *gliders* qui apparaissent dans le jeu de la vie apparaissent aussi à des échelles plus grandes.

Par ailleurs, dans [Piv07] M. Pivato pose la question de la limite continue de tels objets, introduit une notion d'automates euclidiens, et discute un certain nombre de ces possibilités. Tout ceci semble prometteur.

Pour notre part, nous introduisons ici une notion de changement d'échelle générale, valable pour toutes les règles multi-ensemblistes.

Notre formalisme nous permet de prouver formellement que certaines règles d'AC conservent l'universalité lorsqu'on change d'échelle. Ceci n'a rien d'évident car ces propriétés liées au calcul, font intervenir des encodages à priori complexes, et qui n'ont aucune raison à priori d'être maintenus lors des changements d'échelles. En particulier, aucun autre `Larger than Life`, que le `Game of Life` originel n'a été montré universel à ce jour, ceci est discuté dans [Eva05].

Nous construisons d'abord un tel AC en dimension 1, puis nous montrons un résultat légèrement plus faible pour les dimensions supérieures.

3.1 Cas unidimensionnel

Définition 22. *Soit $\mathcal{A} \in \mathcal{MS}$ un automate cellulaire multi-ensembliste d'alphabet $\Sigma_\mathcal{A}$ de taille $n_\mathcal{A}$ et de voisinage connexe centré de taille $k_\mathcal{A}$. Son* rééchelonné *de taille k (ou k-rééchelonné) est l'automate $\mathcal{R}_\mathcal{A}(k)$ de même ensemble d'état, de voisinage de taille k et de règle locale définie par $\forall u \in \Sigma_\mathcal{A}^k$, $\delta_{\mathcal{R}_\mathcal{A}}(u) = \delta_\mathcal{A}(r_{k_\mathcal{A}}(u))$.*

Intuitivement, l'image d'un motif de n'importe quelle taille est l'image de sa k-approximation, avec k la taille de la règle locale de départ.

À priori, on peut de cette manière définir des automates aux rayons, soit plus grands, soit plus petits que celui de départ.

Exemple 6. L'automate `max`, déjà introduit, est un automate multi-ensembliste de voisinage de taille 3. Si on considère son rééchelonné à une taille plus grande, par exemple 5, on obtient un comportement différent. Intuitivement, il y a un nombre minimal de cellules proches qui doivent être dans l'état envahissant pour que celui-ci se propage réellement.

Exemple 7. L'automate $\texttt{majorité}(n,k)$ est défini, sur un alphabet Σ_n et avec un voisinage Moore de taille k, par la règle locale qui associe à un motif l'état le plus petit parmi ceux qui sont les plus représentés dans le voisinage.
On se rend compte facilement que cette règle a une capacité intéressante : rééchelonnée à une taille $k.b+b-1$ avec b une taille quelconque, elle simule le comportement de la règle initiale pour toute configuration initiale formées uniquement de blocs uniformes de taille b. Ces blocs sont maintenus et leur état évolue conformément à la règle de départ.

On le voit sur ces exemples, le comportement d'un rééchelonné peut ressembler dans certains cas au comportement de départ, mais ce n'est pas toujours vrai. On va cependant montrer maintenant que même des propriétés liées à des constructions particulières, très exigeantes du point de vue de la structure peuvent se maintenir par changement d'échelles.

Théorème 2.9. *Il existe un automate cellulaire multi-ensembliste, et une taille k_0, tels que pour tout $k \geq k_0$ impair, son k-rééchelonné est universel.*

Pour construire un automate au comportement similaire à toutes les échelles, on s'intéresse à des simulations sur des sous-shifts faisant intervenir des blocs d'états distincts. Formellement, pour un k_0 impair fixé, on définit donc $\mathcal{D}_{k_0}$ l'ensemble des configurations dont les mots de longueur k_0 contiennent k_0 lettres différentes :

$$\mathcal{D}_{k_0} = \{c \in \Sigma^{\mathbb{Z}} : \forall i,j \in \mathbb{Z}, \text{ si } i \neq j \text{ et } |i-j| \leq k_0, \text{ alors } c(i) \neq c(j)\}.$$

Plus précisément on va considérer un automate multi-ensembliste universel $\mathcal{A}_0$ de voisinage $k_0 = 3$ qui simule un automate cellulaire universel $\mathcal{U}$ sur un sous-shift de simulation inclut dans $\mathcal{D}_{k_0}$. Pour obtenir un tel automate, il suffit de considérer l'image par la fonction Ψ définie au 2.2 d'un automate cellulaire universel de rayon 1 quelconque. Et l'on va montrer que certains automates $\mathcal{A}_k = \mathcal{R}_{\mathcal{A}_0}(k)$ simulent $\mathcal{U}$ sur des configurations particulières.

Pour un $k \geq k_0$ impair donné, soit $b \in \mathbb{N}$ et $1 \leq e \leq k_0+1$ définis de manière unique par $k = b(k_0+1)-e$. Dorénavant, on se limite aux valeurs impairs de k pour lesquelles $b \geq e$. Si $k_0 = 3$, ceci revient à considérer k impair de valeur au moins 7.

On définit pour chaque k la fonction injective $i_k : \Sigma_{n_0} \to \Sigma_{n_0}^b$ telle que $i_k(x) = x^b$.

Soit maintenant $\overline{i_k}$ l'extension uniforme de i_k sur $\Sigma^{\mathbb{Z}}$.

On note $\mathcal{D}_{k_0,b}$ l'image de $\mathcal{D}_{k_0}$ par $\overline{i_k}$. Il s'agit donc des configurations constituées uniquement de *blocs* uniformes de taille exactement b tels que deux blocs séparés par moins de k_0 blocs intermédiaires (chacun nécessairement de longueur b) sont différents.

L'idée va maintenant être d'utiliser les configurations de $\mathcal{D}_{k_0,b}$ pour prouver l'universalité de $\mathcal{A}_k$. Pour comprendre le comportement de ce k-ré

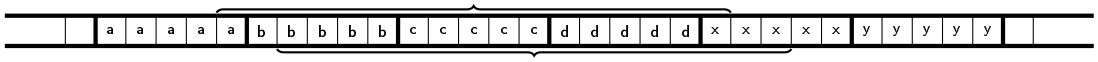


FIGURE 2.3: Configuration par blocs de taille $b = 5$ pour un automate de départ $k_0 = 3$. Les accolades pointent les deux types de facteurs de taille $k = 17$, on a donc $b - e = 2$.

D'après les relation entre k, k_0 et b, et comme l'illustre la figure 2.3, les facteurs de longueur k des configurations de $\mathcal{D}_{k_0,b}$ peuvent être de différents types :

- Ceux qui contiennent k_0 blocs différents complets de taille b. Ceux-ci se subdivisent à nouveau entre :
  - ceux qui ne contiennent rien d'autre, ceci n'arrive que dans le cas $b = e$, et donc $k = bk_0$. Ces facteurs contiennent uniquement k_0 états différents.
  - ceux dont les blocs complets sont précédés et suivis de morceaux de blocs incomplets. Ceux-ci contiennent $k_0 + 2$ états différents.
  - ceux qui sont soit précédés, soit suivis d'un bloc incomplet. Ceux-ci contiennent $k_0 + 1$ états différents.
- Ceux qui contiennent $k_0 - 1$ blocs différents de taille b. Ils sont nécessairement précédés et suivis par des blocs incomplets. Ils contiennent alors $k_0 + 1$ états différents.

Et le lemme intermédiaire suivant garantit que les approximations des multi-ensembles de ces facteurs vont être ceux permettant de simuler le comportement de l'automate de départ $\mathcal{A}_0$.

Lemme 2.10. *Avec les notations précédentes,*

Pour tout $u_i \in \Sigma$, $i \in \{1..n\}$ deux à deux distincts, pour tout $l \in \{0..b-e\}$,

$$r_{k_0}\left(\{\{u_0^l \cdot u_1^b \cdots u_{k_0}^b \cdot u_{k_0+1}^{b-e-l}\}\}\right) = \{\{u_1 u_2 \cdots u_{k_0}\}\}.$$

Et pour tout $l \in \{0..\lfloor \frac{e}{2} \rfloor\}$,

$$r_{k_0}\left(\{\{u_1^{b-l} \cdot u_2^b \cdots u_{k_0+1}^{b-e+l}\}\}\right) = \{\{u_1 u_2 \cdots u_{k_0}\}\}.$$

Preuve : [du lemme] Soit $u = u_0^l \cdot u_1^b \cdots u_{k_0}^b \cdot u_{k_0+1}^{b-e-l}$ satisfaisant les hypothèses du lemme. On note $\pi(u) = (x_1, x_2 \ldots, x_n)$ et $\pi(r_{k_0}(u)) = (y_1, y_2, \ldots, y_n)$.

On remarque d'abord que si pour un certain i, $x_i = 0$ alors $y_i = 0$. C'est une conséquence du lemme 2.1.

De plus $0 \leq x_i \leq \frac{b}{k} \leq \frac{1}{k_0}$. En utilisant de nouveau le lemme 2.1, $y_i = \frac{\gamma_i}{k_0}$ avec $\gamma_i \in \{0, 1\}$. Et dans la mesure où $\sum_{i\in\{1..n\}} y_i = 1$, il y a exactement k_0 valeurs distinctes de i telles que $\gamma_i = 1$. Par conséquent $r_{k_0}(u)$ contient k_0 états distincts choisis dans $\{u_0, u_1, \cdots u_{k_0+1}\}$.

Supposons maintenant par l'absurde que $r_{k_0}(u) \neq \{\{u_1 u_2 \cdots u_{k_0}\}\}$. Dans ce cas, il existe $\alpha \in \{u_0, u_{k_0+1}\}$ qui est un élément de $r_{k_0}(u)$, et il existe $\beta \in \{u_1, \ldots, u_{k_0}\}$ qui n'est pas dans $r_{k_0}(u)$. On note i_α le rang de α dans Σ et i_β celui de β. Soit v le multi-ensemble de taille k_0 obtenu à partir de $r_{k_0}(u)$ en remplaçant α par β, on note $\pi(v) = (z_1, z_2, \ldots, y_n)$. Et comparons maintenant $d_2(u, r_{k_0}(u))$ et $d_2(u, v)$:

$$\begin{aligned} d_2(u, r_{k_0}(u)) - d_2(u, v) &= \sum_{i\in\{1..n\}} (x_i - y_i)^2 - \sum_{i\in\{1..n\}} (x_i - z_i)^2 \\ &= (x_{i_\alpha} - y_{i_\alpha})^2 + (x_{i_\beta} - y_{i_\beta})^2 - (x_{i_\alpha} - z_{i_\alpha})^2 - (x_{i_\beta} - z_{i_\beta})^2 \\ &= \left(x_{i_\alpha} - \frac{1}{k_0}\right)^2 - \left(x_{i_\beta} - \frac{1}{k_0}\right)^2 + (x_{i_\beta})^2 - (x_{i_\alpha})^2 \end{aligned}$$

en développant et car $y_{i_\beta} = z_{i_\alpha} = 0$ et $y_{i_\alpha} = z_{i_\beta} = \frac{1}{k_0}$.

Or $0 \leq x_{i_\alpha} < x_{i_\beta} < \frac{1}{k_0}$, donc $(x_{i_\beta})^2 - (x_{i_\alpha})^2 > 0$ et $\left(x_{i_\alpha} - \frac{1}{k_0}\right)^2 - \left(x_{i_\beta} - \frac{1}{k_0}\right)^2 > 0$, d'où :

$$d_2(u, r_{k_0}(u)) - d_2(u, v) > 0$$

Ce qui contredit la définition de $r_{k_0}(u)$.

La preuve de la seconde partie est complètement similaire : les états de $r_{k_0}(u)$ sont de nouveau les plus représentés dans u.

□

Nous allons montrer que pour chaque $k \geq 7$ considéré, $\mathcal{A}_k$ simule le comportement de $\mathcal{U}$, sur les configurations de $i_k(\mathcal{D}_{k_0})$ de la même manière que $\mathcal{A}_0$ sur les configurations de $\mathcal{D}_{k_0}$ et donc que $\mathcal{A}_k$ est universel.

Preuve :

Soit $b \in \mathbb{N}$ et $e \in \{1..k_0+1\}$ définis par $k = b(k_0+1) - e$.

Soit $\mathcal{A}_k = \mathcal{R}_{\mathcal{A}_0}(k)$, on s'intéresse à l'évolution des configurations $c \in \mathcal{D}_{k_0,b} = \overline{i_k}(\mathcal{D}_{k_0})$. Plus précisément, on veut montrer que $\forall c' \in \mathcal{D}_{k_0}, {\mathcal{A}_k}^{<b,1,0>} \circ \overline{i_k}(c') = \overline{i_k} \circ \mathcal{A}_0(c')$.

Soit c une configuration de $\overline{i_k}(\mathcal{D}_{k_0}) \subseteq \mathcal{D}_{k_0,b}$. On a déjà vu que les multi-ensembles de taille k visibles dans cette configuration sont de l'un des deux types suivants :

- $u_0^l \cdot u_1^b \cdots u_{k_0}^b \cdot u_{k_0+1}^{b-e-l}$ avec $0 \leq l \leq b-e$,
- $u_1^{b-l} \cdot u_2^b \cdots u_{k_0+1}^{b-e+l}$ avec $0 \leq l \leq \lfloor \frac{e}{2} \rfloor\}$.

Grâce au lemme 2.10, sur une configuration de $\mathcal{D}_{k_0,b}$, les multi-ensembles des facteurs centrés sur les différentes cellules d'un même bloc ont la même approximation. Donc les blocs se maintiennent. Et de plus, par le même lemme, le nouvel état d'un bloc d'ancien état $u_{\lceil \frac{k_0}{2} \rceil}$ et entouré de blocs d'états $u_1, \ldots, u_{\lceil \frac{k_0}{2} \rceil - 1}$ à sa gauche et de blocs d'états $u_{\lceil \frac{k_0}{2} \rceil + 1}, \ldots, u_{k_0}$ à sa droite est l'état image par $\delta_{\mathcal{A}_0}$ d'un multi-ensemble $\{\{u_1, \ldots u_{k_0}\}\}$. Par conséquent l'évolution des configurations de $\mathcal{D}_{k_0,b}$ est bien telle que $\forall c' \in \mathcal{D}_{k_0}, {\mathcal{A}_k}^{<b,1,0>} \circ \overline{i_k}(c') = \overline{i_k} \circ \mathcal{A}_0(c')$.

$\mathcal{A}_k$ simule donc exactement le comportement de $\mathcal{A}_0$ sur les configurations de $\mathcal{D}_{k_0}$, par conséquent $\mathcal{A}_k$ simule les automates simulés par $\mathcal{A}_0$ sur un ensemble de configurations inclus dans $\mathcal{D}_{k_0}$, et en particulier l'automate universel $\mathcal{U}$. Ce qui conclut la preuve du théorème.

□

3.2 Cas multi-dimensionnel

La notion d'automates multi-ensemblistes, et celle de changements d'échelles se prête volontiers au passage en dimensions supérieures. Il faut cependant être attentif au choix du voisinage. Ici, nous nous contentons de considérer le cas de voisinages de Moore de diamètre k. Le nombre de cellules dans les voisinages considérés par la fonction locale est alors k^d, avec d la dimension.

Il est tentant de chercher à étendre le résultat précédent à une dimension quelconque. Cependant ceci ne peut se faire directement, à cause de la géométrie induite par la dimension. En effet, en dimension 1, on a vu que pour chaque taille du voisinage initial k_0, et pour chaque taille de voisinage k, on pouvait choisir une taille de blocs b, de manière à ce que les motifs de longueur k centrés sur les différentes cellules d'un même bloc aient la même k_0-approximation.

Ce n'est pas le cas à partir de la dimension 2 comme l'illustre la figure 2.4 : pour certaines valeurs des tailles k et k_0, la structure des blocs uniformes ne peut être maintenue.

Dès cette dimension, avec certaines tailles de voisinages, en décalant selon plusieurs directions, on ne peut plus garantir que le rééchelonné conserve le multi-ensemble attendu (pour simuler l'évolution de $\mathcal{A}_0$).

Ce phénomène, de perte d'information *dans les angles* semble donc indiquer que le changement de la dynamique des automates multi-ensemblistes lorsqu'on les rééchelonne pourrait être plus important pour les dimensions supérieures que pour la dimension 1.

Cependant, pour l'universalité, on peut se prémunir partiellement de ce phénomène. Pour ce faire, on va rigidifier encore un peu plus qu'en dimension 1 l'espace des configurations de simulation que l'on considère, et surtout on va se limiter à certaines tailles de voisinage bien choisies. On s'intéresse ainsi à un automate cellulaire multi-ensembliste $\mathcal{A}_0$ qui simule un automate cellulaire universel $\mathcal{U}$ tel que : $\mathcal{A}_0$ simule $\mathcal{U}$ sur un ensemble de configurations contenant une composante de marqueurs périodiques selon chaque dimension, de période assez grande pour que chaque cellule du voisinage soit distinguable. Ceci est possible pour les mêmes raisons qu'en dimension 1.

Surtout, on se contente ici de considérer les tailles k telles que $k = bk_0 + b - 1$, pour un certain $b \in \mathbb{N}$ impair, et donc k impair également. Le problème précédent est alors évité comme l'illustre la figure 2.5.

Plus précisément, soit $\mathcal{D}_{k_0} \subseteq \Sigma_{\mathcal{U}}^{\mathbb{Z}} \times \{0, 1, \dots k_0 + 1\}^d$ le sous-shift constitué des configurations telles que la $i+1$-ème couche est périodique de période $0 \cdot 1 \cdots k_0 + 1$ selon la i-ème direction et constante selon les autres directions, la figure 2.6 illustre le cas pour la dimension 2 et $k_0 = 3$.

Soit de nouveau i_k la fonction injective $i_k : \Sigma_{\mathcal{A}_0} \to \Sigma_{\mathcal{A}_0}^{b^d}$ telle que $i_k(x) = x^{b^d}$ et $\overline{i_k}$ l'extension uniforme de i_k sur $\Sigma^{\mathbb{Z}^d}$. On note $\mathcal{D}_{k_0,b}$ l'ensemble des configurations images de $\mathcal{D}_{k_0}$ par $\overline{i_k}$. Il s'agit donc des configurations constituées uniquement de *blocs* carrés uniformes de côté b, tels que deux blocs visible dans un même motif de largeur k ont des coordonnées (c'est-à-dire des valeurs pour les d dernières couches) cohérentes avec leurs positions relatives et la période.

Et on s'intéresse aux approximations des multi-ensembles des facteurs *carrés* (c'est-à-dire issus de voisinages de Moore) de côté $k = bk_0 + b - 1$ des configurations de $\mathcal{D}_{k_0,b}$, par des multi-ensembles de diamètre k_0.

Ces multi-ensembles contiennent toujours ${k_0}^d$ blocs complets de taille b^d, ainsi qu'un certain nombre de blocs partiels. Le nombre maximal de blocs partiels, fonction de la dimension peut être calculé, il s'agit du nombre d'hypercubes de taille 1 qui forment la frontière extérieure d'un hypercube de taille k. C'est-à-dire le nombre d'hypercube de taille 1, qui forment la différence entre un hypercube de taille $k+2$ et un hypercube de taille k de même centre. Ce nombre est donné par $(k_0 + 2)^d - {k_0}^d$.

On utilisera surtout le lemme suivant, qui correspond au cas multi-dimensionnel du lemme 2.10. Cependant, il est légèrement simplifié du fait de la relation fixée entre k et k_0.

Lemme 2.11. *Pour tout $u_i \in \Sigma$, $i \in \{1..(k_0+2)^d\}$, et pour $n_j < b^d$ pour chaque $j \in \{{k_0}^d + 1..(k_0+2)^d\}$, on a*

$$r_{{k_0}^d}\left(\{\{u_1^{b^d} \cdot u_2^{b^d} \cdots u_{{k_0}^d}^{b^d} \cdot u_{{k_0}^d+1}^{n_{{k_0}^d+1}} \cdots u_{(k_0+2)^d}^{n_{(k_0+2)^d}}\}\}\right) = \{\{u_1 \cdot u_2 \cdots u_{{k_0}^d}\}\}$$

Preuve : Le raisonnement est le même que pour la preuve du lemme 2.10.

Notons $u = u_1^{b^d} \cdot u_2^{b^d} \cdots u_{{k_0}^d}^{b^d} \cdot u_{{k_0}^d+1}^{n_{{k_0}^d+1}} \cdots u_{(k_0+2)^d}^{n_{(k_0+2)^d}}$, ainsi que, en utilisant la notation vectorielle $(x_1, x_2 \cdots, x_n) = \pi(u)$ et $(y_1, y_2, \cdots, y_n) = \pi(r_{k_0}(u))$.

On remarque d'abord que si pour un certain i, $x_i = 0$ alors $y_i = 0$. C'est une conséquence du lemme 2.1.

De plus, comme les états de deux blocs distincts sont toujours différents, $0 \leq x_i \leq \frac{b^d}{k^d} \leq \frac{1}{{k_0}^d}$. En utilisant de nouveau le lemme 2.1, $y_i = \frac{\gamma_i}{k_0}$ avec $\gamma_i \in \{0, 1\}$. Or $\sum_{i \in \{1..n\}} y_i = 1$, donc il y a exactement ${k_0}^d$ entiers γ_i qui valent 1. Les autres prennent la valeur 0.

Ensuite, exactement comme dans la preuve du lemme 2.10, il suffit de s'assurer que le multi-ensemble à ${k_0}^d$ éléments distincts le plus proche de u est bien celui qui comporte les ${k_0}^d$ éléments

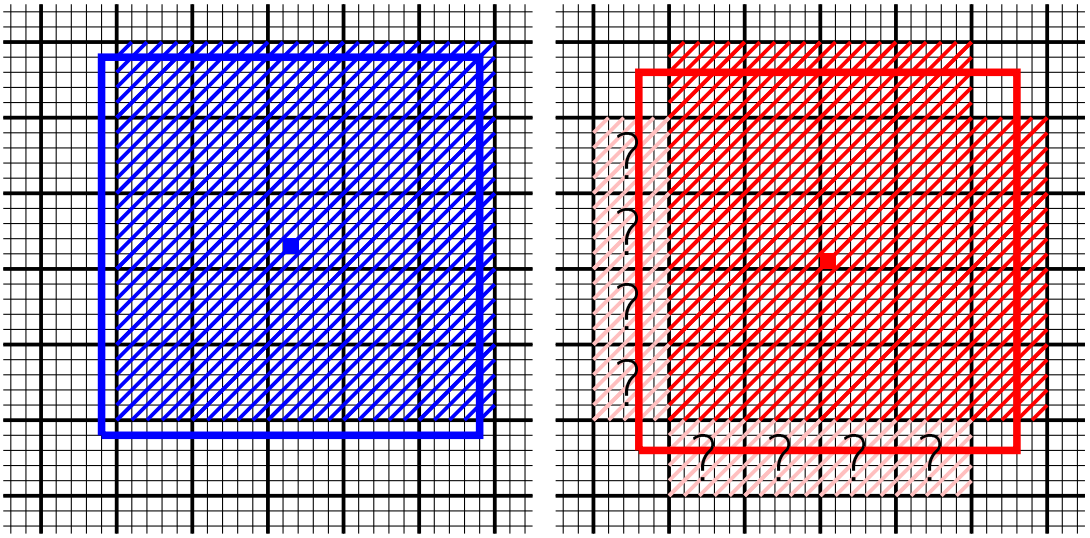


FIGURE 2.4: Pour un diamètre $k_0 = 5$, et $k = 25$. Le cadre délimite le voisinage de la cellule coloriée. Les blocs 5×5 étant uniformes, les blocs hachurés sont ceux dont l'état est conservé par passage au rééchelonné.

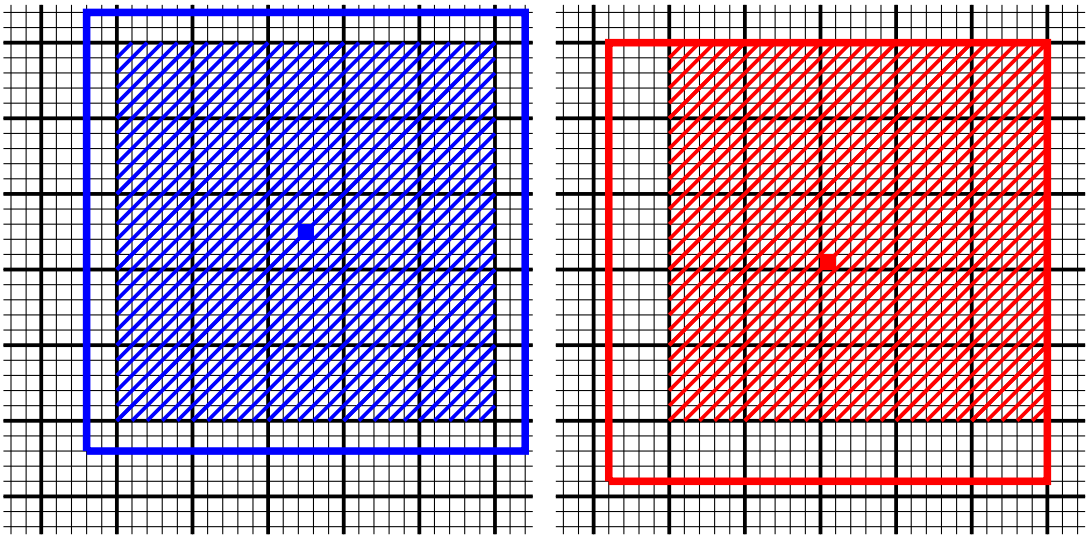


FIGURE 2.5: Pour un rayon $k_0 = 5$, $b = 5$ et $k = 29$ permet de retrouver les multi-ensembles désirés par k_0-approximation des facteurs de la taille du voisinage. Les blocs hachurés sont ceux dont l'état est obtenu.

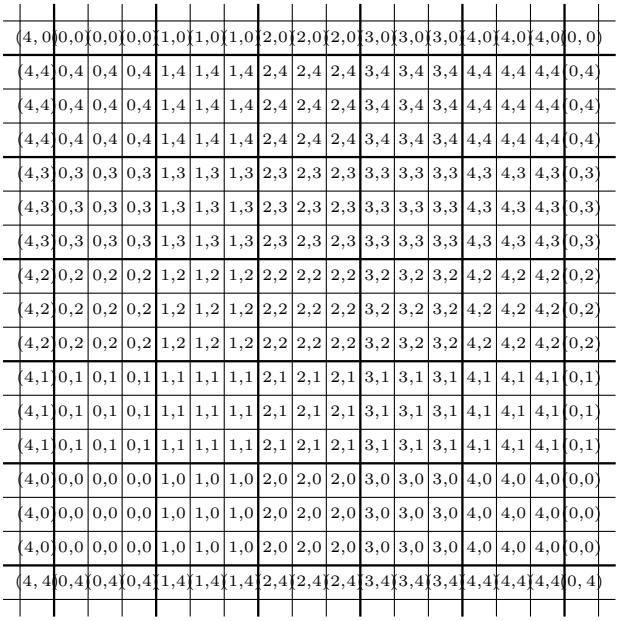


FIGURE 2.6: Couches de coordonnées pour une configuration bien formée, avec $k_0 = 3$, $b = 3$

qui sont strictement les plus représentés dans u.

□

Ce lemme permet de démontrer le résultat suivant :

Théorème 2.12. *Soit $d \geq 1$ un entier. Il existe un automate cellulaire multi-ensembliste de dimension d, tel que pour une infinité de k son k-rééchelonné est intrinsèquement universel.*

Preuve : On considère un automate cellulaire $\mathcal{A}_0$ de rayon 1 et donc de diamètre $k_0 = 3$, qui simule un automate cellulaire universel $\mathcal{U}$ sur un sous-shift de simulation inclus dans $\mathcal{D}_{k_0}$. On rappelle que ce dernier est constitué des configurations de $\Sigma_{\mathcal{U}}^{\mathbb{Z}} \times \{0, 1, \dots k_0 + 1\}^d$ dont la i-ème composante est périodique de période $0 \cdot 1 \cdots k_0 + 1$ selon la i-ème direction et invariante lorsque on bouge selon une autre direction. Cette structure particulière des couches de coordonnées est donc maintenue par $\mathcal{A}_0$.

Pour $k = bk_0 = b - 1$, $b \in \mathbb{N}$, on considère l'automate $\mathcal{A}_k = \mathcal{R}_{\mathcal{A}_0}(k)$, et l'ensemble de configurations $\mathcal{D}_{k_0,b}$ image par i_k de $\mathcal{D}_{k_0}$.

Comme illustré par la figure 2.5, les facteurs rencontrés sont ceux concernés par le lemme 2.11. Et en appliquant celui-ci, on vérifie facilement que $\mathcal{D}_{k_0,b}$ est stable par l'action de $\mathcal{A}_k$, et que les états des blocs évoluent conformément à la règle de $\mathcal{A}_0$.

Par conséquent, $\mathcal{A}_k$ simule $\mathcal{U}$ et il est universel, pour chaque $k = bk_0 + b - 1$, ce qui conclut la preuve du théorème.

□

4 Outer-multi-ensemblistes

On a ainsi vu que la classe des AC multi-ensemblistes présente des propriétés intéressantes. Nous présentons ici rapidement une classe très proche, qui est celle à laquelle appartient l'AC étudié par M. Pivato et K. M. Evans.

4.1 Définition

Dans la littérature, une classe d'automates non réellement totalistiques, mais qui s'en rapproche a été définie. Il s'agit de ceux pour lesquels en plus du total des ensembles d'états, la règle dispose d'une information sur l'état courant de la cellule centrale (c'est-à-dire de la cellule $0 \in \mathbb{Z}^d$ du voisinage).

Cette classe contient en particulier le Jeu de la vie.

On introduit de manière similaire la classe des *outer-multi-ensemblistes* qui sont la classe des automates dont la règle est invariante par permutation des voisins, hormis la cellule centrale.

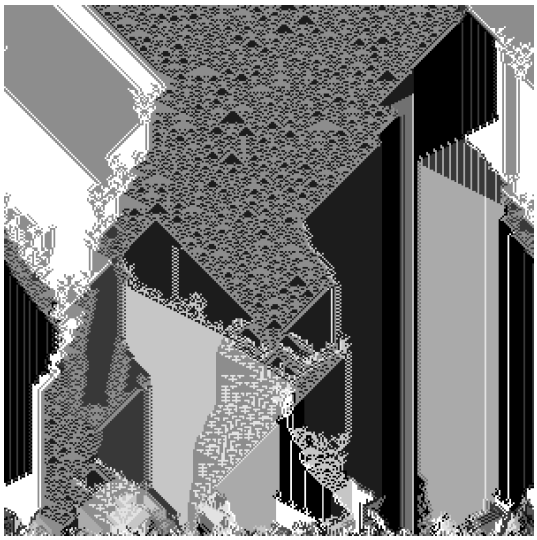
En réalité notre définition est même un peu plus large, introduisant une dépendance arbitraire en un nombre de cellules centrales qui peut être quelconque.

On va dire qu'un automate cellulaire $\mathcal{A} = (\mathbb{Z}^d, \Sigma, \mathcal{V}, \delta)$ est $\mathcal{V}'$ outer-multi-ensembliste si sa règle est indépendante par permutation des cellules de $\mathcal{V} \setminus \mathcal{V}'$.

Formellement, pour $\mathcal{V}' \subseteq \mathcal{V}$, on dit qu'une permutation $\pi : \mathcal{V} \to \mathcal{V}$ laisse $\mathcal{V}'$ invariant si $\forall x \in \mathcal{V}'$, $\pi(x) = x$. On a alors la définition suivante :

Définition 23. *Un AC $\mathcal{A} = (\mathbb{Z}^d, \Sigma, \mathcal{V}, \delta)$ est $\mathcal{V}'$-outer-multi-ensembliste si sa règle locale vérifie : pour toute permutation π de $\mathcal{V}$, laissant $\mathcal{V}'$ invariant, $\delta_{\mathcal{A}}(x_1, \ldots, x_{k_{\mathcal{A}}}) = \delta_{\mathcal{A}}(\pi(x_1), \ldots, \pi(x_{k_{\mathcal{A}}}))$.*

Le diagramme 2.7 illustre un tel automate.



(a) Voisinage de taille 5, ensemble d'état de taille 10

FIGURE 2.7: Diagramme espace-temps d'AC outer-multi-ensemblistes tirés aléatoirement à partir d'un segment initial aléatoire.

Dans la pratique, on considérera surtout des automates cellulaires pour lesquels $\mathcal{V}$ et $\mathcal{V}'$ sont des voisinages de Moore.

On dira qu'une automate cellulaire de voisinage de Moore $k_{\mathcal{A}}$ est ${k_{\mathcal{A}}}'$-outer-multi-ensembliste, s'il vérifie la définition précédente avec $\mathcal{V} = \mathcal{V}_{k_{\mathcal{A}}}$ et $\mathcal{V}' = \mathcal{V}_{k_{\mathcal{A}'}}$.

4.2 Rééchelonné d'un outer-multi-ensembliste

On peut également définir le rééchelonné d'un tel automate. Nous allons introduire la définition formelle pour le cas de $\mathcal{V}'$ réduit à une cellule centrale. Cette définition étend la définition de M. Pivato et K. M. Evans dans [Eva01].

On note $\mathcal{OMS}$ l'ensemble des automates cellulaires 1-outer-multi-ensemblistes.

Pour simplifier les définitions à venir, on introduit la notation suivante pour la règle d'un automate cellulaire outer-multi-ensembliste $\mathcal{A}$: celle-ci sera vu comme une fonction

$$\delta^*_{\mathcal{A}} : \Sigma_{\mathcal{A}} \times \Sigma_{\mathcal{A}}^{(k_{\mathcal{A}})} \to \Sigma_{\mathcal{A}}.$$

la première composante représentant la cellule centrale, mais sa valeur étant reprise dans le multi-ensemble de la seconde composante.

On peut alors définir le rééchelonné d'un outer-multi-ensembliste :

Définition 24. *Soit $\mathcal{A} \in \mathcal{OMS}$ un automate cellulaire multi-ensembliste d'alphabet $\Sigma_{\mathcal{A}}$ de taille $n_{\mathcal{A}}$ et de voisinage connexe centré de taille $k_{\mathcal{A}}$. Son* rééchelonné *de taille k (ou k-rééchelonné) est l'automate $\mathcal{R}_{\mathcal{A}}(k)$ de même ensemble d'état, de voisinage de taille k et de règle locale définie par $\forall u \in \Sigma_{\mathcal{A}}^k$,*

$$\begin{aligned} \delta_{\mathcal{R}_{\mathcal{A}}} : \Sigma_{\mathcal{A}} \times \Sigma_{\mathcal{A}}^{(k)} &\to \Sigma_{\mathcal{A}} \\ (x, u) &\mapsto \delta_{\mathcal{A}}(x, r_{k_{\mathcal{A}}}(u)) \end{aligned}$$

On note que même lorsque l'échelle devient très grande, la cellule centrale reste distinguée et conserve une importance décisive quand bien même toutes les autres cellules n'ont que très peu d'importance.

Ce phénomène qui peut aussi paraître discutable a également motivé l'introduction de la classe des multi-ensemblistes qui n'ont pas ce problème.

Toujours est-il que, un automate de $\mathcal{MS}$ étant un automate de $\mathcal{OMS}$, les résultats suivants sont des conséquences immédiates des théorèmes 2.9 et 2.12.

Corollaire 2.13. *Il existe un automate cellulaire multi-ensembliste, et une taille k_0, tels que pour tout $k \geq k_0$ impair, son k-rééchelonné est universel.*

Corollaire 2.14. *Soit $d \geq 1$ un entier. Il existe un automate cellulaire* outer-multi-ensembliste *de dimension d, tel que pour une infinité de k son k-rééchelonné est intrinsèquement universel.*

Chapitre 3

Densité de propriétés

1 Densité de propriétés 52
1.1 Formalisme 52
1.2 Complexité de Kolmogorov et dénombrement 59
1.3 Quelques objets combinatoires usuels 63
2 Parmi les AC quelconques 65
2.1 Propriétés syntaxiques, propriétés dépendant d'une sous-partie de la table 65
2.2 Propriétés dynamiques 72
3 Parmi des sous-familles d'AC définies syntaxiquement 80
3.1 Méthode générale 81
3.2 À ensemble d'états croissant 83
3.3 À taille de voisinage croissante 85
3.4 Cas les plus généraux 90

L'objectif de cette section est d'essayer de quantifier à quelle point une propriété donnée est répandue, dense, parmi un ensemble d'automates cellulaires.

Ceci permettrait de mieux comprendre la structure de l'ensemble des automates cellulaires, en enrichissant les propriétés, ou les classifications usuelles par des considérations quantitatives. Par exemple, des informations quantitatives sur la répartition des AC entre les classes des différentes classifications permettraient de savoir à quel point celles-ci sont plus ou moins discriminantes. Cette étude peut aussi être vue comme la volonté de donner un sens à la notion d'AC typique, à la compréhension de ce qu'est un AC *en moyenne*.

Cette problématique de la densité de propriétés n'est pas spécifique aux automates cellulaires et a été abordée dans d'autres cas, par exemple pour d'autres modèles de calcul. On peut citer par exemple les travaux sur les machines de Turing ([CS06]).

Pour les automates cellulaires quelques travaux s'y rattachent directement , comme l'étude des automates cellulaires captifs par G. Theyssier ([The04]) ou indirectement comme les travaux de J-C. Dubacq, B. Durand et E. Formenti sur l'utilisation de la complexité de Kolmogorov pour la classification des AC ([JDF01]) auxquels nous feront référence à plusieurs reprises.

Notre approche s'inscrit dans la continuité du formalisme introduit par G. Theyssier, dont nous généralisons le cadre de l'étude et donc les définitions.

1 Densité de propriétés

1.1 Formalisme

Les ensembles considérés

Pour étudier la densité de propriétés parmi un ensemble, il faut commencer par délimiter l'ensemble concerné.

Dans le cadre de ces travaux, nous nous sommes limités à l'étude des automates cellulaires les plus usuels, c'est-à-dire unidimensionnels, et à voisinage connexe et centré. Il ne fait aucun doute que la même approche pourrait, devrait même, être menée en dimensions supérieures. La restriction à des voisinages canoniques, centrés et connexes peut sembler raisonnable, dans la mesure où c'est le cas de la grande majorité des automates étudiés dans la littérature.

De plus nous considérons pour chaque taille n l'alphabet canonique $\Sigma_n = \{0, \dots, n-1\}$. Et on utilisera fréquemment l'ordre induit sur cet alphabet par l'ordre des entiers.

Pour nous, un automate $\mathcal{A}$ sera donc la donnée successive :

- d'une taille d'alphabet $n_{\mathcal{A}}$,
- d'une taille de voisinage $k_{\mathcal{A}}$,
- d'une règle de transition $\delta_{\mathcal{A}} : \Sigma_{n_{\mathcal{A}}}^{k_{\mathcal{A}}} \to \Sigma_{n_{\mathcal{A}}}$.

Il s'agit d'objets syntaxiques : un AC est une description finie.

On note alors $\mathbb{CA}$ l'ensemble des automates, c'est-à-dire l'ensemble des triplets $(n_{\mathcal{A}}, k_{\mathcal{A}}, \delta_{\mathcal{A}})$. On utilisera la notation $\mathbb{CA}_{n,k} \subseteq \mathbb{CA}$ pour désigner l'ensemble des automates avec une taille de voisinage k et une taille d'alphabet n. On a bien $\mathbb{CA} = \bigcup_{n,k} \mathbb{CA}_{n,k}$.

> *Remarque* 4. Tous les AC unidimensionnels (au sens des fonctions globales) sont considérés ici. Par exemple ceux ayant un voisinage non connexe, sont considérés car ils sont également décrits par un voisinage connexe plus grand.
> Cependant ces choix ne sont pas complètement anodins. En effet le poids relatif de chaque fonction globale d'automate n'est a priori pas le même selon le formalisme considéré.
> Par exemple on aurait pu ne considérer que les définitions syntaxiques pour lesquelles le voisinage est minimal (au sens où la règle dépende réellement de chaque cellule du voisinage). De cette manière une fonction globale ne coïnciderait qu'avec une seule définition syntaxique. Ici une règle globale correspond à une définition syntaxique pour chaque taille de voisinage à partir de sa taille minimale.
> De même, en ce qui concerne l'alphabet, on aurait aussi pu choisir de considérer que deux automates égaux à permutation des états de l'alphabet prêt étaient identiques. En ne le faisant pas on augmente le poids relatif des règles très symétriques (c'est-à-dire invariantes par un grand nombre de ces permutations).
> Ces deux aspects sont discutés à la fin de cette partie.

Une fois l'ensemble considéré fixé, on peut expliciter le type de questions que nous nous poserons dans la suite : étant donné une propriété $\mathcal{P}$, c'est-à-dire un sous-ensemble de $\mathbb{CA}$, quelle est la fréquence, la *densité*, de $\mathcal{P}$ dans $\mathbb{CA}$. Nous allons maintenant donner un sens formel à ces questions.

Avant cela, notons également que dans la suite, il nous arrivera de nous intéresser à des densités de propriétés, non pas parmi l'ensemble complet des automates cellulaires $\mathbb{CA}$ mais parmi un sous-ensemble de celui-ci $S \subseteq \mathbb{CA}$. Et nous noterons alors $S_{n,k}$ l'ensemble $S \cap \mathbb{CA}_{n,k}$ des AC de S de taille (n, k).

Vers une énumération *par taille* des automates

Une manière générale de considérer une densité, en accordant le même poids à chaque élément, dans un ensemble dénombrable est d'introduire une énumération quelconque de ses éléments. On considère donc une énumération quelconque des éléments de $\mathbb{CA}$, c'est-à-dire formellement une fonction injective $\phi : \mathbb{N} \to \mathbb{CA}$, qui pourra a priori être surjective ou non.

Si on note $\phi(\{1..i\})$ l'ensemble des i premiers automates dans cette énumération, la densité d'une propriété $\mathcal{P}$ selon l'énumération ϕ est alors la limite, si elle existe, du ratio $\frac{\#(\phi(\{1..i\})\cap\mathcal{P})}{\#(\phi(\{1..i\}))}$ lorsque i croit. Et si elle n'existe pas, on dira que cette densité n'existe pas.

Cependant, dans ce cadre très général, pour n'importe quelle propriété non triviale des fonctions globales d'automates cellulaires (c'est-à-dire une propriété indépendante de la description syntaxique de l'AC et telle que ni la propriété, ni son complémentaire ne soient vides), on peut facilement expliciter une énumération, même surjective, des éléments de $\mathbb{CA}$ pour laquelle la densité de cette propriété est 0, une autre énumération dans laquelle elle vaut 1, et même une troisième dans laquelle cette densité n'existe pas !

En fait, il est même possible d'expliciter une énumération pour obtenir n'importe quelle densité choisie arbitrairement pour notre propriété.

Remarque 5. En effet, on a vu que n'importe quelle fonction globale est décrite par une infinité d'objets de $\mathbb{CA}$. Pour une propriété non triviale $\mathcal{P}$, soit ϕ une énumération de l'ensemble (infini) des AC ayant cette propriété et ϕ' une énumération de ceux n'ayant pas cette propriété. Selon ϕ, et ϕ', les densités sont respectivement 1 et 0.
Mais on peut aussi construire à l'aide de ϕ et ϕ', des énumérations surjectives selon lesquelles on obtient n'importe quelle densité :
- pour obtenir une densité 1 : énumérer 1 élément de ϕ puis 1 de ϕ', puis 2 de ϕ et 1 de ϕ', puis 3 de ϕ et 1 de ϕ',...
- pour obtenir une densité 0, il suffit de faire l'inverse.
- pour obtenir une densité $d \in [0, 1]$ quelconque, il suffit d'approcher d, en énumérant successivement des éléments de ϕ et ϕ'
- enfin pour ne pas obtenir de limite, il suffit d'énumérer 1 élément de ϕ puis 2 de ϕ' puis 6 de ϕ puis 18 de ϕ'... On énumère à chaque fois le double du nombre d'éléments énumérés jusque là.

Se restreindre uniquement aux énumérations calculables ne résout évidemment pas ce problème dans la mesure où on le retrouve, identique, pour les propriétés calculables.

Le choix que nous faisons pour nous affranchir de ce problème et donner un sens à cette notion de densité est d'énumérer les éléments de $\mathbb{CA}$ par tailles. En effet lorsque l'on énumère tous les automates d'une taille donnée simultanément, on contraint fortement l'énumération, on exclut en particulier les problèmes précédents et la notion obtenue a même une certaine robustesse.

ϕ n'est alors plus réellement une énumération, il s'agit maintenant *d'une énumération par taille*, c'est-à-dire qu'il existe $\rho : \mathbb{N} \to \mathbb{N}^2$ et une fonction $t : \mathbb{N} \to \mathbb{N}$ croissante telle que

$$\phi(\{1..i\}) = \cup_{j\in\{0..t(i)\}}\mathbb{CA}_{\rho(j)}.$$

La densité est la limite, si elle existe, de $\frac{\#(\phi(\{1..i\})\cap\mathcal{P})}{\#(\phi(\{1..i\}))}$ lorsque i tend vers l'infini. Et la densité n'existe pas sinon.

Remarque 6. Il reste possible de construire des cas pathologiques, par exemple en utilisant des propriétés dépendant uniquement de la taille des automates (ceci est illustré par l'exemple 8), mais ce n'est en général pas le cas pour les propriétés usuelles, qui peuvent nous intéresser car elles sont en lien avec le comportement global des automates.

Des chemins dans l'espace des tailles

Ceci se formalise à l'aide des définitions suivantes.

Définition 25. *Un* chemin *dans l'espace des tailles est une fonction injective* $\rho : \mathbb{N}^* \to \mathbb{N}^{*2}$.

Intuitivement, $\rho(i)$ est la i-ème taille énumérée. L'injectivité est importante, elle garantit que chaque taille est énumérée au plus une fois.

Souvent on notera ρ_n (respectivement ρ_k) la première (resp. deuxième) composante de ρ, c'est-à-dire celle qui correspond au nombre d'états (resp. à la taille du voisinage) des automates de la taille considérée.

