

Université Paris 7 - Denis Diderot - UFR Sciences du vivant

Mémoire de recherche en vue de l'obtention du diplôme de Master de l'Université Paris 7,
Spécialité Logique, philosophie, histoire et sociologie des sciences - Sciences de la cognition
et des systèmes complexes (LoPHiSS-SC2), Finalité Recherche

EFFECTING A MORE OPEN-ENDED EVOLUTION VIA HETEROGENEOUS CELLULAR AUTOMATA

DAVID MEDERNACH

Soutenu à Paris le 22 juin, devant un jury composé de :

René Doursat : Directeur de mémoire

Taras Kowaliw : Co-directeur

Nicolas Bredeche : Examinateur

Evelyne Lutton : Examinateur

Remerciements

Je tiens tout d'abord à remercier M. René Doursat d'avoir accepté de diriger ce mémoire ainsi que pour ses précieuses recommandations, ses nombreuses corrections ainsi que l'intérêt pour le domaine qu'il a suscité au LoPHiSS-SC2 et à L'ECAL11. De même je tiens à remercier tout particulièrement Taras Kowaliw pour avoir accepté de codiriger ce travail, pour sa disponibilité de tous les instants, ses nombreux conseils avisés, sa passion pour le sujet et sa grande gentillesse. Leurs suggestions m'ont permis de surmonter de nombreuses difficultés et sans eux je n'aurai pas pu mener à bien ce travail.

Un grand merci également à Nicolas Bredeche et Evelyne Lutton d'avoir bien voulu faire partie du jury. Mes remerciements vont aussi à l'Institut des Systèmes Complexes, Paris Île-de-France (ISC-PIF) pour m'avoir accueilli en son sein et à mes amis (Alexandre Conninx et Simon Carrignon) pour leurs discussions toujours enrichissantes. Merci aussi à ma famille (Caroline Medernach) pour ses corrections orthographiques.

J'exprime aussi ma gratitude à toute l'équipe enseignante du LoPHiSS-SC2 ainsi qu'à Madame Françoise Contamina pour avoir rendu possible ces deux années de Master extrêmement enrichissante.

Enfin merci à l'ensemble des chercheurs travaillant dans le domaine de la vie artificielle pour leurs passionnant travaux.

Table des matières

1	Introduction	1
1.1	Évolution et sélection naturelle	1
1.2	Vie artificielle	2
1.3	Évolution “ouverte” (“open ended”)	2
1.4	Notre approche	2
2	État de l’art	3
2.1	Dans la biologie	3
2.1.1	Débat sur la sélection naturelle	3
2.1.2	Caractéristiques de l’évolution ouverte	4
2.2	La vie artificielle	4
2.2.1	Les automates cellulaires	4
2.2.2	La constitution de la vie artificielle en domaine de recherche	6
2.2.3	Les enjeux de la vie artificielle	7
2.2.4	L’autopoïèse	9
2.3	Évolution artificielle	10
2.3.1	Fonction de “fitness” et Algorithme évolutionniste	10
2.3.2	Évolution ouverte	11
2.4	Simulations basées sur des automates cellulaires	15
2.4.1	Automates évolutifs	15
2.4.2	Automates cellulaires et structures autoréplicatrices	20
3	Comment simuler une évolution ouverte	24
3.1	Sans fonction de fitness explicite	24
3.2	Un environnement complexe	25

3.3	Interactions locales	25
3.4	Nombre important d'agents/de réplicateurs	25
3.5	Possibilité de comportements complexes	26
3.6	Bilan	26
4	Le modèle	28
4.1	Présentation rapide	28
4.2	Le concept détaillé	28
4.2.1	Les cellules normales / “vivantes”	30
4.2.2	Cellules en décomposition / Decay	32
4.2.3	Cellules “Quiescent”	32
4.2.4	Notation	33
4.3	Paramètres initiaux	33
4.3.1	Temps de décomposition	34
4.3.2	Age limite	34
4.3.3	Nombre d'états possibles	35
4.3.4	Taux de propagation	35
4.3.5	Mutations	35
4.4	Différents moyens de représenter les génotypes des cellules	36
4.4.1	Tableaux	36
4.4.2	Arbres	36
4.4.3	Représentation issue de la “Linear genetic programing” (LGP)	36
4.5	Les éléments rejetés à l'issues de la phase exploratoire	40
4.5.1	Bonus	40
4.5.2	Maturité	41
4.6	Analogie biologique	41
5	Analyses des résultats	42
5.1	Synthèse des résultats	42
5.1.1	Paramètres importants	43
5.1.2	Émergence de structures à partir de génotypes aléatoires	45
5.1.3	Différencier les phénotypes	47
5.1.4	Interactions locales entre phénotypes	47
5.1.5	Comportement typique du vivant	47