En notant $\mathbb{N}_i$, l'ensemble des entiers au moins aussi grand que i, c'est-à-dire $\mathbb{N}_i = \{x \in \mathbb{N} : x \geq i\}$, on introduit également la définition suivante :

Définition 26. *On dit qu'un chemin* ρ *est un* (n_0, k_0)*-chemin si* $\rho(\mathbb{N}) \subseteq \mathbb{N}_{n_0} \times \mathbb{N}_{k_0}$.

Cette définition sera utile car on cherche souvent à ne considérer que les tailles *au dessus d'une certaine taille minimale* (par exemple pour éviter les AC à 1 état dont le comportement quel que soit le voisinage est assez pauvre).

Une classe particulière de chemins se distingue alors :

Définition 27. *On dit qu'un chemin* ρ *est* (n_0, k_0)*-surjectif si* $\rho(\mathbb{N}) = \mathbb{N}_{n_0} \times \mathbb{N}_{k_0}$.

Une énumération des automates selon un chemin (n_0, k_0)-surjectif est donc une énumération qui considère tous les automates *à partir d'une certaine taille*. Ceci est très utile, dans la mesure où l'on peut ainsi considérer un ensemble d'automates très proche de $\mathbb{CA}$ auquel on a simplement enlevé les automates *trop petits*, souvent problématiques par rapport aux propriétés que l'on veut considérer.

On a vu que l'on notait $\mathbb{CA}_{n,k}$ l'ensemble des automates ayant un alphabet de taille n, et un voisinage de taille k et l'on dispose maintenant également de la notation $\mathbb{CA}_{\rho(i)}$ pour l'ensemble des automates de taille $\rho(i)$. De même, pour un ensemble quelconque $S \subseteq \mathbb{CA}$, $S_{n,k} = S \cap \mathbb{CA}_{n,k}$ et l'on notera $S_{\rho(i)} = S \cap \mathbb{CA}_{\rho(i)}$.

Densités cumulatives et non cumulatives

Avec les notations précédentes et pour un ensemble S, et une propriété $\mathcal{P}$, la densité telle que définie jusque là, que l'on va appeler *densité cumulative* s'exprime avec le formalisme des chemins de la manière suivante. Soit

$$C_{\rho,m}(S, \mathcal{P}) = \frac{\#\left((\cup_{i \leq m} S_{\rho(i)}) \cap \mathcal{P}\right)}{\#\left((\cup_{i \leq m} S_{\rho(i)})\right)}.$$

la proportion d'automates parmi les m premières tailles énumérées. Ceci permet d'écrire la définition suivante :

Définition 28. *La* densité cumulative *d'une propriété* $\mathcal{P}$ *parmi une famille* S *selon un chemin* ρ *est la limite, si elle existe de la fonction* $m \mapsto C_{\rho,m}(S, \mathcal{P})$ *lorsque* m *tend vers l'infini. Dans ce cas on la note*

$$c_\rho(S, \mathcal{P}) = \lim_{m \to \infty} C_{\rho,m}(S, \mathcal{P}) = \lim_{m \to \infty} \frac{\#\left((\cup_{i \leq m} S_{\rho(i)}) \cap \mathcal{P}\right)}{\#\left((\cup_{i \leq m} S_{\rho(i)})\right)}.$$

Cependant, dans la suite, on va principalement utiliser une version *non cumulative* de la densité. On définit la proportion d'automates de S d'une taille donnée ayant la propriété $\mathcal{P}$. Si $S_{n,k}$ n'est pas vide, la proportion d'automates de S de taille (n,k) ayant cette propriété est :

$$D_{n,k}(S,\mathcal{P}) = \frac{\#(S_{n,k} \cap \mathcal{P})}{\#(S_{n,k})}.$$

Et l'on introduit la densité correspondante parmi S selon un chemin ρ : c'est la limite de la proportion $D_{\rho(i)}(S,\mathcal{P})$ lorsque i grandit.

Définition 29. *La* densité non cumulative *d'une propriété $\mathcal{P}$ parmi une famille S selon un chemin ρ est la limite, si elle existe, de la fonction $i \mapsto D_{\rho(i)}(S,\mathcal{P})$ lorsque i tend vers l'infini. Dans ce cas on note*

$$d_\rho(S,\mathcal{P}) = \lim_{i\to\infty} D_{\rho(i)}(S,\mathcal{P}) = \lim_{i\to\infty} \frac{\#\left(S_{\rho(i)} \cap \mathcal{P}\right)}{\#\left(S_{\rho(i)}\right)}.$$

À priori la densité semble donc intimement dépendante du choix du chemin. Cependant, le résultat suivant montre que, au moins pour les chemins surjectifs, la notion de densité, sans plus de précision, a du sens.

Ce résultat simple montre que si une densité non cumulative existe et a une valeur selon *un* chemin surjectif, alors selon *tout* chemin, surjectif ou non, la densité existe et a la même valeur.

Proposition 3.1. *Étant donné $S,\mathcal{P} \subseteq \mathbb{CA}$, s'il existe un chemin (n_0,k_0)-surjectif ρ tel que $d_\rho(S,\mathcal{P})$ est définie, alors* pour tout (n_0,k_0)*-chemin ρ', $d_{\rho'}(S,\mathcal{P})$ existe et $d_{\rho'}(S,\mathcal{P}) = d_\rho(S,\mathcal{P})$.*

Preuve : Soit $S,\mathcal{P} \subseteq \mathbb{CA}$, et ρ (n_0,k_0)-surjectif tel que $d_\rho(S,\mathcal{P})$ existe et vaut d.

Soit ρ' un (n_0,k_0)-chemin et $\epsilon > 0$ quelconque. Par définition de $d_\rho(S,\mathcal{P})$, $\exists i_0, \forall i \geq i_0, |d - D_{\rho(i)}(S,\mathcal{P})| < \epsilon$. Soit $E_{i_0} = \cup_{i<i_0}\{\rho(i)\}$. Par (n_0,k_0)-surjectivité de ρ, on a

$$\forall (n,k) \in (\mathbb{N}_{n_0} \times \mathbb{N}_{k_0}) \setminus E_{i_0}, |d - D_{(n,k)}(S,\mathcal{P})| < \epsilon.$$

Comme E_{i_0} est fini, et comme le chemin ρ est injectif, il existe j_0 tel que $\forall j \geq j_0, \rho'(j) \notin E_{i_0}$. Mais par définition, $\forall j$, $\rho'(j) \in \mathbb{N}_{n_0} \times \mathbb{N}_{k_0}$ donc il existe j_0 tel que $\forall j \geq j_0$, $j \in (\mathbb{N}_{n_0} \times \mathbb{N}_{k_0}) \setminus E_{i_0}$ et donc $|d - D_{\rho'(j)}(S,\mathcal{P})| < \epsilon$. Ce raisonnement est vrai pour tout ϵ, ce qui permet de conclure la preuve de la proposition.

□

Une dernière définition avant de s'intéresser aux autres propriétés de ces densités : on dira qu'une propriété est *négligeable* selon un chemin parmi une famille d'AC si sa densité selon ce chemin et parmi cette famille est 0. On utilisera en général ce vocabulaire dans le cas de densités non cumulatives.

Nous allons voir maintenant que ces deux notions de densité (cumulative ou non) sont reliées directement, c'est ce qu'expriment les lemmes 3.2, 3.3 et 3.4.

Ceci peut se comprendre dans la mesure où le grand nombre d'AC de grande taille fait que, même pour la densité cumulative, c'est surtout la densité parmi ces tailles qui est déterminante.

On montre tout d'abord que l'existence d'une densité non cumulative implique celle d'une densité cumulative, identique.

Lemme 3.2. *Pour une propriété $\mathcal{P} \subseteq \mathbb{CA}$, un chemin ρ et un ensemble $S \subseteq \mathbb{CA}$, si la densité non cumulative $d_\rho(S,\mathcal{P})$ existe, alors la densité cumulative $c_\rho(S,\mathcal{P})$ existe et $c_\rho(S,\mathcal{P}) = d_\rho(S,\mathcal{P})$.*

Preuve : Soit $\mathcal{P}$, S et ρ comme dans les hypothèses du lemme. Et soit $d = d_\rho(S, \mathcal{P})$.

Par définition, $\forall \epsilon > 0$, $\exists i_\epsilon$ tel que si $i \geq i_\epsilon$, alors $|d - D_{\rho(i)}(S, \mathcal{P})| < \epsilon$. On va montrer que $\forall \epsilon > 0$, $\exists m_\epsilon$ tel que si $m \geq m_\epsilon$, alors $|d - C_{\rho(m)}(S, \mathcal{P})| < \epsilon$. Considérons un ϵ fixé, pour tout m, on a :

$$\begin{aligned}
\left|C_{\rho(m)}(S,\mathcal{P}) - d\right| &= \left| d - \frac{\#\left((\cup_{i\leq m} S_{\rho(i)}) \cap \mathcal{P}\right)}{\#\left((\cup_{i\leq m} S_{\rho(i)})\right)} \right| \\
&= \left| \frac{\sum_{i\leq m} \#\left(S_{\rho(i)} \cap \mathcal{P}\right) - d \cdot \sum_{i\leq m} \#\left(S_{\rho(i)}\right)}{\sum_{i\leq m} \#\left(S_{\rho(i)}\right)} \right| \\
&\leq \left| \frac{\sum_{i<i_\epsilon} \left[\#\left(S_{\rho(i)} \cap \mathcal{P}\right) - d \cdot \#\left(S_{\rho(i)}\right)\right]}{\sum_{i\leq m} \#\left(S_{\rho(i)}\right)} \right| \\
&\quad + \left| \frac{\sum_{i=i_\epsilon}^{m} \left[\#\left(S_{\rho(i)} \cap \mathcal{P}\right) - d \cdot \#\left(S_{\rho(i)}\right)\right]}{\sum_{i\leq m} \#\left(S_{\rho(i)}\right)} \right|
\end{aligned}$$

en coupant les sommes en deux au niveau du i_ϵ défini plus haut.

D'une part $\sum_{i<i_\epsilon} \left[\#\left(S_{\rho(i)} \cap \mathcal{P}\right) - d \cdot \#\left(S_{\rho(i)}\right)\right]$ est indépendante de m alors que lorsque m augmente, $\sum_{i\leq m} \#\left(S_{\rho(i)}\right)$ est croissante vers l'infini en m, donc il existe m_ϵ tel que $\forall m \geq m_\epsilon$,

$$\left| \frac{\sum_{i<i_\epsilon} \left[\#\left(S_{\rho(i)} \cap \mathcal{P}\right) - d \cdot \#\left(S_{\rho(i)}\right)\right]}{\sum_{i\leq m} \#\left(S_{\rho(i)}\right)} \right| \leq \epsilon$$

et d'autre part $\forall i \geq i_\epsilon$,

$$\left|\#\left(S_{\rho(i)} \cap \mathcal{P}\right) - d \cdot \#\left(S_{\rho(i)}\right)\right| = \left| \#\left(S_{\rho(i)}\right) \left(\frac{\#\left(S_{\rho(i)} \cap \mathcal{P}\right)}{\#\left(S_{\rho(i)}\right)} - d \right) \right| \leq \#\left(S_{\rho(i)}\right) \cdot \epsilon$$

ce qui entraîne

$$\left| \frac{\sum_{i=i_\epsilon}^{m} \left(\#\left(S_{\rho(i)} \cap \mathcal{P}\right) - d \cdot \#\left(S_{\rho(i)}\right)\right)}{\sum_{i\leq m} \#\left(S_{\rho(i)}\right)} \right| \leq \frac{\sum_{i=i_\epsilon}^{m} \#\left(S_{\rho(i)}\right).\epsilon}{\sum_{i\leq m} \#\left(S_{\rho(i)}\right)} \leq \epsilon \leq \epsilon$$

Donc pour tout ϵ, on a défini plus haut m_ϵ tel que pour tout $m \geq m_\epsilon$,

$$|d - C_{\rho(m)}(S, \mathcal{P})| \leq 2\epsilon$$

ce qui conclut la preuve.

□

La réciproque est fausse dans le cas général. Le plus simple pour s'en convaincre est de construire un contre-exemple à l'aide de propriétés dépendants exclusivement des tailles des automates :

> *Exemple* 8. Soit ρ le chemin tel que pour tout i, $\rho(2i) = (2i, 2^i)$ et $\rho(2i+1) = (2i+1, 2i+1)$. Sur ce chemin la propriété *avoir un ensemble d'états de taille paire* n'a pas de limite dans le cas non cumulatif, mais a une limite (de valeur 1) dans le cas cumulatif.

L'intuition est en effet que l'énumération de tailles pour lesquelles la proportion d'automates satisfaisant la propriété est éloignée de la limite peut être *masquée*, dans le cas cumulatif, par l'énumération de grosses classes alors que ce n'est pas possible dans le cas non cumulatif.

Cependant on peut exhiber des conditions suffisantes pour que la réciproque soit vraie. On prouve de tels résultats pour des cas importants. Ainsi le lemme 3.3 considère les chemins sur lesquels le nombre d'automates dans les tailles énumérées croit très vite (de manière exponentielle), et le lemme 3.4 traite des chemins surjectifs.

Lemme 3.3. *Soit $S \subseteq \mathbb{CA}$ un ensemble et $\rho : \mathbb{N} \to \mathbb{N}^2$ un chemin tels que pour tout $i \geq 1$, $\#\left(S_{\rho(i)}\right) \geq 2 \cdot \#\left(S_{\rho(i-1)}\right)$. Alors pour toute propriété $\mathcal{P}$ telle que $c_\rho(S, \mathcal{P})$ existe, on a $d_\rho(S, \mathcal{P}) = c_\rho(S, \mathcal{P})$.*

Preuve : Soit S, $\mathcal{P}$ comme dans l'énoncé du lemme, et $d = c_\rho(S, \mathcal{P})$. Pour tout $\epsilon > 0$, il existe i_0 tel que si $i \geq i_0$, alors

(1) $d - \epsilon \leq C_{\rho,i}(S, \mathcal{P}) \leq d + \epsilon$

(1) $d - \epsilon \leq C_{\rho,i-1}(S, \mathcal{P}) \leq d + \epsilon$.

Considérons $D_{\rho(i)}(S, \mathcal{P})$.

On a

$$\begin{aligned}
D_{\rho(i)}(S, \mathcal{P}) &= \frac{\#\left((S_{\rho(i)} \cap \mathcal{P})\right)}{\#\left(S_{\rho(i)}\right)} \\
&= \frac{1}{\#\left(S_{\rho(i)}\right)} \cdot \left(\frac{\sum_{j \leq i} \#\left(S_{\rho(j)} \cap \mathcal{P}\right)}{\sum_{j \leq i} \#\left(S_{\rho(j)}\right)} \cdot \sum_{j \leq i} \#\left(S_{\rho(j)}\right) \right. \\
&\qquad \left. - \frac{\sum_{j \leq i-1} \#\left(S_{\rho(j)} \cap \mathcal{P}\right)}{\sum_{j \leq i-1} \#\left(S_{\rho(j)}\right)} \cdot \sum_{j \leq i-1} \#\left(S_{\rho(j)}\right) \right) \\
&= \frac{1}{\#\left(S_{\rho(i)}\right)} \cdot \left(C_{\rho,i}(S, \mathcal{P}) \cdot \sum_{j \leq i} \#\left(S_{\rho(j)}\right) \right. \\
&\qquad \left. - C_{\rho,i-1}(S, \mathcal{P}) \cdot \sum_{j \leq i-1} \#\left(S_{\rho(j)}\right) \right) \\
&= C_{\rho,i}(S, \mathcal{P}) \cdot \frac{\sum_{j \leq i} \#\left(S_{\rho(j)}\right)}{\#\left(S_{\rho(j)}\right)} - C_{\rho,i-1}(S, \mathcal{P}) \cdot \frac{\sum_{j \leq i-1} \#\left(S_{\rho(j)}\right)}{\#\left(S_{\rho(j)}\right)}
\end{aligned}$$

Par conséquent en utilisant les inégalités (1) et (2), d'une part

$$\begin{aligned}
D_{\rho(i)}(S, \mathcal{P}) &\leq (d + \epsilon) \cdot \frac{\sum_{j \leq i} \#\left(S_{\rho(j)}\right)}{\#\left(S_{\rho(j)}\right)} - (d - \epsilon) \cdot \frac{\sum_{j \leq i-1} \#\left(S_{\rho(j)}\right)}{\#\left(S_{\rho(j)}\right)} \\
&\leq d + \epsilon \cdot \left(1 + 2 \frac{\sum_{j \leq i-1} \#\left(S_{\rho(j)}\right)}{\#\left(S_{\rho(j)}\right)} \right) \\
&\leq d + 3\epsilon
\end{aligned}$$

et d'autre part :

$$\begin{aligned} D_{\rho(i)}(S,\mathcal{P}) &\geq (d-\epsilon)\cdot\frac{\sum_{j\leq i}\#\left(S_{\rho(j)}\right)}{\#\left(S_{\rho(j)}\right)} - (d+\epsilon)\cdot\frac{\sum_{j\leq i-1}\#\left(S_{\rho(j)}\right)}{\#\left(S_{\rho(j)}\right)} \\ &\geq d-3\epsilon \end{aligned}$$

Il est donc possible pour tout ϵ d'exhiber i_ϵ tel que pour tout $i \geq i_\epsilon$, alors $d-3\epsilon \leq D_{\rho(i)}(S,\mathcal{P}) \leq d+3\epsilon$. Donc $D_{\rho(i)}(S,\mathcal{P})$ a une limite et converge vers d.

□

Lemme 3.4. *Soit $\mathcal{P}$ et S. Il existe d tel que $d_\rho(S,\mathcal{P}) = d$ sur tout chemin (n_0,k_0)-surjectif ρ si et seulement si $c_\rho(S,\mathcal{P}) = d$ sur tout chemin (n_0,k_0)-surjectif ρ.*

Preuve : Considérant le lemme 3.2 (densité non cumulative implique densité cumulative sur le même chemin) et la proposition 3.1 (densité non cumulative sur un chemin surjectif implique densité non cumulative sur tout chemin), il reste à montrer que l'existence d'une densité $c_\rho(S,\mathcal{P}) = d$ pour tout chemin surjectif implique l'existence d'une densité $d_\rho(S,\mathcal{P}) = d$ pour au moins un chemin surjectif ρ.

On va montrer un peu plus. Supposons par l'absurde qu'il existe un chemin surjectif ρ, et un $\epsilon > 0$ tel que $d_\rho(S,\mathcal{P})$ n'existe pas. Alors par définition, $D_\rho(S,\mathcal{P})$ aurait deux valeurs d'adhérence d' et d'' distinctes. Supposons $d' < d''$, et notons $\epsilon = \frac{d''-d'}{4}$. Soit ρ' et ρ'' deux suites extraites de limite d' et d'' et telles que $\forall i$, $|d'' - D_{\rho''(i)}(S,\mathcal{P})| \leq \epsilon$ et $|ds' - D_{\rho'(i)}(S,\mathcal{P})| \leq \epsilon$.

Ces chemins ne sont pas surjectifs. Mais on peut grâce à eux, construire un chemin ρ_M, surjectif, qui n'a pas de densité cumulative. On va construire ρ_M de manière à ce que pour une infinité de i, $D_{\rho_M(i)}(S,\mathcal{P})$ soit distance moins de 2ϵ de d'' et d'

Pour cela, on va suivre le processus suivant : (à chaque instant, N représente le nombre d'automates déjà énumérés).

- Énumérer une taille non encore énumérée de ρ.
- Énumérer N tailles non encore énumérées de ρ'.
- Énumérer une taille non encore énumérée de ρ.
- Énumérer N tailles non encore énumérées de ρ''.
- Recommencer...

Cette énumération est surjective (car ρ l'est), mais la densité cumulative n'a pas de limite. En effet lorsque l'on énumère à l'étape i autant de tailles de densité ϵ-proche de d', que le nombre d'automates qui ont été considérés jusque la, la densité $D_{\rho_M(i)}(S,\mathcal{P})$ est 2ϵ-proche de d'. Et il en est de même pour d''.

On a donc une contradiction : deux valeurs d'adhérences pour une densité qui devrait converger, ce qui conclut la preuve de notre lemme.

□

À propos de l'utilisation de la définition syntaxique

On a vu (cf remarque 4), que l'on aurait pu choisir, pour caractériser au plus près les règles globales, de ne considérer que les définition syntaxiques pour lesquelles le voisinage est minimal. Le résultat suivant, le premier résultat de densité de propriété que nous introduisons, montre que ceci n'est pas déterminant :

Proposition 3.5. *Selon tout $(2,1)$-chemin ρ, parmi $\mathbb{CA}$, l'ensemble des automates dont le voisinage minimal est strictement plus petit que leur voisinage syntaxique est négligeable.*

Preuve : Pour une taille (n, k) donnée, le nombre total de règles d'AC de cette taille est n^{n^k}. Ceci correspond à la possibilité de choisir chacune des n^k images parmi n. Et parmi ceux-ci le nombre de règles d'AC ne dépendant pas d'au moins une cellule de leur voisinage est inférieure à $k.n^{n^{k-1}}$. Ceci correspond au fait de choisir la cellule en question, puis les images des n^{k-1} motifs obtenus en faisant varier les états des autres cellules. C'est une inégalité car certaines règles pourraient ne pas dépendre de plusieurs cellules.

La proportion d'automates de taille de voisinage strictement inférieure à k parmi $\mathbb{CA}_{n,k}$ est donc bornée par $\frac{kn^{n^{k-1}}}{n^{n^k}} = \frac{k}{n^{n^{k-1}(n-1)}}$. Cette proportion décroit vers 0 lorsque k augmente ou lorsque n augmente. Or sur tout $(2,1)$-chemin ρ, au moins l'une des deux composantes ρ_n ou ρ_k qui paramètrent la taille tend vers l'infini.

□

Ceci montre que les perturbations de nos résultats dues à la redondance des descriptions seront insignifiantes.

Et le résultat suivant, dont la preuve est reportée à la section suivante, montre qu'il en est de même pour le choix de compter plusieurs fois ou non les automates identiques à permutation des états prêt.

Proposition 3.6. *Sur tout $(2,2)$-chemin ρ, l'ensemble des automates cellulaires invariants par au moins une permutation non triviale de son ensemble d'états est négligeable.*

Ces deux résultats permettent de relativiser l'arbitraire de nos définitions, au moins pour les cas où l'on s'intéresse à des densités parmi $\mathbb{CA}$ tout entier. Et des arguments similaires peuvent être avancés pour les densités dans la plupart des sous-classes $S \subseteq \mathbb{CA}$ parmi lesquelles nous serons amenés à considérer des densités.

Pour conclure ces quelques définitions, rappelons que, comme déjà signalé, nous utiliserons uniquement la notion plus forte (d'après le lemme 3.2) de densité non cumulative.

Selon les propriétés et les familles d'automates étudiées, deux cas se présenteront.

- Souvent des contraintes sur les tailles considérées, que ce soit à propos de la taille du voisinage, du nombre d'états, ou du rapport entre les deux apparaitront naturellement. Ce sera particulièrement vrai lorsque l'on s'intéressera à des sous-familles restreintes syntaxiquement d'automates, mais pas uniquement. Dans ce cas, une discussion sur les chemins permettra alors de rendre compte de la diversité des comportements.
- À l'opposé un certain nombre de résultats porteront sur des densités de propriétés identiques *quels que soient les chemins*, marquant alors des résultats particulièrement significatifs. Dans ces cas-là, ce sont les chemins *surjectifs*, en particulier le lemme 3.1 qui nous seront utiles.

1.2 Complexité de Kolmogorov et dénombrement

Nous introduisons maintenant quelques notions sur la complexité de Kolmogorov qui nous seront très utiles pour la suite.

L'idée est d'utiliser le fait que parmi une famille d'objets, la plupart d'entre eux sont aléatoires. On pourra se référer pour plus de détails sur ce sujet au chapitre 6 de [LV93] : *The Incompressibility Method*. Nous ajoutons ici le lien avec la densité.

La complexité de Kolmogorov

La complexité de Kolmogorov est une manière de formaliser la notion d'aléatoires.

Intuitivement, le mot binaire $0101\cdots01$ formé de 20 répétitions du mot 01 semble *moins aléatoire* que le mot $01001100111001000101110100011111101011000$. Pourtant d'un point de vue probabiliste, en tirant successivement 40 bits aléatoires, à l'aide d'une pièce par exemple, on a strictement la même probabilité d'obtenir l'une ou l'autre de ces deux chaînes, soit $\frac{1}{2^{40}}$.

La complexité de Kolmogorov permet de donner un sens à cette intuition. Elle considère qu'un objet est aléatoire s'il est long à définir. Ainsi un objet sera d'autant moins aléatoire qu'il existe une manière efficace de le décrire. Par exemple, pour notre premier mot, on aurait pu dire "20 fois *01*", mais pour le second on ne voit pas de manière vraiment plus efficace que de donner successivement tous les bits.

Bien sur, ceci nécessite de fixer le langage utilisé pour les descriptions de nos objets. Le formalisme usuel utilise des machines de Turing. Dans la suite les machines de Turing utilisées sont les plus simples possibles, prenant un mot binaire fini sur le ruban en entrée, et retournant éventuellement un mot fini, celui présent sur le ruban, si elle entre dans un état d'arrêt. Pour une machine M, on note $\mathsf{M}(y)$ le mot retourné à partir de l'entrée y si celui-ci existe, c'est-à-dire si la machine s'arrête.

On donne alors la définition suivante :

Définition 30. *Étant donné une machine de Turing M, la complexité de Kolmogorov pour la machine M d'un mot $x \in \{0,1\}^*$ est l'entier*

$$\mathsf{K}_{\mathsf{M}}(x) = \min_{p\in\{0,1\}^*} \{|p| : \mathsf{M}(p) = x\}.$$

Si $\{|p| : \mathsf{M}(p) = x\} = \emptyset$, on fixe $\mathsf{K}_{\mathsf{M}}(x) = +\infty$.

Un mot p tel que $\mathsf{M}(p) = x$ est appelé une description de x via M. À priori la complexité de Kolmogorov dépend donc de la machine considérée. Mais un résultat fondamental est justement l'existence d'une machine de Turing universelle pour laquelle la mesure de la complexité de Kolmogorov est optimale à une constante prêt. C'est ce qu'exprime le théorème suivant.

Théorème 3.7. *Il existe une machine de Turing universelle U, dite optimale, c'est-à-dire telle que*

$$\forall \mathsf{M} \exists c_{\mathsf{M}}, \forall x \in \{0,1\}^*, \mathsf{K}_{\mathsf{U}}(x) \leq \mathsf{K}_{\mathsf{M}}(x) + c_{\mathsf{M}}$$

On peut donc fixer une machine de Turing qui satisfait ce théorème, et travailler avec une complexité de Kolmogorov donnée. Nous noterons K la complexité associée à une telle machine U optimale, et c'est cette complexité qui sera utilisée dans la suite.

La notion de mots aléatoires prend alors tout sons sens : ce sont ceux dont les descriptions via U ont a peu près la même taille que x, c'est-à-dire pour lesquels il n'existe pas de description vraiment plus efficace que celle qui consiste à donner successivement toutes les valeurs binaires qui composent le mot.

Définition 31. *Pour tout $c \in \mathbb{N}$, un mot x est dit c-aléatoire, si $\mathsf{K}(x) \geq |x| - c$.*

Les résultats liés à la complexité de Kolmogorov et à cette notion d'aléatoire sont nombreux. On peut par exemple se référer à [LV93].

Notons simplement en ce qui nous concerne que cette complexité n'est pas calculable. Elle est cependant approchable par en haut (formellement l'ensemble des *descriptions* p d'un mot x est énumérable donc l'ensemble de leurs tailles aussi).

Nous allons d'abord expliquer comment nous définissons l'aléatoire pour les AC, puis nous reviendrons sur les propriétés combinatoires qui sont celles que nous utiliserons principalement, après avoir relié l'aléatoire aux tables de transition d'automates cellulaires.

L'aléatoire pour les tables de transitions.

À l'aide de cette définition des mots binaires aléatoires, on souhaite donc donner un sens à l'idée d'automates cellulaires aléatoires. Ce travail est fait en grande partie dans [JDF01].

À chaque automate cellulaire $\mathcal{A}$, d'une taille (n, k) donnée, on associe d'abord le mot $N_{\mathcal{A}}$ sur l'alphabet Σ_n formé des images successives des transitions, en les ordonnant selon l'ordre lexicographique des motifs de voisinage :

$$N_{\mathcal{A}} = \delta_{\mathcal{A}}(0^k) \cdot \delta_{\mathcal{A}}(0^{k-1} \cdot 1) \cdots \delta_{\mathcal{A}}((n-1)^k).$$

On identifie ensuite $N_{\mathcal{A}}$ à l'entier écrit en base n dont il est l'image. La fonction $\mathcal{A} \mapsto N_{\mathcal{A}}$ ainsi obtenue est une bijection entre $\mathbb{CA}_{n,k}$ et $\{0, \ldots, n^{n^k} - 1\}$. Dans le cas des automates cellulaires élémentaires, on retrouve ici leur code de Wolfram. Et cette définition peut être vue comme son extension pour toutes les tailles.

On associe enfin à $\mathcal{A}$ l'écriture binaire de $N_{\mathcal{A}}$. Et l'on note $\beta_{\mathcal{A}}$ le mot binaire de longueur $\lceil n^k log_2(n) \rceil$ obtenu en complétant cette écriture binaire par des 0 le cas échéant. Dans la suite il nous arrivera fréquemment d'identifier la table de transition d'un automate $\mathcal{A}$ donné, voir l'automate lui-même, au mot $\beta_{\mathcal{A}}$ (et par ailleurs, on notera log le logarithme en base 2).

Définition 32. *Pour tout $c \in \mathbb{N}$, un automate cellulaire $\mathcal{A}$ est dit c-aléatoire, si le mot binaire $\beta_{\mathcal{A}}$ qui lui est associé est c-aléatoire.*

Remarque 7. Un entier donné étant associé à un automate de chaque taille à partir d'un certain rang, l'entier $N_{\mathcal{A}}$ associé à un automate n'est pas caractéristique de celui-ci, et il en est donc de même de l'entier binaire qui a la même valeur.

Les propriétés combinatoires liées à la notion d'aléatoire.

L'utilité essentielle de la complexité de Kolmogorov pour notre travail réside dans ses propriétés combinatoires. En effet, et cela rejoint la discussion introductive sur l'intuition de l'aléatoire, la plupart des mots binaires sont c-aléatoires.

Proposition 3.8. *Le nombre de mots binaires de longueur l non c-aléatoire est au plus $2^{l-c} - 1$.*

Preuve : Par définition, un mot de longueur l n'est pas c-aléatoire s'il existe une description de longueur strictement inférieure à $l - c$ de ce mot. Or chaque description est associée à au plus un mot. Et le nombre de description assez courtes, est exactement le nombre de mots binaires de longueur inférieure à 2^{l-c}, soit $\sum_{0 \leq i \leq l-c-1} 2^i = 2^{l-c} - 1$. Ceci fournit donc une borne sur le nombre de mots non c-aléatoires.

□

Dans la suite, c'est de cette propriété simple que découlent la plupart des preuves liées à la complexité de Kolmogorov que nous introduirons.

Souvent, l'idée générale va être de montrer qu'une propriété induit un défaut d'aléatoire important, de manière à en déduire grâce au résultat précédent que cette propriété concerne peu d'automates et est donc négligeable.

Et on formalise ce raisonnement en précisant ce qu'est une *description*.

Une description de l'ensemble des automates est formellement une fonction calculable $\delta : \{0,1\}^* \to \mathbb{CA}$ surjective. On appelle aussi description d'un automate son antécédent par δ. On appelle description *de base* la fonction qui décrit u à l'aide de β_u.

On dit que *le coût d'une description* pour une taille donnée, est la longueur maximale des descriptions des AC de cette taille que l'on considère.

Étant donnée une propriété $\mathcal{P}$ de l'ensemble des AC, le jeu va être d'exhiber une description de l'ensemble des automates telle que les automates de $\mathcal{P}$ aient des antécédents, des *descriptions*, relativement courtes. Le plus souvent, on se contentera d'exhiber une fonction δ surjective sur l'ensemble $\mathcal{P}$, qui pourra être étendue à l'ensemble $\mathbb{CA}$ en ajoutant un bit initial à la description de chaque mot u ; celui-ci indiquera alors si la suite de la description consiste effectivement en une description selon δ ou plutôt en la description de base β_u.

Dans la suite, on utilisera la notion de *gain* d'une description δ, souvent notée $g_\delta : (n,k) \mapsto g_\delta(n,k)$, ou simplement g lorsqu'il n'y aura pas d'ambiguïté sur la description considérée. Il est défini comme la différence entre le coût de la description de base, et celui de δ.

Ainsi, étant donnée une propriété $\mathcal{P}$ des automates cellulaires, qui permet une description δ de l'ensemble des automates de $\mathcal{P}$, le *gain* associé à cette description est

$$g_\delta(n,k) = \lceil n^k.log(n) \rceil - \max_{\mathcal{A} \in \mathcal{P}_{n,k}} \{|\delta(\mathcal{A})|\}.$$

On obtient avec ces notions deux lemmes essentiels pour la suite. Le premier qui relie description et aléatoire :

Lemme 3.9. *Étant donné une propriété $\mathcal{P}$ et un chemin ρ, s'il existe une description δ des automates de $\mathcal{P}$ telle que $\lim_{i\to\infty} g_\delta(\rho(i)) = +\infty$, alors : pour tout $c \in \mathbb{N}$, il existe un rang i_c à partir duquel $\forall i \geq i_c$, $\mathcal{A} \in \mathcal{P}_{\rho(i)}$ n'est pas c-aléatoire.*

Preuve : C'est une conséquence directe des définitions. Si le gain tend vers l'infini cela signifie que les AC $\mathcal{A} \in \mathcal{P}$ sont décrits par des mots de longueur de plus en plus inférieure à $n^k.log(n)$, c'est la définition de l'aléatoire.

□

Et le second qui relie aléatoire et densité :

Lemme 3.10. *Si pour une propriété $\mathcal{P}$ et un chemin ρ, on a : pour tout $c \in \mathbb{N}$, il existe un rang i_c à partir duquel $\forall i \geq i_c$, $\mathcal{A} \in \mathcal{P}_{\rho(i)}$ n'est pas c-aléatoire, alors la propriété $\mathcal{P}$ est négligeable selon le chemin ρ.*

Preuve : Par définition, le nombre d'automates non c-aléatoire de taille (n,k), n'est pas plus grand que le nombre de mots binaires non c-aléatoires de longueur $\lceil n^k.log(n) \rceil$. Utilisant la proposition 3.8, la proportion des automates non c-aléatoires parmi $\mathbb{CA}_{n,k}$ est donc au maximum 2^{-c+1} (le $+1$ est une conséquence de la partie entière).

Or d'après notre hypothèse, pour tout $c \in \mathbb{N}$, il existe un rang i_c à partir duquel $\forall i \geq i_c$, $\mathcal{P}_{\rho(i)}$ est inclus dans l'ensemble des automates non c-aléatoires de taille $\rho(i)$. Ceci se traduit combinatoirement par :

$$\forall c \exists i_c, \forall i \geq i_c, \frac{\#\left(\mathcal{P}_{\rho(i)}\right)}{\mathbb{CA}_{\rho(i)}} \leq 2^{-c+1}$$

Ce qui est exactement la définition du fait que la limite de cette proportion est 0 : $\mathcal{P}$ est négligeable selon ρ.

□

Et la combinaison de ces deux lemmes fournit le résultat suivant :

Lemme 3.11. *Étant donné une propriété $\mathcal{P}$ et un chemin ρ, s'il existe une description δ des automates de $\mathcal{P}$ telle que $\lim_{i\to\infty} g_\delta(\rho(i)) = +\infty$, alors la propriété $\mathcal{P}$ est négligeable selon le chemin ρ.*

Liens entre aléatoire et densité

Ainsi, le lemme précédent fait de la complexité de Kolmogorov un *outil* prometteur, de par ses qualités combinatoires, pour l'étude de la densité des propriétés des automates cellulaires. Le résultat suivant, malgré sa simplicité, est porteur de sens : il nous dit qu'en réalité le lien entre densité et complexité de Kolmogorov est plus fort.

Proposition 3.12. *Si une propriété $\mathcal{P}$ est* récursivement énumérable *et négligeable selon un chemin ρ, alors elle vérifie pour tout $c \in \mathbb{N}$, il existe un rang i_c à partir duquel $\forall i \geq i_c$, $\mathcal{A} \in \mathcal{P}_{\rho(i)}$ n'est pas c-aléatoire.*

Preuve : Soit **M** une machine énumérant $\mathcal{P}$. Il suffit pour obtenir une description efficace d'un AC de $\mathcal{P}$ de donner son rang, parmi ceux de sa taille dans l'énumération par **M**.

□

Il s'agit en fait de la réciproque, pour le cas où la propriété $\mathcal{P}$ est *récursivement énumérable*, du lemme 3.10. Ainsi le fait de ne pas être négligeable apparaît pour les propriétés récursivement énumérables comme une caractérisation des propriétés des objets aléatoires.

Ceci relie encore plus fortement l'aléatoire aux densités des propriétés. Les propriétés à fortes densités apparaissent ici comme les propriétés des automates cellulaires *aléatoires* au sens Kolmogorov.

Ceci donne donc un sens supplémentaire à notre démarche. Viser à quantifier les densités de certaines propriétés, c'est déterminer lesquelles sont celles des objets aléatoires.

1.3 Quelques objets combinatoires usuels

Nous introduisons rapidement dans cette partie des objets usuels en informatique théorique, ainsi que quelques résultats combinatoires qui nous serons utiles dans la suite.

Graphes et arbres

Un graphe non orienté est une paire $G = (V, E)$ avec V l'ensemble des sommets et E un ensemble de couples $\{x, y\}$ avec $x, y \in V$ qui constituent les arrêtes du graphe.

Un *graphe orienté* est une paire $G = (V, E)$ avec V l'ensemble des sommets et $E \subseteq V^2$ l'ensemble des arrêtes, qui sont maintenant orientées.

Pour un tel graphe, on appelle composante faiblement connexe, ou simplement composante connexe un ensemble de sommets $V' \subseteq V$ tel que $\forall x, y \in V'$, il existe une suite $\{x_i\}_{i \in \{0..l\}}$, $x_i \in V'$ telle que $\forall i \in \{0..l-1\}$, $(x_i, x_i + 1) \in E$, et *soit* $x_0 = x$ et $x_l = y$, *soit* $x_0 = y$ et $x_l = x$. C'est-à-dire que pour toute paire, il existe un chemin de l'un à l'autre dans au moins l'un des deux sens.

Un arbre étiqueté est un graphe non orienté, connexe sans cycle.

Il est bien connu, mais on peut se référer à [FS09] pour une preuve à base de fonctions génératrices, que le nombre d'arbres étiquetés à n sommets est donné par n^{n-2}.

Un arbre étiqueté enraciné, est un arbre étiqueté pour lequel un sommet est distingué comme étant la racine. On les note $G = (V, E, r)$ avec $r \in V$ la racine de l'arbre. Le nombre de tels arbres de taille n est alors donné par la formule de Cayley n^{n-1}.

Et le lemme classique suivant se montre facilement.

Lemme 3.13. *Dans un arbre $G = (V, E)$, pour tous sommets $x, y \in V$, il existe un unique chemin sans répétition de x à y, c'est-à-dire une unique suite finie $(x_i)_i \in \{1..l\}$, avec $x_1 = x$, $x_l = y$, $(x_i, x_{i+1}) \in E$ et $x_i = x_j$ ssi $i = j$.*

Pour un graphe $G = (V, E)$, on définit le degré sortant (resp. entrant) d'un sommet comme l'entier $s \in V$ est $\#(\{x \in V : (s, x) \in E\})$ (resp. $\#(\{x \in V : (x, s) \in E\})$).

Un *graphe fonctionnel* est un graphe orienté tel que le degré sortant de chaque sommet est exactement 1. Intuitivement un graphe orienté $G = (V, E)$ peut être vu comme le graphe d'une application $f : V \to V$ avec $f(v) = x$ ssi $(v, x) \in E$.

On se rend compte assez simplement que la forme générale des graphes fonctionnels consiste en une famille d'arbres enracinés sur des cycles, les transitions étant orientées en direction des cycles et un sens unique étant fixé pour les arrêtes composant ceux-ci. Ceci est illustré par la figure 3.1.

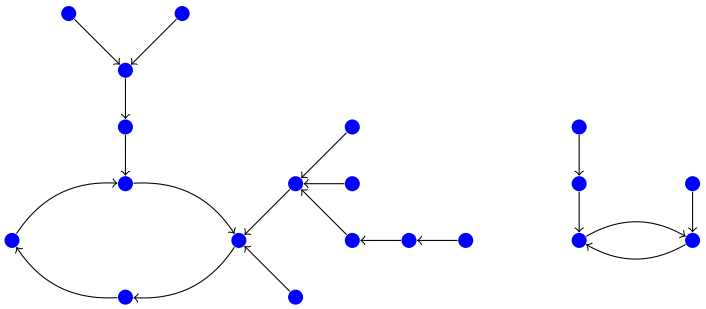


FIGURE 3.1: Exemple de graphe fonctionnel (sans les étiquettes de chaque sommet).

Une composante faiblement connexe est donc constituée d'une famille d'arbres enracinés sur un même cycle. Et inversement, si le graphe contient un seul cycle, il est composé d'une seule composante connexe. Le nombre d'arbres est borné par le nombre de points cycliques, c'est-à-dire le nombre de points sur le graphe.

Ces constatations faites, le lemme suivant paraît évident.

Lemme 3.14. *Le nombre de graphes fonctionnels (V, E), avec $V = \Sigma_n$ ayant pour unique cycle $(0, 0)$ est n^{n-2}.*

Preuve : Pour prouver ce lemme, on constate simplement la bijection entre les arbres étiquetés enracinés en 0 et nos graphes fonctionnels.

En effet, pour un graphe fonctionnel (V, E) ayant pour unique cycle $(0, 0)$, soit $G' = (V, E', 0)$, l'arbre obtenu en oubliant l'orientation des arrêtes de E et en supprimant l'arrête $(0, 0)$: $E' = \{\{x, y\} : (x, y) \in E\} \setminus \{0, 0\}$. On obtient alors un arbre étiqueté enraciné en 0. Considérons maintenant l'opération inverse : à un arbre $G' = (V', E', 0)$, on associe le graphe orienté $G = (V', E)$. Pour définir E, on utilise la propriété 3.13, pour orienter les arrêtes de E' de la manière suivante : soit $\{x, z\} \in E'$, alors $(x, z) \in E$ si et seulement si le chemin de x à 0 passe par z. Par connexité et unicité du chemin, pour toute arrête $\{x, z\}$ de E, soit (x, z) soit (z, x) est élément de E' et pour tout $x \in V$, il existe $y \in V$ unique tel que $(x, y) \in E$. Par conséquent le graphe obtenu est un graphe fonctionnel, dont le seul cycle est $(0, 0)$.

Et on vérifie facilement qu'en composant ces transformations, on retrouve le graphe de départ. Par conséquent ces deux ensembles sont en bijection, le nombre de graphe fonctionnel de taille n ayant pour unique cycle $(0, 0)$ a pour nombre d'éléments le nombre d'arbres étiquetés enracinés en 0, c'est-à-dire n^{n-2}.

□

Les mots de De Bruijn

Nous introduisons simplement la proposition suivante, qui se réfère à [dBE46].

Proposition 3.15. *Pour un alphabet Σ_n de taille n fixée, et pour toute longueur k, il existe un mot de longueur $n^k + k - 1$ qui contient tous les facteurs de taille k possible ; on le nomme mot de De Bruijn.*

On s'en convainc en considérant les graphes de De Bruijn, étiquetés par les mots de Σ_n^k reliés par une arrête si et seulement si on peut passer du premier au second en oubliant la première lettre du premier et en en ajoutant une à la fin du second. On montre facilement que ces graphes sont eulériens (grâce aux degrés entrant et sortant des sommets). On montre ensuite que le caractère eulérien du graphe pour (n, k) est lié au caractère hamiltonien du graphe pour $(n + 1, k)$.

2 Parmi les AC quelconques

On s'intéresse ici aux densités de certaines propriétés parmi l'ensemble $\mathbb{CA}$. On commence par considérer des propriétés syntaxiques, ce qui nous permet de préciser l'utilisation de la complexité de Kolmogorov pour calculer les densités, puis on s'intéresse à la nilpotence et à d'autres propriétés qui traitent du comportement à long terme des AC.

2.1 Propriétés syntaxiques, propriétés dépendant d'une sous-partie de la table

Pour commencer nous allons considérer des propriétés simples, au sens où elles sont syntaxiques, c'est-à-dire lisibles directement dans la description de fonction de transition de l'automate.

Premières propriétés

AC avec un état quiescent

Proposition 3.16. *La densité des automates cellulaires avec au moins un état quiescent parmi l'ensemble de tous les automates cellulaires est :*

(a) $1 - \frac{1}{e}$ *selon les chemins ρ tels que ρ_n a une limite infinie,*

(b) $1 - (1 - \frac{1}{n_0})^{n_0}$ *selon les chemins ρ tels que ρ_n a une limite n_0,*

(c) non définie sur les chemins ρ tels que ρ_n n'a pas de limites.

Preuve : Ceci se prouve simplement par dénombrement.
On compte d'abord parmi les n^{n^k} automates de taille (n, k) ceux n'ayant pas d'état quiescent.
Par définition, il s'agit exactement des automates $\mathcal{A} \in \mathbb{CA}_{n,k}$ tels que :

$$\forall q \in \Sigma_{\mathcal{A}}, \delta_{\mathcal{A}}(q^k) \neq q$$

Choisir $\delta_{\mathcal{A}}$ tel que $\mathcal{A}$ n'ait pas d'état quiescent revient donc à :
(1) choisir arbitrairement les $n^k - n$ transitions ne concernant pas les motifs uniformes,
(2) choisir les n images des transitions uniformes en respectant la contrainte.
Chaque choix (1) se fait parmi les n états images possibles. Chaque choix (2) se fait parmi les $n-1$ états qui n'apparaissent pas dans le mot uniforme concerné. Et tous ces choix sont indépendants.

Le nombre de telles fonctions $\delta_{\mathcal{A}}$ est donc $n^{n^k-n}.(n-1)^n$. Ceci nous donne le nombre de fonctions de transitions de taille (n, k), et donc le nombre d'automates cellulaires de $\mathbb{CA}_{n,k}$ qui n'ont aucun état quiescent.

Ainsi le nombre d'automates ayant au moins un état quiescent est :

$$\#\left(\mathbb{CA}_{n,k}\right) - n^{n^k-n}(n-1)^n = n^{n^k} - n^{n^k-n}(n-1)^n.$$

Et en divisant par $\#\left(\mathbb{CA}_{n,k}\right) = n^{n^k}$, on obtient la proportion d'automates de taille (n,k) ayant un état quiescent : $1-\left(1-\frac{1}{n}\right)^n$. Pour tout chemin ρ, on a donc

$$D_{\rho(i)}(\mathbb{CA}, \textsc{Quiescent}) = 1 - \left(1 - \frac{1}{\rho_n(i)}\right)^{\rho_n(i)}$$

Si ρ_n a une limite infinie selon ρ, alors , $d_\rho(\mathbb{CA}, \textsc{Quiescent})$ existe et vaut $1-\frac{1}{e}$. Si ρ_n a une limite finie n_0, alors $d_\rho(\mathbb{CA}, \textsc{Quiescent})$ existe, est constant à partir d'un certain rang (car $\rho_n(i)$ est entier) et vaut $1-\left(1-\frac{1}{n_0}\right)^{n_0}$. Enfin si ρ_n n'a pas de limite, $D_{\rho(i)}(\mathbb{CA}, \textsc{Quiescent})$ n'a pas de limites.

□

Cette démonstration repose sur l'indépendance entre elles des images de chaque transition. Et on voit que la propriété de quiescence ne dépend que d'une petite zone de la table de transition, celle décrivant les transitions à partir de motifs uniformes.

Allons plus loin en formalisant cette idée de *dépendance à une partie restreinte* de la table de transition.

Formellement, une sous-partie des tables de transitions est une fonction $\mathcal{M} : n,k \mapsto \mathcal{M}_{n,k} \subseteq \Sigma_n^k$ tel que $\forall (n,k)$, $\mathcal{M}_{n,k} \neq \emptyset$.

On note alors $\delta|_{\mathcal{M}_{n,k}}$ la fonction de transition restreinte aux motifs de $\mathcal{M}_{n,k}$.

Définition 33. *On dit qu'une propriété $\mathcal{P} \subseteq \mathbb{CA}$* est déterminée par une sous-partie $\mathcal{M}$ *de la table de transition, si $\forall \mathcal{A}, \mathcal{B} \in \mathbb{CA}_{n,k}$, si $\delta_\mathcal{A}|_{\mathcal{M}_{n,k}} = \delta_\mathcal{B}|_{\mathcal{M}_{n,k}}$ alors $\mathcal{A} \in \mathcal{P}$ ssi $\mathcal{B} \in \mathcal{P}$.*

Définition 34. *Et pour chaque propriété $\mathcal{P}$ et chaque sous-partie $\mathcal{M}$, on note $\mathcal{I}_{\mathbb{CA}_{n,k}}(\mathcal{P}, \mathcal{M}) = \{\delta_\mathcal{A}|_{\mathcal{M}_{n,k}} : \mathcal{A} \in \mathcal{P} \cap \mathbb{CA}_{n,k}\}$ les motifs possibles pour $\mathcal{P}$.*
Et $\mathcal{T}_{\mathbb{CA}_{n,k}}(\mathcal{P}, \mathcal{M}) = \#\left(\mathcal{I}_{\mathbb{CA}_{n,k}}(\mathcal{P}, \mathcal{M})\right)$ est le nombre de tels motifs.

Dans le cas où $\mathcal{P}$ est déterminée par $\mathcal{M}$, $\delta_\mathcal{A}|_{\mathcal{M}_{n,k}} \in \mathcal{I}_{\mathbb{CA}_{n,k}}(\mathcal{P}, \mathcal{M})$ si et seulement si $\mathcal{A} \in \mathcal{P} \cap \mathbb{CA}_{n,k}$. Pour ces propriétés, il suffit de calculer quelle proportion des motifs possibles pour cette sous-partie des tables de transitions entrainent la propriété. En réalité, et cela correspond à l'intuition, il n'est pas nécessaire de s'intéresser à la combinatoire du reste de la table de transition pour obtenir les densités limites. C'est ce que formalise le lemme suivant.

Lemme 3.17. *Soit une propriété $\mathcal{P}$ déterminée par une sous-partie $\mathcal{M}$ de la table de transition, alors la proportion d'automates de $\mathbb{CA}_{n,k}$ vérifiant $\mathcal{P}$ est :*

$$D_{n,k}(\mathbb{CA}, \mathcal{P}) = \frac{\mathcal{T}_{\mathbb{CA}_{n,k}}(\mathcal{P}, \mathcal{M})}{n^{\#(\mathcal{M}_{n,k})}}.$$

Preuve : D'après la définition de la densité, puis celle de $\mathcal{M}$, on a

$$\begin{aligned} D_{n,k}(\mathbb{CA}, \mathcal{P}) &= \frac{\#\left(\mathbb{CA}_{n,k} \cap \mathcal{P}\right)}{\#\left(\mathbb{CA}_{n,k}\right)} \\ &= \frac{\mathcal{T}_{\mathbb{CA}_{n,k}}(\mathcal{P}, \mathcal{M})) \cdot n^{n^k - \#\left(\mathcal{M}_{n,k}\right)}}{n^{n^k}} \\ &= \frac{\mathcal{T}_{\mathbb{CA}_{n,k}}(\mathcal{P}, \mathcal{M})}{n^{\#\left(\mathcal{M}_{n,k}\right)}} \end{aligned}$$

□

Mais même pour les propriétés qui ne sont pas déterminées par une sous-partie, on obtient une inégalité :

$$\begin{aligned} D_{n,k}(\mathbb{CA}, \mathcal{P}) &= \frac{\#\left(\mathbb{CA}_{n,k} \cap \mathcal{P}\right)}{\#\left(\mathbb{CA}_{n,k}\right)} \\ &\leq \frac{\mathcal{T}_{\mathbb{CA}_{n,k}}(\mathcal{P}, \mathcal{M})) \cdot n^{n^k - \#\left(\mathcal{M}_{n,k}\right)}}{n^{n^k}} \\ &\leq \frac{\mathcal{T}_{\mathbb{CA}_{n,k}}(\mathcal{P}, \mathcal{M})}{n^{\#\left(\mathcal{M}_{n,k}\right)}} \end{aligned}$$

Et celle-ci se traduit en terme de densité :

Lemme 3.18. *Pour toute propriété $\mathcal{P}$, tout chemin ρ et toute sous-partie $\mathcal{M}$ de la table de transition, si $\lim_{i\to\infty} \frac{\mathcal{T}_{\mathbb{CA}_{\rho(i)}}(\mathcal{P},\mathcal{M})}{\rho_n(i)^{\#\left(\mathcal{M}_{\rho(i)}\right)}} = 0$ alors $d_{\rho(i)}(\mathbb{CA}, \mathcal{P}) = 0$.*

Preuve : C'est le passage à la limite de l'inégalité précédente.

□

On peut alors introduire une version plus fine du lemme 3.11 qui relie le gain à la densité :

Lemme 3.19. *Pour toute propriété $\mathcal{P}$, tout chemin ρ et toute partie $\mathcal{M}$ de la table de transition, si on bénéficie d'une description de l'ensemble $\mathcal{I}_{\mathbb{CA}_{\rho(i)}}(\mathcal{P}, \mathcal{M})$ telle que le gain apporté par cette description, c'est-à-dire la différence entre $\rho_n(i)^{\#\left(\mathcal{M}_{\rho(i)}\right)} log(\rho_n(i)$ et la taille de cette description croit à l'infini lorsque i croit, alors la propriété $\mathcal{P}$ est négligeable.*

Preuve : D'après le lemme précédent, $D_{\rho(i)}(\mathbb{CA}, \mathcal{P}) \leq \frac{\mathcal{T}_{\mathbb{CA}_{\rho(i)}}(\mathcal{P},\mathcal{M})}{{\rho_n}^{\#\left(\mathcal{M}_{\rho(i)}\right)}}$. Dire que l'on possède une description dont le gain tend vers l'infini, c'est précisément dire (par passage à l'exponentielle) que la quantité de droite de cette inégalité tend vers 0, ce qui prouve le lemme.

□

Équipé de ce lemme, il sera possible dans la suite de se focaliser sur certaines zones bien délimitées de la table de transition, et de montrer, en ignorant complètement le reste de la table, la négligeabilité de certaines propriétés.

AC avec un état envahissant

Proposition 3.20. *La densité des automates cellulaires avec un état envahissant parmi l'ensemble de tous les automates cellulaires est 0 selon tout $(2, 1)$-chemin.*

Preuve : Ce résultat peut se montrer par comptage.

On s'intéresse d'abord au nombre d'automates de taille (n, k) pour lesquels le premier état de l'alphabet, 0, est envahissant.

Par définition, cette propriété est caractérisée par le fait que toutes les transitions dont le motif contient l'état 0 ont pour image cet état 0. Le nombre de telles transitions est $n^k - (n-1)^k$. Les $(n-1)^k$ autres transitions ne subissent aucune contrainte particulière et peuvent être choisies librement. Par conséquent, le nombre d'automates ayant 0 comme état envahissant est $n^{(n-1)^k}$.

Par symétrie des états, et comme un automate a au plus un seul état envahissant, le nombre d'automates ayant un état envahissant est $n^{(n-1)^k+1}$.

La proportion de ces automates dans $\mathbb{CA}_{n,k}$ est donc :

$$\frac{n^{(n-1)^k+1}}{n^{n^k}} = \frac{1}{n^{n^k-(n-1)^k-1}}.$$

Or sur tout chemin, soit ρ_n, soit ρ_k a une limite infinie, et cette proportion tend vers 0 dans chacun de ces deux cas, pour des valeurs de nombre d'états au moins égales à 2, et une taille de voisinage au moins égale à 1. Donc la densité limite existe et vaut 0 selon tout $(2, 1)$-chemin.

□

Il est intéressant sur cet exemple de présenter une seconde preuve, simple également, mais qui utilise les outils de complexité de Kolmogorov introduits plus haut.

Il s'agit donc à nouveau d'un argument de comptage, mais formalisé différemment. Sur le fond l'idée est similaire, mais sur la forme, on y gagne en évitant des calculs. Et ceci permet d'illustrer les liens entre densité et aléatoire Kolmogorov.

Preuve : Rappelons que décrire un automate de taille (n, k), quelconque nécessite exactement la taille de la description de sa table de transition, c'est-à-dire $\lceil n^k.log(n) \rceil$ bits.

Par contre, décrire un automate qui possède un état envahissant nécessite beaucoup moins de bits. En effet, il suffit de :

(1) décrire cet état envahissant, ce qui prend $\lceil log(n) \rceil$ bits.

(2) décrire les $(n-1)^k$ transitions qui n'impliquent pas cet état. C'est possible car ces transitions sont identifiées par l'état envahissant. Ceci coûte $\lceil (n-1)^k log(n) \rceil$ bits.

La description totale occupe donc $log(n) + (n-1)^k log(n)$ bits à une constante additive près.

Le gain entre la description d'un AC quelconque et celle d'un AC avec état envahissant est donc de $(n^k - (n-1)^k - 1).log(n) + c$ bits.

Ce gain croit de manière non bornée sur tout chemin. En utilisant le lemme 3.11, ceci suffit à prouver le résultat.

□

Ici, on ne peut pas directement faire plus simple en utilisant le lemme 3.19, la partie concernée de la table de transition étant l'ensemble de la table.

AC invariants par permutation

On peut maintenant prouver la proposition 3.6 évoquée plus haut.

Proposition 3.6. *Sur tout $(2, 2)$-chemin ρ, l'ensemble des automates cellulaires invariants par au moins une permutation non triviale de son ensemble d'états est négligeable.*

Preuve : Pour une taille (n, k) donnée, on introduit une description des automates invariants par au moins une permutation π des états :

- description des images des états par π : $\lceil n.log(n) \rceil$,
- description des différentes images non encore décrites par la règle locale : $\lceil N.log(n) \rceil$ pour un certain N.

En fait, on donne les images dans l'ordre habituel (lexicographique), mais on s'assure auparavant que le motif de la transition n'est pas l'image par la permutation d'un motif dont la transition a déjà été décrite. Si c'est le cas, on obtient l'image de la transition en appliquant π à l'image déjà décrite.

De cette manière, comme une permutation envoie au moins 1 état de l'alphabet vers un état plus grand dans l'ordre de l'alphabet, on *économise* au moins la description des n^{k-1} transitions dont le motif commence par la lettre image en question, dans la mesure où les transitions dont le motif commence par la lettre antécédent de celle-ci par π ont été énumérées.

Le gain est donc, à une constante près, d'au moins $n^{k-1}.log(n) - n.log(n)$. Il tend vers l'infini selon tout chemin dès que $n \geq 2$ et $k \geq 2$, la propriété est bien négligeable.

□

AC additifs

Les résultats de négligeabilité de cette propriété ainsi que le suivant sont présents dans [JDF01]. Plus précisément ces résultats montrent que toute taille (n, k), les tables de transitions des AC additifs ne sont pas aléatoires.

Proposition 3.21. *La densité des automates cellulaires additifs parmi l'ensemble de tous les automates cellulaires est 0 selon tout $(2, 2)$ chemin.*

Preuve : Ceci peut se prouver simplement en utilisant à nouveau les outils de description *à la Kolmogorov*.

Soit $\times : \Sigma_n^2 \to \Sigma_n$ l'opération et $e \in \Sigma_n$ l'état neutre pour cette opération. On décrit un AC de la manière suivante :

- description de e, qui nécessite $\lceil log(n) \rceil$ bits,
- description de la loi $\times$, qui nécessite de décrire les n^2 images ordonnées de chaque produit et nécessite donc $\lceil n^2.log(n) \rceil$ bits,
- description des images des $k.n$ transitions formées uniquement de l'état e sauf en une position où l'état peut être quelconque. Ceci nécessite donc $\lceil n.k.log(n) \rceil$ bits.

Le cout d'une telle description est donc inférieur à : $(n.k + n^2 + 1).log(n)$ bits à une constante additive près. Le gain face à la description basique de cout $\lceil n^k.log(n) \rceil$ est $g(n, k)$ donné par :

$$g(n,k) = \left(n^k - n.k - n^2 - 1\right).log(n) + c$$

Et il est donc croissant de limite infinie selon n'importe quel chemin, dès lors que $n \geq 2$ et $k \geq 2$, ce qui permet de conclure en utilisant le lemme 3.11.

□

AC surjectifs

Proposition 3.22. *La densité des automates cellulaires 1-équilibrés parmi l'ensemble de tous les automates cellulaires est 0 selon tout $(2, 2)$-chemin.*

Ce résultat est en substance également présent dans [JDF01]. Les auteurs l'avaient déjà combiné avec le théorème 1.4 pour montrer que les tables d'AC surjectifs ou injectifs ne sont pas aléatoires.

Proposition 3.23. *La densité des automates cellulaires surjectif, ou injectifs parmi l'ensemble de tous les automates cellulaires est* 0 *selon tout* $(2,2)$*-chemin.*

Preuve : [de la proposition 3.22] À nouveau, comme cette propriété est syntaxique, elle est simple à dénombrer précisément. Un automate est 1-équilibré si ses n^k transitions se répartissent équitablement entre les n images possibles. Le nombre d'automates 1-équilibrés de taille (n,k) est donc identique au nombre de partitions de n^k éléments en n parties, chacune de taille n^{k-1}, soit $\frac{n^k!}{(n^{k-1}!)^n}$. Ce qui donne une proportion de $\frac{n^k!}{(n^{k-1}!)^n.n^{n^k}}$ pour ces AC.

Soit $\rho : i \mapsto (\rho_n(i), \rho_k(i))$ un chemin quelconque. La proportion pour la taille $\rho(i)$ est donc $\frac{\rho_n(i)^{\rho_k(i)}!}{(\rho_n(i)^{\rho_k(i)-1}!)^{\rho_n(i)}\rho_n(i)^{\rho_n(i)^{\rho_k(i)}}}$. Lorsque i croit, $\rho_n(i)^{\rho_k(i)}$ et $\rho_n(i)^{\rho_k(i)-1}$ tendent vers l'infini, on peut donc utiliser l'équivalent de Stirling de la factorielle à l'infini ($x! \sim_\infty \sqrt{2\pi x}\left(\frac{x}{e}\right)^x$) pour obtenir un équivalent de cette proportion sur le chemin ρ :

$$\frac{\rho_n(i)^{\rho_k(i)}!}{(\rho_n(i)^{\rho_k(i)-1}!)^{\rho_n(i)}\rho_n(i)^{\rho_n(i)^{\rho_k(i)}}} \sim_\infty \frac{1}{(2\pi)^{(\rho_n(i)-1)/2}.\rho_n(i)^{(\rho_n(i)\rho_k(i)-\rho_n(i)-\rho_k(i))/2}}$$

qui tend vers 0 lorsque i croit si $\rho_n(i)\rho_k(i) - \rho_n(i) - \rho_k(i) \geq 0$. Par conséquent, selon n'importe quel $(2,2)$-chemin ρ, la densité limite est nulle.

□

Sous-automates et facteurs

Nous traitons maintenant deux propriétés particulièrement intéressantes dans la mesure où ce sont celles qui constituent les relations locales des simulations usuelles.

AC ayant un sous-automate

On dit qu'un sous-automate d'un AC de taille (n,k) est non trivial si l'ensemble d'états de celui-ci est de taille m, avec $1 < m < n$.

Proposition 3.24. *Selon n'importe quel* $(1,3)$*-chemin, la densité des automates cellulaires ayant au moins un sous-automate non trivial est* 0*.*

Ce résultat avait déjà été démontré par Guillaume Theyssier dans le cas de chemins à taille du voisinage constant, on le montre ici pour le cas général.

Preuve :

On veut savoir quelle proportion parmi les automates de taille (n,k) a un sous-automate de taille m, avec $1 < m < n$.

On va utiliser une description à la Kolmogorov. En effet pour décrire un automate ayant un sous-automate, il suffit de donner successivement :

- la taille de ce sous-automate, ce qui nécessite $log(n)$ bits
- les différents états qui le composent, ce qui nécessite $m.log(n)$ bits
- la table de transitions de ce sous-automate c'est-à-dire $m^k.log(m)$
- les transitions restantes $(n^k - m^k).log(n)$

Le gain d'une telle description est donné par

$$\begin{aligned} g(n,k,m) &= \lceil n^k log(n) \rceil - \Big[(1+m).\lceil log(m) \rceil + \lceil m^k.log(m) \rceil + \lceil (n^k - m^k).log(n) \rceil\Big] \\ &\geq (m^k - m - 1).log(n) - m^k.log(m) - 4. \end{aligned}$$

La dérivée de $g(n,k,m)$ selon m est la fonction

$$\begin{aligned} h(n,m,k) &= (k.m^{k-1}-1).log(n) - k.m^{k-1}.log(m) - m^{k-1} \\ &= (k.log(\frac{n}{m}) - 1).m^{k-1} - log(n). \end{aligned}$$

Sa dérivée seconde est

$$i(n,k,m) = (k.(k-1).log\left(\frac{n}{m}\right) - 2.k + 1).m^{k-2}.$$

Donc $\forall k \geq 3$, $\forall 1 < m < n$, $i(n,k,m) > 0$. Et la fonction $m \mapsto h(n,k,m)$ est croissante, pour toute valeur de $k \geq 3$ et $n \geq 3$ sur l'intervalle $[2, m-1]$.

Or

$$h(n,k,2) = \left(k.2^{k-1} - 1\right) log(n) - 2^{k-1}\left(k.log(2) - 1\right)$$

On résout facilement $h(n,k,2) < 0$ en fonction de n, et on obtient

$$n < 2^{\frac{2^{k-1}(k-1)}{k.2^{k-1}-1}}$$

Or si $k \geq 3$, $\frac{1}{k.2^{k-1}} \leq \frac{1}{12}$, donc $\frac{2^{k-1}(k-1)}{k.2^{k-1}-1} \leq \frac{log(2)}{1-\frac{1}{k.2^{k-1}}} \leq \frac{12}{11}$.

Donc $h(n,k,2) < 0$ implique $n < 3$. Par conséquent , dans les cas où la recherche de sous-automate est non triviale ($n \geq 3$), $m \mapsto h(n,k,m)$ est positive en $m = 2$. Et comme sa dérivée est positive, elle est positive pour tout l'intervalle $[2, n-1]$.

Par conséquent, $m \mapsto g(n,k,m)$ est croissante à son tour sur cet intervalle. On s'intéresse maintenant à $g(n,k,2) = (2^k - 3).log(n) - 2^k.log(2)$, celui-ci tend vers l'infini lorsque $\min(n,k)$ croit à l'infini, et c'est le cas pour tout chemin.

Ainsi, le gain de notre description sur tout $(3,3)$-chemin ρ, est minoré par une fonction qui tend vers l'infini sur ce chemin, on en déduit donc que la propriété est négligeable sur ces chemins. Comme il n'y a aucun automate avec un sous-automate non trivial pour $n = 1$ ou $n = 2$, ceci conclut la preuve.

□

AC ayant un facteur

On considère maintenant la seconde relation locale qui intervient pour les simulations, mais cette fois sur une famille plus restreinte de chemins.

Comme pour les sous-automates, on dit qu'un facteur de taille m d'un automate de taille (n,k) est non trivial si $1 < m < n$.

Proposition 3.25. *Selon tout chemin ρ à ρ_n constant, l'ensemble des automates possédant un facteur non trivial est négligeable.*

Preuve : Considérons l'ensemble des automates de taille (n,k) possédant un facteur de taille m avec $1 < m < n$.

Pour chaque $\mathcal{A}$ de cet ensemble, on note $\mathcal{B}$ un facteur de taille m, et $\pi : \Sigma_{\mathcal{A}} \to \Sigma_n$ la fonction telle que $\mathcal{B} \trianglelefteq_\pi \mathcal{A}$.

On peut décrire un automate $\mathcal{A}$ de cette ensemble de la manière suivante :

(1) description de m, ainsi que des images par π des états de $\Sigma_{\mathcal{A}}$: ceci nécessite $\lceil log(n) \rceil + \lceil n.log(m) \rceil$.
(2) description de la règle de $\mathcal{B}$: $\lceil m^k.log(m) \rceil$
(3) description, pour chaque transition de $\mathcal{A}$ de son image en utilisant la compatibilité avec la règle de $\mathcal{B}$: $\lceil n^k.log(n-m+1) \rceil$.

Le troisième point nécessite quelques explications. Après le (1), on dispose non seulement des images à travers π mais aussi, par conséquent d'une numérotation des états de $\pi^{-1}(x)$ pour chaque $x \in \Sigma_{\mathcal{B}}$. Donc pour décrire une transition $\delta_{\mathcal{A}}(x_1, \cdots, x_{k_{\mathcal{A}}}) = y$, il suffit de regarder d'abord l'image $z = \delta_{\mathcal{B}}(\pi(x_1), \cdots, \pi(x_{k_{\mathcal{A}}}))$, puis de donner le numéro de y parmi $\pi^{-1}(z)$. Hors, π étant surjectif le cardinal de chaque $\pi^{-1}(x)$ est borné par $n-m+1$.

Le gain de cette description est donc donné par

$$\begin{aligned} g(n,k,m) &= \lceil n^k.log(n) \rceil - \lceil log(n) \rceil - \lceil n.log(m) \rceil - \lceil m^k.log(m) \rceil - \lceil n^k.log(n-m+1) \rceil \\ &\geq n^k.log\left(\frac{n}{n-m+1}\right) - (m^k+n).log(m) - log(n) - 5 \end{aligned}$$

Pour n constant, et donc k croissant, ce gain tend vers l'infini, ce qui prouve notre résultat. □

Ces deux résultats sont intéressants car les propriétés qu'ils considèrent sont utilisées pour étudier la structure de l'ensemble $\mathbb{CA}$. Cependant, pour pouvoir en déduire quelque chose sur cette structure, pour quantifier certaines classes d'automates liées aux pré-ordres induits, il faudrait arriver à faire intervenir les changements d'échelles, ce qui n'est pas le cas pour l'instant.

2.2 Propriétés dynamiques

On va maintenant s'intéresser à des propriétés qui caractérisent le comportement global des automates cellulaires. Pour cela on introduit un dernier outil qui permet de relier les densités sur différents chemins.

"Recollement" de densités

Plus précisément, ce lemme, permet de relier des densités selon les chemins à ρ_n et ρ_k constants à une densité plus générale selon des chemins quelconques

Lemme 3.26. *Étant donnée une propriété $\mathcal{P}$ de $\mathbb{CA}$, telle que*
(1) selon tout chemin ρ à ρ_n constant de valeur au moins n_0, on a $d_\rho(S, \mathcal{P}) = 0$,
(2) selon tout chemin ρ à ρ_k constant de valeur au moins k_0, on a $d_\rho(S, \mathcal{P}) = 0$,
(3) de plus la décroissance en ρ_n est uniforme, c'est-à-dire
$\forall \epsilon > 0$, $\exists n_\epsilon$ *tel que* $\forall n \geq n_\epsilon$, $\forall k$, $D_{n,k}(S, \mathcal{P}) < \epsilon$.
Alors on a selon tout (n_0, k_0)-chemin ρ, $d_\rho(S, \mathcal{P}) = 0$.

Preuve : Soit $\mathcal{P}$ une propriété satisfaisant nos hypothèses, ρ un chemin et $\epsilon > 0$ quelconques.
Soit n_ϵ, la valeur définie par le lemme.
D'après (1), pour chaque valeur de $n \leq n_\epsilon$, il existe $k_{n,\epsilon}$ tel que si $k \geq k_{n,\epsilon}$ alors $D_{n,k}(S, \mathcal{P}) < \epsilon$. Posons $k_\epsilon = \max_{n \leq n_\epsilon}\{k_{n,\epsilon}\}$.
Donc pour tout ϵ, il existe n_ϵ, k_ϵ tel que si $n \geq n_\epsilon$ ou $k \geq k_\epsilon$ alors $D_{n,k}(S, \mathcal{P}) < \epsilon$.
Et comme souvent, on utilise le fait que pour tout (n,k), pour tout chemin ρ, il existe i_ϵ tel que si $i \geq i_\epsilon$ alors $\rho_n(i) \geq n$ ou $\rho_k(i) \geq k$. Par conséquent, il existe i_ϵ tel que si $i \geq i_\epsilon$ alors

$D_{\rho(i)}(S, \mathcal{P}) < \epsilon$. Ce qui montre que $d_\rho(S, \mathcal{P}) = 0$.

□

On peut bien évidemment montrer un résultat symétrique, mais il ne nous sera pas utile ici.

La nilpotence

Nilpotents

Proposition 3.27. *L'ensemble des automates nilpotents est négligeable.*

Preuve : On commence par étudier le cas ρ_k constant. On s'intéressera ensuite au cas ρ_n constant avant de s'appuyer sur ces résultats partiels pour montrer le résultat général en utilisant le lemme 3.26.

Cas ρ_k constant

Commençons donc par le cas ρ_k constant de valeur k, et donc ρ_n croissant à l'infini.

Rappelons qu'un automate nilpotent a toujours un et un seul état quiescent. On compte tout d'abord le nombre d'automates nilpotents de taille (n, k), dont l'état quiescent est 0. On appelle ces automates 0-*nilpotents*.

Et l'on s'intéresse aux transitions $\alpha 0^{k-1} \to \beta$, pour $\alpha, \beta \in \Sigma_n$.

À chaque automate $\mathcal{A} \in \mathbb{CA}_{n,k}$ dont 0 est le seul état quiescent, on associe le graphe $\mathcal{G}_\mathcal{A} = (V_\mathcal{A}, E_\mathcal{A})$ tel que :

- $V_\mathcal{A} = \Sigma_n$
- $E_\mathcal{A} = \{(\alpha, \beta) : \delta_\mathcal{A}(\alpha 0^{k-1}) = \beta\}$,

Ces graphes sont des graphes fonctionnels, c'est-à-dire que le degré sortant de chaque sommet est au plus 1 (voir le paragraphe 1.3). Et chacun de ces graphes contient la transition $0 \to 0$.

De plus, d'une part chacun de ces graphes fonctionnels contenant la transition $0 \to 0$ est associé à exactement n^{n^k-n} automates, ceux dont les n transitions de motif $\alpha 0^{k-1}$ correspondent aux transitions du graphe, les autres transitions étant quelconques. Et d'autre part si $\mathcal{A}$ est 0-nilpotent, alors le graphe $\mathcal{G}_\mathcal{A}$ ne contient pas d'autres cycles que $0 \to 0$. Par conséquent, le nombre d'automates 0-nilpotents de taille (n, k) est majoré par le nombre de graphes fonctionnels avec le seul cycle $0 \to 0$, multiplié par n^{n^k-n}.

D'après le lemme 3.14 le nombre de tels graphes est n^{n-2}. Par conséquent le nombre d'automates 0-nilpotents est d'au plus n^{n^k-2}. Par symétrie du choix de l'état quiescent (parmi n possibles), le nombre total d'automates cellulaires nilpotent est donc borné par n^{n^k-1}. Ce qui donne une proportion d'automates cellulaires nilpotents de taille (n, k) inférieure à $\frac{1}{n}$, et montre que la nilpotence sur ces $(1, 1)$-chemins à ρ_k constant est négligeable.

Cas ρ_n constant

On s'intéresse maintenant au cas ρ_n constant, et donc ρ_k croissant à l'infini.

Considérons d'abord les automates de taille (n, k) fixée. À chaque mot $u \in \Sigma_n^*$, de longueur $p < k/2$ on associe le sous-shift $S_u = \{\sigma^i({}^\omega u^\omega) : 0 \leq i \leq p-1\}$ des configurations périodiques formées de la répétition de ce mot et le langage $\mathcal{L}_{u,k}$ des facteurs de longueur k de S_u. On a $\mathcal{L}_{u,k} = \{u_{[i,p-1]} u^{\lfloor \frac{k}{p} \rfloor} u_{[0,k-p\lfloor \frac{k}{p} \rfloor - i]} : 0 \leq i \leq p-1\}$.

Dès lors que u n'est pas uniforme (c'est-à-dire contient au moins deux états), un AC nilpotent $\mathcal{A}$ ne stabilise pas S_u : pour un mot u non uniforme, $\mathcal{A}({}^\omega u^\omega) \notin S_u$. On va utiliser cette propriété pour décrire plus facilement les transitions impliquées dans le calcul de l'image de u.

Si la longueur p de u est première, alors $\mathcal{L}_{u,k}$ contient p mots différents. Comme aucun S_u n'est stabilisé, pour un u connu, on peut décrire l'ensemble des p images des transitions issues de ces mots en utilisant $log(n^p - p)$ bits au lieu des $log(n^p)$ habituellement nécessaires pour décrire p images.

On gagne un peu en utilisant cette description, mais pour augmenter le gain on peut considérer différents mots u, à condition que les transitions impliquées dans l'évolution des différents ${}^{\omega}u^{\omega}$ soient indépendantes, c'est-à-dire que les langages associés soient disjoints.

Pour deux mots u et v non-uniformes *et de longueurs premières différentes*, on considère les langages $\mathcal{L}_{u,k}$ et $\mathcal{L}_{v,k}$. Soit $x = u_{[i,|u|-1]}u^{\lfloor \frac{k}{|u|-1} \rfloor}u_{[0,k-|u|-1\lfloor \frac{k}{|u|-1} \rfloor - i]}$, alors ce mot ne peut pas s'écrire $v_{[i,|v|-1]}v^{\lfloor \frac{k}{|v|-1} \rfloor}v_{[0,k-|v|-1\lfloor \frac{k}{|v|-1} \rfloor - i]}$. Donc l'intersection de $\mathcal{L}_{u,k}$ et $\mathcal{L}_{v,k}$ est vide.

De même pour deux mots u et v *différents mais de même longueur* p, la primalité de p, et la non-uniformité de u et v permettent de garantir que les langages $\mathcal{L}_{u,k}$ et $\mathcal{L}_{v,k}$ sont égaux si et seulement si les deux mots sont égaux à rotation prêt, et d'intersection vide sinon.

Par conséquent, on peut cumuler les gains concernant les $\mathcal{L}u, k$ pour tous les mots de même taille non uniformes et non égaux à rotation prêt, c'est-à-dire pour $\frac{n^p-n}{p}$ mots différents. Et on peut aussi cumuler ces gains pour chaque taille première différente.

Le gain d'une description combinant tous les gains possibles est alors donné par la formule :

$$\begin{aligned} g(n,k) &\geq \sum_{p \text{ premier, } p \leq k/2} (log(n^p) - \log(n^p - p)) \\ &\geq log\left(\prod_{p \text{ premier, } p \leq k/2} \left(1 + \frac{p}{n^p - p}\right)^{(n^p-p)/p)} \right) \end{aligned}$$

Or le terme général de ce produit tend vers e, qui est une constante plus grande que 1.

Par conséquent ce gain tend vers l'infini lorsque la taille k du voisinage tend vers l'infini dès lors que n vaut au moins 2. Sur tout $(2,1)$ chemin à ρ_n constant, l'ensemble des AC nilpotents est négligeable.

On peut maintenant utiliser le lemme 3.26 pour conclure directement.

□

Ce résultat pose également un certain nombre de questions par rapport aux problèmes de décidabilité. En effet, la nilpotence est une propriété connue pour son indécidabilité. Mais ce résultat de densité montre que sa non calculabilité ne concerne qu'une faible partie des automates. En effet, ce problème est visiblement *simple en moyenne* : lorsque l'on considère des automates de plus en plus grands, seule une fraction négligeable d'entre eux sont nilpotents. Ceci pose donc la question d'une étude de la calculabilité en moyenne des propriétés des automates cellulaires, à l'instar de ce qui est esquissé pour les machines de Turing ([Ryb07, HM06]).

Autres propriétés

Dans la suite on continue à s'intéresser à des propriétés dynamiques. Comme toujours dans les automates, il y a une difficulté à passer des propriétés syntaxiques à des propriétés globales. Les résultats présentés ici concernent les propriétés globales. Cependant, on peut dire qu'ils donnent un certain nombre d'informations qui, en un sens, sont plus intermédiaires entre le niveau local et le niveau global, tant la quantification syntaxique est présente de manière forte.

AC avec de fortes contraintes sur l'ensemble limite

Proposition 3.28. *Pour toute constante m, selon tout chemin ρ, tel que $\lim_{i\to\infty} \rho_n(i) = \infty$, les automates cellulaires dont le langage limite contient moins de m lettres sont négligeables.*

Plus précisément, on va montrer le résultat suivant.

Proposition 3.29. *Pour toute constante m, selon tout chemin ρ, tel que $\lim_{i\to\infty} \rho_n(i) = \infty$, les automates cellulaires dont l'ensemble limite contient moins de m configurations uniformes différentes sont négligeables.*

Preuve :

Soit m une constante fixée.

On va utiliser une construction similaire à celle introduite dans le cas de la nilpotence pour obtenir une description économe d'une partie de la table des automates cellulaires de taille (n, k) ayant moins de m configurations dans leur ensemble limite.

On s'intéresse ici uniquement à la sous-partie de la table de transitions constituée des transitions uniformes $\alpha^k \to \beta$ avec $\alpha, \beta \in \Sigma_n$.

Formellement $\mathcal{M}_{n,k} = \{\alpha^k : \alpha \in \Sigma_n\}$.

À chaque motif $\delta : \mathcal{M}_{n,k} \to \Sigma_n$ de la sous-partie, on associe le graphe $\mathcal{G}_\delta = (V_\delta, E_\delta)$ défini comme suit :

- $V_\delta = \Sigma_n$
- $E_\delta = \{(\alpha, \beta) : \delta(\alpha^k) = \beta\}$.

Les graphes obtenus sont des graphes fonctionnels, (voir le paragraphe 1.3), que l'on peut se représenter comme une famille d'arbres enracinés sur des cycles.

Remarquons que les sommets qui appartiennent à des cycles correspondent exactement aux états dont la configuration uniforme associée appartient à l'ensemble limite. Ces états appartiennent eux-même à l'ensemble limite.

Supposons maintenant que moins de m points soient sur ces cycles, on peut donner une description efficace de ce graphe :

- d'abord décrire la forêt (ou l'ensemble d'arbres) à n noeuds qui consiste en notre graphe fonctionnel auquel on a enlevé les transitions entre les points des cycles. Le nombre de telles forêts est $(n+1)^{n-1}$, c'est le nombre d'arbres étiquetés à $n+1$ sommets dans lequel on supprime le sommet de plus grand indice. Donc le cout de la description est de $\lceil (n-1).log(n+1) \rceil$ bits.
- ensuite décrire les transitions entre les points des cycles. Le cout d'une telle description est au maximum de $\lceil (m+1).log(m) \rceil$ bits.

On peut donc décrire un tel graphe fonctionnel en $(n-1).log(n+1) + (m+1).log(m) + 2$ bits au lieu des $n.log(n)$ bits nécessaires à une description dans le cas quelconque.

De la description de ce graphe, on déduit une description des n transitions uniformes de notre automate, c'est-à-dire de la sous-partie de la table que nous considérons. Le gain associé à cette description est de

$$\begin{aligned} g(n,m) &= n.log(n) - (n-1).log(n+1) - (m+1).log(m) \\ &= log(\frac{n^n}{(n+1)^{n-1}}) - (m+1).log(m) \end{aligned}$$

Sur un chemin ρ, ce gain croit donc à l'infini lorsque ρ_n croit, ce qui suffit, grâce au lemme 3.19 à prouver nos deux propositions.

□

De plus, on observe dans la preuve que l'on peut obtenir le même résultat sur un chemin pour un nombre d'états dans l'ensemble limite non plus constant mais logarithmique en la taille de l'ensemble d'états.

Pour l'autre paramètre de taille, on montre le résultat suivant :

Proposition 3.30. *Pour toute constante l, selon tout chemin ρ, tel que $\lim_{i\to\infty} \rho_k(i) = \infty$, l'ensemble des automates cellulaires dont l'ensemble limite ne contient pas tous les mots de taille l est négligeables.*

Preuve :

Choisissons un tel l. On va introduire une description de l'ensemble des automates dont l'ensemble limite évite au moins un mot u de longueur l.

Pour un automate donné, l'existence d'un tel u signifie en particulier qu'aucun des sous-shifts qui contiennent u n'est stabilisé par cet automate.

On peut choisir un tel u, non uniforme, quitte à le choisir de longueur $l+1$. On considère alors les mots $w = u \cdot v$ de longueurs totale p premières, $l+1 < p \leq k$, formés en concaténant u avec un mot v de longueur $p-l$. Et l'on va utiliser une description efficace de l'ensemble des transitions qui interviennent dans le calcul de l'image des configurations ${}^{\omega}w^{\omega}$.

La primalité de p, associée à la non-uniformité de u, garantit que l'ensemble $\mathcal{L}_w$ des mots de taille k apparaissant dans ces configurations est de taille p : si on écrit $k = pq+r$ la division entière de k par q, on a :

$$\mathcal{L}_w = \{w_{[i,p-1]} \cdot w^{q-1} \cdot (ww)_{[0,r+i-1]}; i \in \{0..p-1\}\}$$

Si u n'est pas dans l'ensemble limite, cela signifie que pour chaque v, le $\mathcal{L}_{uv}$ n'est pas stabilisé. Ceci signifie que l'on peut décrire les p transitions impliquées dans le calcul de son image par $log(n^p - p)$ bits au lieu des $log(n^p)$ du cas général.

De plus deux ensembles $\mathcal{L}_w$ et $\mathcal{L}_{w'}$ sont disjoints, sauf si $w = uv$ et $w' = uv'$ sont égaux à rotation prêt, c'est-à-dire si v et v' sont égaux. Donc le nombre de tels ensembles $\mathcal{L}_w$ disjoints de taille p est n^{p-l}.

On peut donc décrire simplement les transitions qui servent au calcul de ${}^{\omega}w^{\omega}$:

- on décrit un mot u qui n'apparait pas dans l'ensemble limite : ce qui coute $\lceil (l+1).log(n+1) \rceil$ bits
- on décrit ensuite, pour tout p, pour chacun des n^{p-l} mots v satisfaisant notre contrainte (ceux-ci peuvent être ordonnées facilement selon un ordre canonique), les p transitions issues de motifs de $\mathcal{L}_w$. Ceci nécessite $log(n^p - p)$ bits pour chaque groupe de p transitions, le cout pour chaque taille p est donc d'environ $n^{p-l}.log(n^p - p)$. Une description de toutes ces transitions nécessite donc $\lceil \sum_{p\ premier, p\leq k} n^{p-l}.log(n^p - p) \rceil$ bits.

Par conséquent, le gain total de cette description par rapport à la description basique des pn^{p-l}

transitions pour toutes les tailles p est donnée à une constante près, par

$$\begin{aligned} g(n,k) &= \sum_{p\ premier, p\leq k} pn^{p-l}.log(n^p) - n^{p-l}.log(n^p-p) \\ &= \sum_{p\ premier, p\leq k} n^{p-l}.(log(n^p) - log(n^p-p)) \\ &= \sum_{p\ premier, p\leq k} n^{p-l} log\left(\frac{n^p}{n^p-p}\right) \\ &= n^{-l}. \sum_{p\ premier, p\leq k} n^p log\left(1+\frac{p}{n^p-p}\right) \end{aligned}$$

Par conséquent, lorsque p croit, le terme général de la somme est strictement plus grand que 1, de l'ordre de p. Sur un chemin ρ tel que ρ_k n'est pas borné, p non plus, et le gain total tend vers l'infini : on peut utiliser le lemme 3.19 pour conclure.

□

Comme précédemment, il serait possible d'élargir (un tout petit peu ici) les chemins considérés en acceptant de faire varier légèrement n et l en fonction de k.

En réalité, ce que montrent les deux résultats précédents peut se reformuler de la manière suivante.

- L'alphabet limite d'un automate typique de taille (n,k) croit lorsque la taille n considérée augmente (pour une voisinage constant).
- De même la taille du plus petit mot qui n'est pas dans l'ensemble limite d'un AC de taille (n,k) croit lorsque la taille k considérée augmente (pour un ensemble d'états constant).

AC ayant un Mot d'Eden petit

On peut également montrer des résultats similaires pour les Mots d'Eden :

Proposition 3.31. *Selon les chemins à ρ_k croissant, pour toute taille l fixée, l'ensemble des automates cellulaires admettant un mot d'Eden de longueur inférieure ou égale à l est négligeable.*

Preuve : Encore une fois, nous introduisons une description plus efficace de l'ensemble étudié. Celle-ci utilise les mots de De Bruijn introduits dans la proposition 3.15.

Rappelons que pour (n,k) fixé, il s'agit de mots sur Σ_n, de longueur n^k+k-1 qui contient tous les facteurs de taille k possible.

Il est donc possible de décrire la table de transition d'un automate cellulaire par un mot w de longueur n^k dont la i-ème lettre décrit l'image de la transition qui correspond au i-ème facteur de longueur k du mot de De Bruijn pour (n,k) L'avantage est que cela garantit qu'un mot apparaissant dans w est un mot de l'ensemble image de l'automate.

Notre description efficace pour les automates ayant un petit mot d'Eden u, c'est-à-dire dont l'ensemble image évite un mot u de taille l est alors

- une description de u, cela nécessite $\lceil (l+1).log(n) \rceil$ bits
- une description de w qui code la table de transition, à l'aide de blocs de longueur l à choisir parmi les n^l-1 blocs différents de u : cela coute $\lceil \frac{n^k}{l}.log(n^l-1) \rceil$ bits

Le gain de notre description est (toujours à une constante près) donné par :

$$g(n,k) = n^k log(n) - (l+1)log(n) - \frac{n^k}{l} log(n^l - 1) = n^k log\left(\frac{n}{(n^l-1)^{\frac{1}{l}}}\right) - (l+1)log(n)$$

qui tend vers l'infini avec k puisque $\frac{n}{(n^l-1)^{\frac{1}{l}}} > 1$.

□

En réalité, on voit bien que ce résultat peut être élargi à des mots u de longueur variable, mais négligeable devant k dès lors que la complexité de Kolmogorov, c'est-à-dire le cout de description du mot u est très faible.

AC avec propagation sur un fond uniforme

On cherche maintenant à aborder la propagation d'information dans l'évolution des AC. L'objectif d'une telle approche est d'arriver à quantifier des propriétés telles que la sensibilité ou au contraire le fait d'avoir des points d'équicontinuité. Et plus généralement de comprendre la circulation d'information dans les évolutions d'automates cellulaires typiques.

Cependant nos résultats sont plus modestes, dans la mesure où on se restreint pour l'instant à des fonds uniformes, et à l'existence d'une information se propageant.

Formellement, on dit qu'un ensemble d'états $X = (x_i)_{i\in\{0..p-1\}} \subseteq \Sigma_n$ décrit un fond uniforme de taille p pour l'AC $\mathcal{A}$, si $\mathcal{A}^i({}^\omega(x_0)^\omega) = {}^\omega(x_{i \bmod p})^\omega$, et si les x_i sont distincts. Et l'on supposera dans la suite que x_0 est le plus petit état de X selon l'ordre de Σ_n (afin d'éviter de considérer plusieurs fois chaque fond).

Et l'on dit qu'un état $x \in \Sigma_\mathcal{A}$ se propage vers la droite sur un fond uniforme $X = (x_i)_{i\in\{0..p-1\}}$ si $\mathcal{A}^t({}^\omega(x_0)^\omega)_{\lfloor\frac{k_\mathcal{A}-1}{2}\rfloor(t+1)} \neq \mathcal{A}^t({}^\omega(x_0)\cdot x \cdot (x_0)^\omega)_{\lfloor\frac{k_\mathcal{A}-1}{2}\rfloor(t+1)}$ (avec x en position 1).

Proposition 3.32. *Selon tout chemin ρ tel que $\lim_{i\to\infty}\rho_n = \infty$, la densité limite des AC possédant un état se propageant à vitesse maximale sur un fond uniforme est* 1.