5.2	Compétition entre les phénotypes	55
5.3	Génération continue de nouveaux phénotypes	58
5.3.1	Système de mesure	58
5.3.2	Résultats	62
6	Potentiels développements futurs	71
6.1	Analyse des génotypes	71
6.2	Rajouter de l'échange génétique	71
6.3	Comparer l'évolutivité des GP-CA aux autres méthodes pour coder le génotype des automates	71
6.4	Modifier le voisinage	72
6.5	Des types de cellules spécifiques	72
6.6	Explorer le développement	73
6.7	Des changements environnementaux	73
6.8	Questionner la nature de l'évolution	73
6.9	Structures autoréplicatrices	74
7	Conclusion	75
8	Annexes	79

Résumé

The purpose of this work is to study open-ended evolution via heterogeneous cellular automata. We begin with a definition of open ended evolution as it exists in the natural world, and a review of previous attempts to reproduce it in artificial life. To achieve our goal, we present a model of heterogeneous cellular automata with local transition rules encoded in the cells' genotypes. These genotypes are a modified form of linear genetic programming. Our heterogeneous cellular automata also use two additional novel concepts : life expectancy and decay. We explore several types of statistical measures of the open endedness of long-term evolution for cellular automata, and choose two which correspond to our intuitions regarding phenotypic diversity. Via these measures and qualitative analysis, we show that our model choices lead to greater long-term phenotypic variability than several controls. These controls include two standard systems known for their emergent behaviours : the Game of Life (well known for its emergent behavior) and classical 2D cellular automata, along with several variants of our model designed to validate our design choices.

Chapitre 1

Introduction

1.1 Évolution et sélection naturelle

Depuis Darwin, on considère la sélection naturelle comme le mécanisme principal de l'évolution. Elle est caractérisée par trois éléments :

- L'hérédité : la présence de réplicateurs capables de se multiplier en créant de nouvelles entités partageant un nombre important de caractéristiques communes.
- Les variations : l'apparition régulière de variations dans les caractéristiques transmissibles des réplicateurs, en particulier lors de leur réplication.
- La sélection : un environnement hostile et compétitif ne permettant pas la survie de tous les réplicateurs et dans lequel les caractéristiques transmissibles des réplicateurs sont susceptibles d'influer sur leurs chances de survie.

Il en résulte un processus adaptatif. Il est à noter que l'évolution telle qu'elle existe dans le monde du vivant ne se limite pas à ces trois éléments (caractérisant la sélection naturelle).

Daniel W. McShea [19] propose d'ajouter une règle concernant l'évolution de la complexité. Il donne pour cela une définition particulière de la complexité comme étant le nombre de types de composants à un certain niveau d'organisation (par exemple l'organisation interne de la cellule est un certain niveau d'organisation tandis que l'organisation des cellules entre elles en est un autre). Cet auteur propose donc d'ajouter ce qu'il appelle la ZEFL (Zero Evolution Force Law) qui implique, qu'en l'absence de contraintes s'exerçant sur la complexité ou la diversité, celles-ci auront tendance à augmenter avec le temps dans un système évolutif :

In any evolutionary system in which there is variation and heredity, in absence of natural selection, other forces, and constraints acting on diversity or complexity, diversity and complexity

will increase on average.[19]

1.2 Vie artificielle

Le domaine de la vie artificielle est un champ de recherche apparu à la fin des années 1980 et consistant à essayer de modéliser, simuler voire recréer la vie sur un support différent. Il inclut des domaines aussi divers que la robotique, la biologie de synthèse, la modélisation des systèmes vivants et nombre de simulations évolutionnistes. C'est principalement à cet aspect de la vie artificielle que nous nous intéresserons ici, et plus particulièrement aux tentatives de recréer une sélection naturelle non dirigée avec des caractéristiques proches de celles observées dans le monde du vivant, plutôt que de réutiliser le principe de la sélection naturelle comme système d'optimisation. De telles simulations permettront à terme, nous l'espérons, de mieux comprendre les caractéristiques de l'évolution telle qu'on l'observe en biologie.