Preuve : On s'intéresse d'abord à la probabilité pour un automate d'avoir propagation d'un état sur un fond uniforme donné, de période temporelle p sous l'action de l'automate.

Étant donné un automate cellulaire $\mathcal{A} \in \mathbb{CA}$, de taille (n,k) et un ensemble d'états $X = \{x_0, x_1, ...x_{p-1}\} \subseteq \Sigma_\mathcal{A}$.

À chaque ensemble d'états cycliques X de taille p de l'automate cellulaire $\mathcal{A}$, on associe le graphe $\mathcal{G}(\mathcal{A}, X) = (V, E)$ définit comme suit ;

- l'ensemble d'états est $V = X \times \Sigma_\mathcal{A}$,
- l'ensemble des transitions est $E = \{((x_i,\alpha),(x_{i+1 \bmod p},\beta)) : \delta_\mathcal{A}(\alpha x_i^{k-1}) = \beta\}$.

$\mathcal{G}(\mathcal{A}, X)$ détermine exactement la sous-partie de la table de transition de taille np, constituée des transitions issues des motifs αx_i^{k-1}. La fonction $\mathcal{G} : (\mathcal{A}, X) \mapsto \mathcal{G}(\mathcal{A}, X)$ a deux propriétés importantes :

- Chaque graphe de son image a exactement n^{n^k-np} automates antécédents.
- $\mathcal{A}$ a un état se propageant sur les configurations uniformes formées d'états de X si et seulement si $\mathcal{G}(\mathcal{A}, X)$ a un cycle différent du cycle uniforme.

Par conséquent la proportion de cycles de taille p admettant un état propageant est exactement la proportion de graphes dans l'image de $\mathcal{G}$ ayant au moins deux cycles.

Nous nous intéressons donc à la combinatoire de l'ensemble image de $\mathcal{G}$. Il s'agit d'une sous-famille particulière de l'ensemble des graphes fonctionnels.

D'abord, un tel graphe contient le graphe uniforme. Ensuite on peut voir le reste du graphe comme p sous-ensembles indicés de 0 à $p-1$, la i-ème partie, contenant les $n-1$ éléments (x_i, α). Et pour chaque tel élément, le graphe contient soit une arrête vers un des $n-1$ éléments de la $(i+1 \bmod p)$-ème partie soit une arrête vers l'élément correspondant $(x_{i+1 \bmod p}, x_{i+1 \bmod p})$ du cycle uniforme. Ceci se visualise sur la figure 3.2.

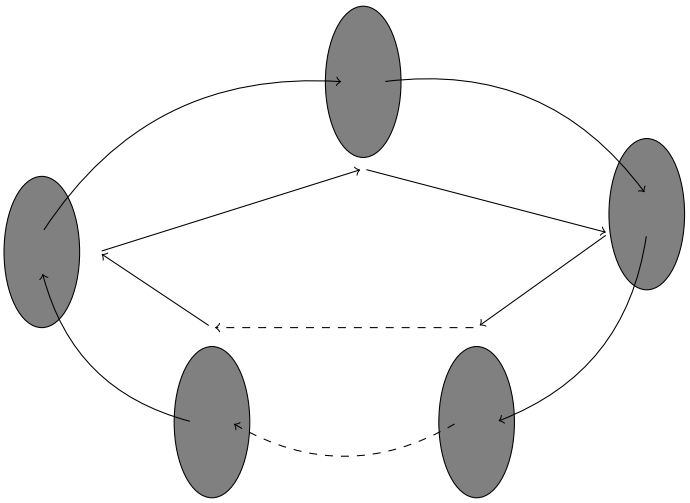


FIGURE 3.2: Le type de graphes fonctionnels intervenants dans la construction.

Il y a donc $n^{(n-1)p}$ tels graphes différents.

Parmi ceux-ci, comptons ceux qui contiennent un cycle supplémentaire, de longueur p, passant par le point $(x_0, 0)$. Pour décrire un tel cycle, il faut choisir un point intermédiaire parmi les $n-1$ choix possibles dans chacune des $p-1$ parties intermédiaires. Pour décrire le reste du graphe, il faut ensuite choisir une image, dans la partie suivante, pour chacun des $(n-2)p$ points restants. La proportion de tels graphes est donc $P_1 = \frac{(n-1)^{p-1} n^{p(n-2)}}{n^{(n-1)p}} = \frac{(n-1)^{p-1}}{n^p}$.

Si on souhaite maintenant compter le nombre de ceux qui contiennent deux cycles de longueur p, l'un passant par $(x_0, 0)$ et l'autre par $(x_0, 1)$, il faut choisir un couple d'états dans chaque partie intermédiaire, puis fixer les autres transitions. On obtient une proportion $P_2 = \frac{(n-1)^{p-1}(n-2)^{p-1}}{n^{2p}}$ de tels graphes.

On généralise ce raisonnement pour obtenir la proportion P_i de ceux qui contiennent

$$P_i = \frac{\prod_{j=1}^{i} (n-j)^{p-1}}{n^{ip}}$$

Or ces résultats sont indépendants du choix des points de passage des cycles dans la première partie. On peut donc en déduire la proportion P des graphes ayant au moins un second cycle de longueur p. D'après le principe d'inclusion-exclusion :

$$P = \sum_{i<n} (-1)^{i+1} \binom{n-1}{i} P_i$$

Pour estimer cette somme, commençons par remarquer que $\forall j < \frac{n}{2}$

$$\binom{n-1}{2j+1}P_{2j+1} - \binom{n-1}{2j+2}P_{2j+2} = \binom{n-1}{2j+1}P_{2j+1}\left(1 - \frac{(n-2j-2)^p}{(2j+2)n^p}\right)$$
$$> 0$$

On peut évaluer plus finement la somme des deux premiers termes :

$$\binom{n}{1}P_1 - \binom{n}{2}P_2 = \frac{n^p}{(n-1)^p}\left(1 - \frac{(n-2)^p}{2n^p}\right)$$

Si $p = o(n)$, lorsque n croît vers l'infini, le premier terme du produit tend vers 1, et le second vers $1/2$. En particulier cette différence est strictement supérieure à une constante strictement positive dès que n est assez grand, et toujours pour un p négligeable devant n. Et comme P s'écrit comme une somme de termes positifs, il est au moins aussi grand que cette différence.

Par conséquent, lorsque n croit, dans l'ensemble image de $\mathcal{G}$, une proportion $P > 0$ des éléments ont au moins deux cycles. Donc une proportion $P > 0$ des couples $(\mathcal{A}, X)$ possèdent un état se propageant sur les configurations uniformes d'états pris dans X selon la règle de $\mathcal{A}$.

De plus pour un automate donné, les ensembles des transitions qui interviennent dans la propagation sur chacun de ses cycles uniformes sont disjoints.

Et il est connu que l'espérance du nombre de cycles de taille inférieure ou égale à $\frac{n}{log(n)}$ dans les cycles de taille n croit lorsque n augmente.

On peut par exemple se référer à [FO90] dans lequel les auteurs prouvent, entre autre nombreux résultats, que l'espérance du nombre de cycles de taille r fixée tend vers $1/r$ lorsque le nombre de sommets dans le graphe augmente.

Par conséquent, lorsque le nombre d'états augmente, presque tous les AC ont un état se propageant sur un fond uniforme.

□

On a donc pu jusqu'ici quantifier un certain nombre de propriétés. On a vu que les propriétés syntaxiques peuvent être quantifiées, souvent facilement. On a vu aussi que l'on arrive à quantifier certaines propriétés à plus long terme, mais que ceci nécessite plus d'effort. Les derniers résultats évoqués, en particulier celui sur la propagation appellent des développements et généralisations ultérieures.

3 Parmi des sous-familles d'AC définies syntaxiquement

Nous abordons maintenant une approche complémentaire de l'étude menée dans la partie précédente : il s'agit de l'exploration d'ensembles restreints d'automates cellulaires. On s'intéresse maintenant à la densité de propriétés non plus parmi l'ensemble $\mathbb{CA}$ en entier mais sur des sous-familles de celui-ci.

Ce type d'exploration a été initié par Guillaume Theyssier [The04] qui a étudié les automates cellulaires captifs de cette manière. Il a ainsi montré qu'à voisinage fixé et donc à nombre d'états croissants, la densité d'automates cellulaires intrinsèquement universels parmi les automates cellulaires captifs est 1.

Nous reprenons ici à notre compte deux aspects importants de son travail :

– D'abord le type de sous-familles considérées. Il s'était limité aux AC captifs, nous élargissons l'étude mais en conservant le caractère syntaxique des familles considérées : pour chacune, l'appartenance ou non à la classe est *lisible* dans la description finie de l'AC.

– Ensuite l'étude de l'universalité intrinsèque comme propriété que nous quantifions dans ces familles.

Cependant nous élargissons l'étude par la généralité des sous-familles considérées, ainsi que par l'utilisation de chemins plus généraux.

Comme dans le cas des automates cellulaires captifs, nous observons pour la quasi-totalité des classes restreintes *naturelles* que nous avons considérées, que l'universalité intrinsèque est une propriété répandue. Toutefois, ces résultats sont rarement vrais pour tout chemin : dans la plupart des cas, un paramètre de taille privilégié est induit par la restriction syntaxique.

Signalons aussi ici que dans le cas des automates cellulaires quelconques, la question de la densité de l'universalité intrinsèque reste ouverte.

Cette étude fournit donc un éclairage sur le lien entre règle locale et comportement global des automates cellulaires. En effet ce que nous montrons ici pour nos sous-familles est un lien *statistique* entre leurs restrictions syntaxiques et l'universalité intrinsèque qui alors qu'elle semble pourtant une propriété globale très forte est présente presque partout.

3.1 Méthode générale

Les preuves de ces différents résultats ont un schéma similaire. L'idée est de montrer deux choses. D'une part, que la classe considérée contient un automate cellulaire intrinsèquement universel. D'autre part que les automates aléatoires de cette classe ne peuvent, *à partir d'une certaine taille*, éviter de simuler un automate donné de la classe. Ces deux idées sont déjà présentes dans l'article sur les captifs [The04].

Le second point se formalise en terme d'*une loi* 0-1 sur chaque classe considérée.

Définition 35. *Une propriété $\mathcal{P}$ est dite* croissante *si pour tout automate $\mathcal{B} \in \mathcal{P}$, alors $\mathcal{B} \preccurlyeq_{\sqsubseteq} \mathcal{A}$ implique $\mathcal{A} \in \mathcal{P}$.*

De même une propriété est dite décroissante *si pour tout automate $\mathcal{A} \in \mathcal{P}$, alors $\mathcal{B} \preccurlyeq_{\sqsubseteq} \mathcal{A}$ implique $\mathcal{B} \in \mathcal{P}$.*

On parlera de *loi* 0-1 pour une classe $\mathcal{C}$ si toute propriété croissante non triviale dans la classe $\mathcal{C}$ a densité 1, toute propriété décroissante non triviale a densité 0.

L'universalité étant une propriété croissante, ceci nous permettra de conclure.

L'idée centrale pour montrer un tel résultat est que *les gros automates de la classe sont très peu nombreux à éviter de simuler un automate donné, petit, de cette classe*. Pour obtenir ce type de résultat, on utilise certaines propriétés propres à la simulation intrinsèque, en particulier le fait que la simulation d'un automate se réalise sur un sous-shift par bloc donné du simulateur.

L'essentiel de nos constructions, pour montrer qu'un gros automate simule presque surement un petit automate donné, consiste à exhiber un grand nombre de sous-shifts sur chacun desquels les gros automates ont une probabilité non nulle de simuler le petit. Il suffit ensuite de montrer que la probabilité qu'un même *gros* automate évite en même temps toutes ces possibilités diminue lorsque la taille de l'automate croît.

Pour ce faire nous utiliserons le plus souvent l'*indépendance* des sous-shifts de simulation considérés. Il nous suffira alors de montrer que sur les chemins considérés, le nombre de sous-shift où la simulation peut intervenir croit plus vite que la probabilité de simulation sur un sous-shift donné ne décroit.

Formellement, cette indépendance est définie sur les motifs de taille k apparaissant dans les sous-shifts. Pour un ensemble de motifs de transition $E \subseteq \Sigma_{\mathcal{A}}^{k_{\mathcal{A}}}$, on note $\delta_{\mathcal{A}}|_E$ la restriction de $\delta_{\mathcal{A}}$

à E. Et l'on dit qu'une famille d'ensembles de motifs $\{E_j\}_{j\in J}$ est indépendante pour une classe d'automates $\mathcal{C}$ si pour chaque taille (n,k), on a

$$\#\left(\{\delta_{\mathcal{A}}|_{\cup_{j\in J}E_j} : \mathcal{A}\in\mathcal{C}_{n,k}\}\right) = \prod_{j\in J}\#\left(\{\delta_{\mathcal{A}}|_{E_j} : \mathcal{A}\in\mathcal{C}_{n,k}\}\right).$$

Remarque 8. Cette indépendance correspond à l'intuition de l'indépendance probabiliste. Et en effet elle est liée à l'indépendance des événements "les images des motifs de E_j ont *telle* valeur fixée" pour chaque E_j.

Par extension, on dira qu'une famille de sous-shift caractérisée par des ensembles de motifs $\{E_j\}_j$, est indépendante si la famille $\{E_j\}_j$ est indépendante. Et de même pour une famille de fonctions de simulation $\{i_j\}_{j\in J}$, avec $i_j : \Sigma^{\mathbb{Z}}_{\mathcal{A}_0} \to \Sigma^{\mathbb{Z}}_{\mathcal{A}}$, si la famille des sous-shifts $\{i_j(\Sigma^{\mathbb{Z}}_{\mathcal{A}_0})\}_{j\in J}$ est indépendante.

Cette notion nous permet d'introduire le lemme suivant qui formalise un raisonnement récurrent dans les preuves de lois 0-1.

Rappelons d'abord que $\mathcal{S}_{\mathcal{A}}$ désigne l'ensemble des automates qui simulent $\mathcal{A}$ et $\mathcal{S}_{\mathcal{A},X}$ l'ensemble de ceux qui le simulent sur un sous-shift de simulation inclus dans X.

Lemme 3.33. *Soit $\mathcal{A}_0 \in \mathcal{C}_{n_0,k_0}$ un automate donné, et $\{X_i\}_{i\in I}$ une famille de sous-shifts indépendants pour $\mathcal{C}$, tels que pour chacun d'eux, il existe $\alpha_i \geq 0$ et*

$$\frac{\#\left(\mathcal{S}_{\mathcal{A}_0,X_i}\cap\mathcal{C}_{n,k}\right)}{\#\left(\mathcal{C}_{n,k}\right)} \geq \alpha_i$$

alors,

$$\frac{\#\left(\mathcal{S}_{\mathcal{A}_0}\right)\cap\mathcal{C}_{n,k}}{\#\left(\mathcal{C}_{n,k}\right)} \geq 1-\prod_{i\in I}(1-\alpha_i)$$

Preuve : C'est simplement la définition de l'indépendance des X_i renormalisée par le nombres d'AC de la classe.

□

Nous allons donc passer en revue quelques classes, en s'intéressant à chaque fois aux chemins sur lesquels on obtient une densité importante d'automates universels.

À chaque fois les descriptions, que ce soit celles concernant la loi 0-1, ou celles montrant l'existence d'AC universels, sont différentes et intimement liées à la restriction syntaxique considérée. Et justement, il semble extrêmement compliqué, au delà du schéma général décrit jusque là de *factoriser* ces preuves en dégageant des étapes communes.

En réalité un éclairage pourrait être donné sur ces résultats si on arrivait à quantifier l'universalité intrinsèque parmi l'ensemble $\mathbb{CA}$. En effet, une densité 1 amènerait à penser qu'en un sens nos sous-classes sont représentatives de $\mathbb{CA}$. Par contre une densité 0, signifierait réellement que le fait de se restreindre à certains types de règles introduit fortement *de la structure*, qui amplifie les capacités de simulation des AC. Mais pour l'instant, cette question reste ouverte.

Un dernier point important est à noter. Dans la partie précédente, nous nous sommes attachés à donner explicitement, autant que possible, des bornes précises sur les chemins, en précisant toujours s'il s'agissait de $(1,1)$-chemin ou plutôt de $(2,3)$-chemins... Dans la suite nous ne le faisons plus.

La raison en est que les bornes fournies par nos constructions sont sans soute bien plus larges. Nous n'avons en effet pas cherché à les raffiner particulièrement, et au contraire nous avons privilégié des constructions relativement simples. En effet ce n'est pas du tout sur la recherche de bornes précises que nous nous sommes focalisés. *Il faudra donc comprendre dans la suite les résultats de la forme "*pour tout chemin*" comme des "*il existe (n, k) tel que pour tout (n, k)-chemin*".*

3.2 À ensemble d'états croissant

On l'a déjà évoqué, Guillaume Theyssier a montré que selon tous chemins à taille de voisinage fixée, la densité des automates cellulaires captifs était 1. Nous reprenons ici la preuve de ce résultat.

Les captifs

Proposition 3.34. *Selon tout chemin ρ tel que la taille du voisinage ρ_k est constante, parmi les AC captifs, l'universalité intrinsèque a densité 1.*

Ceci se montre à l'aide des deux lemmes suivants :

Lemme 3.35. *Il existe un automate cellulaire captif qui est intrinsèquement universel.*

Preuve : On sait qu'il existe un automate cellulaire universel $\mathcal{U}$, d'ensemble d'états $\Sigma_\mathcal{U}$ et de voisinage de taille 3.

On va construire un automate cellulaire captif qui le simule. Il existe de multiples manières de le faire. Le plus simple est de considérer un automate $\mathcal{A}$ de même ensemble d'états $\Sigma_\mathcal{A} = \Sigma_\mathcal{U}$, et de rayon $3 + 2n_\mathcal{U}$.

On appelle *bibliothèque*, et on note $L_{\Sigma_\mathcal{U}}$ le mot formé de la concaténation de tous les états de $\Sigma_\mathcal{U}$ dans l'ordre de cet alphabet : si on note $\Sigma_\mathcal{U} = \{0, 1, \dots, n_\mathcal{U} - 1\}$, alors $L_{\Sigma_\mathcal{U}} = 0 \cdot 1 \cdots n_\mathcal{U} - 1$.

On considère alors la fonction qui à chaque état de $\Sigma_\mathcal{U}$ associe le mot formé de cet état concaténé à une telle bibliothèque.

$$\begin{aligned} i : \Sigma_\mathcal{U} &\to \Sigma_\mathcal{A}^{n_\mathcal{U}+1} \\ x &\mapsto x \cdot L_{\Sigma_\mathcal{U}} \end{aligned}$$

On peut alors choisir la règle locale suivante pour $\delta_\mathcal{A}$: si le motif considéré contient trois états séparés par deux bibliothèques, son image par $\delta_\mathcal{A}$ est l'image par $\delta_\mathcal{U}$ du motif formé par les trois états ; sinon l'état de la cellule ne change pas. Toutes ces transitions sont captives, grâce aux bibliothèques.

Et l'on vérifie facilement que $\mathcal{A}$ simule $\mathcal{U}$ à travers le codage i et avec $t_\mathcal{A} = t_\mathcal{U} = l_\mathcal{U} = 1$ et $l_\mathcal{A} = n_\mathcal{U} + 1$:

$$\mathcal{U} \sqsubseteq_i \mathcal{A}^{<n_\mathcal{U}+1,1,0>}.$$

Par conséquent $\mathcal{A}$ est universel.

□

Lemme 3.36. *Selon tout chemin à taille de voisinage k constante, parmi les AC captifs, toute propriété $\mathcal{P}$, croissante et telle que $\exists \mathcal{A}_0 \in \mathcal{P}_{n_0,k_0}$ avec $k_0 \leq k$, est de densité 1.*

Preuve : Soit un automate $\mathcal{A}_0$, captif, de taille (n_0, k_0). Étant donnée une taille $n \geq n_0$, $k \geq k_0$, on s'intéresse à l'ensemble $\mathcal{S}_{\mathcal{A}_0} \cap \mathcal{C}_{n,k}$.

Comme la propriété est croissante, on peut toujours choisir $k = k_0$. En effet, pour tout $\mathcal{A}_0$ de rayon k_0, et tout $k \geq k_0$, il existe $\mathcal{A}_0{}'$ de rayon k, tel que $\mathcal{A}_0 \preccurlyeq_\sqsubseteq \mathcal{A}_0{}'$: il suffit d'étendre le rayon en adaptant la règle.

On considère donc $\mathcal{S}_{\mathcal{A}_0} \cap \mathcal{C}_{n,k_0}$.

Σ_n est l'ensemble d'états de tout automate $\mathcal{A}$ de cet ensemble. Pour chaque sous-ensemble S de taille n_0 de Σ_n, $\mathcal{A}$ simule $\mathcal{A}_0$ sur les configurations formées exclusivement d'états de S, et à travers l'isomorphisme $i_S : \Sigma_{\mathcal{A}_0} \to S$ préservant l'ordre lexicographique, si et seulement si la règle locale $\delta_{\mathcal{A}}$ de $\mathcal{A}$ est compatible avec la règle locale de $\mathcal{A}_0$ à travers i_S.

Formellement, par définition, $\mathcal{A}_0 \preccurlyeq_{\sqsubseteq i_S} \mathcal{A}$ si $\forall c \in \Sigma_{\mathcal{A}_0}^{\mathbb{Z}}$, $\overline{i_S} \circ \mathcal{A}_0(c) = \mathcal{A} \circ \overline{i_S}(c)$. Au niveau des règles locales cela s'écrit : $\forall x_1, \dots, x_{k_0} \in \Sigma_{\mathcal{A}_0}$, $i_S(\delta_{\mathcal{A}_0}(x_1, \dots, x_{k_0})) = \delta_{\mathcal{A}}(i_S(x_1), \dots, i_S(x_{k_0}))$.

Ceci revient à dire que $\mathcal{A}_0 \preccurlyeq_{\sqsubseteq i_S} \mathcal{A}$ si et seulement si les ${n_0}^{k_0}$ transitions de $\delta_{\mathcal{A}}$ correspondantes sont fixées de l'unique manière compatible avec $\delta_{\mathcal{U}}$.

Chaque transition a donc une seule valeur compatible parmi les au plus k_0 possibles (grâce à la captivité des automates considérés). Et de plus ces transitions sont toutes différentes. La fraction des automates $\mathcal{A}$ de taille (n, k_0) qui réalisent une simulation $\mathcal{A}_0 \preccurlyeq_{\sqsubseteq i_S} \mathcal{A}$ pour un i_S fixé est donc d'au moins $1/{k_0}^{{n_0}^{k_0}}$. Et on note que cette fraction est indépendante de n. On a

$$\frac{\mathcal{S}_{\mathcal{A}_0,i} \cap \mathcal{C}_{n,k}}{\mathcal{C}_{n,k}} \geq \frac{1}{{k_0}^{{n_0}^{k_0}}}.$$

Considérons maintenant une famille de tels ensembles S, et les sous-shifts associés. Si les ensembles d'états considérés sont tous disjoints, les sous-shifts de simulation sont indépendants, cela se vérifie simplement par dénombrement.

Ainsi, si on considère simplement $\lfloor \frac{n}{n_0} \rfloor$ ensembles disjoints, en prenant par exemple $S_i = \{in_0 + 1, \cdots in_0 + n_0\}$ pour chaque $i \in \{0, \dots, .., \frac{n}{n_0} \rfloor\}$, l'inégalité précédente s'applique pour chacun, et on peut utiliser le lemme 3.33 pour obtenir :

$$\frac{\mathcal{S}_{\mathcal{A}_0} \cap \mathcal{C}_{n,k}}{\mathcal{C}_{n,k}} \geq 1 - (1 - \frac{1}{{k_0}^{{n_0}^{k_0}}})^{\lfloor \frac{n}{n_0} \rfloor}$$

.

Par conséquent, lorsque l'on considère des chemins ρ pour lesquels ρ_n a une limite infinie, cette proportion tend vers 1.

Donc selon les chemins à ρ_k constant, la densité limite des automates simulant un AC donné tend vers 1, ce qui prouve que les densités croissantes ont densité 1 selon ces chemins.

□

Remarque 9. Comme on le verra souvent dans la suite, on se rend compte qu'en réalité, la même construction permet d'étendre le résultat à des chemins pour lesquels la taille du voisinage croit légèrement au lieu d'être constante. Ici, il faut par exemple une croissance de k_0 en $log(log(n))$ pour maintenir la convergence.

Ce résultat a donc été le premier concernant des sous-familles d'AC.

On peut également étendre ce résultat à un ensemble de sous-classes des automates cellulaires captifs, mais il faut alors prendre garde à maintenir l'indépendance des sous-shifts.

Les AC invariants par permutation de ses états

Dans la suite, on va s'intéresser à des familles symétriques par rapport à des permutations des positions dans le voisinage, comme les multi-ensemblistes évoqués au chapitre 2.

Il semble donc logique de se poser la question de l'analogue pour les ensembles d'états.

Définition 36. *Un AC est dit* invariant par permutation de ses états *si sa règle locale est invariante par toutes permutations de l'ensemble d'états. Formellement, $\mathcal{A}$ est state-symetric si pour toute permutation $\pi : \Sigma_n \to \Sigma_n$, $\forall x_1, \ldots, x_{k_\mathcal{A}}, \in \Sigma_n$, $\delta_\mathcal{A}(x_1, \ldots, x_{k_\mathcal{A}}) = \pi^{-1}(\delta_\mathcal{A}(\pi(x_1), \ldots, \pi(x_{k_\mathcal{A}})))$.*

On peut aussi remarquer le lien qui existe avec les AC captifs :

Lemme 3.37. *Soit (n, k) tel que $1 \leq k \leq n-2$. Alors les AC de cette classe de taille (n, k) sont captifs.*

Preuve : Supposons par l'absurde qu'il existe un AC de cette classe dont une des transitions ne soit pas captive. Par notre hypothèse sur la taille de l'alphabet, on peut alors considérer une permutation laissant l'ensemble des états apparaissant dans le motif de la transition invariant, et changeant l'image. Cet automate est donc changé par la permutation...

□

On note également que sur tout chemin ρ à taille de voisinage ρ_k constante à partir d'un certain rang, et donc nombre d'états croissant, le nombre d'AC de cette classe est borné par une constante (fonction de ρ_k). En effet, une fois que les transitions dont les motifs impliquent les ρ_k premières lettres de l'alphabet ont été décrites, l'ensemble des transitions de l'automate est connu.

Par conséquent, on a le résultat suivant :

Proposition 3.38. *Selon tout chemin à taille de voisinage constante, parmi l'ensemble des AC state-symetric, soit $\mathcal{P}$ une propriété représentée parmi toutes les tailles de ce chemin sauf éventuellement un nombre fini d'entre elles. Alors si la densité de $\mathcal{P}$ existe, elle est strictement inférieure à1.*

Preuve : C'est une conséquence directe du fait que le nombre d'AC state-symetric pour chaque tailles est globalement bornée.

□

L'intérêt de cette classe est d'illustrer que, tout de même, rien n'est évident. Nous serons amené dans la partie suivante, à considérer une famille dont le comportement à ensemble d'états constant est similaire. Cependant pour la majorité des familles que nous introduisons maintenant , ce n'est pas le cas.

3.3 À taille de voisinage croissante

Il est en effet possible d'exhiber un certain nombre de classes ayant des propriétés similaires à celle des captifs, mais cette fois lorsque la taille du voisinage croit..

La première que nous considérons est celle introduite au chapitre 1.

Les multi-ensemblistes

Rappelons qu'un AC est dit multi-ensembliste si sa règle de transition est donnée par une fonction du multi-ensemble des états présents dans le motif de voisinage. Intuitivement, l'automate oublie l'information positionelle lorsqu'il applique la règle : il connaît le nombre d'occurrences de chaque état dans le voisinage mais ne sait pas de quelle cellule chacun vient.

Nous montrons le résultat suivant :

Proposition 3.39. *Selon tout chemin à nombre d'états constant, parmi les AC multi-ensemblistes, la densité des universels est 1.*

Ceci repose à nouveau sur deux lemmes :

Lemme 3.40. *Il existe un automate cellulaire multi-ensembliste, et intrinsèquement universel.*

Lemme 3.41. *Selon tout chemin à nombre d'états ρ_n constant, parmi les AC multi-ensemblistes, les propriétés croissantes contenant au moins un AC de taille (n_0, k_0) avec $n_0 \leq \rho_n - 2$ ont densité 1.*

Preuve : Pour cela, on montre que étant donné un automate cellulaire $\mathcal{A}_0$ de taille (n_0, k_0), la densité des automates cellulaires qui simulent cet automate est 1 selon les chemins à nombre d'états ρ_n constant, avec $\rho_n \geq n_0 + 2$.

Comme dans le cas des captifs, nous allons pour une taille donnée (n, k) (avec $n \geq n_0 + 2$, $k \geq k_0$) exhiber un certain nombre de sous-shifts disjoints sur lesquels la simulation aura une certaine probabilité de se produire.

Il y a de nombreux choix possibles pour de tels sous-shifts. Nous considérons ceux formés de la manière suivante.

Soit l'entier $l = \lfloor \frac{k-k_0}{k_0-1} \rfloor$, et pour tout entier $k_0 \leq i \leq l - k_0$, v_i, le mot $v_i = 0^i \cdot 1^{l-i}$. On note encore $o = k - k_0 - (k_0 - 1)l$ le *décalage*, on remarque que $0 \leq o < k_0 - 1$.

Et pour chaque i considéré ($k_0 \leq i \leq l - k_0$), on définit le sous-shift S_i comme l'ensemble des configurations alternant v_i et un état de $\Sigma_{\mathcal{A}_0}$. Par exemple pour $k_0 = 3$ et $k = 28$, alors $l = 12$ et $i \in [3..12]$; avec $x, y, z, t \in \Sigma_{\mathcal{A}_0}$,la configuration suivante est dans S_7.

$$\cdots 0011111\ x\ \underbrace{000000011111}_{v_7}\ y\ \underbrace{000000011111}_{v_7}\ z\ \underbrace{000000011111}_{v_7}\ t\ 0000001\cdots$$

Pour fixer les idées, un motif de longueur k centré sur les états est :

$$\cdots 0011111\ \underbrace{x\ \ 000000011111\ y\ 000000011111\ z\ 0}\,00000011111\ t\ 0000001\cdots$$

o est alors le nombre de cellules à l'extérieur de la zone délimitée par les états de $\Sigma_{\mathcal{A}_0}$ extrémaux dans le voisinage, ici $o = 2$. L'intervalle des valeurs de i (donné par $o < k_0 \leq i$ et $l - i \leq k_0 < o$) contraint fortement les nombres de 0 et de 1 qui peuvent être vus en plus des v_i complets.

Plus précisément, les motifs de longueur k qui apparaissent dans les configurations de S_i sont les suivants :

- les motifs "*centrés sur les états*". On les définit comme ceux qui contiennent exactement $k_0 - 1$ mots v_i, $\lceil \frac{o}{2} \rceil$ états 0 et $\lfloor \frac{o}{2} \rfloor$ états 1 supplémentaires, ainsi que k_0 états de $\Sigma_{\mathcal{A}_0}$.
  On note ces motifs $C(i, a_1, a_2, \cdots, a_{k_0})$, les a_i étant les états de $\Sigma_{\mathcal{A}_0}$ présents dans le voisinage.
- les motifs décalés, avec k_0 états de $\Sigma_{\mathcal{A}_0}$. Ils contiennent donc k_0 états de $\Sigma_{\mathcal{A}_0}$, $k_0 - 1$ mots v_i *complets* (au sens multi-ensembliste, on ne sait rien de leur position), ainsi que j états 0 et $o - j$ états 1, pour un certain j avec $0 \leq j \leq o$.
  On note ces motifs $D1(i, j, a_1, a_2, \cdots, a_{k_0})$, les a_i étant les états de $\Sigma_{\mathcal{A}_0}$ présents dans le voisinage.
- les motifs décalés avec $k_0 - 1$ états de $\Sigma_{\mathcal{A}_0}$. Ils contiennent donc k_0 états de $\Sigma_{\mathcal{A}_0}$, $k_0 - 1$ mots v_i *complets* (au sens multi-ensembliste, on ne sait rien de leur position), ainsi que j états 0 et $o + 1 - j$ états 1, pour un certain j avec $0 \leq j \leq o + 1$.
  On note ces motifs $D2(i, j, a_1, a_2, \cdots, a_{k_0-1})$, les a_i étant toujours les états de $\Sigma_{\mathcal{A}_0}$ présents dans le voisinage.

Pour qu'un AC de $\mathcal{MS}_{n,k}$ simule $\mathcal{A}_0$ sur un sous-shift S_i, avec i fixé, il est alors suffisant que cet AC respecte les transitions suivantes :

- $C(i, a_1, a_2, \cdots, a_{k_0}) \to \delta_{\mathcal{A}}(a_1, a_2, \cdots, a_{k_0})$
- si $j > \lceil \frac{o}{2} \rceil$, $D1(i, j, a_1, a_2, \cdots, a_{k_0}) \to 0$
- si $j < \lceil \frac{o}{2} \rceil$, $D1(i, j, a_1, a_2, \cdots, a_{k_0}) \to 1$
- si $j \geq \lceil \frac{o+1}{2} \rceil$, $D2(i, j, a_1, a_2, \cdots, a_{k_0-1}) \to 0$
- si $j \leq \lfloor \frac{o+1}{2} \rfloor$, $D2(i, j, a_1, a_2, \cdots, a_{k_0-1}) \to 1$

Ces transitions sont bien toutes multi-ensemblistes, indépendantes, et compatibles entre elles (c'est-à-dire qu'elles reposent sur des multi-ensembles différents). Et elles permettent de stabiliser (avec éventuellement un décalage) les motifs v_i, et de faire évoluer les états de $\Sigma_{\mathcal{A}_0}$ conformément à $\delta_{\mathcal{A}}$. On a bien :

$$\mathcal{A}_0 \sqsubseteq_i \mathcal{A}^{<l+1,1,\epsilon>}$$

avec un certain ϵ, nul dans le cas où k est impaire, valant $\frac{l}{2}$ dans le cas paire.

Le nombre de transitions à fixer *pour un S_i donné* est exactement le nombre total de motifs multi-ensemblistes différents, c'est-à-dire :

- ${n_0}^{k_0}$ motifs de type C différents
- ${n_0}^{k_0}(o+1)$ motifs de type $D1$ différents
- ${n_0}^{k_0}(o+2)$ motifs de type $D2$ différents

Soit un total de ${n_0}^{k_0}(2o+3)$ multi-ensembles dont il faut fixer l'image d'une manière unique spécifiée. Or $o < k_0 - 1$, donc le nombre de transitions à fixer est inférieur ou égal à ${n_0}^{k_0}(k_0+1)$.

La proportion d'automates de $\mathcal{MS}_{n,k}$ vérifiant ces transitions et donc simulant $\mathcal{A}_0$ sur S_i est donc au moins $1/{n_0}^{{n_0}^{k_0}(k_0+1)}$. On a :

$$\frac{\#\left(\mathcal{S}_{\mathcal{A}_0,i} \cap \mathcal{MS}_{n_0+2,k}\right)}{\#\left(\mathcal{MS}_{n_0+2,k}\right)} \geq \frac{1}{n^{{n_0}^{k_0}(k_0+1)}}$$

Mais la famille $\{S_i\}_i$ est indépendante. Par conséquent s'après le lemme 3.33, la densité d'automates simulant $\mathcal{A}_0$ dans $\mathcal{MS}_{n,k}$ vérifie :

$$\frac{\#\left(\mathcal{S}_{\mathcal{A}_0}\right) \cap \mathcal{MS}_{n,k}}{\#\left(\mathcal{MS}_{n,k}\right)} \geq 1 - \left(1 - \frac{1}{n^{{n_0}^{k_0}(k_0+1)}}\right)^{l-2k_0}$$

Or $l = \lfloor \frac{k-k_0}{k_0-1} \rfloor$ croit vers l'infini lorsque k croit donc

$$\frac{\#\left(\mathcal{S}_{\mathcal{A}_0}\right) \cap \mathcal{MS}_{n,k}}{\#\left(\mathcal{MS}_{n,k}\right)} \to_{k\to\infty} 1$$

□

Remarque 10. Comme dans le cas captif, le résultat peut être étendu légèrement, c'est-à-dire à des chemins pour lesquels la taille du voisinage croit lentement devant la croissance de l'ensemble d'états.

Les ensemblistes

On considère maintenant une restriction encore plus drastique de la règle : en plus de la position, celle-ci n'exploite dorénavant même plus le nombre d'occurrences de chaque état présent dans le voisinage.

Un automate cellulaire est donc dit ensembliste lorsque l'image, par la règle de transition, d'un état ne dépend que de l'ensemble des états présents dans le motif local. L'automate ne tient pas compte pour calculer le nouvel état ni de quelles cellules ont fourni un état donné, ni même du nombre de cellules ayant fourni les différents états ; il tient finalement juste compte de la présence, ou non d'un état dans le voisinage. Formellement, un automate $\mathcal{A}$ est ensembliste si sa règle locale vérifie : $\forall u, v \in \Sigma_{\mathcal{A}}^{k_{\mathcal{A}}}$, si $\{u\} = \{v\}$ alors $\delta_{\mathcal{A}}(u) = \delta_{\mathcal{A}}(v)$.

Ceci parait très restrictif, et pourtant :

Lemme 3.42. *Il existe un automate cellulaire ensembliste, et intrinsèquement universel.*

Preuve : En réalité les automates multi-ensemblistes et intrinsèquement universels construits dans la preuve du lemme 3.40 sont également ensemblistes.

□

Mais on montre tout de même que :

Proposition 3.43. *Selon tout chemin ρ tel que le nombre d'états ρ_n reste borné par une constante k_0, parmi les AC ensembliste, la densité des universels, si elle existe, est strictement inférieure à 1.*

Exactement comme dans le cas de la proposition 3.38, ceci repose sur la finitude des ensembles considérés : le nombre de règles ensemblistes pour chaque taille est bornée par une constante globale. Ceci se montre très simplement :

Preuve : Pour une taille (n, k) donnée, le nombre de règles ensemblistes est très faible. En effet, dès lors que $k \geq n$, le nombre d'ensembles à au plus k états choisis parmi n est donné par 2^n.

Par conséquent pour une taille n plus petite que n_0, le nombre d'AC de taille (n, k) différents est bornée par la constante ${n_0}^{2^{n_0}}$.

Comme on sait que certains automates ne sont pas universels, comme les automates nilpotents, ou encore plus simplement les AC dont la règle envoie tout les motifs vers le même état, on peut conclure directement.

□

Les totalistiques

Rappelons qu'un AC est dit totalistique si la règle de transition ne dépend que de la somme des états présents dans le voisinage. Formellement, $\mathcal{A}$ est dit totalistique s'il existe $\phi : \mathbb{N} \to \Sigma_{\mathcal{A}}$ tel que $\delta_{\mathcal{A}}(x_1, \ldots, x_{k_{\mathcal{A}}}) = \phi(x_1 + \cdots + x_{k_{\mathcal{A}}})$.

On montre alors un résultat du même type que les précédents :

Proposition 3.44. *Selon tout chemin à nombre d'états constant, parmi les AC totalistiques, la densité des universels est 1.*

Et l'on s'appuie de nouveau sur les deux lemmes d'existence et de loi 0-1.

Lemme 3.45. *Il existe un automate cellulaire totalistique et intrinsèquement universel.*

Ce résultat est bien connu, on l'a évoqué au 4.3. On le montre ici en se ramenant simplement à un AC multi-ensembliste universel.

Preuve : Pour construire un automate totalistique qui simule un automate multi-ensembliste donné, il suffit de définir un totalistique tel que l'écart entre les états utilisés pour la simulation suffise à reconstituer le motif multi-ensembliste.

Concrètement, pour simuler un multi-ensembliste de taille (n_0, k_0), il suffit de construire un totalistique de même voisinage et avec un ensemble d'états de taille n^{k_0}. On simule alors le comportement de l'automate en utilisant les états $1, n, n^2, ...n^{k_0-1}$.

Sur les sous-shifts constitués uniquement de tels états, la valeur associée à un mot de taille k_0 permet de reconstituer l'unique multi-ensemble qui a cette valeur. On peut donc définir les images des transitions totalistiques issues de tels motifs à partir de la règle de transition multi-ensembliste de l'universel précédent.

À travers l'injection état par état $i : x \mapsto n^x$, l'automate totalistique simule le multi-ensembliste de départ et est donc universel si celui-ci l'est.

□

Remarque 11. Avec l'universalité intrinsèque, le fait de choisir ou non un ensemble d'états connexes n'a aucune importance dans la mesure où la simulation se fait toujours sur un sous-shift restreint particulier. Il suffit de ne pas utiliser les états rajoutés pour obtenir la connexité.

Lemme 3.46. *Sur l'ensemble des automates cellulaires totalistiques, les propriétés croissantes ont densité* 1 *selon les chemins à nombre d'états constant.*

Preuve : On va adapter la preuve du cas multi-ensembliste et ne donner que les idées importantes.

On considère un automate cellulaire $\mathcal{A}_0$ de taille (n_0, k_0), et l'on se demande combien d'automates de taille n_0, k vont simuler $\mathcal{A}_0$.

Pour cela, on considère à nouveau l'entier $l = \lfloor \frac{k-k_0}{k_0-1} \rfloor$, et pour tout entier $k_0 \leq i \leq l-k_0+-1$, v_i, le mot $v_i = 0^i \cdot v^{l-i}$, dans lequel v est l'état de valeur $k_0 n_0$, et 0 celui de valeur 0.

Et pour chaque i considéré ($k_0 \leq i \leq l-k_0+-1$), le sous-shift S_i est formé des configurations alternant v_i et un état d'une valeur prise parmi celles de $\Sigma_{\mathcal{A}_0}$. Par exemple la configuration suivante, avec $k = 28$, $l = 12$, $i = 7$ et $x, y, z, t \in \Sigma_{\mathcal{A}_0}$.

$$\cdots 00vvvvv\ x\ \underbrace{0000000vvvvv}_{v_7}\ y\ \underbrace{0000000vvvvv}_{v_7}\ z\ \underbrace{0000000vvvvv}_{v_7}\ t\ 000000v\cdots$$

La simulation intervient si l'AC satisfait les mêmes motifs de transition que pour le cas multi-ensembliste (en remplaçant les 1 par des v).

Le point crucial est alors que les motifs de longueur k qui interviennent pour chaque S_i sont distinguables. Un automate avec $n_0.k_0$ états et un voisinage k simule l'automate $\mathcal{A}_0$ sur S_i s'il vérifie un ensemble de ${n_k}^{k_0}.(k_0+1)$ transitions, qui est constant lorsque k augmente.

Comme le nombre de tels sous-shifts augmente avec k, leur indépendance fait que la proportion d'AC de taille $(n_0 k_0, k)$ qui simule $\mathcal{A}_0$ tend vers 1.

□

Automates avec un état persistant

Rappelons que cette famille est constituée des AC qui possèdent un état persistant, c'est-à-dire tel qu'une cellule dans cet état n'en changera pas sous l'action de l'automate.

Proposition 3.47. *Selon tout chemin à nombre d'états constant, parmi les automates cellulaires ayant un état persistant, l'universalité à densité* 1.

Lemme 3.48. *Il existe un automate cellulaire universel avec un état persistant.*

Preuve : Par exemple l'automate universel pour la simulation facteur construit au chapitre 5 est aussi intrinsèquement universel.

Sinon, considérons un automate cellulaire universel quelconque $\mathcal{U} \in \mathbb{CA}_{n,k}$. Soit $\mathcal{A}$ l'automate de même voisinage, d'ensemble d'états $\Sigma_\mathcal{A} = \Sigma_\mathcal{U} \cup \{p\}$ et dont la règle locale est obtenue en étendant celle de $\mathcal{U}$ de la manière suivante : si $u \in \Sigma_\mathcal{U}^{k_\mathcal{U}}$, $\delta_\mathcal{A}(u) = \delta_\mathcal{U}(u)$, sinon $\delta_\mathcal{A}(u) = p$.

Et bien cet automate $\mathcal{A}$ a un état persistant (et même envahissant) p, et il est universel car il simule $\mathcal{U}$ via sa restriction sur $\Sigma_\mathcal{U}$.

□

Lemme 3.49. *Selon tout chemin à nombre d'états constant, parmi l'ensemble des AC ayant un état persistant, les propriétés croissantes ont une densité* 1.

Preuve : On va commencer par montrer cette propriété sur l'ensemble des automates ayant l'état 0 comme état persistant. Par symétrie des états, on en déduira le lemme.

Considérons un automate $\mathcal{A}_0 \in \mathcal{P}_{n_0,k_0}$. On cherche à dénombrer les automates de taille n_0, k, qui le simulent.

Pour cela pour chaque $b \in \mathbb{N}$, $\lfloor \frac{k-k_0}{k_0} \rfloor \leq b \leq \lfloor \frac{k-k_0}{k_0-1} \rfloor$, on considère la fonction $i_b : \Sigma_{\mathcal{A}_0} \to \Sigma_\mathcal{A}^b$ définie par $\forall x \in \Sigma_{\mathcal{A}_0}$, $i_b(x) = x \cdot 0^{b-1}$, où 0 est l'état persistant.

Soit $\mathcal{A}$ de taille (n_0, k) avec un état persistant 0. Pour que $\mathcal{A}$ simule $\mathcal{A}_0$ à travers i_b, il suffit que $\delta_\mathcal{A}$ vérifie $\forall x_1, x_2, ..., x_{k_0} \in \Sigma_{\mathcal{A}_0}$, $\delta_\mathcal{A}(x_1 0^{b-1} x_2 \cdots 0^{b-1} x_{k_0} 0^{k-b.k_0}) = \delta_{\mathcal{A}_0}(x_1 x_2 \cdots x_{k_0})$. En effet, comme 0 est persistant, cela suffit à assurer que i_b sera compatible avec les règles des automates. Ceci revient à fixer ${n_0}^{k_0}$ transitions différentes d'une manière unique. Donc pour chaque b, la probabilité que $\mathcal{A} \preccurlyeq_{\sqsubset_{i_b}} \mathcal{A}_0$ est plus grande qu'une constante $(1/{n_0}^{{n_0}^{k_0}})$.

Or le nombre de valeurs possibles pour b croit avec le voisinage de l'automate et les ensembles de transitions considérés pour chaque b sont disjoints, ceci permet d'utiliser le lemme 3.33 pour conclure.

□

Outer-multiset et autres

Il est également possible de considérer des variantes de ces familles, à base de symétries partielles. Le cas le plus immédiat est celui des *outer-multiset*, définis au chapitre 2 et obtenus en considérant les règles dépendant uniquement du multi-ensemble des états présents dans le voisinage ainsi que de manière arbitraire d'une partie plus restreinte de leur voisinage.

Intuitivement, ces automates tiennent compte de la position des états sur une petite sous-partie de leur voisinage.

Ce cas est intéressant car il contient des représentants très célèbre comme le jeu de la vie (qui est en 2 dimensions).

On obtient des résultats similaires à ceux obtenus pour les familles à symétrie totale, à condition de limiter la zone de dépendance à une partie très restreinte du voisinage (par exemple pour une zone centrale en $log(log(k))$ avec k la taille du voisinage.

Ceci se montre avec les même constructions que précédemment, mais en introduisant un décompte différent des transitions à fixer.

3.4 Cas les plus généraux

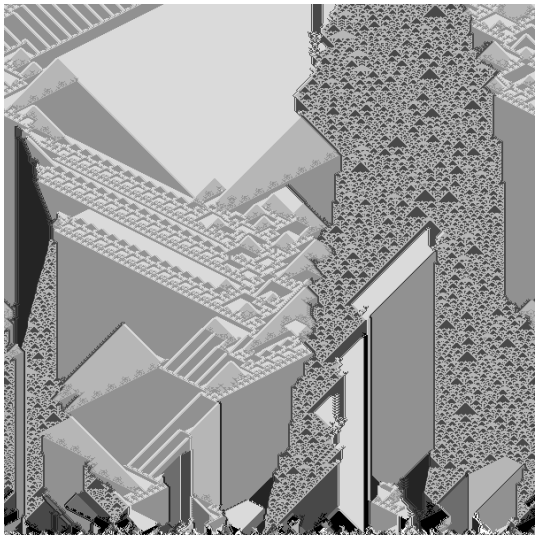
Jusque là on a considéré des familles pour lesquelles le plus souvent, l'universalité ne devenait majoritaire que lorsque seulement un des deux paramètres de taille variait. Intuitivement, on

explorait l'ensemble des tailles *horizontalement* ou *verticalement*.

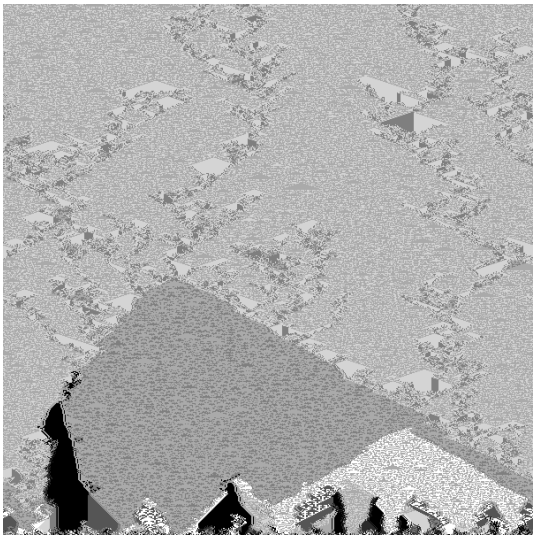
On va maintenant introduire les quelques classes pour lesquelles on connait parfaitement la densité de l'universalité intrinsèque selon n'importe quel chemin.

Les multi-ensemblistes captifs

Un automate est dit multi-ensembliste captif s'il est multi-ensembliste et captif. C'est-à-dire que sa règle de transition ne dépend que du multi-ensemble des états présents dans le voisinage, et que le nouvel état est choisi parmi les états présents dans ce multi-ensemble. Les diagrammes espace-temps 3.3 représentent de tels automates choisis aléatoirement.



(a) Voisinage de taille 5, ensemble d'états de taille 8



(b) Voisinage de taille 7, ensemble d'états de taille 7

FIGURE 3.3: Diagrammes espace-temps d'AC multi-ensemblistes captifs tirés aléatoirement à partir d'un segment initial aléatoire.

On montre le résultat suivant

Proposition 3.50. *Selon tout chemin, parmi les AC multi-ensemblistes captifs, l'ensemble des AC non universels est négligeable.*

Rappelons qu'une formulation précise de ce résultat est :
il existe une taille (n_0, k_0) telle que sur tout (n_0, k_0)-chemin, parmi les AC multi-ensemblistes captifs, l'ensemble des AC non universels est négligeable.

À l'aide d'un premier lemme :

Lemme 3.51. *Il existe un automate cellulaire multi-ensembliste captif intrinsèquement universel.*

Preuve : Pour cela, on associe à chaque AC, un AC multi-ensembliste et captif qui le simule.

On reprend l'idée de la construction multi-ensembliste (considérer un produit cartésien dont la seconde composante sert de marqueur positionnel) et pour garantir le caractère captif, on va ajouter entre chaque état des configurations de la construction précédente une *bibliothèque*, de manière à ce que tous les états soient visibles dans le voisinage d'une configuration.

Soit $\mathcal{A}$ un automate cellulaire d'ensemble d'états $\Sigma_\mathcal{A}$ de taille $n_\mathcal{A}$ et de voisinage de taille $k_\mathcal{A}$. Nous décrivons un automate $\Psi(\mathcal{A})$, multi-ensembliste et captif tel que $\Psi(\mathcal{A})$ simule $\mathcal{A}$.

L'ensemble d'états de $\Psi(\mathcal{A})$ est donné par $\Sigma_{\Psi(\mathcal{A})} = \Sigma_\mathcal{A} \times \{0, \dots, k_\mathcal{A}+1\}$ et son voisinage est centré, de taille $k_{\Psi(\mathcal{A})} = n_\mathcal{A}(k_\mathcal{A}-1)+k_\mathcal{A}$.

Pour $0 \leq i \leq k_\mathcal{A}+1$, on définit la i-ème bibliothèque, L_i comme le mot de longueur $n_\mathcal{A}$ qui contient successivement, dans l'ordre lexicographique tous les état de $\Sigma_{\Psi(\mathcal{A})}$ dont la seconde composante est i : en notant $\Sigma_\mathcal{A} = \{0, 1, \cdots, n_\mathcal{A}-1\}$, $L_i = (0,i)\cdot(1,i)\cdots(n_\mathcal{A}-1,i)$.

La règle de transition de $\Psi(\mathcal{A})$ va simuler le comportement de $\mathcal{A}$ sur les configurations alternant une bibliothèque L_i et un état, dit *utile* de type i, puis une bibliothèque du type suivant $i+1 \bmod k_\mathcal{A}+2$, et un état de type $i+1 \bmod k_\mathcal{A}+2$, etc...

Par exemple pour $n_\mathcal{A} = 4$ et $k_\mathcal{A} = 3$, un extrait d'une telle configuration ressemble à la figure suivante (en superposant les deux couches du produit cartésien, avec $x, y, z, t \in \Sigma_\mathcal{A}$). Le motif souligné est un motif de voisinage de $\Psi(\mathcal{A})$, et le motif encadré correspond à L_4 :

$$\cdots 123x0123y0123z0123x0123t0123y\boxed{0123}y0123x012\cdots$$
$$\cdots 333344444000001111122222333334444400000111\cdots$$

Notons $B_{i,b}$ le début de la bibliothèque L_i contenant b éléments et $E_{i,e}$ la fin de L_i contenant $n_\mathcal{A}-e$ éléments. Plus précisément,

$B_{i,b} = (0,i)\cdot(1,i)\cdots(b-1,i)$

$E_{i,e} = (e,i)\cdot(e+1,i)\cdots(n_\mathcal{A}-1,i)$.

Les multi-ensembles des mots de longueur $n_\mathcal{A}\cdot(k_\mathcal{A}-1)+k_\mathcal{A}$ qui interviennent dans une telle configuration peuvent être vu comme étant de deux types :

1) ceux centrés sur un état *utile*, qui voient un multi-ensemble contenant $k_\mathcal{A}-1$ bibliothèques complètes d'indices consécutifs, c'est-à-dire pour un certain j ($0 \leq j \leq k_\mathcal{A}+1$), $L_{j+1 \bmod k_\mathcal{A}+2}, \dots, L_{j+k_\mathcal{A}-1 \bmod k_\mathcal{A}+2}$ ainsi que $k_\mathcal{A}$ états de types $j, \dots, j+k_\mathcal{A}-1 \bmod k_\mathcal{A}+2$. Pour plus de commodités, on note ces multi-ensembles $C(j, x_j, x_{j+1}, \cdots, x_{j+k_\mathcal{A}-1})$ (on omet les modulos)avec pour chaque i, x_i la composante dans $\Sigma_\mathcal{A}$ de l'état de type i.
2) ceux centrés sur un état de bibliothèque, qui voient un multi-ensemble contenant : $k_\mathcal{A}-1$ états de types consécutifs j, $j+1 \bmod k_\mathcal{A}+2$,..., $j+k_\mathcal{A}-2 \bmod k_\mathcal{A}+2$, ainsi que la fin de la bibliothèque d'indice j c'est-à-dire $E_{j,e}$, le début de celle d'indice $j+k_\mathcal{A}-2 \bmod k_\mathcal{A}+2$ c'est-à-dire $B_{j+k_\mathcal{A}-2 \bmod k_\mathcal{A}+2,e+1}$, ainsi que $k_\mathcal{A}-2$ bibliothèques complètes d'indices $j+1 \bmod k_\mathcal{A}+2$,...,$j+k_\mathcal{A}-3 \bmod k_\mathcal{A}+2$.
On note de même ces multi-ensembles $D(j, e, x_{j+1}, \cdots, x_{j+k_\mathcal{A}-2})$ avec x_i dans $\Sigma_\mathcal{A}$ pour chaque i.

Ces motifs de voisinage caractéristiques de ces transitions sont distinguables, les multi-ensembles utilisés étant tous disjoints.

On décrit le comportement de $\Psi(\mathcal{A})$ sur ces multi-ensembles. Sur ceux du premier type, on applique la règle de transition de $\mathcal{A}$, en reconstituant le motif de transition :

$$\delta_{\Psi(\mathcal{A})}(C(j, x_j, x_{j+1}, \cdots, x_{j+k_\mathcal{A}-1})) = (\delta_\mathcal{A}(x_j, x_{j+1}, \cdots, x_{j+k_\mathcal{A}-1}), j+(k_\mathcal{A}-1)/2)\,.$$

Sur les transitions du second type, on maintient les bibliothèques en place, la valeur de e permettant de maintenir le positionnement correct :

$$\delta_{\Psi(\mathcal{A})}(D(j, e, x_{j+1}, \cdots, x_{j+k_\mathcal{A}-2})) = (e, j+(k_\mathcal{A}+1)/2)\,.$$

Toutes ces transitions sont bien multi-ensemblistes et captives. Et l'on peut compléter la règle de transition de $\Psi(\mathcal{A})$ de n'importe quelle manière. Cela ne change rien à la simulation.

On peut maintenant vérifier que $\Psi(\mathcal{A})$ simule bien $\mathcal{A}$ à travers la fonction injective

$$i : \Sigma_{\Psi(\mathcal{A})}^{(k_\mathcal{A}+2)\cdot(n_\mathcal{A}+1)} \to \Sigma_\mathcal{A}^{k_\mathcal{A}+2}.$$

C'est une conséquence directe de la construction. Plus précisément, on a :

$$\mathcal{A}^{<k_\mathcal{A}+2,1,0>} \sqsubseteq_i \Psi(\mathcal{A})^{<(k_\mathcal{A}+2)(n_\mathcal{A}+1),1,0>}$$

En particulier, il existe des automates cellulaires intrinsèquement universels multi-ensemblistes et captifs, c'est le cas des images par notre transformation des automates cellulaires intrinsèquement universels quelconques.

□

Il reste maintenant à montrer le lemme de densité suivant.

Lemme 3.52. *Selon tout chemin, parmi l'ensemble des AC multi-ensemblistes captifs, les propriétés croissantes ont densité* 1.

Preuve : Pour cela on reprend la construction introduite pour prouver le lemme de densité 3.41 qui concerne les multi-ensemblistes.

On considère maintenant un automate cellulaire $\mathcal{A}_0 \in \mathcal{MSC}_{n_0,k_0}$ et on s'intéresse à l'ensemble $\mathcal{S}_{\mathcal{A}_0} \cap \mathcal{MSC}_{n,k}$.

À nouveau l est l'entier $l = \lfloor \frac{k-k_0}{k_0-1} \rfloor$, et pour tout i, $k_0 \leq i \leq l-k_0$, v_i est le mot $v_i = 0^i \cdot 1^{l-i}$. On note encore $o = k_\mathcal{A} - k_0 - (k_0-1)n_0$ le *décalage*, et on a toujours $0 \leq o < k_0 - 1$.

On introduit de plus $m = \lfloor \frac{n-2}{n_0} \rfloor$, et h tel que $0 \leq h \leq m$. L'alphabet $\mathcal{Q}_h \subseteq \Sigma_n$ à n_0 éléments consécutifs tel que :

$$\mathcal{Q}_h = \{hn_0 + 2, hn_0 + 3, ...(h+1)n_0 + 1\}.$$

Et pour chaque i et h considéré ($k_0 \leq i \leq l - k_0$ et $0 \leq h \leq m$), on définit le sous-shift $S_{i,m}$ comme l'ensemble des configurations alternant v_i et un état de $\mathcal{Q}_h$.

On peut maintenant reprendre le raisonnement du cas multi-ensembliste, portant sur les S_i dans sa quasi-intégralité. On obtient alors au plus ${n_0}^{k_0}(k_0+1)$ transitions à fixer d'une manière unique pour qu'un automate de taille (n,k) simule $\mathcal{A}_0$ sur un sous-shift $S_{i,h}$ donné. Elles correspondent aux motifs de types C, $D1$, et $D2$ définis dans le cas multi-ensembliste.

Cependant grâce à la captivité, fixer une des transitions de chacun de ces types est nettement moins couteux : l'état image est en effet choisi parmi les au plus $k_0 + 2$ états présents dans le voisinage.

Par conséquent, la proportion d'automates multi-ensemblistes captifs de taille (n,k) ($n \geq n_0+2$ et $k \geq k_0$) pour lesquels les ${n_0}^{k_0}(k_0+1)$ transitions sont fixées de manière à ce qu'ils simulent $\mathcal{A}_0$ sur $S_{i,m}$ vérifie :

$$\frac{\#\left(\mathcal{S}_{\mathcal{A}_0,i,m} \cap \mathcal{MSC}_{n,k}\right)}{\#\left(\mathcal{MSC}_{n,k}\right)} \geq \frac{1}{(k_0+2)^{{n_0}^{k_0}(k_0+1)}}$$

Mais les $S_{i,h}$ sont indépendants. Par conséquent s'après le lemme 3.33, et comme i a $l - 2k_0$ valeurs possibles, et h en a m, la densité d'automates simulant $\mathcal{A}_0$ dans $\mathcal{MSC}_{n,k}$ vérifie :

$$\frac{\#\left(\mathcal{S}_{\mathcal{A}_0}\right) \cap \mathcal{MSC}_{n_0+2,k}}{\#\left(\mathcal{MSC}_{n_0+2,k}\right)} \geq 1 - \left(1 - \frac{1}{(k_0+2)^{{n_0}^{k_0}(k_0+1)}}\right)^{(l-2k_0)m}$$

Or l croit vers l'infini linéairement en k, et m linéairement en n. Or, sur n'importe quel chemin ρ, la fonction $m_\rho : x \mapsto \min(\rho_n(x), \rho_k(x))$ croit vers l'infini. C'est donc aussi le cas pour $\min(l, m)$ et donc pour $(l - 2k_0)m$, on a :

$$\lim_{x\to\infty} \frac{\#\left(\mathcal{S}_{\mathcal{A}_0}\right) \cap \mathcal{MSC}_{\rho(x)}}{\#\left(\mathcal{MSC}_{\rho(x)}\right)} \to 1$$

ce qui conclut la preuve du lemme.

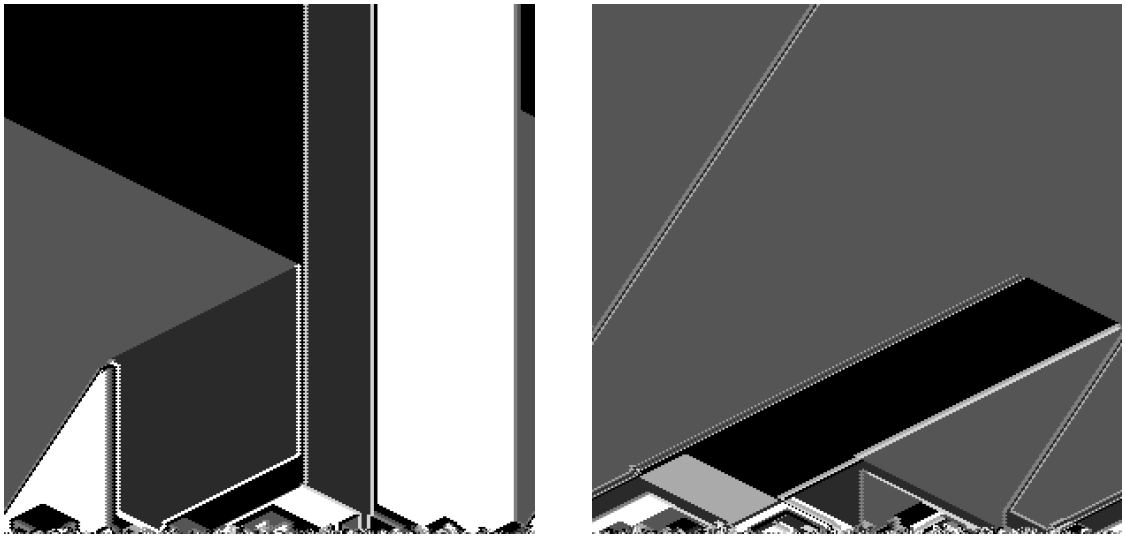
□

Le résultat précédent est donc particulièrement fort car il est valable pour tout chemin.

On traite maintenant le cas d'une famille où les densités sont assez bien connues, mais ne sont pas les mêmes selon les différentes chemins.

Les ensemblistes captifs

Un AC est dit ensembliste captif s'il est ensembliste et captif. Rappelons que cela signifie que la règle de transition ne tient compte que du fait que les états sont présents ou non dans le voisinage (sans tenir compte ni des positions ni de la multiplicité) et que le nouvel état est forcément choisi parmi les états présents dans le voisinage. La figure 3.4 montrent des diagrammes de tels automates choisis aléatoirement.



(a) Voisinage de taille 5, ensemble d'états de taille 7

(b) Voisinage de taille 5, ensemble d'états de taille 7

FIGURE 3.4: Diagrammes espace-temps d'AC ensemblistes captifs tirés aléatoirement à partir d'un segment initial aléatoire.

Nous montrons d'abord les deux lemmes usuels, avant de formuler le résultat plus général :

Lemme 3.53. *Il existe un automate cellulaire ensembliste captif intrinsèquement universel.*

Preuve : On va légèrement modifier la fonction Ψ introduite dans le cas multi-ensembliste captif pour en faire une transformation ensembliste (et toujours captive).

On reprend donc les notations introduites dans la preuve du lemme 3.51. Le problème est que la fonction Ψ, dans l'état actuel n'est pas ensembliste, en effet, les états utiles sont *masqués* par les bibliothèques.

L'idée pour résoudre ce problème est de conserver l'alternance entre bibliothèques et états utiles, mais d'ajouter un décalage entre les types des bibliothèques, et ceux des états utiles de manière à distinguer ces derniers des états des bibliothèques. Plus précisément, le nombre de types différents est maintenant $2k_{\mathcal{A}} - 1$, et la bibliothèque L_i sera suivie d'un état de type $i + k_{\mathcal{A}} \bmod 2k_{\mathcal{A}} - 1$.

De plus pour maintenir ce décalage des types, on ajoute une étape intermédiaire, de décalage des bibliothèques, à chaque pas de la simulation. Celle-ci ne se fait donc plus en temps réel mais à la moitié de celui-ci.

Nous avons aussi besoin de modifier légèrement la définition des bibliothèques de sorte que l'ensemble d'états est maintenant donné par $\Sigma_{\Psi(\mathcal{A})} = (\Sigma_{\mathcal{A}} \cup \{\#\}) \times \{0, ..., 2k_{\mathcal{A}} - 2\}$, la taille du voisinage de $\Psi(\mathcal{A})$ est toujours $k_{\Psi(\mathcal{A})} = n_{\mathcal{A}}(k_{\mathcal{A}} - 1) + k_{\mathcal{A}}$, et la bibliothèque L_i est donnée par :

$$L_i = (\#, i) \cdot (0, i) \cdot (1, i) \cdots (n_{\mathcal{A}} - 1, i)(\#, i)$$

On a alors
$B_{i,b} = (\#, i) \cdot (0, i) \cdot (1, i) \cdots (b - 1, i)$
$E_{i,e} = (e, i) \cdot (e + 1, i) \cdots (n_{\mathcal{A}} - 1, i) \cdot (\#, i)$.

Et la simulation se passe sur des configurations alternant une bibliothèque L_i complète et un état de type $i + k_{\mathcal{A}} \bmod 2k_{\mathcal{A}} - 1$. Par exemple pour $n_{\mathcal{A}} = 4$ et $k_{\mathcal{A}} = 3$, un extrait d'une telle configuration ressemble à la figure suivante (en superposant les deux couches du produit cartésien, avec $x, y, z, t \in \Sigma_{\mathcal{A}}$). Le motif souligné est un motif de voisinage de $\Psi(\mathcal{A})$, et le motif encadré correspond à L_4 :

$$\cdots 123x0123y0123z0123x0123t0123y\boxed{0123y}0123x012\cdots$$
$$\cdots 3331444420000\underline{3111142222033331}\boxed{4444}200003111\cdots$$

Lors du calcul de l'image de cette configuration, la transition du simulé est appliquée mais les états utiles de type i deviennent des états utiles de type $i + k_{\mathcal{A}} - \lfloor \frac{k_{\mathcal{A}}}{2} \rfloor \bmod 2k_{\mathcal{A}} - 1$, et les bibliothèques de type i prennent le type $i + \lceil \frac{k_{\mathcal{A}}}{2} \rceil - 1 \bmod 2k_{\mathcal{A}} - 1$. Pour fixer les idées, l'image de la configuration précédente est donnée par :

$$\cdots 123t0123y0123t0123z0123x0123t0123t0123y012\cdots$$
$$\cdots 4443000041111022221333324444300004111152 22\cdots$$

Il faudrait alors utiliser l'étape supplémentaire pour remettre le rapport initial entre les types des bibliothèques et ceux des états utiles. Les états utiles ne changent pas d'état, mais L_i devient $L_{i+\lfloor \frac{k_{\mathcal{A}}}{2} \rfloor}$.

$$\cdots 123t0123y0123t0123z0123x0123t0123t0123y012\cdots$$
$$\cdots 000311114222203333144442000031111422220333\cdots$$

Une fois ce mécanisme décrit, il sera possible de montrer que $\Psi(\mathcal{A})$ simule $\mathcal{A}$. Cependant, il faut d'abord s'assurer que ce mécanisme est effectivement ensembliste et captif. Pour cela, nous allons d'abord décrire les ensembles des motifs de longueurs $k_{\Psi(\mathcal{A})}$ qui interviennent dans les configurations que nous avons décrites, nous assurer qu'ils sont tous distinguables, puis spécifier leur images (captives) par $\delta_{\Psi(\mathcal{A})}$.

Les motifs intervenant dans la première étape de la simulation sont de deux types :

- Celles contenant exactement $k_{\mathcal{A}}-1$ bibliothèques de types consécutifs $i+k_{\mathcal{A}}$, $i+k_{\mathcal{A}}+1 \bmod 2k_{\mathcal{A}}-1$,...,$i+2k_{\mathcal{A}}-2 \bmod 2k_{\mathcal{A}}-1$, ainsi que $k_{\mathcal{A}}$ états isolés de types consécutifs : x_i,$x_{i+1 \bmod 2k_{\mathcal{A}}-1}$,..., $x_{i+k_{\mathcal{A}} \bmod 2k_{\mathcal{A}}-1}$, l'indice correspondant au type de chaque état.
  On les notera $C1(i, x_i, x_{i+1}, \ldots, x_{i+k_{\mathcal{A}}})$ en omettant les modulos.
- Celles contenant une fin de bibliothèque, $E(i+k_{\mathcal{A}}+1 \bmod 2k_{\mathcal{A}}-1, e)$, un début de bibliothèque $B(i+2 \bmod 2k_{\mathcal{A}}-1, e)$, ainsi que $k_{\mathcal{A}}-2$ bibliothèques de types consécutifs $i+k_{\mathcal{A}}+2 \bmod 2k_{\mathcal{A}}-1$,...,$i+1 \bmod 2k_{\mathcal{A}}-1$, et $k_{\mathcal{A}}-1$ états isolés de types consécutifs : x_i,$x_{i+1 \bmod 2k_{\mathcal{A}}-1}$,..., $x_{i+k_{\mathcal{A}}-2 \bmod 2k_{\mathcal{A}}-1}$.
  On les notera $D1(i, e, x_i, ..., x_{i+k_{\mathcal{A}}-1})$.

Et les motifs intervenant dans la seconde sont :

- Celles contenant exactement $k_{\mathcal{A}}-1$ bibliothèques de types consécutifs $i-\lfloor\frac{k_{\mathcal{A}}}{2}\rfloor \bmod 2k_{\mathcal{A}}-1$,...,$i-\lfloor\frac{k_{\mathcal{A}}}{2}\rfloor+k_{\mathcal{A}}-2 \bmod 2k_{\mathcal{A}}-1$, ainsi que $k_{\mathcal{A}}$ états isolés de types consécutifs : x_i,$x_{i+1 \bmod 2k_{\mathcal{A}}-1}$,..., $x_{i+k_{\mathcal{A}} \bmod 2k_{\mathcal{A}}-1}$.
  On les notera $C2(i, x_i, x_{i+1}, \ldots, x_{i+k_{\mathcal{A}}})$ en omettant les modulos.
- Celles contenant une fin de bibliothèque, $E(i-\lfloor\frac{k_{\mathcal{A}}}{2}\rfloor-1 \bmod 2k_{\mathcal{A}}-1, e)$, un début de bibliothèque $B(i-\lfloor\frac{k_{\mathcal{A}}}{2}\rfloor+k_{\mathcal{A}}-2 \bmod 2k_{\mathcal{A}}-1, e)$, ainsi que $k_{\mathcal{A}}-2$ bibliothèques de types consécutifs $i-\lfloor\frac{k_{\mathcal{A}}}{2}\rfloor \bmod 2k_{\mathcal{A}}-1$,...,$i-\lfloor\frac{k_{\mathcal{A}}}{2}\rfloor+k_{\mathcal{A}}-3 \bmod 2k_{\mathcal{A}}-1$, et $k_{\mathcal{A}}-1$ états isolés de types consécutifs : x_i,$x_{i+1 \bmod 2k_{\mathcal{A}}-1}$,..., $x_{i+k_{\mathcal{A}}-1 \bmod 2k_{\mathcal{A}}-1}$.
  On les notera $D2(i, e, x_i, ..., x_{i+k_{\mathcal{A}}-2})$.

Ces motifs sont bien distinguables au sens ensembliste du terme : les ensembles qu'ils définissent sont tous distincts, en effet on peut d'une part distinguer si les bibliothèques sont complètes grâce à la présence ou non de $\#$, et l'on peut d'autre part distinguer les ensembles de la première étape de ceux de la seconde car ces derniers sont plus courts. En effet, un des états est masqué par une bibliothèque.

On spécifie donc pour $\delta_{\Psi(\mathcal{A})}$ les transitions suivantes, qui correspondent au comportement souhaité sont donc ensemblistes (et captives) :

- $C1(i, x_i, x_{i+1}, \ldots, x_{i+k_{\mathcal{A}}}) \to (\delta_{\mathcal{A}}(x_i, x_{i+1}, \ldots, x_{i+k_{\mathcal{A}}}), i+1 \bmod 2k_{\mathcal{A}}-1)$
- $D1(i, e, x_i, ..., x_{i+k_{\mathcal{A}}-2}) \to (e, i+k_{\mathcal{A}}-1 \bmod 2k_{\mathcal{A}}-1$
- $C2(i, x_i, x_{i+1}, \ldots, x_{i+k_{\mathcal{A}}}) \to (x_{i+\lfloor\frac{k_{\mathcal{A}}}{2}\rfloor+1 \bmod 2k_{\mathcal{A}}-1}, i+\lfloor\frac{k_{\mathcal{A}}}{2}\rfloor+1 \bmod 2k_{\mathcal{A}}-1$
- $D2(i, e, x_i, ..., x_{i+k_{\mathcal{A}}-2}) \to (x, i+\lfloor\frac{k_{\mathcal{A}}}{2}\rfloor \bmod 2k_{\mathcal{A}}-1$

Le comportement est captif. Et même le troisième type de transitions est ensembliste dans la mesure où la détermination de x est unique.

Le comportement correspond bien à l'application aux étapes impaires de la règle de $\mathcal{A}$ et après les étapes paires, les types sont corrects.

Et il suffit de compléter la règle de manière à ce qu'elle soit ensembliste captive, par exemple en ordonnant les états de $\Sigma_{\Psi(\mathcal{A})}$ et en fixant un comportement *max*, pour obtenir un AC ensembliste et captif bien défini.

On vérifie facilement que :

$$\mathcal{A}^{<2k_{\mathcal{A}}-1,1,0>} \preccurlyeq_{\sqsubseteq_i} \Psi(\mathcal{A})^{<(n_{\mathcal{A}}+2)(2k_{\mathcal{A}}-1),2,\epsilon>}$$

□

Lemme 3.54. *Pour tout chemin ρ tel que ρ_n a une limite infinie, parmi les AC ensemblistes-captifs, toute propriété croissante a densité 1 sur ρ.*

Preuve : Soit $\mathcal{A}_0 \in \mathcal{ESC}_{n_0,k_0}$, un automate donné, on s'intéresse à $\mathcal{S}_{\mathcal{A}_0} \cap \mathcal{ESC}_{n,k}$, pour $n \geq n_0$ et $k \geq k_0$.

Selon le schéma maintenant familier, on va définir des sous-shifts sur lesquels la simulation a lieu si la règle locale satisfait certaines propriétés.

On suppose $n \geq 2k_0 + 4 + n_0$. Soit 0_i et 1_i pour $i \in \{0..k_0 + 1\}$, les $2k_0 + 4$ premiers états de Σ_n. 0_i est le $2i$-ème état et 1_i est le $2i + 1$-ème état de Σ_n. Soit $\mathcal{Q}_j \subseteq \Sigma_n$ l'alphabet de taille n_0 contenant les $2k_0 + 4 + jn_0$-ème,... $2k_0 + 4 + jn_0 + n_0 - 1$-ième éléments de Σ_n.

Soit maintenant $l = \lfloor \frac{k-k_0}{k_0-1} \rfloor$, et $o = k - k_0$, et $v_i \in \Sigma_n^l$ défini par $v_i = 0_i^o \cdot 1_i^{l-o}$. Le sous-shift S_j est celui alternant des occurrence successives et ordonnées (circulairement, c'est-à-dire avec des modulors) selon i de v_i, avec des éléments de $\mathcal{Q}_j$.

Pour $k_0 = 3$, $k =$... Ceci donne des configurations de la forme :

Les ensembles des mots de longueur k qui apparaissent dans ces sous-shifts sont de trois types :

- ceux centrés sur un état, qui contiennent k_0 états de $\mathcal{Q}_j$, ainsi que k_0+1 types de 0_i différents et k_0 types de 1_i différents.
  On voit donc l'ensemble $\{a_1, ...a_{k_0}\} \cup \{0_i, 1_i, ..., 1_{i+k_0-1 \bmod 2k_0+4}, 0_{i+k_0 \bmod 2k_0+4}\}$.
  On note cet ensemble $C(i, \{a_1, ...a_{k_0}\})$.
- ceux décalés *vers la gauche*, contiennent k_0 ou $k_0 - 1$ états de $\mathcal{Q}_j$, ainsi que k_0 types de 0_i différents et k_0 types de 1_i différents, avec décalage des indices : on voit l'ensemble $\{a_1, ...a_{k_0} + e\} \cup \{1_{i-1 \bmod 2k_0+4}, 0_i, 1_i, ..., 1_{i+k_0-1 \bmod 2k_0+4}, 0_{i+k_0 \bmod 2k_0+4}\}$ avec $e \in \{0, -1\}$ et $a_x \in \mathcal{Q}_j$.
  On note cet ensemble $D1(i, \{a_1, ...a_{k_0} + e\})$.
- ceux décalés *vers la droite*, contiennent k_0 ou $k_0 - 1$ états de $\mathcal{Q}_j$, ainsi que k_0 types de 0_i différents et k_0 types de 1_i différents, sans décalage des indices : on voit l'ensemble $\{a_1, ...a_{k_0} + e\} \cup \{0_i, 1_i, ..., 0_{i+k_0 \bmod 2k_0+4}, 1_{i+k_0 \bmod 2k_0+4}\}$ avec $e \in \{0, -1\}$ et $a_x \in \mathcal{Q}_j$.
  On note cet ensemble $D2(i, \{a_1, ...a_{k_0} + e\})$.

Ces ensembles sont tous distinguables, en s'intéressant aux indices des 0 et des 1. Et pour qu'un automate simule $\mathcal{A}_0$ sur S_j, il suffit qu'il satisfasse les transitions suivantes :

- $C(i, \{a_1, ...a_{k_0}\}) \to \delta_{\mathcal{A}_0}(\{a_1, ...a_{k_0}\})$
- $D1(i, \{a_1, ...a_{k_0} + e\}) \to 1_{i+\lfloor k_0/2 \rfloor}$
- $D2(i, \{a_1, ...a_{k_0} + e\}) \to 0_{i+\lfloor k_0/2 \rfloor}$

En effet dans ce cas, la structure générale des configurations est maintenue, S_j est stable, et les états de $\mathcal{Q}_j$ évoluent selon $\delta_{\mathcal{A}_0}$. On peut alors expliciter la fonction $i_j : \Sigma_{\mathcal{A}_0}^{2k_0+4} \to \Sigma_n^{l(2k_0+4)}$ compatible avec les évolutions de l'automate simulateur et de $\mathcal{A}_0$.

Satisfaire ces motifs de transition pour un sous-shift S_j donné revient à fixer un nombre donné, indépendant de n et k, de transitions.

Mais le nombre d'états différents apparaissant dans chacune de ces transitions est inférieur à $3k_0 + 2$. Par conséquent, la proportion d'automates d'une taille donnée avec plus de $2k_0 + 4 + jn_0$ états qui simulent $\mathcal{A}_0$ sur le sous-shift S_j est d'au moins p avec p une constante indépendante de j, n et k.

Or lorsque n croit, le nombre de sous-shifts S_j à considérer croit (linéairement en n), et est indépendant de k.

On peut utiliser le lemme 3.33 pour conclure la preuve.

□

Lemme 3.55. *Sur les chemins à nombre d'états borné, une propriété $\mathcal{P}$ telle que pour tout (n, k), $\mathcal{P} \cap \mathcal{ESC}_{n,k} \neq \emptyset$ n'est pas négligeable.*

Preuve : Le nombre d'ensemble de taille au plus k à n éléments disjoints, est inférieur à 2^n.

Donc le nombre d'automates de $\mathcal{ESC}_{n,k}$ est inférieur à 2^n. Par conséquent, avec l'hypothèse que $\mathcal{P} \cap \mathcal{ESC}_{n,k} \neq \emptyset$ et $\mathcal{P} \cap \mathcal{ESC}_{n,k} \neq \mathcal{ESC}_{n,k}$ pour chaque (n,k), on a :

$$\frac{1}{2^n} \leq \frac{\#\left(\mathcal{P} \cap \mathcal{ESC}_{n,k}\right)}{\#\left(\mathcal{ESC}_{n,k}\right)}$$

Donc, si n est borné par une constante c, il existe $\epsilon > 0$ tel que pour tout (n,k), on a

$$0 < \epsilon \leq \frac{\#\left(\mathcal{P} \cap \mathcal{ESC}_{n,k}\right)}{\#\left(\mathcal{ESC}_{n,k}\right)}$$

Ce qui suffit à conclure.

□

On peut maintenant formuler notre proposition :

Proposition 3.56. *Selon un chemin ρ, parmi les AC ensemblistes captifs, la densité de l'universalité vérifie :*

(1) si ρ_n tend vers l'infini alors elle vaut 1,

(2) si ρ_n est bornée par une constante positive, si elle existe, est strictement inférieure à 1,

(3) si ρ_n n'est pas bornée, mais ne tend pas vers l'infini, alors elle n'existe pas.

Preuve : Le (1) est une conséquence, comme d'habitude, des deux lemmes 3.53 et 3.54. Le (2) vient du lemme 3.55, en l'appliquant à la classe des AC nilpotents.

Enfin le (3) nécessite d'utiliser le (1) et le (2). En effet, un chemin tel que ceux considérés au (3) possède deux suites extraites l'une bornée, et l'autre tendant vers l'infini, en appliquant le (1) et le (2) à ces suites, on peut conclure.

□

Pour conclure cette partie, on peut signaler que la recherche sur toutes les problématiques présentées dans ce chapitre est encore active. D'une part celles concernant l'ensemble des AC, mais aussi celles concernant la quantification parmi des classes restreintes. En particulier, l'utilisation de comparaisons entre les classes permettrait sans doute d'affiner ces résultats et d'en amener des nouveaux. Ceci nécessite de construire des fonctions d'arités régulières entre certaines classes. C'est sans doute une des manières les plus prometteuses d'aborder la question de la densité de l'universalité parmi l'ensemble des AC général.

Chapitre 4

Propriétés des ensembles μ-limites

1 Définitions 100
1.1 μ-nilpotence 100
1.2 Murs et sensibilité 102
1.3 Langages μ-limites des automates cellulaires non sensibles 103
2 Résultats de calculabilité 104

La partie précédente a permis d'introduire un formalisme et d'apporter quelques résultats concernant le comportement d'un automate cellulaire pris au hasard, ou si l'on veut le comportement d'un *automate cellulaire typique*, en quantifiant la densité de certaines propriétés, soit sur l'ensemble total des automates cellulaires, soit parmi des sous-classes définies à partir de restrictions syntaxiques.

Dans ce chapitre, nous avons une approche complémentaire, en utilisant des notions qui permettent en un sens d'étudier le *comportement à long terme typique d'un automate cellulaire fixé*.

Nous avons évoqué dans l'introduction, et parfois utilisé dans les chapitres précédents, les ensembles limites d'automates cellulaires. Ceux-ci sont couramment utilisés pour chercher à appréhender le comportement à long terme des AC. Leur défaut cependant est d'accorder la même importance à toute les configurations, en particulier un ensemble limite peut être considéré comme très *complexe* simplement en raison de la structure particulière d'un ensemble réduit de configurations qui se maintiennent, et cela même si elles ont finalement très peu de *chances* d'émerger dans un calcul.

Pour tenter d'apporter une réponse à ce problème, et d'appréhender le comportement à long terme *typique* d'un AC, P. Kůrka et A. Maas [KM00] ont introduit la notion d'*ensembles limites associés à une mesure de probabilité* μ, que nous appellerons dans la suite ensembles μ-limites. Pour un automate donné, son ensemble μ-limite est un sous-shift constitué des configurations dont les facteurs (au sens de sous-mot connexe) ont une probabilité d'apparition qui ne décroit pas vers 0 lorsque le nombre d'étapes de calcul augmente, et en démarrant d'une *configuration aléatoire* suivant la distribution μ.

Ils ont étudiés ces propriétés, principalement pour des mesures de Bernoulli, μ à support total, et ont en particulier exhibé des AC pour lesquels, l'ensemble μ-limite coïncide avec l'ensemble limite. C'est le cas en particulier pour tous les AC surjectifs possédant un point d'équicontinuité.

En ce qui nous concerne, nous introduisons la notion d'AC μ-nilpotent, analogue de la nilpotence pour les ensembles limites. Et nous nous intéressons surtout aux propriétés de calculabilité

des ensembles μ-limites. Nous montrons que l'ensemble des automates μ-nilpotents n'est ni récursivement énumérable, ni co-récursivement énumérable. Nous montrons également l'existence d'AC dont les langages μ-limites ne sont pas récursifs.

1 Définitions

Dans toute la suite nous considérons des automates cellulaires unidimensionnels $\mathcal{A}$, dont l'ensemble des configurations $\Sigma_{\mathcal{A}}^{\mathbb{Z}}$ est muni d'une mesure de probabilité μ.

Pour tout $t \geq 0$, on note $\mathcal{A}^t\mu$ la mesure de probabilité définie sur tout borélien $U \subseteq \Sigma_{\mathcal{A}}^{\mathbb{Z}}$ par $\mathcal{A}^t\mu(U) = \mu(\mathcal{A}^{-t}(U))$. Le théorème 1.1 du à Hedlund ([Hed69]) garantit qu'il s'agit toujours d'une mesure. Ces mesures permettent de donner un sens à la probabilité d'apparition d'un ensemble de configurations après un certain nombre d'étapes de calcul de l'automate.

Dans la suite, nous limitons l'ensemble des mesures considérées aux mesures σ-invariantes les plus simples : les mesures de Bernoulli.

Une mesure μ est dite *de Bernoulli* s'il existe un vecteur de probabilités $(p_1, p_2, \ldots, p_{\#(\Sigma_{\mathcal{A}})})$ (c'est-à-dire tel que $0 \leq p_i \leq 1$ et $\sum p_i = 1$), qui vérifie : pour tout mot $u \in \Sigma_{\mathcal{A}}^*$ et pour tout $i \in \mathbb{Z}$, $\mu([u]_i) = \prod_{x \in \Sigma_{\mathcal{A}}} p_x^{|u|_a}$. Une mesure de Bernoulli est dite *complète* si pour tout i, $p_i \neq 0$.

Ce sont donc ces mesures de Bernoulli à support total (c'est-à-dire complète) que nous utilisons pour nos définitions principales :

Définition 37. *Soit $\mathcal{A}$ un automate cellulaire et μ une mesure de Bernoulli sur $\Sigma_{\mathcal{A}}^{\mathbb{Z}}$.*

(1) Un mot u est dit μ-limite *pour $\mathcal{A}$ si sa probabilité d'apparaître après t itérations ne tend pas vers 0 lorsque t croit vers l'infini.*

(2) Le langage μ-limite $\Lambda_\mu(\mathcal{A})$ *est défini comme l'ensemble de ces mots :*

$$u \notin \Lambda_\mu(\mathcal{A}) \Leftrightarrow \lim_{t \to \infty} \mathcal{A}^t\mu([u]_0) = 0.$$

(3) Enfin l'ensemble μ-limite *de $\mathcal{A}$ est le sous-shift $\Omega_\mu(\mathcal{A})$ défini par ce langage :*

$$\Omega_\mu(\mathcal{A}) = \{c \in \Sigma_{\mathcal{A}}^{\mathbb{Z}} : L(c) \subseteq \Lambda_\mu(\mathcal{A})\}.$$

Il faut donc noter que pour un mot $u \in \Lambda_\mu(\mathcal{A})$, il y a deux cas de figure : soit la limite de $\mathcal{A}^t\mu([u]_0)$ lorsque t croit a une valeur différente de 0, soit cette limite n'existe pas. Intuitivement, il s'agit donc des mots dont la probabilité d'apparition après un calcul arbitrairement long, démarrant d'une configuration aléatoire, ne devient pas négligeable à toutes les étapes.

En ce sens, il est plausible qu'ils capturent l'intuition des motifs qui apparaissent après un certain temps lorsqu'on fait évoluer un automate cellulaire à partir de configurations aléatoires.

Remarque 12. Comme noté dans [KM00], il serait sans doute raisonnable d'étendre les notions considérées à une classe de mesure plus large. L'extension la plus immédiate est peut-être la classe des mesures dites de Markov évoquées dans [MS03], ou même la classe plus large des mesures à mémoire finie.

1.1 μ-nilpotence

La totalité de ce chapitre est consacrée à l'étude des langages et des ensembles μ-limites. Par analogie avec les ensembles limites traditionnels, pour lesquels la nilpotence est une des propriétés les plus étudiées, il parait conséquent d'introduire une notion de μ-nilpotence.

Définition

Définition 38. *Soit $\mathcal{A}$ un automate quelconque et μ une mesure de Bernoulli. On dit que $\mathcal{A}$ est μ-nilpotent si $\Omega_\mu(\mathcal{A})$ est réduit à un singleton.*

On vérifie facilement par σ-invariance que dans ce cas l'ensemble $\Omega_\mu(\mathcal{A})$ est constitué d'une configuration uniforme.

Intuitivement, les automates μ-nilpotents sont donc ceux dont l'évolution mène *presque toujours* vers cette unique configuration uniforme.

L'automate `Just Gliders` de l'exemple 2 est un AC μ-nilpotent pour la mesure μ uniforme.

Cette notion fait fortement penser à la classification de Wolfram et semble correspondre à sa classe 1. En effet, la μ-nilpotence donne un sens formel au fait de mener à partir de presque toute configuration initiale vers une unique configuration uniforme.

Ensemble μ-limite fini non réduit à un singleton

Cependant, contrairement à ce qui se passe dans le cas de la nilpotence classique, associée aux ensembles limites, un ensemble μ-limite peut non seulement être réduit à un singleton et être infini, mais il peut aussi être réduit à un ensemble fini de cardinal strictement plus grand que 1.