1.3 Évolution “ouverte” (“open ended”)

Une autre caractéristique de l'évolution en science naturelle est sa nature ouverte, sans fin, ou “open ended”. Cela recouvre essentiellement deux caractéristiques :

- Créer perpétuellement de la nouveauté
- Ne pas cycler et recréer plusieurs fois des entités identiques (Ou génotypiquement très proches sans être pour autant apparentées à des formes intermédiaires)

Nous aurons l'occasion de passer en revue quelques tentatives de recréer une évolution de ce type déjà effectuées en vie artificielle.

1.4 Notre approche

Notre approche consiste ici, pour simuler une forme d'évolution ouverte, à utiliser un outil très répandu en vie artificielle : les automates cellulaires, mais en utilisant des règles de transition locales (automates cellulaires hétérogènes) et variables. Ces règles de transitions, potentiellement différentes pour chaque cellule, deviennent alors l'analogie, non plus des règles physiques du monde comme pour les automates cellulaires classiques homogènes¹, mais des équivalents des génotypes des organismes vivant dans un écosystème².

1. Les équivalent éventuels des génotypes (par exemple dans le réplicateur universel de John Von Neumann) étant dans ce cas les états d'une ou plusieurs cellules.

2. Les états d'une ou plusieurs cellules étant ici le phénotype (développement) du génotype considéré

Chapitre 2

État de l'art

2.1 Dans la biologie

2.1.1 Débat sur la sélection naturelle

S'il est généralement admis que le moteur principal de l'évolution est la sélection naturelle, l'importance d'autres mécanismes tels que la dérive génétique ou les facteurs épigénétiques (qui peuvent, eux aussi d'ailleurs, s'expliquer dans le cadre de la sélection naturelle) reste à évaluer. Il existe aussi plusieurs façons de considérer la sélection naturelle : Richard Dawkins va, par exemple, proposer une vision de l'évolution centrée essentiellement sur les gènes [6] (où ils sont les réplicateurs sur lesquels s'exerce la pression de sélection naturelle), tandis que Stephen Jay Gould va insister sur des phénomènes ayant lieu au niveau des phénotypes (tel que l'exaptation Figure 2.1) ou au niveau des dynamiques qui existent dans l'évolution (équilibres ponctués [8]).

FIGURE 2.1: Rôle de l'exaptation - Stephen Jay Gould [12]

Process	Character	Usage
Natural selection shapes the character for a current use—adaptation	adaptation	function
A character, previously shaped by natural selection for a particular function (an adaptation), is coopted for a new use—cooption	exaptation	aptation
A character whose origin cannot be ascribed to the direct action of natural selection (a nonadaptation), is coopted for a current use—cooption		effect

2.1.2 Caractéristiques de l'évolution ouverte

Créer continuellement de la nouveauté L'évolution telle que nous l'observons dans le monde des sciences naturelles est un processus d'adaptation qui, s'il est susceptible de subir des périodes d'accélération brutale et de relative stagnation (cf. équilibres ponctués de Stephen Jay Gould), ne s'arrête jamais. On voit donc indéfiniment de nouveaux génotypes et phénotypes apparaître, survivre et remplacer les formes précédentes.

Pas de répétitions S'il existe dans le monde du vivant des cycles de types "proies-prédateurs" qui vont amener différentes populations à croître, puis décroître cycliquement, l'évolution ne se répète pas et ne mène donc pas à la création d'espèces identiques (ne crée pas des entités similaires génétiquement mais non apparentées). Ce qui n'empêche pas certaines convergences phénotypiques entre des espèces non apparentées, mais vivant dans le même milieu : un dauphin est plus proche génétiquement d'un chien que d'un requin, néanmoins sa forme fuselée et adaptée au déplacement aquatique rappelle davantage celle d'un requin que celle d'un chien.