Exemple 9. Soit $\mathcal{A}$ l'automate cellulaire élémentaire 184 selon la notation de Wolfram. Il s'agit d'un automate cellulaire à deux états $\Sigma_\mathcal{A} = \{0,1\}$, de rayon 1. Sa règle locale est donnée par $\forall x \in \{0,1\}$, $\delta_\mathcal{A}(1,0,x) = \delta_\mathcal{A}(x,1,1) = 1$ et $\delta_\mathcal{A}(0,0,x) = \delta_\mathcal{A}(x,1,0) = 0$.
Cet automate est souvent présenté comme un modèle très simple d'évolution pour les embouteillages : les *voitures* (1) avancent si elles ont un *espace* (0) immédiatement à leur droite, et restent en place sinon.
On montre d'abord que, pour la mesure de Bernoulli uniforme μ_0, ni 00, ni 11 ne sont dans le langage μ_0-limite. En effet, il est simple de vérifier que si le mot $u = u_0u_1 \dots u_{2n+1}$ est dans $\mathcal{A}^{-n}(00)$, alors $u_0 = 0$ et le mot $u_1 \dots u_{2n+1}$ est un préfixe d'un mot bien parenthésé (avec 0 parenthèse ouvrante, et 1 fermante). En effet, lors de l'évolution, un motif 00 avance vers la droite sur un fond alternant 0 et 1. Il disparait, pour donner un motif alternant 0 et 1 s'il rencontre un motif 11. Ceci se ramène à dire que pour qu'un motif 00 soit l'image d'un mot u, il faut que dans tout préfixe de u, le nombre de 0 soit strictement plus grand que le nombre de 1.
Or la proportion de tels mots parmi les mots de longueur $n+2$ tend vers 0 lorsque n tend vers l'infini. Ceci se montre par exemple en considérant une bijection avec les chemins de Dyck ([FS09]). Par conséquent $\lim_{n\to\infty} \mathcal{A}^n(00) = 0$, et donc 00 n'est pas persistant. Et l'on montre que le mot 11 n'est pas non plus dans l'ensemble limite, soit avec une preuve similaire, soit directement en utilisant l'antisymétrie de l'automate (il est invariant par symétrie des voisins et permutation des deux états).
Cependant, pour chaque l il y a au moins un mot de longueur l dans le langage μ-limite, $\Lambda_{\mu_0}(\mathcal{A})$ mais comme $\Lambda_{\mu_0}(\mathcal{A})$ est stable par sous-mot, et ne contient ni 00 ni 11, on a $\Lambda_{\mu_0}(\mathcal{A}) = (0+\epsilon)(10)^*(1+\epsilon)$ et $\Omega_{\mu_0}(\mathcal{A}) = \{^\omega(01)^\omega, ^\omega(10)^\omega\}$.

La définition de la μ-nilpotence peut donc être discutée, et l'on pourrait également choisir de considérer ainsi tous les automates dont l'ensemble μ-limite est un ensemble fini. Par exemple l'automate 184, dont l'ensemble μ-limite est limité à deux configurations qui sont les images l'une de l'autre à la fois par l'automate et par le décalage parait très simple. Cependant, les ensembles

μ-limites finis qui ont été construits dernièrement se montrent de plus en plus complexes. Peut-être la bonne propriété est-elle intermédiaire, par exemple en s'intéressant aux AC dont le comportement sur l'ensemble μ-limite se réduit à un cycle.

En l'absence d'argument décisif jusque là, nous nous contenterons de travailler avec la définition précédente qui limite la μ-nilpotence au cas d'un ensemble μ-limite réduit à une seule configuration, donc forcément uniforme.

Cependant, nous insistons sur le fait que les résultats énoncés dans la suite restent *tous* valides si l'on remplace cette définition par une autre plus large.

1.2 Murs et sensibilité

Une notion essentielle pour la compréhension des ensembles μ-limite est la notion de *murs*. En dimension 1, celle-ci est intimement liée à l'existence de points d'équicontinuité et donc à la *sensibilité* aux conditions initiales.

Intuitivement, une *base de mur* en dimension 1 est un mot dont la présence dans une configuration fixe la présence d'un motif, centré sur la base du mur, à chaque étape de l'évolution de la configuration (c'est cette suite de motifs qui constitue le *mur*). Et l'idée est que si ce mur est partout assez large, il empêche toute communication entre les deux parties de la configuration qu'il déconnecte, d'où le lien avec les propriétés topologiques.

Pour un automate cellulaire $\mathcal{A}$. Pour tout $u \in \Sigma_{\mathcal{A}}^*$, on note $[u]_{mid}$ l'ensemble de configurations suivant :

$$[u]_{mid} = \begin{cases} [u]_{-\frac{|u|}{2}} & \text{si |u| est paire} \\ [u]_{-\frac{|u|}{2}+1} & \text{si |u| est impaire} \end{cases}$$

. Il s'agit donc du cylindre déterminé par le mot u, placé dans une position *à peu près* centrée.

Définition 39 (Murs). *Un mur pour $\mathcal{A}$ est une suite $\mathcal{W} = (w_n)_{n\geq 0}$ de mots non vides de $\Sigma_{\mathcal{A}}^*$ tels que*

(1) $\forall c \in [w_0]_{mid}, \forall n \geq 1 : \mathcal{A}^n(c) \in [w_n]_{mid}$

(2) la suite $(|w_n|)_{n\geq 0}$ est bornée.

Remarquons que le mur $\mathcal{W} = (w_n)_{n\geq 0}$ est forcément ultimement périodique dans la mesure où $[w_0]_{mid}$ contient des configurations spatialement périodiques. Le mot w_0 est appelé la *base* du mur $\mathcal{W}$. Un mot est une *base de mur* pour $\mathcal{A}$ si c'est la base d'un mur de $\mathcal{A}$.

Un mot est appelé *brique du mur $\mathcal{W}$* si il est dans la période de $\mathcal{W}$. Formellement, u est une brique du mur $\mathcal{W} = (w_n)_{n\geq 0}$ s'il existe p et n_0 tel que pour tout $n \in \mathbb{N}$, $w_{pn+n_0} = u$. Un mot $u \in \Sigma_{\mathcal{A}}^*$ est appelé *une brique de mur* pour $\mathcal{A}$ si c'est une brique pour un mur de $\mathcal{A}$.

Par définition l'ensemble des briques est stable par passage au *facteur* (au sens de sous-mot connexe).

Une propriété bien connue relie l'existence de briques de mur pour un automate à la sensibilité, ou non, de celui-ci ([Kůrka97]).

Proposition 4.1. *Un automate cellulaire unidimensionnel $\mathcal{A}$ de rayon r est sensible aux conditions initiales si et seulement s'il ne possède aucune brique de mur de taille r.*

Celle-ci repose sur la propriété exprimée par le lemme suivant :

Lemme 4.2. *Soit $\mathcal{A}$ un automate cellulaire de rayon r. Si w est la base d'un mur de $\mathcal{A}$ ayant une brique de taille au moins r, alors pour tout mot $u \in \Sigma_{\mathcal{A}}^*$, il existe un mur de $\mathcal{A}$ dont le pied est wuw et qui a une brique de taille au moins $|u|$.*

1.3 Langages μ-limites des automates cellulaires non sensibles

Il faut tout d'abord expliciter un lemme qui formalise l'idée selon laquelle dans les ensembles d'antécédents d'un mot μ-limite, tout facteur (au sens de sous-mot connexe) apparaît à une certaine position de façon récurrente. Ce lemme apparait sous une autre forme dans [BT00].

Lemme 4.3. *Soit $\mathcal{A}$ un automate cellulaire de rayon r et μ une mesure de Bernoulli complète sur $\Sigma_{\mathcal{A}}^{\mathbb{Z}}$. Alors, pour chaque $u \in \Lambda_\mu(\mathcal{A})$ et pour tout mot $w \in \Sigma_{\mathcal{A}}^*$, il existe des entiers positifs k_1 et k_2, ainsi qu'une suite strictement croissante d'entiers positifs $(n_j)_{j\geq 0}$ tels que*

$$\forall j \geq 0 : \mathcal{A}^{-n_j}(u) \cap \left(\Sigma_{\mathcal{A}}^{rn_j-k_1-|w|} \cdot \{w\} \cdot \Sigma_{\mathcal{A}}^{k_1+k_2+|u|} \cdot \{w\} \cdot \Sigma_{\mathcal{A}}^{rn_j-k_2-|w|}\right) \neq \emptyset$$

Preuve : Par l'absurde supposons qu'il existe un mot $u \in \Lambda_\mu(\mathcal{A})$ qui ne vérifie pas le lemme. Alors on a $\forall k \geq 0$, $\exists n_k \geq 0$, $\forall n \geq n_k$:

$$\mathcal{A}^{-n}(u) \subseteq \Sigma_{\mathcal{A}}^{n-k|w|} \cdot \left(\Sigma_{\mathcal{A}}^{|w|} \setminus \{w\}\right)^k \cdot \Sigma_{\mathcal{A}}^{|u|} \cdot \left(\Sigma_{\mathcal{A}}^{|w|} \setminus \{w\}\right)^k \cdot \Sigma_{\mathcal{A}}^{n-k|w|}.$$

Nous avons donc pour tout k, et pour chaque $n \geq n_k$ $\mathcal{A}^n\mu([u]_0) \leq (1-\mu([w]_0))^{2k}$. Par conséquent, $\mathcal{A}^n\mu(u) \to 0$ lorsque $n \to \infty$ et $u \notin \Lambda_\mu(\mathcal{A})$.

□

Nous allons montrer à l'aide ce lemme le théorème suivant :

Théorème 4.4. *Soit $\mathcal{A}$ un automate cellulaire non sensible aux conditions initiales et μ une mesure de Bernoulli complète. Alors $\Lambda_\mu(\mathcal{A})$ est exactement l'ensemble des briques de murs de $\mathcal{A}$.*

Preuve : Tout d'abord, soit u une brique de mur pour $\mathcal{A}$. Par définition, il existe un mot $w \in \Sigma_{\mathcal{A}}^*$ et des entiers positifs n_0 et p, tels que $\forall c \in [w]_{mid}$ et $\forall n \geq 0$, $\mathcal{A}^{np+n_0}(c) \in [u]_{mid}$. Donc $\mathcal{A}^{np+n_0}\mu([u]_0) \geq \mu([w_0])$, ce qui prouve que $u \in \Lambda_\mu(\mathcal{A})$.

Réciproquement, supposons $u \in \Lambda_\mu(\mathcal{A})$. D'après la proposition 4.1, comme $\mathcal{A}$ n'est pas sensible aux conditions initiales, $\mathcal{A}$ admet une brique de mur de taille au moins r (avec r le rayon de $\mathcal{A}$), associée à un mur $\mathcal{W} = (w_n)_{n\geq 0}$. En appliquant le lemme 4.3 au mot w_0, on en déduit qu'il existe des entiers positifs k_1 et k_2, ainsi qu'une suite strictement croissante d'entiers positifs $(n_j)_{j\geq 0}$ tels que :

$$\forall j \geq 0 : \mathcal{A}^{-n_j}(u) \cap \left(\Sigma_{\mathcal{A}}^{rn_j-k_1-|w_0|} \cdot \{w_0\} \cdot \Sigma_{\mathcal{A}}^{k_1+k_2+|u|} \cdot \{w_0\} \cdot \Sigma_{\mathcal{A}}^{rn_j-k_2-|w_0|}\right) \neq \emptyset$$

Comme $\Sigma_{\mathcal{A}}^{k_1+k_2+|u|}$ est fini, il est possible d'extraire de $(n_j)_{j\geq 0}$ une sous-suite $(n_{j_k})_{k\geq 0}$ telle que pour un certain $v \in \Sigma_{\mathcal{A}}^{k_1+k_2+|u|}$, on ait :

$$\forall k \geq 0 : \mathcal{A}^{-n_{j_k}}(u) \cap \left(\Sigma_{\mathcal{A}}^{rn_{j_k}-k_1-|w_0|} \cdot \{w_0\} \cdot \{v\} \cdot \{w_0\} \cdot \Sigma_{\mathcal{A}}^{rn_{j_k}-k_2-|w_0|}\right) \neq \emptyset$$

Et en utilisant le lemme 4.2, le mot w_0vw_0 est la base d'un mur de $\mathcal{A}$, avec une brique de taille au moins $|v|$. Donc u est un facteur (au sens de sous-mot connexe) d'une brique de mur, c'est une brique lui-même car l'ensemble des briques est stable par passage au facteur.

□

Ce théorème entraîne en particulier le résultat suivant :

Corollaire 4.5. *L'ensemble μ-limite d'un automate non sensible au conditions initiales est le même pour toute mesure de Bernoulli à support total.*

Comme l'ont montré P. Kůrka et A. Mass, il existe des automates cellulaires, nécessairement sensibles, dont l'ensemble limite dépend de la mesure de Bernoulli considérée. C'est le cas de l'automate `Just Gliders`, introduit à l'exemple 2. On peut en effet montrer que s'il est μ-nilpotent pour la mesure de Bernoulli uniforme, ce n'est pas le cas pour une mesure pour laquelle les probabilités des états $\rightarrow$ et $\leftarrow$ sont différentes.

2 Résultats de calculabilité

Dans la suite μ dénotera la mesure uniforme. Hormis le théorème 4.9 pour lequel la preuve nécessite une adaptation ad hoc pour les autres mesures, les automates construits ne sont pas sensibles aux conditions initiales. Les résultats concernés peuvent donc être étendus à une mesure de Bernoulli à support total quelconque à travers l'utilisation du lemme 4.3 et le théorème 4.4

Remarque 13. Dans sa preuve de l'indécidabilité de la nilpotence, Kari ([Kar92]) a montré l'indécidabilité de la nilpotence pour les automates avec un état envahissant. Par ailleurs, il découle du théorème 4.4 qu'un automate cellulaire avec un état envahissant est μ-nilpotent, avec comme unique configuration limite, la configuration uniforme ne contenant que l'état envahissant. Par conséquent, déterminer si un automate nilpotent est μ-nilpotent ou non est indécidable.

Dans la suite on montre que l'ensemble des AC μ-nilpotents n'est ni énumérable (théorème 4.6), ni co-récursivement énumérable (théorème 4.9). Et l'on montre qu'il existe un automate au langage μ-limite non récursif (théorème 4.8).

Théorème 4.6. *L'ensemble des AC μ-nilpotent n'est pas récursivement énumérable.*

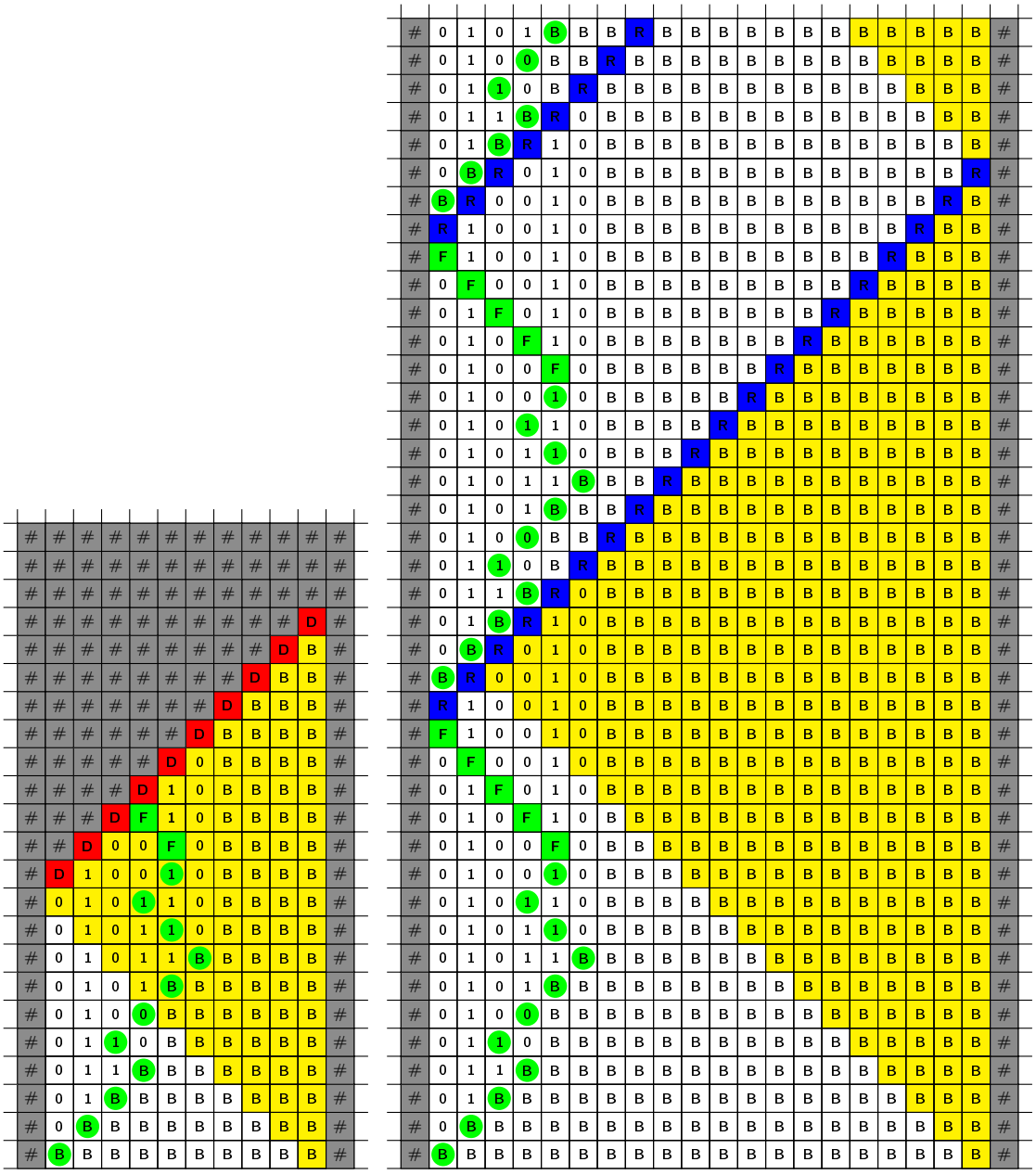
Preuve : Soit M une machine de Turing quelconque d'états Q_M (avec q_f son état final et q_0 son état initial), et d'alphabet de ruban $\Sigma_M = \{0, 1, B\}$, travaillant sur un ruban semi-infini. On construit un automate cellulaire $\mathcal{A}_M$ de rayon 1 qui est μ-nilpotent si et seulement si M ne s'arrête pas sur une entrée vide.

L'idée est de lancer des calculs de la machine M sur l'entrée vide, dans une zone délimitée par deux $\#$, et de lancer en parallèle une horloge qui borne la durée du calcul. Si la machine ne termine pas dans l'espace ou le temps qui lui est alloué, la zone est effacée, remplacée par des $\#$. Si au contraire la machine termine pour une certaine taille et un certain temps, le calcul est relancé périodiquement, garantissant alors une certaine richesse de l'ensemble μ-limite.

L'ensemble d'état de $\mathcal{A}_M$ est $\{\#\} \cup (S_{\text{simul}} \times S_{\text{sign}})$ où

- $\#$ est un état persistant (c'est-à-dire que si une cellule est dans cet état, elle ne prendra jamais aucun autre état).
- $S_{\text{simul}} = (Q_M \cup \{\cdot\}) \times \Sigma_M$ est l'ensemble des état utilisés pour simuler le comportement de M :
  - $(\cdot, \alpha)$ représente une case du ruban contenant la lettre α et sans tête de la machine,
  - (q, α) représente une case du ruban contenant la lettre α avec la tête de la machine dans l'état q sur cette case.
- $S_{\text{sign}} = \{-, L, F, R, D\}$ est un ensemble de signaux dont la signification et le comportement seront expliqués plus bas.

La fonction locale de transition de l'automate $\mathcal{A}_M$ peut être décrite pas les règles suivantes, qui sont illustrées par les figures 4.1 :



(a) Voisinage trop petit pour terminer

(b) Voisinage assez grand pour terminer

FIGURE 4.1: Comportement de l'AC dans deux cas différents. On représente ici différentes informations issues des différentes couches. Le rond vert représente la tête de la MT simulée, les rectangles jaunes des signaux L. Les autres signaux D, R et F sont appelés par leur nom.

- Les $\#$ sont inaltérables. Comme l'automate est de rayon 1, ils se comportent comme des murs : aucune information ne peut les traverser.
  Une succession continue de cellules dans des états autres que $\#$ entre deux $\#$ sera appelée un *segment*. La longueur d'un tel segment est le nombre de cellules entre les deux $\#$.
  Remarquons que $\#^b$ étant un mot bloquant pour tout b, d'après la proposition 4.1, l'automate n'est pas sensible.
- A chaque étape, une cellule dans un état autre que $\#$ simule, par sa première composante, le comportement de M. Les problèmes d'incohérence (par exemple deux têtes qui veulent se déplacer vers une même cellule) sont traités d'une manière arbitraire quelconque dans la mesure où cela n'a pas d'impact pour notre construction (on ne s'intéressera en fait qu'aux simulations démarrant d'une entrée vide correcte). Si à une étape donnée, la tête veut se déplacer sur une cellule qui est dans l'état $\#$, la tête est détruite et le calcul ne peut terminer.
- Le signal $-$ signifie qu'il n'y a pas de signal particulier sur la cellule.
- Si à une étape donnée l'état final de M, q_f apparaît, la cellule où celui-ci est apparu génère un signal F (sur sa composante de signaux).
- Le signal F se déplace vers la gauche à vitesse maximale. Quand il atteint le bord gauche du segment (marqué par un $\#$), il se transforme en un signal R.
- Le signal R se déplace vers la droite et réinitialise le calcul sur le segment qu'il traverse, c'est-à-dire qu'il met la première cellule dans l'état (q_0, B), et toutes les autres dans l'état $(\cdot, B)$ au fur et a mesure qu'il les traverse. Comme ce signal se déplace à vitesse maximale, la simulation de M se passe sans problème sur un ruban réinitialisé.
- Lorsque le signal R atteint l'extrémité droite du segment, il disparait.
- Pendant tout ce temps, la cellule la plus à droite d'un segment (c'est-à-dire en réalité n'importe quelle cellule qui voit un $\#$ à sa droite) génère un signal L à chaque étape.
- Les signaux L se déplacent vers la gauche à vitesse maximale. Lorsqu'un de ces signaux atteint le bord gauche d'un segment, il génère un signal D.
- Le signal D détruit la totalité du segment en se déplaçant vers la droite et en mettant toutes les cellules traversées dans l'état $\#$. Les signaux D disparaissent en atteignant le bord droit du segment (c'est-à-dire le premier $\#$ rencontré).

Tous les signaux introduits se déplacent à vitesse maximale (une cellule par étape) dans l'une ou l'autre des directions. On interdit à deux signaux de se croiser, lorsque deux signaux se rencontrent l'un disparaît et l'autre n'est pas modifié (il continue sa propagation). La priorité sur les signaux est donnée par l'ordre suivant :

$$L < F < R < D$$

Par exemple si un signal R croise un signal L, le signal L est détruit et le signal R n'est pas perturbé.

Supposons maintenant que la machine M s'arrête en t étapes, et considérons un segment de longueur $2t$, tel qu'aucune cellule ne contienne un signal, et tel que la première cellule soit dans l'état (q_0, B) et toutes les suivantes dans l'état $(\cdot, B)$. Sur ce segment, la simulation de M démarre d'une configuration bien formée, et l'état q_f est atteint après t étapes de calcul, un signal F est alors généré qui se propage vers la gauche.

La longueur du segment étant $2t$, les signaux L générés à son extrémité droite n'atteignent pas la gauche du segment avant le signal F. Ce dernier initialise donc un signal R qui réinitialise le calcul et efface les signaux L.

Un nouveau cycle de calcul se lance, qui aura assez de temps pour finir et effacer à nouveau les L.

Le segment étant isolé du reste de la configuration par les $\#$ extrémaux, ce cycle peut continuer indéfiniment, aucun $\#$ n'apparaîtra sur le segment. Le segment étant une zone finie, son comportement finira par devenir périodique.

L'évolution de ce segment décrit donc un mur. D'après le théorème 4.4, au moins une de ses images que nous noterons $\#u\#$ avec $u \in (\Sigma_{\mathcal{A}_M} \setminus \{\#\})^{2t}$ est une brique de mur, et appartient au langage limite. En réalité pour tout p le mot $(\#u\#)^p$ est dans le langage μ-limite, et donc la configuration périodique de période $\#u\#$ est dans l'ensemble μ-limite. Or il existe de tels segments pour toutes les tailles $2t$, l'ensemble μ-limite de $\mathcal{A}_M$ est donc infini, et $\mathcal{A}_M$ n'est pas μ-nilpotent.

Nous allons maintenant supposer que M ne s'arrête pas sur l'entrée vide, et montrer que dans ce cas un segment de longueur n est effacé par des $\#$ en moins de $5n$ étapes. En réalité, on va voir qu'au plus tard $4n$ étapes après l'étape initiale un signal D apparaît sur la première cellule du segment.

Éventuellement un tel signal D pouvait être présent dans le segment à l'étape initiale. Dans ce cas, celui-ci va couper le segment en deux parties en effaçant la moitié droite à l'aide de $\#$. Par conséquent, on peut supposer que le segment considéré ne contient pas de D à l'instant initial, quitte à se ramener à ce cas en considérant un segment plus petit.

On se place donc dans ce dernier cas. Après au plus n étapes, tous les signaux R présents à l'étape initiale ont disparus. Au plus tard à cette étape-là, les signaux L générés par la cellule la plus à droite du segment vont pouvoir se propager vers la gauche. Le seul moyen, pour qu'ils n'atteignent pas la cellule la plus à gauche et ne génèrent pas de signal D, est qu'un nouveau signal R apparaisse avant l'étape $2n$. Dans ce cas, le calcul est réinitialisé, et une simulation de M sur une entrée vide démarre à partir de l'étape $2n+1$.

Cependant, nous avons supposé que M ne s'arrête pas sur l'entrée vide. Par conséquent, aucun signal F ne sera généré et donc aucun autre signal R. Lorsque le signal R généré avant l'étape $2n$, a atteint le bord droit, c'est-à-dire avant l'étape $3n$, il disparait, et un signal L est généré. Celui-ci ne sera pas arrêté et un signal D apparaît, à l'étape au plus $4n$, et commence à détruire le segment. Tout ceci s'est passé en $5n$ étapes.

Il reste simplement à montrer que dans ce cas (si M ne s'arrête pas) aucun état autre que $\#$ n'est une brique de mur. Pour cela, considérons un mur quelconque $\mathcal{W} = (w_i)_{i\in\mathbb{N}}$. Soit c_{w_0} la configuration formée de w_0 au centre, et entouré de $\#$ partout ailleurs. Par définition c_{w_0} ne contient aucun segment de longueur plus grande que $|w_0|$, donc pour $n \geq 5|w_0|$, $\mathcal{A}_M^n(c_{w_0})$ est la configuration uniforme ${}^\omega\#^\omega$. Donc pour $n \geq 5|w_0|$, $w_n \in \#^*$: les briques de mur sont toutes des mots de $\#^*$. En utilisant le théorème 4.4, l'automate $\mathcal{A}_M$ est μ-nilpotent.

Par conséquent, une machine de Turing énumérant les automates cellulaires μ-nilpotent pourrait servir à construire une machine qui énumèrerait les machines de Turing qui ne s'arrêtent sur l'entrée vide, ce qui conclut la preuve du théorème.

□

Remarque 14. Comme discuté un peu plus haut, la construction précédente montre également qu'il est indécidable de savoir si un ensemble μ-limite est fini ou non.
De fait, l'ensemble μ-limite des automates construits est soit infini soit réduit à une seule configuration.

Corollaire 4.7. *Étant donné un automate cellulaire $\mathcal{A}$, l'ensemble $\{(\mathcal{A}, w) : w \notin \Lambda_\mu(\mathcal{A})\}$ n'est pas récursivement énumérable.*

Preuve : On a montré dans la preuve précédente que l'ensemble des automates $\mathcal{A}$ d'ensemble μ-limite $\Omega_\mu(\mathcal{A})$ réduit à un singleton n'est pas récursivement énumérable. Or ces automates sont exactement ceux dont le langage limite $\Lambda_\mu(\mathcal{A})$ est de la forme $\{x\}^*$ pour un certain $x \in \Sigma_\mathcal{A}$, c'est-à-dire ceux dont l'ensemble μ-limite contient un et un seul état.

Si l'on pouvait énumérer le langage $\{(\mathcal{A}, w) : w \notin \Lambda_\mu(\mathcal{A})\}$, on pourrait énumérer les automates ayant tous leurs états hormis un en dehors de leur langage μ-limite. Ceci contredirait le résultat précédent.

□

Le théorème suivant, qui repose sur une construction adaptée de celle de la preuve précédente, a également un équivalent pour les langages limites [Hur87].

Théorème 4.8. *Il existe un automate cellulaire avec un langage μ-limite non récursif.*

Preuve : On reprend la construction précédente, en la modifiant de la manière suivante :

L'ensemble d'états est désormais $\{\#\} \cup (S_{\text{simul}} \times S_{\text{mémo}} \times S_{\text{sign}})$.

$\#$, S_{sign}, et S_{simul} sont comme précédemment, et $S_{\text{mémo}}$ est maintenant une couche de sauvegarde $S_{\text{mémo}} = \Sigma_M$.

Le comportement de l'automate est le même sauf que dorénavant, lorsque R réinitialise un calcul, l'état placé sur le ruban de calcul n'est plus B mais l'état contenu dans la couche $S_{\text{mémo}}$.

Pour les mêmes raisons que dans la preuve précédente, les segments non uniformément $\#$ qui font partie du langage μ-limite sont ceux dont la couche mémoire correspond à un mot wB^k tel que M termine à partir de l'entrée w, en utilisant moins de $|w| + k$ cases de son ruban. Or les langages persistants sont stables par facteur, donc un mot $\#wB$ est la couche mémoire d'un mot de $\Lambda_\mu(\mathcal{A}_M)$ si et seulement si M termine en démarrant sur l'entrée w.

Or si le langage $\Lambda_\mu(\mathcal{A}_M)$ est récursif, le langage $\{\#wB : M \text{ s'arrête sur } w\}$ l'est aussi. Il suffit de tester l'appartenance à $\Lambda_\mu(\mathcal{A}_M)$ de tous les mots $\#u$ tels que wB est la composante mémoire de u, et ils sont en nombre fini.

Il suffit donc, pour obtenir un langage μ-limite non calculable de considérer l'automate $\mathcal{A}_M$ associé à une machine M dont le domaine n'est pas récursif, une machine de Turing universelle par exemple.

□

Théorème 4.9. *L'ensemble des AC μ-nilpotent n'est pas co-récursivement énumérable.*

Preuve : Comme dans la preuve du théorème 4.6, on va associer à une machine de Turing M un automate cellulaire $\mathcal{A}_M$ de rayon 1 qui la simule. $\mathcal{A}_M$ sera μ-nilpotent si et seulement si M s'arrête sur l'entrée vide. Comme précédemment, la configuration sera divisée en segments, séparés par des $\#$. Cependant, ces $\#$ ne seront plus vraiment persistants. L'idée est que si une simulation ne s'arrête pas sur un segment donné, le $\#$ de droite sera supprimé, de manière à pouvoir relancer la simulation sur un segment agrandi. A l'opposé, si un signal de fin de calcul apparaît, le segment est détruit, recouvert de $\#$.

De cette manière, si M s'arrête presque tous les segments seront détruits, nous montrerons la μ-nilpotence de la machine. Mais si elle ne s'arrête pas, la simulation de la machine continuera

sur de larges zones. Il nous faudra alors montré que l'ensemble μ-limite est assez riche. Pour cela nous ajoutons aux états non-# une couche binaire $\{0,1\}$ de bits qui n'interagiront pas avec leurs voisins (sauf éventuellement pour être effacés par un #), mais qui alterneront leur valeur à chaque étape temporelle.

La construction est très similaire à la précédente (preuve du théorème 4.6). L'ensemble d'états est presque le même : $\{\#\} \cup (S_{\mathsf{simul}} \times S_{\mathsf{sign}} \times S_{\mathsf{bin}})$ avec # l'état délimiteur, $S_{\mathsf{simul}} = (Q_M \cup \{\cdot\}) \times \Sigma_M$ et $S_{\mathsf{bin}} = \{0,1\}$. De plus l'ensemble des signaux est modifié. On a maintenant $S_{\mathsf{simul}} = \{-, L, R, D, D_L, D_R, C_L, C_R\}$.

Et la règle locale d'évolution, illustrée par 4.2 et 4.3 de l'automate peut être décrite comme suit :

- Les # sont *presque* persistant, concrètement seul un signal particulier (D) peut les détruire. Nous continuons à utiliser la notion de segment (une suite continue de cellules non # délimitée par deux #).
- Sur chaque segment, les cellules simulent M de la même manière que dans la construction précédente.
- Les bits de S_{bin} de chaque cellule (non #) ne changent jamais de valeur.
- Les signaux L apparaissent toujours au niveau de la cellule la plus à droite d'un segment (en fait pour chaque cellule qui voit un # à sa droite) et se propagent vers la gauche.
- Lorsqu'un signal L rencontre le bord gauche d'un segment (un #), un signal D est généré.
- Un signal D se déplace vers la droite. Il détruit tout les signaux L qu'il rencontre. S'il rencontre une cellule ayant l'état q_f (état final de M dans sa couche de simulation, il génère deux signaux D_L et D_R qui vont détruire le segment en se propageant chacun dans une direction et en transformant toutes les cellules en # jusqu'à atteindre les extrémités du segment (marquées par des # déjà présents). Sinon, lorsque le signal D atteint l'extrémité droite du segment, il supprime le # qui marque cette extrémité. Il met la cellule qui était dans l'état # dans un état calculant $(\cdot, B, 0)$ et envoie deux signaux C_L et C_R vers la gauche et la droite.
- Les signaux C_L et C_R se déplacent respectivement vers la gauche et la droite, ils effacent le contenu du ruban de manière à relancer un calcul sur l'espace délimité par le même # à gauche et le suivant à droite. Ils détruisent également les signaux qu'ils pourraient rencontrer, mais ne touchent pas à la couche S_{bin}. Lorsque le signal C_L atteint le bord gauche du segment, il se transforme en signal R. Lorsque le signal C_R atteint le bord droit il disparaît.
- Le signal R initialise le calcul sur le nouveau segment comme dans la construction précédente. Il détruit également les signaux L. Mais il ne touche pas non plus à S_{bin}.

Nous allons maintenant prouver le théorème de manière similaire à la preuve du théorème 4.6.

Supposons que M ne s'arrête pas sur l'entrée vide. Dans ce cas, un calcul de $\mathcal{A}_M$ simulant le comportement de M à partir d'une entrée vide ne fera jamais apparaître l'état final de M q_f et donc ne fera pas apparaître les signaux D_L ou D_R. Sur un segment ne contenant initialement ni simulation de M, ni signaux, un signal L va atteindre le bord gauche du segment, et initialiser un calcul, en étendant éventuellement le segment. Ce calcul ne terminant pas, par choix de M, le segment va continuer à calculer, et à s'étendre en envahissant vers la droite. De même il peut aussi se retrouver envahi par la gauche. Dans tous les cas, le segment ne sera jamais détruit, et une simulation continuera à se dérouler sur la zone considérée, une simulation correcte dans la mesure où des signaux C_L et C_R ont été générés..

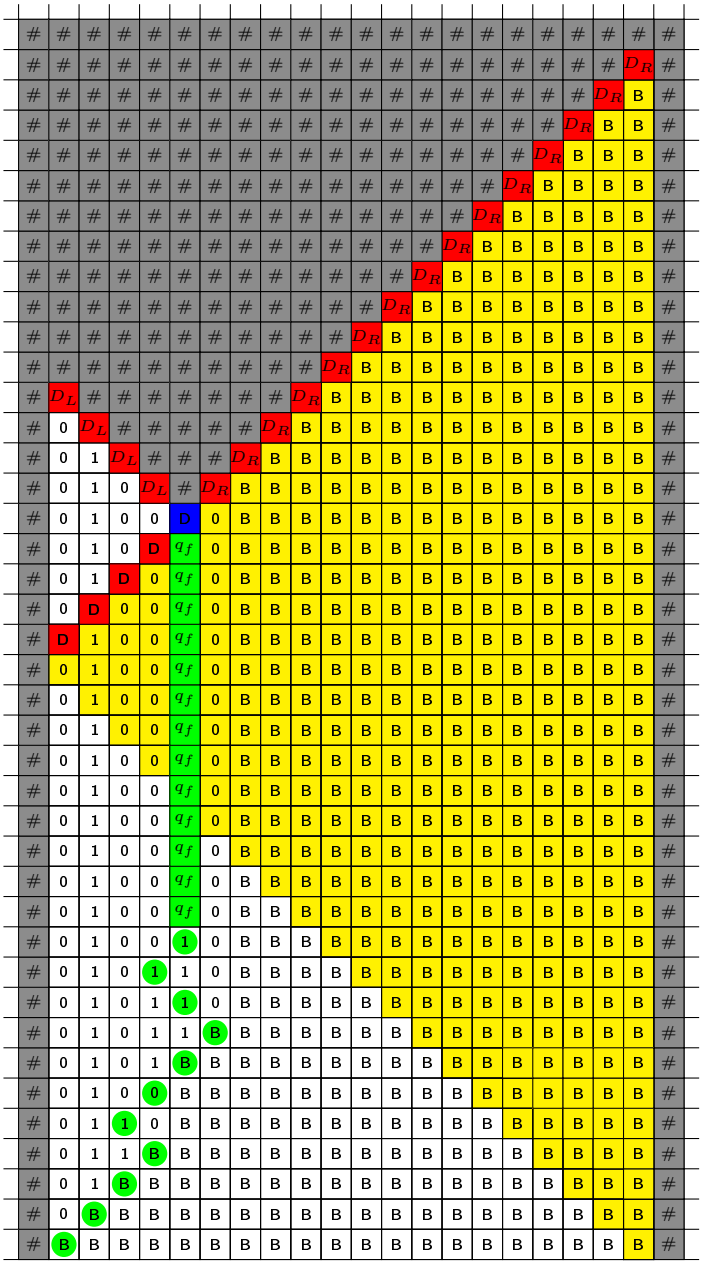


FIGURE 4.2: Comportement de l'AC avec un voisinage assez grand pour terminer. On représente une superposition de certaines informations issues de différentes couches. Le rond vert représente la tête de la MT simulée, les rectangles jaunes des signaux L. Les autres signaux sont appelés par leur nom.

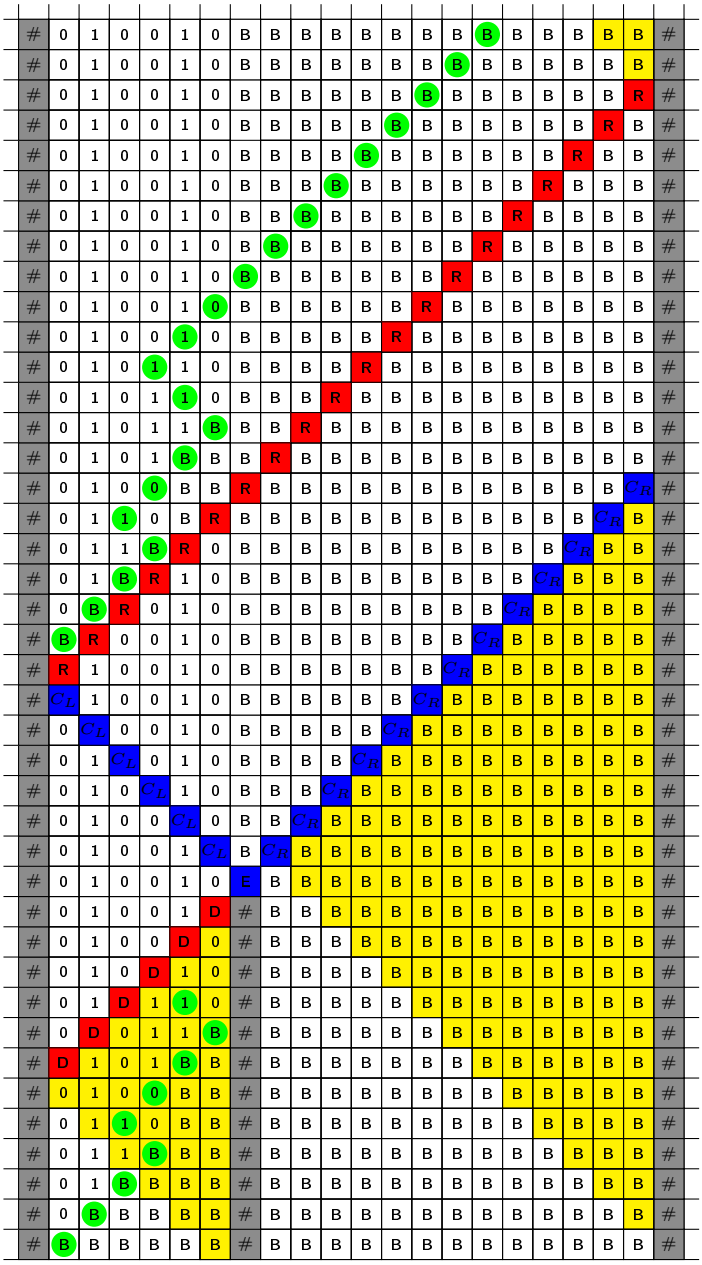


FIGURE 4.3: Comportement de l'AC avec un voisinage trop petit pour terminer. On représente une superposition de certaines informations issues de différentes couches. Le rond vert représente la tête de la MT simulée, les rectangles jaunes des signaux L. Les autres signaux sont appelés par leur nom.

En résumé, si M ne s'arrête pas, un segment inactif dans la configuration initiale va grandir, et continuer à calculer sans que les cellules qui le composent n'entrent jamais dans l'état $\#$, maintenant en particulier la couche S_{bin}.

Considérons maintenant un tel segment s, de taille $2k+1$ (en incluant les bords $\#$), alors pour tout $n \in \mathbb{N}$, et tout mot $w \in \Sigma^*_{\mathcal{A}_M}$,

$$wsw \in \cup_{q \in \Sigma_{\mathcal{A}_M} \setminus \{\#\}} \mathcal{A}_M^{-n-k}(q)$$

Dit autrement, les antécédents des états non $\#$ sont très nombreux. Plus précisément, pour tout $n \in \mathbb{N}$, $\sum_{q \in \Sigma_{\mathcal{A}_M} \setminus \{\#\}} \mathcal{A}_M^{n+k} \mu(q) \geq \mu([s]_0)$. Ce qui signifie qu'au moins un des états non $\#$ est dans le langage μ-limite, notons-le q_1.

Or cet état a une couche binaire S_{bin}, de valeur $x \in \{0,1\}$, dont il a hérité (à travers les alternances successives) de la cellule centrale de s dans la mesure où la cellule n'est jamais entrée dans l'état $\#$. Cette valeur binaire n'entrant pas en compte autrement dans l'évolution du segment wsw, l'état obtenu en inversant la valeur binaire de q_1 est également dans le langage μ-limite. Par conséquent, $\mathcal{A}_M$ n'est pas μ-nilpotent.

De plus on peut considérer des mots s de longueur croissante $2k+l$, pour tout $l \in \mathbb{N}^+$ et considérer les antécédents des mots de longueur l. On obtient alors 1, puis 2^l mots de longueur l dans le langage μ-limite, en considérant toutes les valeurs possibles de la couche S_{bin}. Ceci permet de garantir que l'ensemble μ-limite est infini.

Considérons maintenant une machine M qui s'arrête en t étapes à partir de l'entrée vide. Soit s un segment de longueur $l \geq 2t$ qui ne contient aucune simulation de M, et tel que le seul signal présent est un signal R sur la première cellule. Si ce segment n'est pas envahi, il va faire un calcul correct de M, faire apparaître un état q_f, et un état D va l'atteindre et générer D_L et D_R. Et le segment va être détruit, remplacé par des $\#$.

L'élément important est que ce segment n'envahira jamais le segment à sa droite. Il en est de même si lui-même a été envahi (par la gauche nécessairement). En effet dans ce cas, des signaux C_L et C_R auront été envoyé, garantissant qu'un calcul correct a été lancé. L'espace étant au moins aussi grand, le calcul n'entrainera jamais l'invasion vers la droite. Un tel segment, qui n'envahit jamais son voisin est appelé *non-invasif*. Et de tels segments existent.

Considérons maintenant un segment, entouré du côté gauche d'un segment non-invasif à distance moins de d_l (la distance est le nombre de cellules entre le segment considéré et l'extrémité droite du segment non invasif), et du côté droit par un $\#$ à distance d_r (de son bord droit) avec $d_r \geq 2t$.

Si le segment n'est pas envahi, il va après un certain temps lancer une simulation, à moins d'avoir été complètement effacé. A partir de cet instant, il va continuer à simuler, puis à envahir ses voisins tant que le calcul ne terminera pas. Comme nous avons supposé la présence d'un $\#$ à distant au moins $2t$, la simulation va finir par terminer et le segment sera alors détruit.

La seule chose qui pourrait repousser cette destruction est une succession d'invasion par la gauche. Mais la présence d'un segment non-invasif garanti que celles-ci ne peuvent intervenir qu'en nombre limité.

Par conséquent, la simulation va finir par se produire sur un segment assez grand pour faire terminer le calcul, et détruire ce segment.

En résumé, on a montré que si M termine sur l'entrée vide, il existe une fonction $\tau : \mathbb{N}^2 \to \mathbb{N}$ telle qu'un segment ayant un segment non-invasif à sa gauche à distance d_1 et un $\#$ à sa droite à distance $d_2 \geq 2t$ disparait en temps moins de $\tau(d_1, d_2)$. Par conséquent, pour tout $n \geq \tau(d_1, d_2)$,

et tout état $q \in \Sigma_{\mathcal{A}_M} \setminus \{\#\}$ un mot de $\mathcal{A}_M^{-n}(q)$ ne contient pas de segment non-invasif dans les positions à gauche de la position $-d_1$ et ne contient pas deux états $\#$ dans les positions incluses entre $2t$ et d_2. Ceci suffit à montrer à l'aide du lemme 4.3 qu'aucun des états de $\Sigma_{\mathcal{A}_M} \setminus \{\#\}$ n'est dans le langage μ-limite.

De plus dans le cas ou la machine s'arrête, comme il existe des segments non invasifs, le mot formé de deux tels segments consécutifs est un pied de mur, dont les murs ont au moins la largeur du second segment (en réalité ce sont des briques de murs uniformément formés de $\#$).

Par conséquent, dans le cas où la machine M s'arrête, cet automate n'étant pas sensible, d'après le théorème 4.4, il sera μ-nilpotent pour toute mesure de Bernoulli complète.

□

Remarque 15. Comme au théorème 4.6, l'ensemble μ-limite obtenu est soit fini réduit à une seule configuration, soit infini. Ceci nous garantit donc des changements de définition.
Par ailleurs, pour des mesures de Bernoulli non uniformes, mais à support complet, comme nous ne montrons que l'automate n'est pas sensible dans le cas où la machine ne s'arrête pas, nous ne pouvons pas invoquer le théorème 4.4. Cependant, la construction précédente peut être adaptée, mais la présence d'un support total est essentielle.

Corollaire 4.10. *Étant donné un automate cellulaire $\mathcal{A}$, l'ensemble $\{(\mathcal{A}, w) : w \in \Lambda_\mu(\mathcal{A})\}$ n'est pas récursivement énumérable.*

Preuve : Un automate est μ-nilpotent si et seulement s'il son langage limite est réduit à une seule lettre. Son langage limite $\Lambda_\mu(\mathcal{A})$ est alors de la forme $\{x\}^*$ pour un certain $x \in \Sigma_{\mathcal{A}}$. Supposons qu'on dispose d'une énumération de l'ensemble $\{(\mathcal{A}, w) : w \in \Lambda_\mu(\mathcal{A})\}$. Dans ce cas, chaque fois que pour un automate $\mathcal{A}$ fixé, on a énuméré deux mots $(\mathcal{A}, u)$ et $(\mathcal{A}, v)$, tels que l'union des alphabets de u et v contient au moins deux lettres on sait que $\mathcal{A}$ n'est pas μ-nilpotent. Réciproquement, si un automate $\mathcal{A}$ n'est pas μ-nilpotent, son langage $\Lambda_\mu(\mathcal{A})$ n'est pas réduit à un $\{x\}^*$, il contient soit un mot u qui contient deux lettres, soit une paire de mots dont les alphabets sont différents.

Par conséquent, on peut déduire d'une énumération de $\{(\mathcal{A}, w) : w \in \Lambda_\mu(\mathcal{A})\}$ une énumération de $\{\mathcal{A} : \forall x \in \Sigma_{\mathcal{A}}, \Lambda_\mu(\mathcal{A}) \neq \{x\}^*\}$, ce qui caractérise les AC non μ-nilpotent et contredit donc le théorème 4.9

□

Chapitre 5

Universalité par facteurs

1 Universalité parmi les AC surjectifs 2D 116

2 Universalité parmi les AC persistants 1D 116

2.1 Les AC persistants 116

2.2 Existence d'un AC persistant-universel 117

On s'intéresse ici aux propriétés de la simulation par facteurs et plus précisément à la notion d'universalité qui lui est associée, *dite universalité par facteurs*. Rappelons que celle-ci a été introduite dans [The05].

Cette relation est aussi fortement liée à la notion mathématique de facteurs entre systèmes dynamiques, expliquée dans [LM95]. Et l'on peut de ce fait utiliser une définition similaire pour travailler avec des sous-shifts ([BT10]).

On a évoqué précédemment le fait que cette simulation ne repose pas sur un encodage des orbites du simulé dans celles du simulateur, mais plutôt sur une interprétation de toutes les orbites du simulateur comme des orbites du simulé. En quelques sortes, la simulation se produit et à du sens sur toutes les orbites pour tous les calculs du simulateur.

Cette simulation s'éloigne donc de la notion usuelle en informatique d'*encodage*. En contrepartie, on obtient une manière de simuler plus souple, qui ne repose pas sur des sous-ensembles de simulation définis de manière très rigide en concaténant des blocs précis.

En ce sens cette simulation parait donc d'une part plus *robuste* en ce qui concerne le calcul, et d'autre part permet de comparer des automates non pas en se focalisant sur certains ensembles de configurations bien formés, mais en s'intéressant à leur dynamique prise dans son ensemble.

Cependant cette simulation reste moins bien connue que la simulation intrinsèque. Et nous avons évoqué au chapitre 1 le fait que la question de l'existence d'un automate universel reste ouverte.

Sans répondre à cette question, nous apportons quelques éléments en se restreignant à des sous-classes. Les résultats que l'on apporte concernent l'existence d'automate universel pour des sous-classes, c'est-à-dire d'un AC qui soit lui-même dans une telle classe, et qui soit capable de simuler tous les AC de celle-ci. Nous montrons un résultat négatif, la non existence d'automates cellulaires universels par facteur pour les surjectifs en dimension 2, puis un résultat positif, l'existence d'un automate cellulaire universel par facteur pour les AC persistants en dimension 1.

1 Universalité parmi les AC surjectifs 2D

Pour la simulation intrinsèque, l'ensemble des automates cellulaires surjectifs forme un idéal, c'est-à-dire que tout AC simulé par un surjectif est lui-même surjectif.

En dimension 2, cet idéal ne possède pas d'élément maximal : il n'existe pas d'AC intrinsèquement universel pour les surjectifs. En effet, il suffirait d'énumérer les automates cellulaires simulés par celui-ci pour obtenir une énumération des automates surjectifs. Or ceci contredirait le théorème de Kari (théorème 1.5).

Il est possible de suivre le même raisonnement pour la simulation par facteur.

Théorème 5.1. *Il n'existe pas d'automate cellulaire* surjectif-*universel pour la simulation facteur en dimension 2.*

Preuve : Notons pour commencer que les AC simulés par facteur par un automate cellulaire surjectif sont également surjectifs. En effet, si $\mathcal{A} \preccurlyeq_{\trianglelefteq} \mathcal{B}$ ils vérifient une relation de la forme

$$\mathcal{A}^{t_{\mathcal{A}}} \circ \Phi = \Phi \circ \mathcal{B}^{t_{\mathcal{B}}}$$

avec une fonction Φ surjective. Par conséquent si $\mathcal{B}$ est surjectif, c'est aussi le cas de $\mathcal{A}$.

Par ailleurs, on peut énumérer les automates simulés par un automate donné. En effet les paramètres de groupages, tout comme les fonctions de coloriage peuvent être énumérées. Par conséquent, l'existence d'un automate cellulaire universel surjectif permettrait d'énumérer l'ensemble des automates cellulaires surjectifs de dimension 2 qui serait donc récursivement énumérable. Ce qui contredirait le théorème 1.5.

□

2 Universalité parmi les AC persistants 1D

2.1 Les AC persistants

On élargit légèrement la définition des automates persistants, par rapport à la définition des automates *avec un état persistant* utilisée dans les chapitres précédents.

Définition 40. *Un automate cellulaire $\mathcal{A}$ est dit persistant, noté $\mathcal{A} \in \mathbb{P}$, s'il possède un mot non vide $u \in \Sigma_{\mathcal{A}}^*$ tel que pour un certain entier t, pour toute configuration $c \in \Sigma_{\mathcal{A}}^{\mathbb{Z}}$ et toute position $x \in \mathbb{Z}$, si $c_{[x..x+|u|-1]} = u$ alors $\mathcal{A}^t(c)_{[x..x+|u|-1]} = u$.*

L'intuition est simple : il s'agit des automates qui possèdent un certain mot qui, s'il apparait quelque part dans un diagramme espace-temps, réapparaîtra périodiquement dans le futur, à la même position.

L'ensemble des automates *avec un état persistant* évoqué au chapitre précédent est le cas particulier où, avec les notations de la définition $t = |u| = 1$.

La définition de ces automates à état persistant est beaucoup plus syntaxique, même lisible directement dans la table de transition. Ce n'est pas le cas pour les automates persistants :

Proposition 5.2. *L'ensemble des automates cellulaires persistants n'est pas récursif.*

Preuve : Pour le montrer, on va réduire le problème de la nilpotence pour les automates de rayon 1 possédant un état envahissant à celui-ci.

Soit $\mathcal{A}$ un automate de voisinage centré de taille 3, d'alphabet $\Sigma_\mathcal{A}$ et possédant un état envahissant noté q_0.

Soit $\mathcal{B} = \sigma_3 \circ \mathcal{A}$.

$\mathcal{B}$ déplace le cone d'état envahissant vers la gauche.

On va montrer que :

(1) $\mathcal{B}$ est persistant ss'il existe l tel que q_0^l est persistant pour $\mathcal{B}$.

(2) $\mathcal{A}$ est nilpotent ss'il existe l tel que q_0^l est persistant pour $\mathcal{B}$.

On montre d'abord le (1). Supposons qu'il existe un mot $u \in \Sigma_\mathcal{A}^l$ persistant, c'est-à-dire qu'il existe $t > 0$ tel que $\forall c \in [u]_0$, $\mathcal{A}^t(c) \in [u]_0$. Or si $c \in [u \cdot q_0^{5t}]_0$, alors $\mathcal{A}^t(c) \in [q_0^l]_0$. Donc $u = q_0^l$.

On montre maintenant le (2). Si $\mathcal{A}$ est nilpotent, alors $\mathcal{B}$ l'est également, et il existe un temps t tel que $\forall c \in \Sigma_\mathcal{B}^\mathbb{Z}$, $\mathcal{B}^t(c) = {}^\omega q_0^\omega$. Donc q_0^l est persistant.

Inversement, si q_0^l est persistant pour $\mathcal{B}$, il existe t tel que $\forall c \in \Sigma_\mathcal{A}^\mathbb{Z}$, $\forall x \in \mathbb{Z}$, si $c_{[x..x+l-1]} = q_0^l$, alors $\mathcal{B}(c)_{[x..x+l-1]} = q_0^l$, et donc par définition de $\mathcal{B}$, $\mathcal{A}(c)_{[x-3t..x-3t+l-1]} = q_0^l$. Soit maintenant $v \in \Sigma_\mathcal{A}^{2t+1}$ quelconque. Et soit c telle que $c_{[-t..t]} = u$, et $c_{[3t..3t+l-1]} = q_0^l$. On a donc $\delta_\mathcal{A}^{2t}(v) = c_0 = q_0$. Et c'est valable pour tout mot v, donc $\mathcal{A}$ est nilpotent. Par conséquent $\mathcal{A}$ est nilpotent si et seulement si $\mathcal{B}$ est persistant. Ceci prouve que la persistance n'est pas récursivement énumérable. □

Combinatoirement parlant le nombre d'automates *avec un état persistant* est de $n^{n^{k-1}.(n-1)}$. C'est-à-dire en particulier que cet ensemble est négligeable parmi l'ensemble des automates cellulaires lorsque la taille croit. Cependant, cette classe possède une grande richesse de comportement. En particulier du point de vue calculatoire, il possède des automates cellulaires intrinsèquement universels. Et ceux-ci ont même une densité 1 dans cette classe lorsque le rayon croît.

La classe des automates persistants regroupe donc des comportements assez riches et variés.

2.2 Existence d'un AC persistant-universel

On va maintenant montrer l'existence parmi cette classe d'un automate cellulaire universel pour la simulation par facteur.

Théorème 5.3. *Il existe un automate $\mathbb{P}$-universel.*

Pour cela on distingue tout d'abord un sous-ensemble de $\mathbb{P}$.

Soit $\mathbb{P}_0$, l'ensemble des automates cellulaires de rayon 1, avec un ensemble d'états de taille 2^p pour un p donné et pour lesquels 0 est un état persistant.

Lemme 5.4. *Pour tout automate $\mathcal{A} \in \mathbb{P}$, il existe $\mathcal{B} \in \mathbb{P}_0$ tel que $\mathcal{A} \preccurlyeq_\trianglelefteq \mathcal{B}$.*

Preuve : Soit $\mathcal{A}$ un automate persistant de rayon $r_\mathcal{A}$ et alphabet $\Sigma_\mathcal{A}$, on va construire un tel $\mathcal{B}$. Soit $u \in \Sigma_\mathcal{A}^l$ un mot persistant de $\mathcal{A}$, et $t \in \mathbb{N}^+$ tel que $\forall x \in \mathbb{Z}$, $\forall c \in \Sigma_\mathcal{A}^\mathbb{Z}$, si $c_{[x..x+l-1]}$ alors $\mathcal{A}^t(c)_{[x..x+l-1]} = u$.

Soit $m \in \mathbb{N}^+$ le plus petit entier multiple de l et plus grand que $r_\mathcal{A} t$. Soit $\mathcal{A}'$ l'automate obtenu en groupant $\mathcal{A}$ temporellement selon t et spatialement selon m.

$$\mathcal{A}' = \mathcal{A}^{<m,t,0>}$$

Alors $\mathcal{A}'$ a un rayon 1, et il possède un état persistant qui correspond à un multiple de u, de plus on sait que le morphisme entre $\Sigma_\mathcal{A}^m$ et $\Sigma_{\mathcal{A}'}$ fait de 0 l'état persistant..

On va maintenant définir $\Sigma_\mathcal{B}$, de taille 2^p au moins aussi grande que $\Sigma_{\mathcal{A}'}$, et $\mathcal{B}$ de même comportement que $\mathcal{A}'$. Pour cela, il aura sur les motifs de $\Sigma_{\mathcal{A}'}$ la même règle de transition que

$\mathcal{A}'$, et on la complète de manière à ce que les états ajoutés induisent le même comportement que l'état 0. $\mathcal{B}$ a donc même voisinage que $\mathcal{A}$ et alphabet $\Sigma_\mathcal{B} = \Sigma_{\mathcal{A}'} \cup \{x_i\}_{i\in I}$ avec I fini. Si on note $\pi_0 : \Sigma^*_\mathcal{B} \to \Sigma^*_{\mathcal{A}'}$ le morphisme lettre à lettre qui remplace toutes les occurrences des x_i dans un mot sur $\Sigma_\mathcal{B}$ par l'élément minimal $0 \in \Sigma_{\mathcal{A}'}$, on définit $\delta_\mathcal{B}$ par $\delta_\mathcal{B}(u) = \delta_{\mathcal{A}'}(\pi_0(u))$.

On vérifie alors $\mathcal{A}' \preccurlyeq_\lhd \mathcal{B}$. Et ce dernier a bien un état persistant.

□

Pour prouver le théorème 5.3, on va décrire un automate universel $\mathcal{U}$, de rayon 1, d'alphabet $\Sigma_\mathcal{U}$ et de règle $\delta_\mathcal{U}$.

Et d'après le lemme 5.4, il suffira de montrer que, pour chaque automate $\mathcal{A} \in \mathbb{P}_0$, il existe un temps $\tau_\mathcal{A}$ et une fonction locale et surjective $\Psi_\mathcal{A} : \Sigma^\mathbb{Z}_\mathcal{U} \to \Sigma^\mathbb{Z}_\mathcal{A}$ tels que $\forall c \in \Sigma^\mathbb{Z}_\mathcal{A}$, $\mathcal{A} \circ \Psi_\mathcal{A}(c) = \Psi_\mathcal{A} \circ \mathcal{U}^{\tau_\mathcal{A}}$.

Malgré les différences entre les notions de simulation, on va s'inspirer des ingrédients classiques, issus des constructions d'automates intrinsèquement universels. On va les combiner avec l'utilisation de l'état persistant pour les adapter aux contraintes liées à l'universalité par facteur.

L'ingrédient de base de notre construction est issu de la description d'automates cellulaires intrinsèquement universels. Il s'agit de traduire chaque état de l'automate simulé $\mathcal{A}$ par un motif d'états de $\mathcal{U}$ (c'est-à-dire un mot de $\Sigma^*_\mathcal{U}$) dont les longueurs sont identiques (pour un automate $\mathcal{A}$ donné). Un bloc de cellules de $\mathcal{U}$ dans un tel état émulera le comportement d'une cellule de $\mathcal{A}$. On l'appellera *macro-cellule pour* $\mathcal{A}$. Chaque macro-cellule contient des informations : au moins un encodage de l'état courant de la cellule simulée et un encodage de la règle de transition $\delta_\mathcal{A}$ de l'automate simulé.

Dans le cas de l'universalité intrinsèque classique, il suffit de gérer, par la règle de transition de l'automate universel, l'évolution des états courant des macro-cellules. Ceci se fait en suivant la règle de l'automate encodée dans celles-ci. Et surtout, cela se fait uniquement pour les *configurations constituées exclusivement de macro-cellules correspondant toutes au même automate* $\mathcal{A}$.

À l'opposé, pour la simulation facteur, il ne suffit pas de garantir une évolution correcte pour cet ensemble restreint de configurations bien connues. Si un automate $\mathcal{A}$ simule un automate $\mathcal{B}$, à travers une fonction Ψ, il faut que l'évolution de *n'importe quelle configuration* soit interprétable à travers Ψ comme une orbite correcte de l'automate simulé.

Et si l'on considère un automate facteur-universel, ceci doit être vrai pour n'importe quel automate : l'évolution de toute configuration doit être interprétable, à travers une fonction $\Psi_\mathcal{A}$, comme une évolution correcte de $\mathcal{A}$ pour n'importe quel $\mathcal{A}$.

Pour utiliser tout de même cette notion de macro-cellules, on va se servir de l'état persistant de l'automate simulé comme *interprétation* de toutes les zones qui ne correspondent pas à des macro-cellules bien formées pour cet automate $\mathcal{A}$. Il faudra, de plus, que l'évolution des macro-cellules soit conforme à l'automate simulé, même si elles ne sont pas entourées de macro-cellules correspondant au même automate, et même si elles ne sont pas entourées de macro-cellules du tout.

Pour chaque automate avec un état persistant, on va donc distinguer d'une part les macro-cellules qui correspondent à cet automate, et d'autre part les autres mots de la même longueur, mais qui ne sont pas des macro-cellules correctes. Les premières seront interprétées en fonction de l'état courant encodé dedans, et les autres seront toujours interprétées comme l'état persistant.

Comme parfois dans le cas de l'universalité intrinsèque, on va gérer l'évolution de ces macro-cellules à l'aide de signaux se déplaçant sur une couche *au dessus* de la couche de calcul principal. Ceci se formalise en faisant de l'ensemble d'états $\Sigma_{\mathcal{U}}$ de notre automate universel un produit cartésien. Dans ce cas, chaque état de l'automate universel contient une superposition d'une information issue de chacune des couches.

Dans la suite, on distinguera la couche principale qui contient la plupart des informations relatives aux macro-cellules, et les autres couches, utilisées pour le calcul. L'ensemble d'état $\Sigma_{\mathcal{U}}$ aura la forme suivante :

$$\Sigma_{\mathcal{U}} = (M \times S \times C) \cup \{q_p\}$$

L'état q_p est un état persistant. Il n'interviendra pas de manière utile dans la simulation : les macro-cellules seront formées sur l'alphabet $\Sigma_{\mathcal{U}} \setminus \{q_p\}$, et celui-ci n'est l'image d'aucune transition autre que celles assurant sa persistance.

M est la couche principale, les autres couches seront détaillées plus loin. Pour un mot $x \in \Sigma_{\mathcal{U}}^*$, on notera $\pi_M(x)$ le mot constitué de la couche principale des états.

Macro-cellules

On va d'abord décrire précisément la couche principale qui caractérise les macro-cellules correspondant à un automate $\mathcal{A}$ donné.

On introduit d'abord une notion de *motifs calculant* que l'on affinera ensuite légèrement pour caractériser précisément les macro-cellules. Pour un automate $\mathcal{A}$, les motifs calculant sont les mots de $\Sigma_{\mathcal{U}}$ dont la couche principale est constituée de la succession de :

- Un délimiteur $\#$ marquant le début du motif.
- Un état de contrôle C_i, $i \in \{0..7\}$ qui évoluera au cours des différentes étapes de chaque cycle de simulation.
- Une description de la table de transition de $\mathcal{A}$.
- Une description de l'état courant du motif calculant, choisi dans $\Sigma_{\mathcal{A}}$.
- Une zone de mémoire utile pour le calcul.
- Un délimiteur $\#$ marquant la fin du motif.

Les délimiteurs n'apparaissent ou ne disparaissent jamais.

L'état de contrôle servira à initialiser les étapes successives de la simulation. Il sert à la synchronisation temporelle. De plus sont aussi inclus dans cet état deux bits, $b_g, b_d \in \{0, 1\}$ qui vont contenir au cours du calcul de l'information respectivement sur les voisinages de gauche et de droite.

La description de la table de transition de $\mathcal{A}$ se fait en utilisant un alphabet à trois éléments $\{0_{tt}, 1_{tt}, B_{tt}\}$. Les deux premiers états servent à décrire la valeur binaire correspondant à l'image d'une transition, et le troisième à délimiter ces différentes images. Ceci nécessite donc $log(n_{\mathcal{A}})+1$ bits pour chaque transition (comme on considère des automates $\mathcal{A}$ de $\mathbb{P}_0$, $\log(n_{\mathcal{A}})$ est un entier). Ces images sont ordonnées selon un ordre bien spécifié, de manière à pouvoir les retrouver ! Rappelons que $\Sigma_{\mathcal{A}}$ est ordonné. On ordonne deux images $\delta_{\mathcal{A}}(x_1, x_2, x_3)$ et $\delta_{\mathcal{A}}(y_1, y_2, y_3)$ en comparant $x_2x_1x_3$ et $y_2y_1y_3$ selon l'ordre lexicographique induit par celui-ci. La longueur totale de la description de la table de transition est donc ${n_{\mathcal{A}}}^3(log(n_{\mathcal{A}})+1)$. On peut signaler enfin que lorsque la couche principale d'une cellule est dans un des états 0_{tt}, 1_{tt}, ou B_{tt}, cette valeur ne changera jamais.

La description de l'état courant se fait par l'écriture dans un alphabet binaire $\{0_{cs}, 1_{cs}\}$ du rang de l'état dans $\Sigma_{\mathcal{A}}$. Ceci occupe donc $log(n_{\mathcal{A}})$ cellules. Et une cellule dont la couche principale est de ce type n'en changera pas : ce sous-alphabet est stable sous l'action de $\mathcal{U}$.

La zone de mémoire utilise un troisième alphabet $\{0_m, 1_m, B_m\}$, stable également sous l'action de $\mathcal{U}$. La zone utilisée pour le calcul proprement dit est de seulement $2.log(n_\mathcal{A})$ bits. Mais on va imposer que la mémoire occupe une zone de longueur $({n_\mathcal{A}}^3+1)log(n_\mathcal{A})+1$ de manière à occuper plus de la moitié de la longueur $l_\mathcal{A}$ des motifs calculant, ce qui sera utile pour la suite. Comme pour les deux autres sous-alphabets de la couche principale, celui-ci est également stable.

Une *macro-cellule* est simplement un motif calculant dont l'état de contrôle est C_0, c'est-à-dire à l'étape de démarrage d'un cycle de mise à jour. On note $\mathcal{C}_\mathcal{A} \subseteq \Sigma_\mathcal{U}^*$ l'ensemble des macro-cellules associées à un automate $\mathcal{A}$. Et l'on note donc $l_\mathcal{A}$ *la longueur d'un motif calculant* ou d'une macro-cellule associée à $\mathcal{A}$. On a $l_\mathcal{A} = 4 + (2{n_\mathcal{A}}^3 + 2)log(n_\mathcal{A})$.

L'alphabet de la couche principale est

$$M = \{C_i\}_{i\in\{0..6\}} \times \{0,1\}^2 \cup \{0_{tt}, 1_{tt}, B_{tt}, 0_{cs}, 1_{cs}, 0_m, 1_m, B_m, \#\}$$

On retrouve les états de contrôle et les bits les accompagnent, les sous-alphabets de chaque zones, ainsi que les délimiteurs.

L'évolution de ces macro-cellules est gérée par la règle de transition de l'automate universel de manière à mimer le comportement de l'automate simulé.

On va commencer par décrire la règle de l'automate à partir d'une macro-cellule correcte. Puis on verra comment on complète la règle ensuite, ce qui n'est pas anodin dans la mesure où le calcul doit être cohérent avec les règles des automates simulés sur n'importe quelle configuration.

Comportement d'une macro-cellule

Considérons donc d'abord une macro-cellule de $\mathcal{C}_\mathcal{A}$. Une étape de simulation, c'est-à-dire un cycle de mise à jour de son état courant, se décompose en plusieurs opérations successives.

1. D'abord la macro-cellule va tester si elle est bordée à gauche, ou à droite, par des motifs calculant de même taille qu'elle-même et qui sont synchronisés avec elle, c'est-à-dire telles que l'état C_0 est apparu au même moment.
2. Puis pour chaque voisinage qui satisfait ces critères, elle va tester si les différentes zones (table de transition, état courant, mémoire) coïncident avec les siennes, et si le contenu de la table de transition est identique à la sienne.
3. Ensuite elle peut enfin copier dans sa zone de mémoire l'état se chacun de ses voisins, en écrivant à la place l'état persistant si un des tests précédents a échoué.
4. Elle calcule finalement son nouvel état courant en utilisant l'ancienne valeur de celui-ci, et les valeurs écrites dans sa zone de mémoire, pour chercher dans la description de la table de transition l'image correspondant à celles-ci. Une fois localisée, elle la copie dans sa zone d'état courant.

On l'a déjà dit, tout ceci est géré en utilisant des signaux. Il s'agit d'états se déplaçant sur des couches spécifique à chaque type de signaux (c'est-à-dire ne détruisant pas systématiquement les autres couches), et interagissant entre signaux, et avec la couche principale. On dit qu'un signal se déplace pour dire que son comportement est celui d'un décalage spatial périodique à une certaine

vitesse. Il est classique de construire des signaux se déplaçant avec pour vitesse n'importe quelle fraction de la vitesse maximale.

Tous les signaux sont émis par un état de contrôle, ou par un autre signal. Dans notre construction, on peut toujours rattacher un signal émis après l'instant initial à une macro-cellule. Si ce signal est engendré par un état de contrôle, on dit qu'il appartient à la macro-cellule correspondant à celui-ci. s'il est engendré par une collision entre deux signaux appartenant à la même cellule, où à une interaction entre un signal et un $\#$, il appartient à la même macro-cellule que les signaux impliqués. Et de même, on verra que les cas où un signal est généré par l'interaction entre deux signaux de macro-cellules différentes sont très limités, et que dans chaque cas, on peut rattacher ce signal à une macro-cellule. De plus la règle de transition va nous garantir quelques propriétés de ces signaux.

- Un signal ne quitte jamais la zone constituée de sa macro-cellule et des *voisins* de celle-ci : il ne pourra jamais être séparé de sa macro-cellule par plus de 3 $\#$.
- Un signal peut toujours déterminer s'il appartient à la macro-cellule sur laquelle il est, ou à sa voisine de droite ou de gauche.
- Tout signal qui rencontre une cellule dans l'état q_p est détruit.

Enfin, le dernier ingrédient nécessaire à notre construction est *un système d'horloges*, reposant sur le lemme suivant :

Lemme 5.5. *Pour tout $k, h \in \mathbb{N}\backslash\{1\}$,$m \in \mathbb{N}$, il existe un automate cellulaire $\mathcal{A}$ avec $\{\#, q_s, q_f\} \subset \Sigma_{\mathcal{A}}$ qui vérifie :*
$\forall l \in \mathbb{N}^+$, $\forall c \in [\#\Sigma_{\mathcal{A}}^l\#]$, $\forall t \in \mathbb{N}$, $t \geq kl^2 + hl + m$, si $\mathcal{A}^t(c) \in [\#q_f\Sigma_{\mathcal{A}}^{l-1}\#]$ alors
(1) $\mathcal{A}^{t-kl^2-hl-m}(c) \in [\#q_s\Sigma_{\mathcal{A}}^{l-1}\#]$
(2) $\forall x \in \mathbb{N}$, $t - kl^2 - hl - m < x < t$, $\mathcal{A}^x(c) \notin [\#q_f\Sigma_{\mathcal{A}}^{l-1}\#]$

Ce lemme nous permettra de spécifier d'une part la durée de chacune des opérations d'une étape de simulation, et d'autre part la durée totale d'une étape. Il permettra de garantir, dans le cas où un C_0 apparait à une étape, l'existence d'un C_0 et d'un démarrage de calcul correct à une étape antérieure.

Preuve : [du lemme 5.5] On construit d'abord un automate qui répond au problème pour $k = 2$ et $h = m = 0$.

Ses états, autres que $\#$, q_s et q_f, contiendront une couche binaire, et une couche de signaux.

L'idée générale est simple : lorsque q_s apparaît, il lance un signal qui va osciller entre les deux $\#$. Au premier passage, celui-ci initialise la zone délimitée par les $\#$, en mettant le premier bit à 1, et tous les autres à 0. Ensuite le signal continue à faire des aller-retours entre les deux $\#$. Et à chaque passage vers la droite, il met le premier bit de valeur 0 qu'il rencontre à 1.

Lorsque la couche binaire de la cellule la plus à droite est mise à 1, un nouveau signal est envoyé vers la gauche, qui va entraîner l'apparition de q_f lorsqu'il atteindra le bord gauche.

Si deux signaux se rencontrent, ils se détruisent. La seule exception est celle des signaux initialisant qui détruisent n'importe quel autre signal, mais survivent. Ceux-ci ne peuvent se rencontrer vu qu'il n'y en a qu'une sorte, et qu'ils se déplacent donc tous dans le même sens.

Par conséquent, durant une séquence de $2l$ étapes de calcul consécutives, le nombre total de 1 pourra ne pas grandir uniquement dans les cas suivants :

- s'il n'y a pas de signaux du tout.
- Si toutes les valeurs binaires étaient à 1, et dans ce cas q_f a été généré.
- Si un signal initialisant a été lancé.

En particulier, si un q_f est apparu à une étape donnée, alors un 1 est apparu dans les $4l$ étapes précédentes. Et dans chaque séquence de $2l$ étapes consécutives, au moins un 1 est apparu.

Donc dans la séquence des $2l^2$ étapes consécutives précédentes, un signal d'initialisation a été lancé, et donc un q_s généré. Mais par construction lorsque c'est le cas, le premier q_f apparaît seulement exactement $2l^2$ étapes de calcul plus tard.

Ceci permet de conclure la preuve dans le cas $k = 2$ et $h = m = 0$. Pour d'autres valeurs de k, il suffit de ralentir le signal qui va de la droite vers la gauche, en divisant sa vitesse par $k - 1$. Pour d'autres valeurs de h, il suffit de, après la fin de la partie quadratique, avant de faire apparaître q_f, lancer un signal vers la droite à vitesse maximale, qui revient après avoir rencontré le $\#$ à une vitesse ralentie d'un facteur $h - 1$, et qui génère le signal q_f lorsqu'il atteint le $\#$ gauche. Enfin pour d'autres valeurs de m, il suffit d'ajouter encore m états, qui réalisent un compte à rebours avant de faire apparaître q_f après la fin de la partie linéaire.

□

Nettoyage

Rappelons que l'on considère l'évolution à partir d'une macro-cellule, correspondant donc à un automate donné $\mathcal{A}$.

À l'instant initial, l'état de contrôle de la cellule est dans l'état C_0. À l'instant suivant, cet état devient C_1 et un certain nombre de signaux sont envoyés qui démarrent les tests de longueur et synchronisation, et initialisent les couches de signaux. Détaillons cette initialisation.

Un signal s_1 est envoyé à vitesse maximale (1) vers la gauche et un autre s_4 vers la droite. Ils détruisent tous les signaux appartenant à la même macro-cellule qu'eux-même qu'ils rencontrent. Le but de cette opération est d'éviter que des signaux présents dans la macro-cellule ou dans les zones voisines ne perturbent le calcul. Comme la table de transition est invariante, seul l'état courant encodé peut être modifié avant le calcul, c'est lui dont on cherche à empêcher les modifications en détruisant ces signaux.

Malheureusement cette initialisation ne peut pas être instantanée. Le contenu de la zone d'état courant peut être modifié avant l'arrivée du signal nettoyant. Cependant la longueur de la zone de mémoire, qui est au moins la moitié de celle de la macro-cellule, permet de garantir qu'aucun signal qui n'était pas présent sur la macro-cellule à l'étape initiale n'atteint la zone de description de l'état courant avant d'être détruit par s_1. Le nettoyage garantit donc qu'aucun signal externe à la macro-cellule à l'étape initiale ne modifie l'état courant.

Par conséquent, même si on ne peut pas empêcher qu'il y ait une différence entre l'état courant pris en compte lors du calcul et celui de l'instant initial, la valeur de l'état considéré au cours du calcul peut être déterminée à partir du contenu de la macro-cellule à l'instant initial, indépendamment du voisinage de la macro-cellule : pour cela il faut considérer non seulement la couche principale, mais aussi les signaux contenus sur les autres couches à l'instant initial, et les modifications qu'ils vont induire sur l'état courant.

Définition 41. *L'état ainsi déterminé est appelé* valeur associée *à la macro-cellule, c'est celui pris en compte au moment du calcul. Il s'agit aussi de l'état encodé dans le motif calculant à l'étape* $l_{\mathcal{A}}/2$.

On note $v(u) \in \Sigma_{\mathcal{A}}$ *la valeur associée à une macro-cellule* $u \in \Sigma_{\mathcal{U}}^{l_{\mathcal{A}}}$.

Et cet état va effectivement pouvoir servir d'*interprétation* à la macro-cellule (c'est-à-dire d'image par $\Psi_{\mathcal{A}}$ dans la mesure où il ne dépend que des états présents dans la zone de la macro-cellule.

Test de la longueur et de la synchronisation des voisinages

Nous décrivons maintenant l'étape de comparaison des longueurs et des tables de transitions. Rappelons que le mécanisme décrit traite le cas d'une macro-cellule correcte.

Cette opération est illustrée, dans un cas de succès, par la figure 5.1.

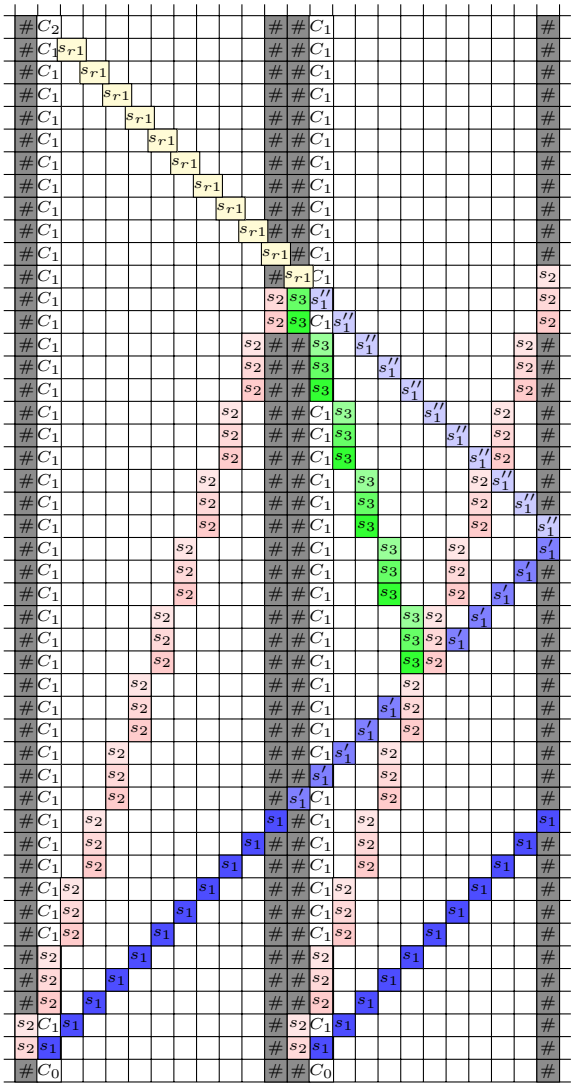


FIGURE 5.1: Test du voisinage de droite. Dans ce schéma et les suivants, on représente une petite partie de l'information des états, issue de différentes couches de l'automate. En particulier on ne représente pas le détail de la couche principale des états sauf éventuellement à l'étape initiale.

La longueur Commençons par le côté droit. On a vu qu'un signal s_1 (qui sert aussi au nettoyage) est envoyé à vitesse maximale vers la droite. Un second signal s_2 est envoyé dans la même direction à un tiers de cette vitesse.

Le premier va traverser les deux premiers # et un état de contrôle C_1, se transformant en un signal s_1', puis rebondir sur le troisième # qu'il rencontre pour devenir s_1''. s'il ne rencontre pas le

deuxième # puis un état de contrôle C_1 après le premier #, il devient un signal s_1''' qui continue le nettoyage mais disparaît sans rebondir sur le troisième # qu'il rencontre.

On s'intéresse à la position où les deux signaux s_1'' et s_2 se rencontrent. Il est simple de se rendre compte que ces deux signaux se rencontrent *exactement* sur le deuxième # de la frontière si et seulement si les deux macro-cellules ont la même longueur (et un état C_1). En effet la distance parcourue par le premier est alors exactement le triple de la longueur $l_\mathcal{A}$ des macro-cellules.

La synchronisation Pour tester la synchronisation il suffit d'ajouter un signal : lorsque s_1' croise le signal s_2 de la cellule de droite, un signal s_3 est émis à vitesse un tiers vers la gauche. Et ce signal rencontre les deux autres exactement là où ils se croisent, si les signaux ont été émis lorsque C_0 est devenu C_1, c'est-à-dire si les macro-cellules sont synchronisées.

Il est alors possible d'émettre, si la collision de ces trois signaux a lieu, un signal s_{r1}, à vitesse maximale, en direction de la gauche.

Si la macro-cellule considérée possède à sa droite un motif de même longueur entre deux # et qui a un état de contrôle synchronisé avec le sien, un signal de succès du test pour ce voisin s_{r1} est reçu par la cellule dans l'état de contrôle C_1 de notre macro-cellule exactement $4l_\mathcal{A} - 2$ étapes plus tard.

Si à ce moment-là, elle ne reçoit rien, elle peut en conclure que son voisin de droite n'est pas une macro-cellule de même taille et de même synchronisation. Pour cela il faut donc initialiser un compteur qui soit lancé lorsque C_0 apparaît, et qui lève un signal pour la cellule de contrôle exactement $4l_\mathcal{A}$ étapes plus tard. C'est possible grâce au lemme 5.5.

Test du voisinage de gauche

Cette opération est illustrée sur la figure 5.2, dans le cas d'un succès.

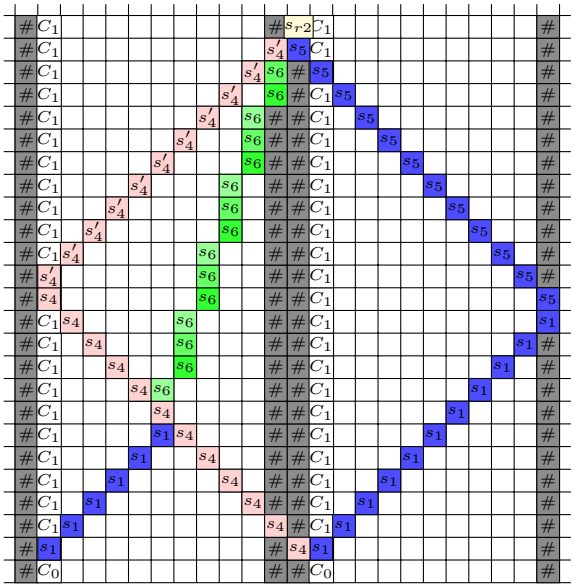


FIGURE 5.2: Test du voisinage de droite.

Du côté gauche, le mécanisme est similaire. Un signal s_4 est émis à vitesse maximale vers la gauche, et se réfléchit sur le troisième # qu'il rencontre, si celui-ci jouxte un état de contrôle C_1 (ce

signal devient donc $s'4$ puis s''_4). Par ailleurs un signal s_5 est émis vers la gauche lorsque s_1 (le signal initialisant vers la droite) traverse son premier $\#$. Ces deux signaux se rencontrent sur la frontière entre les deux macro-cellules lorsque les deux zones délimitées par les $\#$ ont même longueur. La synchronisation est testée grâce à un troisième signal émis à vitesse un tiers vers la droite par la collision du signal s_4 avec le signal s_1 de la cellule de gauche. Ces trois signaux se rencontrent simultanément et sur la frontière si et seulement si les macro-cellules sont synchronisées.

Et comme précédemment un signal s_{r2} est émis en direction du signal de contrôle si le résultat est positif. Le cas échéant, ce résultat parvient à l'état de contrôle en exactement $2l_{\mathcal{A}}+2$ étapes. Si aucun résultat n'est parvenu à l'état de contrôle après $4l_{\mathcal{A}}$ étapes alors celui-ci en déduit que le voisin de gauche n'est pas une macro-cellule de même longueur synchronisée avec elle-même.

Dans tous les cas le résultat de cette étape est stocké par l'état de contrôle dans les deux bits correspondants. Et l'état de contrôle devient C_2 $4l_{\mathcal{A}}$ étapes après l'apparition de C_0

Pour garantir le comportement de l'automate jusque là, on montre le lemme suivant.

Lemme 5.6. *Exactement $4l_{\mathcal{A}}$ étapes après que la cellule la plus à gauche d'une zone de longueur $l_{\mathcal{A}}$ délimitée par deux $\#$ soit entrée dans l'état C_0, cette cellule devient C_2 et le bit de contrôle associé au côté droit (resp. gauche) vaut 1 si et seulement si le voisinage de droite (resp. gauche) de la zone est constitué d'une zone de longueur $l_{\mathcal{A}}$ délimitée par deux $\#$ et qui est synchronisée, c'est-à-dire dont la cellule la plus à gauche est entrée dans l'état C_0 au même moment que celle de la zone centrale.*

Formellement, soit $c \in [u]_0$ avec $u \in \mathcal{C}_{\mathcal{A}}$. Alors :
- $\pi_M(\mathcal{A}^{4l_{\mathcal{A}}}(c)_1) = (C_2, b_g, b_d)$,
- $b_g = 1$ *ssi* $\pi_M(c_{[-l_{\mathcal{A}},-1]}) = \# \cdot (C_0, x', y') \cdot v \cdot \#$ *avec* $v \in M \setminus \{\#\}$
- $b_d = 1$ *ssi* $\pi_M(c_{[l_{\mathcal{A}},2.l_{\mathcal{A}}-1]}) = \# \cdot (C_0, x'', y'') \cdot v \cdot \#$ *avec* $v \in M \setminus \{\#\}$

En réalité, ce lemme repose non seulement sur la construction décrite ici, mais également sur des propriétés plus globales de la table de transition. Pour le prouver, il faut en effet garantir que les signaux impliqués dans cette étape ne peuvent apparaître autrement que dans les circonstances déjà décrites, et qu'ils ne peuvent interagir avec d'autres signaux. Avec cette hypothèse, on peut prouver le lemme :

Preuve : Rappelons que le nettoyage de la macro-cellule et de son voisinage, c'est-à-dire l'effacement de tous les signaux parasites, a été effectué. On va prouver le lemme pour le côté droit, le côté gauche est similaire.

Grâce à l'initialisation, un signal de succès ne peut être que le résultat d'une collision des trois signaux voulus, au niveau du premier $\#$ de la frontière. Deux signaux s'''_1 et s_2 se rencontrent en cette position si et seulement si les longueurs de la macro-cellule et celle de la voisine sont identiques. Et grâce à l'initialisation, si deux tels signaux existent ils sont le produit du processus décrit plus haut.

Pour le signal s_3 c'est un peu différent, en effet celui-ci est le produit de l'interaction entre un signal de notre cellule et le signal s_2 de la macro-cellule voisine. Ce dernier n'a donc pas été affecté par le nettoyage. Cependant, cette interaction n'a lieu que si le voisinage de droite contient un état de contrôle (sinon s'_1 serait devenu s'''_1). Or les vitesses relatives des différents signaux font que la collision de s_3 avec s_2 et s''_1 ne peut avoir lieu que si le signal s_2 de la zone de droite est issu de cet état de contrôle à l'étape initiale, c'est-à-dire si la synchronisation est bonne.

□

Test de la table de transition, et des types des différents sous-alphabets des cellules du voisinage

Pour chacun des voisins pour lesquels le test précédent à eu un résultat positif, on va tester si la table de transition et les longueurs des différentes zones sont les mêmes que pour notre macro-cellule.

Ceci se fait assez simplement. Lorsque C_2 apparait, il devient immédiatement C_3 et lance pour chaque voisin dont le bit de contrôle correspondant vaut 1 un signal qui démarre le test. Et les bits de contrôle sont remis à 0.

En une étape le signal se trouve sur la première cellule de la table de transition. L'idée est alors de propager en direction du voisin auquel on veut se comparer les états de sa couche principale. La comparaison se fait alors état par état, et un signal d'erreur est levé si une différence non acceptable est détectée.

Plus précisément, on propage les états complets $0_t t$, $1_t t$, ou $B_t t$ pour la table de transition, mais uniquement les types des états cs ou m pour les autres sous-alphabets. Les valeurs propagées sont comparées à celle du voisin concerné.

Par exemple en direction du voisin de droite, les informations vont être propagées de la manière suivante. La cellule la plus à gauche lance un signal, avec sa valeur et son type, vers la droite. Celui-ci active la seconde et continu à se propager vers la droite, la seconde fait de même en activant la troisième, etc... Lorsque le premier signal atteint la première cellule après l'état de contrôle du voisin, il compare les valeurs propagées à celles rencontrées, puis disparait. Le second à son tour est compare aux valeurs suivantes, etc... Une erreur est signalée, si une valeur de la table de transition est différente de celle propagée correspondante, ou si un type propagé ne correspond pas à celui de la cellule recevant l'information. Si aucune erreur n'est détectée, c'est-à-dire si les tables de transition sont identiques, et si les différentes zones sont les mêmes, un signal de succès est envoyé lorsque la dernière cellule de la mémoire a été comparée.

En direction du voisin de gauche, le principe est similaire.

Ces mécanismes sont illustrés par les figures 5.3 et 5.4, pour les voisinages de droite et de gauche, dans le cas d'un succès, et celui d'un échec.

Pour chacun des tests, la cellule dans l'état de contrôle conserve le résultat dans les bits b_g et b_d. Le test en lui-même prend au plus $3l_{\mathcal{A}}$ étapes, auxquelles il faut ajouter $2l_{\mathcal{A}}$ étapes pour le retour du signal avec le résultat..