2.2 La vie artificielle

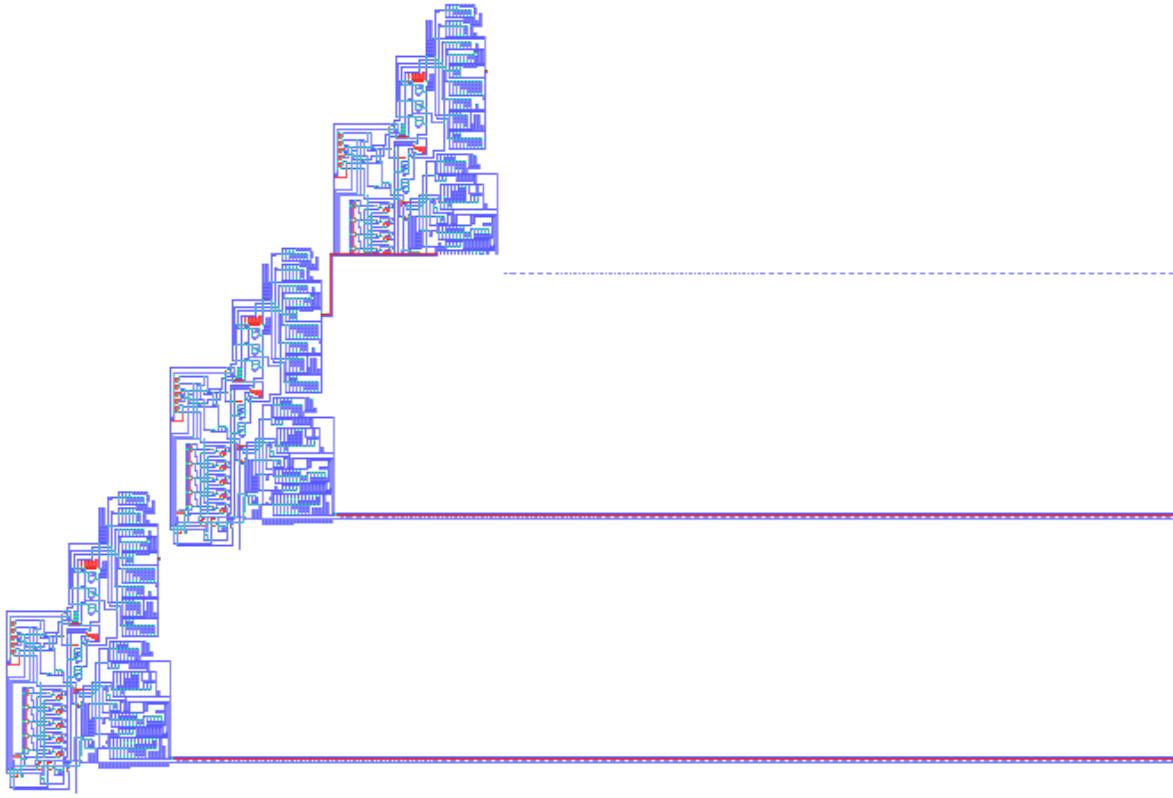
2.2.1 Les automates cellulaires

Historique Bien avant qu'apparaisse l'appellation VIE ARTIFICIELLE, dans les années 40 John von Neumann (l'un des pères de l'informatique avec Alan Turing) inventa le principe des automates cellulaires. Il cherchait à modéliser des systèmes dynamiques et à réaliser des machines capables de créer une copie améliorée d'elles-même et cela l'amena naturellement à essayer de concevoir un système artificiel autoréplicateur. Ses travaux n'aboutissant pas, il se résolut à n'étudier que l'aspect théorique de la chose et, suivant les conseils de Stanisław Ulam, s'intéressa aux automates cellulaires.

Principe Le principe des automates cellulaires consiste en une matrice ou grille, généralement à deux dimensions, comportant des cases (cellules) pouvant être dans différents états (le plus souvent représentés par des chiffres ou différentes couleurs). Le changement d'état se fait de manière discrète, selon une fonction de transition prédéfinie (spécifique à chaque automate cellulaire) et en fonction de l'état de la cellule ainsi que de ceux des cellules "voisines". La définition du voisinage dépend de l'automate cellulaire dans le cas d'un automate à deux dimensions, c'est le plus souvent le "voisinage de von Neumann"¹ $VN(p)$ ou le "voisinage

1. les quatre cellules adjacentes à la cellule considérée donc sans compter les diagonales

FIGURE 2.2: Implémentation d’Umberto Pesavento de l’automate de von Neumann (capture d’écran de wikipedia du logiciel golly disponible sur <http://golly.sourceforge.net>)



de Moore”² $VM(p)$.

L’exemple le plus connu est le “jeu de la vie” de John Horton Conway où il n’existe que deux états, appelés “vivant” et “mort”, où l’on utilise le voisinage de Moore, et où les règles sont : une case vivante “meurt” si elle a moins de deux voisines vivantes ou plus de trois et une case morte devient vivante si elle possède exactement trois voisines vivantes. Le jeu de la vie est un exemple d’automate cellulaire intéressant car on peut y voir émerger des formes complexes, dont certaines semblent se déplacer sur la grille, à partir d’une distribution aléatoire.

Intérêt des automates cellulaires L’objectif initial de von Neumann, réaliser une machine autoréplicatrice, reste très présent dans ses travaux sur les automates cellulaires (ces automates contiennent par exemple une “tête d’écriture”). Dans les années 40, alors que le rôle de l’ADN n’était pas encore parfaitement connu (avant la découverte de sa structure en 1953), von Neumann créa une structure autoréplicatrice, son constructeur universel, contenant au sein de sa structure une représentation de lui-même, sorte “d’enregistrement”

2. les huit cases adjacentes donc en comptant les diagonales

permettant à la structure de se répliquer, et éventuellement d'introduire des variations transmissibles lors de cette réPLICATION, ce qui a été étudié plus récemment par A. R. Yinsua [35]. Cet enregistrement, même s'il ressemble davantage à un plan de montage utilisé par un premier constructeur universel pour construire une seconde entité, fait penser au rôle que joue l'ADN en tant que support d'information pour le vivant. La première implémentation informatique du réPLICATEUR de von Neumann ne fut réalisée qu'en 1995 par Umberto Pesavento [22] (Figure 2.2).

Les automates cellulaires présentent plusieurs intérêts qui en font des outils très utiles de la vie artificielle :

- Comme nous l'avons vu précédemment (travaux de John von Neumann) et comme ce fut par la suite, entre autre, développé par les travaux sur les boucles autoréPLICATRICES de Christopher Langton et de John Byl, ils permettent la simulation d'entités autoréPLICATRICES.
- Comme nous le verrons par la suite, ils permettent aussi de simuler des systèmes autopoïétiques (ce qu'a fait Francesco Varela).
- Les automates cellulaires peuvent être Turing universel. C'est ce qu'a prouvé Matthew Cook[5] en étudiant l'automate à deux dimensions, dit de la règle 110. Par la suite et plus récemment il fut prouvé en 1998 par B. Durand et Z. Róka[7], que le jeu de la vie est lui aussi Turing universel. De même, dans les automates beaucoup plus complexes, l'automate cellulaire créé par John von Neumann est lui aussi Turing complet.
- Par ailleurs les automates cellulaires se sont révélés être très efficaces pour modéliser des systèmes dynamiques biologiques comme par exemple des réseaux de vascularisation [16] ou un ensemble d'autres éléments biologiques (croissance de micro-organismes, formation de motifs, colonies de termites, etc.)
- Les automates cellulaires sont aussi utilisés dans la modélisation de systèmes physiques (comme par exemple la modélisation de fluides, en se basant sur les équations de Navier et Stokes)
- Enfin les automates cellulaires peuvent être utilisés dans des tâches de développement artificiel[16]. Le développement artificiel consiste à concevoir des systèmes capables non pas de résoudre directement un problème précis, mais de créer un processus qui aboutira à terme à une résolution du problème en fonction de l'environnement et des contraintes contextuelles. Cette approche présente l'avantage de proposer des solutions davantage évolutives et extensibles. Par ailleurs, modéliser des systèmes se développant est aussi une étape importante dans la modélisation du vivant.

2.2.2 La constitution de la vie artificielle en domaine de recherche

Ce champ de recherche a été nommé ainsi en 1987 à la suite d'un article fondateur de Christopher Langton [17] et d'une conférence, ALIFE I, qu'il organisa à Los Alamos. Les actes de la conférence ne furent publiés

que deux ans plus tard et sont considérés, ainsi qu'un article de Langton, qu'il y ajouta et qu'il nomma ARTIFICIAL LIFE, comme les textes fondateurs de la vie artificielle. La prochaine conférence Alife, ALIFE XIII aura lieu en août 2012 au Michigan.

Christopher Langton est le créateur d'un automate autoréplicateur (Figure 2.3) significativement plus petit que ceux de John von Neumann, inspiré par les travaux de ce dernier et de Tedd Codd³. Christopher Langton a aussi travaillé sur le contexte favorable à l'apparition de la vie et a développé l'idée, se basant sur les travaux de Stephen Wolfram, que les systèmes complexes (et donc la vie) ne peuvent apparaître que dans une transition de phase critique entre l'ordre et le chaos. Le domaine apparu alors que l'Intelligence artificielle était en pleine crise. La vie artificielle peut être vue comme l'une des réponses, avec le connexionnisme, à l'échec relatif de l'intelligence artificielle (échec dans les tentatives de reproduire une intelligence similaire à l'intelligence humaine). L'artificial life postule qu'une telle intelligence ne peut se concevoir dissociée d'un corps et d'une organisation du "cerveau" adéquate. Jean-Claude Heudin résume ainsi [14] :