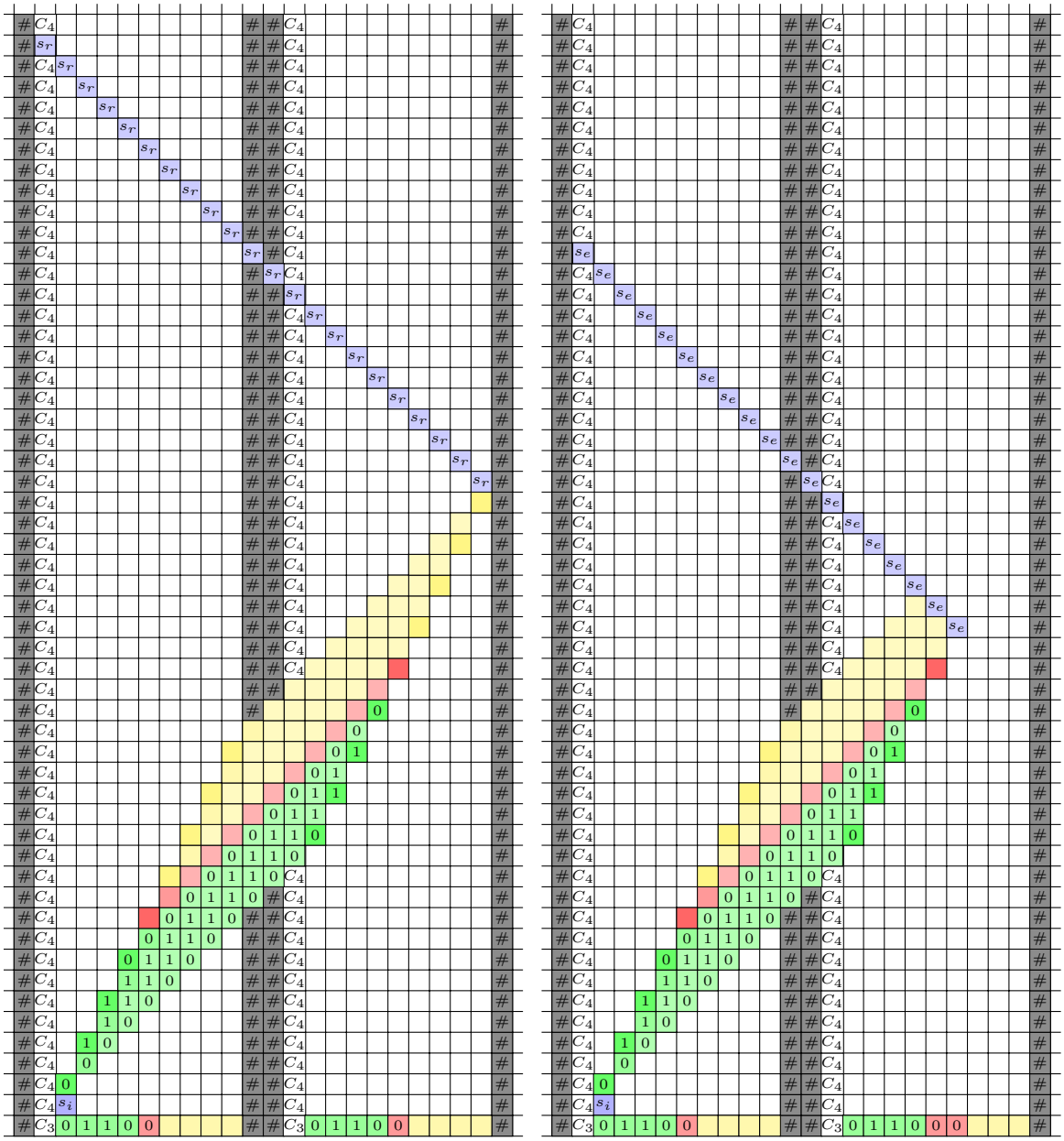
Dans tous les cas, c'est une horloge de durée $5l_{\mathcal{A}}$, initialisée à l'apparition de C_2, qui signifie le moment où C_4 doit apparaître et marque la fin de cette opération. Ceci permet de maintenir la synchronisation que les voisins soient à tester ou non.

Et comme pour l'opération précédente, on souhaite avoir la garantie suivante :

Lemme 5.7. *Exactement $5l_{\mathcal{A}}$ étapes après que C_2 soit apparu à gauche d'un motif calculant issu d'une macro-cellule correcte, l'état de contrôle devient C_4 et le bit de contrôle associé au côté droit (resp. gauche) vaut 1 si et seulement si le voisinage de droite (resp. gauche) est constitué d'un motif calculant correspondant au même automate que la macro-cellule centrale, et de même synchronisation.*

Formellement, pour tout $c \in \Sigma_{\mathcal{U}}^{\mathbb{Z}}$ tel que $c_{[0..l_{\mathcal{A}}]} \in \mathcal{C}_{\mathcal{A}}$,

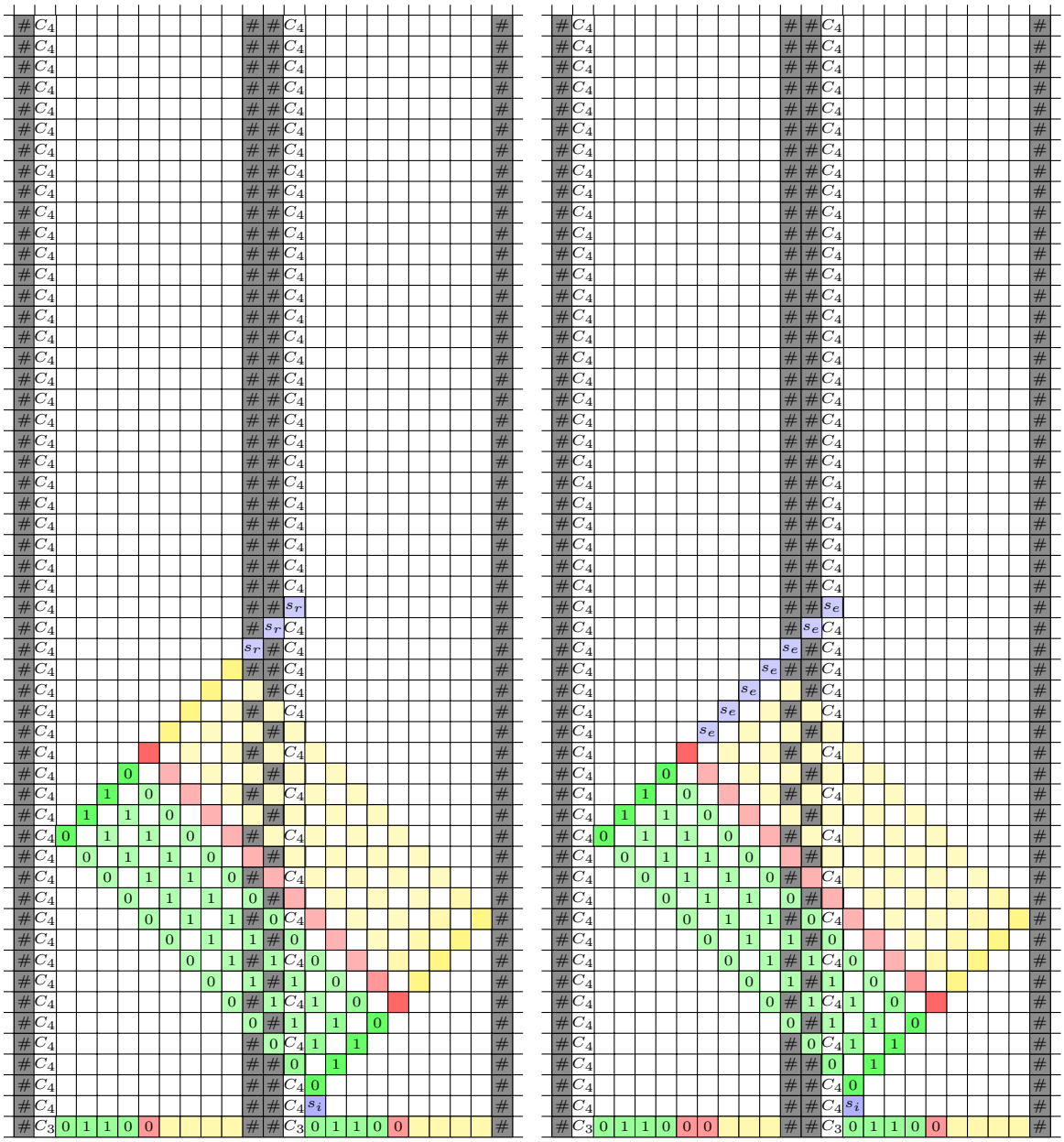
- $(\mathcal{A}^{9l_{\mathcal{A}}}(c))_1 = (C_4, b_g, b_d)$
- $b_g = 1$ *ssi* $c_{[-l_{\mathcal{A}},-1]} \in \mathcal{C}_{\mathcal{A}}$
- $b_d = 1$ *ssi* $c_{[l_{\mathcal{A}},2l_{\mathcal{A}}-1]} \in \mathcal{C}_{\mathcal{A}}$



(a) Exemple de succès

(b) Exemple d'échec

FIGURE 5.3: Test du voisinage de droite.



(a) Exemple de succès (b) Exemple d'échec

FIGURE 5.4: Test du voisinage de gauche.

De nouveau le sens direct nécessite des garanties supplémentaires sur l'ensemble de la règle de $\mathcal{U}$. On suppose donc encore une fois que les signaux qui apparaissent au cours de cette opération, ne sont le résultat d'aucune autre transition que celles spécifiées jusque là. De plus il faut garantir qu'ils n'interagissent pas avec d'autres signaux que ceux que l'on a explicité ici.

Preuve : Il est simple de prouver qu'un voisinage constitué d'une macro-cellule correcte et synchronisée amène le bit de contrôle à avoir la valeur 1 : il s'agit de l'objectif de la construction. Et le nettoyage a assuré qu'il n'y avait pas de signaux parasites.

Cependant, le comportement de l'automate est tel que la présence de l'état de contrôle C_2 implique nécessairement un lancement du test. Et le test est fait de tel manière que si le motif testé n'est pas un motif calculant de même table de transition, une erreur est signalée. Et le signal d'erreur ne peut être modifié avant d'avoir atteint l'état de contrôle de notre macro-cellule. Par conséquent, si le motif testé n'est pas calculant le bit de contrôle vaut 0. Et il ne peut devenir 1 que si le test précédent avait été un succès, ce qui garantit la synchronisation.

□

Il est utile pour la suite de définir l'état détecté d'un voisinage, par une macro-cellule. Cette notion est relative à la macro-cellule centrale considérée, et correspond à la valeur que la macro-cellule va prendre en compte pour appliquer la règle de transition et mettre à jour son état courant.

Définition 42. *La valeur détectée associée au voisinage de droite ou de gauche d'une macro-cellule de $\mathcal{C}_\mathcal{A}$ est :*

- *soit sa valeur associée $v(u)$ si le voisinage considéré est une macro-cellule $u \in \mathcal{C}_\mathcal{A}$.*
- *soit l'état persistant $p_\mathcal{A}$ sinon.*

D'après le lemme précédent, lorsqu'elle entre dans C_4, les bits b_g et b_d d'une macro-cellule permettent de déterminer lequel des deux cas de la définition doit être utilisé.

Écriture dans la mémoire des états détectés des voisins

Lorsque l'état de contrôle C_4 apparaît, il se transforme en C_5 et un mécanisme est initié pour écrire les états détectés des voisins dans la mémoire.

Marquage de la mémoire

Tout d'abord on initialise la zone mémoire. Pour cela, on écrit deux blocs de 0_m de longueur $log(n_\mathcal{A})$, séparés par des B_m. Il suffit donc de calculer les $log(n_\mathcal{A})+1$-ième et $2log(n_\mathcal{A})+1$-ème positions en utilisant la longueur de la zone d'état courant ($log(n_\mathcal{A})$) comme témoin. Ceci peut se faire en envoyant deux signaux vers la droite depuis le début de la zone d'état courant. L'un à vitesse $1/2$ et l'autre à vitesse $1/3$. Lorsque le premier atteint la fin de la zone d'état courant, il passe à la vitesse $1/4$, et les deux signaux se croisent après avoir parcouru la même distance que la longueur de la description de l'état courant. Ceci est illustré par la figure 5.5

Et ceci peut se répéter pour initialiser le deuxième bloc.

Copie des états courants des voisins vers la mémoire

Si le bit de contrôle correspondant à un voisinage vaut 0, alors la valeur détectée de celui-ci, qui est celle de l'état persistant, $0^{log(n_\mathcal{A})}$, est déjà écrit dans la zone de la mémoire correspondant. Il n'y a rien à faire.

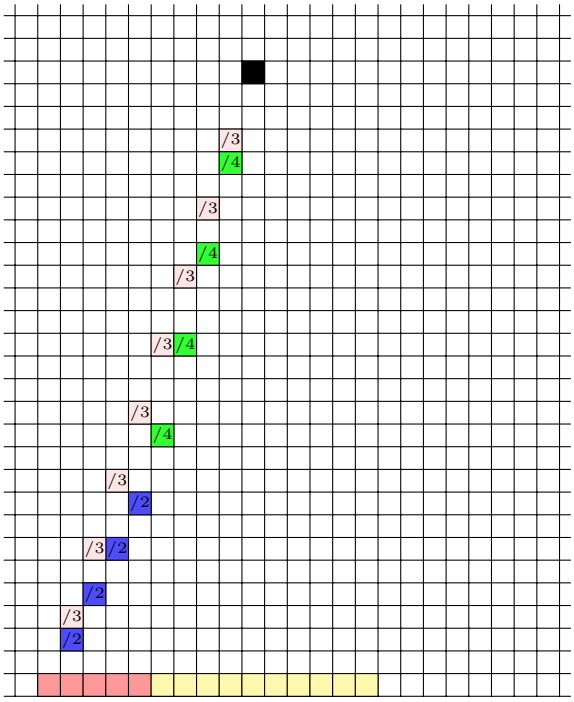


FIGURE 5.5: exemple de signaux s'intersectant pour marquer le double d'une distance.

Sinon, pour chaque voisinage, dont le bit de contrôle correspondant vaut 1, on va chercher l'état courant de la macro-cellule correspondante et le copier dans la zone mémoire.

Le mécanisme utilisé pour amener l'information vers la mémoire est le même que pour propager les états en vue des tests.

Par exemple pour la macro-cellule de droite. Un signal est envoyé, qui lorsqu'il atteint la cellule la plus à gauche de la description de l'état courant de la macro-cellule de droite initialise une propagation de la valeur de cet état vers la zone mémoire de la macro-cellule centrale. Le mécanisme est similaire à celui utilisé pour la comparaison, mais lorsque les valeurs atteignent la cellule de la mémoire correspondante, où doit se placer la valeur du voisin, les cellules de la mémoire prennent les états de valeur correspondante.

Et un mécanisme similaire peut se dérouler simultanément pour chacun des deux voisins. Cette opération dure moins de $6l_{\mathcal{A}}$ étapes.

Lemme 5.8. *$6l_{\mathcal{A}}$ étapes après que C_4 soit apparu dans une macro-cellule ayant des bits de contrôle valides, l'état de contrôle devient C_6 et la mémoire de la macro-cellule contient les valeurs détectées des voisins, c'est-à-dire la valeur associée s'il s'agit d'une macro-cellule du même automate, ou bien l'état $0^{log(n_{\mathcal{A}})}$ qui correspond à l'état persistant de l'automate si ce n'est pas le cas.*

Ici, il faut encore avoir la garantie que l'automate ne fait pas apparaître des signaux perturbateurs. Ce lemme repose également sur le fait que les couches principales des voisins dont on va effectivement chercher à utiliser la valeur associée, ont été protégées du fait qu'il s'agit de macro-cellules correctes.

Mise à jour de l'état courant

À partir de là, il reste à mettre à jour l'état courant de la macro-cellule en transformant la valeur binaire constituée de la juxtaposition des trois états (d'abord l'état courant puis les deux états des voisins) en l'adresse unaire de l'état image dans la table de transition puis en copiant la valeur à cette adresse vers la zone d'état courant. Cette opération est simplifiée par le fait que les transitions de la table de transition sont ordonnées selon l'ordre lexicographique de ces triplets.

Calcul de l'indice de l'image

On commence donc la lecture du mot binaire par les bits de poids fort.

- Le mot est parcouru jusqu'au premier bit de valeur 1, là on envoie un signal qui pose une marque au début de la dernière transition de la table (la marque restera sur un B_{tt}).
- Ensuite, chaque bit suivant est lu et l'opération suivante est effectuée :
  - Si le bit vaut 0, on déplace la marque pour doubler la distance entre la fin de la table et la cellule marquée. Ceci se fait simplement à l'aide de signaux, comme pour marquer les zones de la mémoire.
  - Si le bit vaut 1, on fait de même mais ensuite on décale la marque d'une transition vers la gauche.

À la fin de la lecture du triplet, la transition marquée contient l'image correspondante.

Copie du nouvel état courant

Il reste maintenant à copier la valeur de l'image de la transition dans la zone d'état courant. Ceci se fait en propageant comme précédemment les $log(n_{\mathcal{A}})$ valeurs binaires codant le nouvel état depuis la table de transition vers la zone d'état courant.

Fin du cycle

Le calcul de la position, puis la propagation de l'état se font en un temps inférieur à $l_{\mathcal{A}}^2$. Exactement $l_{\mathcal{A}}^2$ étapes après l'apparition de C_6, l'état de contrôle redevient C_0. Ceci est garanti par une horloge d'une durée $l_{\mathcal{A}}^2$. Mais on ajoute aussi une autre condition.

La durée totale d'une étape de mise à jour est $\tau_{\mathcal{A}} = 13l_{\mathcal{A}} + l_{\mathcal{A}}^2$. Et bien on utilise en plus une horloge globale, lancée lorsque C_0 est apparu et qui doit lever un signal $\tau_{\mathcal{A}}$ étapes plus tard pour que celui-ci puisse réapparaître.

Ceci est essentiel pour les preuves à venir.

Il est maintenant possible de spécifier l'ensemble d'états de l'automate cellulaire universel. Il s'agit donc d'un produit cartésien :

$$\Sigma_{\mathcal{U}} = (M \times S \times C) \cup \{q_p\}$$

Chaque état contient la superposition d'une information issue de chaque couche :

- de la couche principale M :

$$M = \{C_i\}_{i\in\{0..6\}} \times \{0,1\}^2 \cup \{0_{tt}, 1_{tt}, B_{tt}, 0_{cs}, 1_{cs}, 0_m, 1_m, B_m, \#\}$$

  On retrouve les états de contrôle et les bits les accompagnant, les sous-alphabets de chaque zones, ainsi que les délimiteurs.

– de celles des signaux S, qui est elle-même une superposition d'un certain nombre de couches :

$$S = \times_{i \in I} \{s_i, 0\}$$

Les signaux sont ceux introduits dans la construction, et qui n'ont pas toujours été détaillés. Mais l'élément essentiel est que le nombre de types de signaux différents est fini.

– et celle des horloges C :

$$C = \times_{j \in J} \{c_i, 0\}_{i \in I_j}$$

Chaque type d'horloge (c'est-à-dire chaque durée), a sa propre couche de calcul, et de signaux.

Au vue de l'ensemble d'états, il apparaît que l'on n'a spécifié jusque là qu'une petite partie de la règle de transition de notre automate cellulaire universel. Il faut maintenant préciser le reste de la table de transition. Ceci ne peut se faire n'importe comment dans la mesure où on doit continuer à satisfaire les hypothèses qui ont été nécessaires aux preuves des lemmes. Et puis surtout on veut garantir la cohérence de la simulation pour tous les automates $\mathcal{A}$.

Cet objectif repose sur deux propriétés :

– D'une part les zones correspondant à des macro-cellules évoluent correctement. C'est l'objet de la description précédente.
– D'autre part les zones incorrectes, c'est-à-dire celles qui ne correspondent pas à des macro-cellules, ne deviennent pas correctes. C'est pour cette dernière condition qu'il faut être attentif à la manière dont on complète la table de transition.

Pour tout cela on fixe les contraintes suivantes à la règle de transition :

1. la couche principale n'est pas modifiée par autre chose que des transitions déjà définies ;
2. en particulier les # ne disparaissent ou n'apparaissent jamais ;
3. hormis dans les cas déjà spécifiés, aucun signal n'apparaît et deux signaux n'interagissent pas : ils sont transparents les uns pour les autres. Enfin un signal ne traverse jamais de # autrement que par les mécanismes déjà décrits, et disparaît s'il en rencontre un.

Ceci suffit à satisfaire les hypothèses utilisées pour montrer les lemmes.

On peut maintenant décrire précisément la fonction *d'interprétation* associée à chaque automate $\mathcal{A}$. Il s'agit d'une fonction $\Psi_{\mathcal{A}} : \Sigma_{\mathcal{U}}^{\mathbb{Z}} \to \Sigma_{\mathcal{A}}^{\mathbb{Z}}$ continue et surjective. Et celle-ci est associée à une fonction locale $\psi_{\mathcal{A}}$ définie sur les motifs de longueur $l_{\mathcal{A}}$ de $\Sigma_{\mathcal{U}}$. Plus précisément, on a :

– $\forall u \notin \mathcal{C}_{\mathcal{A}}$, $\psi_{\mathcal{A}} = p_{\mathcal{A}}$
– $\forall u \in \mathcal{C}_{\mathcal{A}}$, $\psi_{\mathcal{A}} = v(u)$ avec $v(u)$ la valeur associée à la macro-cellule u définie plus haut.

Tout ceci nous permet d'énoncer et de prouver les deux propriétés suivantes qui synthétisent les résultats de la simulation :

D'une part les macro-cellules évoluent de manière à former périodiquement des macro-cellules pour le même automate, et leur évolution est conforme à la règle de l'automate.

Lemme 5.9. *$\forall c \in \Sigma_{\mathcal{U}}^{\mathbb{Z}}$, $t_0 \in \mathbb{N}$, si $\mathcal{U}^{t_0}(c)_{[0, l_{\mathcal{A}}-1]} \in \mathcal{C}_{\mathcal{A}}$, alors $v = \mathcal{U}^{t_0+\tau_{\mathcal{A}}}(c)_{[0, l_{\mathcal{A}}-1]} \in \mathcal{C}_{\mathcal{A}}$, et $\psi_{\mathcal{A}}(v) = \delta_{\mathcal{A}}(\psi_{\mathcal{A}}(c_{[-l_{\mathcal{A}}, -1]}), \psi_{\mathcal{A}}(c_{[0, l_{\mathcal{A}}-1]}), \psi_{\mathcal{A}}(c_{[l_{\mathcal{A}}, 2 \cdot l_{\mathcal{A}}-1]}))$*

Ce lemme consiste simplement à mettre bout à bout les différents lemmes intermédiaires de la construction.

Et d'autre part, si un mot n'est pas une macro-cellule pour un automate donné, alors son image, après une étape de la simulation par notre automate universel de cet automate, ne sera toujours pas une macro-cellule correcte.

Lemme 5.10. *Si* $\exists t \geq \tau_{\mathcal{A}}$, $c \in \Sigma_{\mathcal{U}}^{\mathbb{Z}}$ *tels que* $u = \mathcal{U}^t(c)_{[0,l_{\mathcal{A}}-1]} \in \mathcal{C}_{\mathcal{A}}$ *alors* $v = \mathcal{U}^{t-\tau_{\mathcal{A}}}(c)_{[0,l_{\mathcal{A}}-1]} \in \mathcal{C}_{\mathcal{A}}$.

Preuve : Ce lemme-ci est aussi une conséquence de la construction, il repose sur la synchronisation globale des macro-cellules. Remarquons tout d'abord que les alphabets des différentes sous-zones étant stables, celles-ci (état de contrôle, table de transition, état courant et mémoire), ainsi que les valeurs de la table de transition (qui n'est jamais modifiée) font de v un motif calculant pour l'automate $\mathcal{A}$. Il reste simplement à vérifier que l'état de contrôle était dans l'état C_0 à l'étape $t - \tau_{\mathcal{A}}$.

Ceci se fait grâce à l'horloge globale et au lemme 5.5. En effet, on sait que l'état C_0 disparaît en une étape, il est donc apparu au temps t. Mais par construction, il ne peut apparaître que si un signal a été levé par l'horloge globale. Et celle-ci, par le lemme 5.5, ne lève un signal que si elle a été initialisée exactement $\tau_{\mathcal{A}}$ étapes plus tôt, c'est-à-dire si l'état de contrôle était dans l'état C_0 à cette étape-là. Ce qui permet de conclure la preuve du lemme.

□

Preuve : [du théorème 5.3] Il reste maintenant à montrer l'universalité de notre automate en utilisant ces deux lemmes.

Soit $\mathcal{A} \in \mathbb{P}_0$, on va montrer que $\mathcal{A} \preccurlyeq_{\trianglelefteq} \mathcal{U}$. Plus précisément, soit $l_{\mathcal{A}}$ la longueur définie dans la construction et $\Psi_{\mathcal{A}}$ la fonction définie plus haut. Cette dernière est locale, par définition à partir de $\psi_{\mathcal{A}}$. Elle est également surjective, car pour n'importe quel $x \in \Sigma_{\mathcal{A}}$, la macro-cellule correcte de $\mathcal{C}_{\mathcal{A}}$, d'état courant x est envoyée sur x par $\Psi_{\mathcal{A}}$; donc n'importe quelle configuration peut être atteinte par concaténation de telles macro-cellules.

De plus on a bien $\Psi_{\mathcal{A}} \circ \mathcal{U}^{\tau_{\mathcal{A}}} = \mathcal{A} \circ \Psi_{\mathcal{A}}$. Pour montrer cela, on peut se contenter de raisonner sur le motif central de taille $l_{\mathcal{A}}$, et conclure grâce à l'invariance par des translations multiples de $l_{\mathcal{A}}$. Pour tout $c \in \Sigma_{\mathcal{U}}$, soit $u = c_{[0,l_{\mathcal{A}}-1]}$ et $v = \mathcal{U}^{\tau_{\mathcal{A}}}(c)_{[0,l_{\mathcal{A}}-1]}$. De deux choses l'une :

– soit $u \in \mathcal{C}_{\mathcal{A}}$ et dans ce cas d'après le lemme 5.9

$$\psi_{\mathcal{A}}(v) = \delta_{\mathcal{A}}(\psi_{\mathcal{A}}(c_{[-l_{\mathcal{A}},-1]}, u, \psi_{\mathcal{A}}(c_{[l_{\mathcal{A}},2\cdot l_{\mathcal{A}}-1]}))$$

– soit $u \notin \mathcal{C}_{\mathcal{A}}$ et dans ce cas d'après le lemme 5.10

$$\psi_{\mathcal{A}}(v) = p_{\mathcal{A}} = \delta_{\mathcal{A}}(\psi_{\mathcal{A}}(c_{[-l_{\mathcal{A}},-1]}, u, \psi_{\mathcal{A}}(c_{[l_{\mathcal{A}},2\cdot l_{\mathcal{A}}-1]}))$$

Ce qui prouve

$$\mathcal{A} \trianglelefteq \mathcal{U}^{<l_{\mathcal{A}},t_{\mathcal{A}},0>}$$

Pour conclure la preuve du théorème, il faut remarquer que grâce à l'état q_p, l'automate $\mathcal{U}$ est bien dans $\mathbb{P}$.

□

Il faut noter également que l'AC ainsi construit est *intrinsèquement universel*. Il est capable de simuler au sens de la simulation *intrinsèque* (ou simulation injective injective) n'importe quel automate cellulaire.

En effet, étant donné un AC $\mathcal{A}$, si l'on considère la fonction $i : \Sigma_{\mathcal{A}} \to \mathcal{C}_{\mathcal{A}} \subseteq \Sigma_{\mathcal{U}}$ qui associe à un état q de l'automate, la macro-cellule qui a comme état interne ce même q, qui n'a aucun signal sur sa couche de signal, et dont la zone mémoire est vide. Cette fonction est injective et compatible avec l'évolution des automates : $\mathcal{A} \sqsubseteq_i \mathcal{U}^{<l_{\mathcal{A}},t_{\mathcal{A}},0>}$.

Perspectives

Pour conclure ce manuscrit, nous présentons brièvement quelques directions de recherche qui nous intéressent particulièrement, en nous limitant cependant à la continuité des travaux qui ont été présentés ici.

En ce qui concerne le comportement typique d'un automate cellulaire fixé, les travaux à mener pour mieux comprendre les ensembles μ-limites restent nombreux.

Pour commencer, le nombre d'AC dont l'ensemble μ-limite est connu explicitement est très limité. Il faut relier ce constat à l'absence d'outils permettant de systématiser ce calcul, même pour des clases très restreintes. Dans les exemples connus, on se contente d'appliquer la définition, et le moins que l'on puisse dire est que celle-ci n'est pas facilement manipulable, même lorsqu'on se restreint au cas de la mesure uniforme. Développer ces exemples, y compris en s'interessant à des mesures plus variées permettrait de préciser la pertinence de la notion, ainsi que sa compréhension. En particulier ceci aiderait à comprendre quels ensembles de mesures apportent des informations pertinentes sur les AC.

À l'inverse, l'ensemble des sous-shifts qui peuvent être obtenus comme ensembles μ-limites d'AC n'est pas parfaitement connu. Ce travail s'articule autour de deux axes : la recherche d'ensembles ne pouvant pas être atteints comme ensembles μ-limite, et la construction, pour certaines classes de sous-shifts, d'AC dont ils constituent les ensembles μ-limites. Des travaux en cours de soumission apportent quelques réponses, généralisant le résultat selon lequel l'ensemble limite de l'automate MAX ne peut être atteint comme ensemble μ-limite, et construisant des AC ayant comme μ-limite de larges classes de sous-shifts. Cependant la caractérisation exacte reste à faire.

Une autre direction naturelle d'extension de ces travaux est la recherche d'un *théorème de Rice* pour les ensembles μ-limites, pour lequel certains des outils introduits pour répondre aux questions précédentes semblent pouvoir être adaptés.

Pour la simulation par facteur, comme nous l'avons vu, la question de l'existence d'un automate cellulaire universel reste ouverte. De nombreuses approches ont été envisagées, que ce soit pour tenter de construire un tel automate, ou pour essayer d'exhiber des contradictions en supposant son existence... Les idées, qui mèneraient peut-être à l'impossibilité de l'existence d'un tel automate sont sans doute à rechercher du côté de la propagation, et du croisement, de l'information. En particulier, est-il possible de construire un AC simulant par facteur tous les produits cartésiens de translations ? De tels résultats permettraient de clarifier la possibilité d'obtenir, ou non, en chaque position, la connaissance maximale d'informations à partir d'une configuration quelconque.

En ce qui concerne l'approche développée ici, la construction d'un AC universel pour une sous-classe aussi large que possible d'AC, le pas naturel suivant pourrait être la recherche d'un AC universel pour les AC avec un point d'équicontinuité. Ils constituent en effet une généralisation des

AC avec possédant un mot persistant. Cependant, les choses sont différentes pour eux : l'initialisation change profondément le comportement et rend impossible jusqu'à présent la réutilisation des idées introduites ici.

Les automates multi-ensemblistes, qui sont apparus au cours de ce travail comme une famille capturant de manière satisfaisante la notion d'isotropie (dans le sens de l'indépendance par rapport aux positions des états dans le voisinage) semblent prometteurs pour cette raison.

L'étude des changements d'échelles, par définition propre à cette classe, est intéressante en elle-même. Un éclairage sur ces transformations pourrait sans doute être apporté par le questionnement autour de l'existence et de la nature de la limite continue de nos rééchelonnés. Nous avons évoqué l'étude menée par M. Pivato autour de *Real Life*. La notion qu'il introduit d'Automates Euclidiens correspond-elle en général à la limite de nos changements d'échelles ? Dans quelle cas cette limite existe-t-elle ? De manière complémentaire, il semble possible, à partir de certaines classes de ces systèmes, continus en espace mais discret en temps, de parcourir le chemin inverse. On obtiendrait alors des objets continus aux comportements, en un sens, approchables par des AC. Ceci pourrait donner un cadre formel au passage du continu au discret, en ne considérant que la discrétisation spatiale, et dans les cas isotropes. Ceci fournirait alors une compréhension utile aux approches de modélisation par AC dans lesquelles cette problématique est omniprésente.

D'autre part, en ce qui concerne l'étude de la densité de propriétés, le cadre formel précisé dans ce manuscrit constitue une base solide qui permet d'envisager de nombreux prolongements. En particulier, l'universalité, les propriétés dynamiques (sensibilité, expansivité) restent à quantifier. Une approche prometteuse pour ces propriétés à long terme serait la généralisation des résultats concernant la propagation d'information. D'abord à un fond périodique, puis surtout à un fond quelconque.

Par ailleurs, en ce qui concerne l'étude de l'universalité parmi des sous-classes, une technique qui n'a pas été utilisée ici, est l'introduction de comparaisons entre les densités dans différentes classes. Celles-ci peuvent reposer sur des transformations, entre ces classes, conservant l'universalité, et ayant des propriétés combinatoires intéressantes. Des classes telles que les AC ayant un état envahissant, semblent ainsi pouvoir être comparées au cas des AC quelconques. Et il en est de même pour les AC avec un état persistant. Étendre cette approche à des cas plus subtils, en quantifiant par exemple la proportion de transitions menant à l'état envahissant ou persistant pourrait mener à des résultats pour le cas des AC quelconques.

De plus, nous nous sommes surtout focalisés pour justifier ces questions de densité, sur les propriétés les plus classiques des automates. L'étude de la densité des propriétés que nous avons considérées par ailleurs, par exemple la μ-nilpotence ou bien même l'universalité par facteur, n'a pas été abordée dans ce travail.

Il reste également à s'intéresser sérieusement aux questions de décidabilité en moyenne. Nous avons montré que la négligeabilité de la nilpotence en fait un problème presque partout décidable. Est-ce un cas isolé ? Est-ce la règle ? En effet, et ceci a été fait pour l'arrêt des machines de Turing ([HM06, Ryb07]), il serait intéressant d'aborder les problèmes de décidabilité sur les AC en lien avec la notion de densité. Existe-t-il des propriétés presque partout indécidables ? Sont-elles répandues ? Tout ceci reste à explorer.

Enfin il reste à comprendre ce phénomène pour le moins intrigant qui fait qu'une restriction syntaxique suffit à rendre presque tout automate intrinsèquement universel. Les AC sont-ils presque tous intrinsèquement universels en général ? Ou bien l'introduction de structure locale est-elle

déterminante ? Y a-t-il alors, au-delà de la recherche de sous-shifts de simulation indépendants, un point commun caché entre toutes nos constructions ? Dans les deux cas des réponses à ces questions fourniraient des informations qui aideraient à mieux comprendre la structure de l'ensemble des AC.

Bibliographie personnelle

[BDS10] L. Boyer, M. Delacourt, and M. Sablik. Construction of μ-limit sets. In *JAC*, 2010.

[BPT06] L. Boyer, V. Poupet, and G. Theyssier. On the complexity of limit sets of cellular automata associated with probability measures. In R. Kralovic and P. Urzyczyn, editors, *MFCS*, volume 4162 of *Lecture Notes in Computer Science*, pages 190–201. Springer, 2006.

[BT09] L. Boyer and G. Theyssier. On local symmetries and universality in cellular automata. In S. Albers and JY. Marion, editors, *STACS*, volume 3 of *LIPIcs*, pages 195–206. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, 2009.

[BT10] L. Boyer and G. Theyssier. On factor universality in symbolic spaces. In P. Hlinený and A.ín Kucera, editors, *MFCS*, volume 6281 of *Lecture Notes in Computer Science*, pages 209–220. Springer, 2010.

Bibliographie

[AC87] J. Albert and K. Culik. A simple universal cellular automaton and its one-way and totalistic version. *Complex Systems*, 1 :1–16, 1987.

[AP72] S. Amoroso and Y.N. Patt. Decision Procedures for Surjectivity and Injectivity of Parallel Maps for Tesselation Structures. *Journal of Computer and System Sciences*, 6 :448–464, 1972.

[BCG82] E. R. Berlekamp, J. H. Conway, and R. K. Guy. *Winning Ways for your Mathematical Plays*, volume 2. Academic Press, 1982. chapter 25.

[BDS10] L. Boyer, M. Delacourt, and M. Sablik. Construction of μ-limit sets. In *JAC*, 2010.

[BPT06] L. Boyer, V. Poupet, and G. Theyssier. On the complexity of limit sets of cellular automata associated with probability measures. In R. Kralovic and P. Urzyczyn, editors, *MFCS*, volume 4162 of *Lecture Notes in Computer Science*, pages 190–201. Springer, 2006.

[BT00] F. Blanchard and P. Tisseur. Some properties of cellular automata with equicontinuity points, 2000.

[BT09] L. Boyer and G. Theyssier. On local symmetries and universality in cellular automata. In S. Albers and JY. Marion, editors, *STACS*, volume 3 of *LIPIcs*, pages 195–206. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, 2009.

[BT10] L. Boyer and G. Theyssier. On factor universality in symbolic spaces. In P. Hlinený and A.ín Kucera, editors, *MFCS*, volume 6281 of *Lecture Notes in Computer Science*, pages 209–220. Springer, 2010.

[CD98] B. Chopard and M. Droz. *Cellular automata modeling of physical systems*. Collection Aléa-Saclay : Monographs and Texts in Statistical Physics. Cambridge University Press, Cambridge, 1998.

[CFMM97] G. Cattaneo, E. Formenti, L. Margara, and J. Mazoyer. A new shift-invariant metric on $S^{\mathbb{Z}}$ inducing a non-trivial topology. In I. Privara and P. Rusika, editors, *Mathematical Foundations of Computer Science (MFCS'97)*, volume 1295 of *Lecture Notes in Computer Science*. Springer-Verlag, August 1997.

[CFMM00] G. Cattaneo, E. Formenti, G. Manzini, and L. Margara. Ergodicity, transitivity, and regularity for linear cellular automata over $\mathsf{z_m}$. *Theor. Comput. Sci.*, 233(1-2) :147–164, 2000.

[CPY89] K. Culik, J. Pachl, and S. Yu. On the limit sets of cellular automata. *SIAM Journal on Computing*, 18(4) :831–842, August 1989.

[CS06] C. S. Calude and M. Stay. Most programs stop quickly or never halt. *CoRR*, abs/cs/0610153, 2006.

[CY88] K. Culik and S. Yu. Undecidability of ca classification schemes. *Complex Syst.*, 2(2) :177–190, 1988.

[dBE46] N.G. de Bruijn and P. Erdös. On a combinatorial problem. *Indagationes Math*, 10, 1946.

[DMOT10a] M. Delorme, J. Mazoyer, N. Ollinger, and G. Theyssier. Bulking i : an abstract theory of bulking. *CoRR*, 2010.

[DMOT10b] M. Delorme, J. Mazoyer, N. Ollinger, and G. Theyssier. Bulking ii : Classifications of cellular automata. *CoRR*, abs/1001.5471, 2010.

[DRT04] C. Dürr, I. Rapaport, and G. Theyssier. Cellular automata and communication complexity. *Theoretical Computer Science*, 322, 2004.

[Dur94] B. Durand. *Automates cellulaires : réversibilité et complexité*. PhD thesis, École Normale Supérieure de Lyon, février 1994.

[Eva01] K. M. Evans. Larger than life : Digital creatures in a family of two-dimensional cellular automata. In *Discrete Mathematics and Theoretical Computer Science Proceedings, Vol. AA (Discrete Models : Combinatorics, Computation, and Geometry*, pages 177–192, 2001.

[Eva05] K. M. Evans. Is bosco's rule universal ? In Maurice Margenstern, editor, *Machines, Computations, and Universality*, volume 3354 of *Lecture Notes in Computer Science*, pages 188–199. Springer Berlin / Heidelberg, 2005.

[FO90] P. Flajolet and A. M. Odlyzko. Random mapping statistics. In *In Advances in Cryptology*, pages 329–354. Springer Verlag, 1990.

[FS09] P. Flajolet and R. Sedgewick. *Analytic Combinatorics*. Cambridge University Press, 2009.

[Gil87] R. H. Gilman. Classes of linear automata. *Ergodic Theory and Dynamical Systems*, 7 :105–118, 1987.

[Gor87] D. Gordon. On the computational power of totalistic cellular automata. *Mathematical Systems Theory*, 20(1) :43–52, 1987.

[GR10] P. Guillon and G. Richard. Revisiting the rice theorem of cellular automata. *CoRR*, abs/1001.0253, 2010.

[Hed69] G. A. Hedlund. Endomorphisms and Automorphisms of the Shift Dynamical Systems. *Mathematical Systems Theory*, 3(4) :320–375, 1969.

[HM06] J. D. Hamkins and A. Miasnikov. The halting problem is decidable on a set of asymptotic probability one. *Notre Dame J. Formal Logic*, 47, 2006.

[Hur87] L. P. Hurd. Formal language characterizations of cellular automaton limit sets. *Complex Systems*, 1 :69–80, 1987.

[IÔN83] M. Itô, N. Ôsato, and M. Nasu. Linear cellular automata over zm. *Journal of Computer and System Sciences*, 27(1) :125 – 140, 1983.

[JDF01] JC.Dubacq, B. Durand, and E. Formenti. Kolmogorov complexity and cellular automata classification. *Theor. Comput. Sci.*, 259(1-2) :271–285, 2001.

[Kar92] J. Kari. The Nilpotency Problem of One-dimensional Cellular Automata. *SIAM Journal on Computing*, 21 :571–586, 1992.

[Kar94a] J. Kari. Reversibility and Surjectivity Problems of Cellular Automata. *Journal of Computer and System Sciences*, 48(1) :149–182, 1994.

[Kar94b] J. Kari. Rice's theorem for the limit sets of cellular automata. *Theoretical Computer Science*, 127 :229–254, 1994.

[KM00] P. Kůrka and A. Maass. Limit Sets of Cellular Automata Associated to Probability Measures. *Journal of Statistical Physics*, 100(5-6) :1031–1047, 2000.

[Kůrka97] P. Kůrka. Languages, equicontinuity and attractors in cellular automata. *Ergodic Theory and Dynamical Systems*, 17 :417–433, 1997.

[LM95] D. Lind and B. Marcus. *An introduction to symbolic dynamics and coding*. Cambridge University Press, 1995.

[LV93] M. Li and P. M. B. Vitányi. *An Introduction to Kolmogorov Complexity and its Applications*. Springer, 1993.

[MK76] A. Maruoka and M. Kimura. Condition for Injectivity of Global Maps for Tessellation Automata. *Information and Control*, 32 :158–162, 1976.

[Moo62] E. F. Moore. Machine Models of Self-Reproduction. In *Proceedings of Symposia in Applied Mathematics*, volume 14, pages 17–33. American Mathematical Society, 1962.

[Moo64] E. F. Moore. The firing squad synchronization problem. In E. F. Moore, editor, *Sequential Machines - Selected Papers*, pages 213–214. Addison-Wesley, 1964.

[MR98a] J. Mazoyer and I. Rapaport. Additive cellular automata over Z_p and the bottom of $(CA, \leq)$. In *Mathematical Foundations of Computer Science*, pages 834–843. Lecture Notes in Computer Science, 1998.

[MR98b] J. Mazoyer and I. Rapaport. Inducing an Order on Cellular Automata by a Grouping Operation. In *Symposium on Theoretical Aspects of Computer Science*. Lecture Notes in Computer Science, 1998.

[MS03] A. Maass and S.Martínez. Evolution of probability measures by cellular automata on algebraic topological markov chains. *Biol Res*, 36(1) :113–8, 2003.

[Myh63] J. Myhill. The Converse of Moore's Garden-of-Eden Theorem. In *Proceedings of the American Mathematical Society*, volume 14, pages 658–686. American Mathematical Society, 1963.

[Oll02a] N. Ollinger. *Automates Cellulaires : structures*. PhD thesis, École Normale Supérieure de Lyon, décembre 2002.

[Oll02b] N. Ollinger. The quest for small universal cellular automata. In *International Colloquium on Automata, Languages and Programming*, pages 318–330. Lecture Notes in Computer Science, 2002.

[Oll03] N. Ollinger. The intrinsic universality problem of one-dimensional cellular automata. In *Symposium on Theoretical Aspects of Computer Science*, pages 632–641. Lecture Notes in Computer Science, 2003.

[Piv07] M. Pivato. Reallife : The continuum limit of larger than life cellular automata. *Theor. Comput. Sci.*, 372(1) :46–68, 2007.

[PY02] M. Pivato and R. Yassawi. Limit measures for affine cellular automata. *Ergodic Theory & Dynamical Systems*, 22(4) :1269–1287, 2002.

[Rap97] I. Rapaport. *Inducing an Order on Cellular Automata by a Grouping Operation*. PhD thesis, École Normale Supérieure de Lyon, juin 1997.

[Rók94] Z. Róka. *Cellular automata on Cayley graphs*. PhD thesis, École Normale Supérieure de Lyon, 1994.

[Rók99] Z. Róka. Simulations between cellular automata on cayley graphs. *Theoretical Computer Science*, 225, 1999.

[Rók00] Z. Róka. The firing squad synchronization problem on cayley graphs. *Theor. Comput. Sci.*, 244(1-2) :243–256, 2000.

[Ryb07] A. Rybalov. Note : On the strongly generic undecidability of the halting problem. *Theor. Comput. Sci.*, 377(1-3) :268–270, 2007.

[Sab08] M. Sablik. Directional dynamics for cellular automata : A sensitivity to initial condition approach. *Theor. Comput. Sci.*, 400(1-3) :1–18, 2008.

[ST08] M. Sablik and G. Theyssier. Topological dynamics of 2d cellular automata. In A. Beckmann, C. Dimitracopoulos, and B. Löwe, editors, *CiE*, volume 5028 of *Lecture Notes in Computer Science*, pages 523–532. Springer, 2008.

[Sut90] K. Sutner. Classifying circular cellular automata. *Physica D*, 45 :386, 1990.

[The04] G. Theyssier. Captive cellular automata. In *Mathematical Foundations of Computer Science*, pages 427–438. Lecture Notes in Computer Science, 2004.

[The05] G. Theyssier. *Automates Cellulaires : un modèle de complexité*. PhD thesis, École Normale Supérieure de Lyon, décembre 2005.

[Wol84] S. Wolfram. Universality and complexity in cellular automata. *Physica D*, 10 :1–35, 1984.