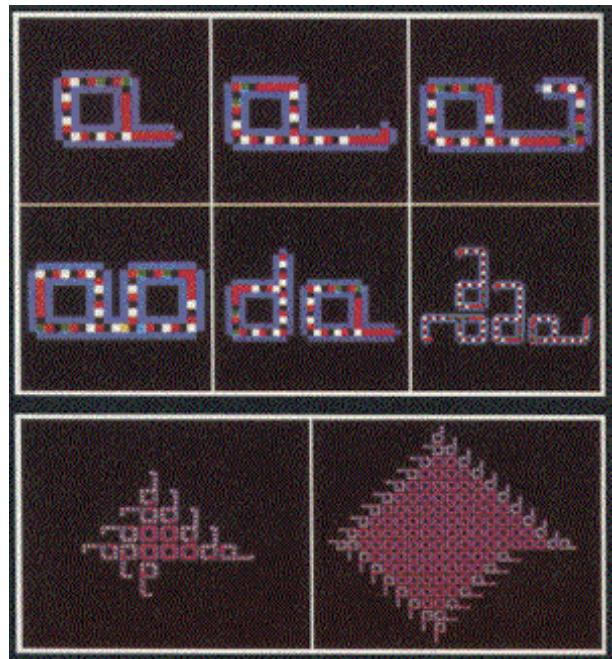
La construction d'un "corps" que l'on pourrait dire vivant doit donc précéder celle d'un être que l'on pourrait dire pensant.

2.2.3 Les enjeux de la vie artificielle

La vie artificielle est un champ de recherche pluridisciplinaire dont l'objectif est de créer des systèmes artificiels s'inspirant des systèmes vivants et donc de pouvoir élargir le champ de la biologie à l'étude de la vie telle qu'elle pourrait se développer en dehors de notre système solaire (exobiologie). L'idéal, "l'interprétation forte", de la vie artificielle est définie ainsi par Christopher Langton :

The ultimate goal of the study of artificial life would be to create 'life' in some other medium, ideally a virtual medium where the essence of life has been abstracted from the details of its implementation in any particular hardware. We would like to build models that are so

FIGURE 2.3: Boucle de Langton (source : S. Levy, Artificial Life., Penguin, 1992 - <http://www.rennard.org>)



³. auteur d'une version simplifiée de l'automate de von Neumann basée sur le même principe, mais utilisant des symétries pour réduire la taille du réplicateur

life-like that they cease to be models of life and become examples of life themselves.[18]

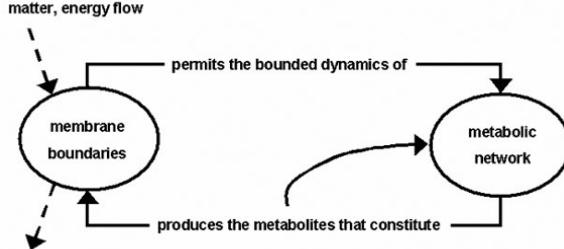
A l'inverse, dans son “interprétation faible”, la vie artificielle se donne pour objection de simuler, et non pas recréer, la vie sur un nouveau médium. Reste néanmoins à définir ce qu'est l'essence de la vie et ce n'est pas du tout une question aisée.

Le principe “méthodologique” de la vie artificielle repose sur deux étapes : extraire les principes fondamentaux du vivant, puis les réaliser sur un support arbitraire.

Il est donc nécessaire, lors de la première étape, de caractériser la vie. Cette tâche est rendue d'autant plus complexe que la nature continue ou discontinue du vivant n'a pas été réellement déterminée. Pour Richard Dawkins et d'autres biologistes de l'évolution, le vivant émerge par accumulation du processus de sélection naturelle. Il en résulte qu'il est très difficile d'établir une délimitation stricte entre le vivant et le non vivant (il y a donc continuité entre vivant et non vivant), mais certaines définitions du vivant, se référant par exemple à un seuil de complexité, s'accommodeent mal de cette potentielle continuité. Il est de même possible de considérer que la vie est essentiellement une structure dissipative ou système autopoïétique (que l'on définira un peu plus loin). On peut aussi avancer que la caractéristique principale de la vie est d'être un système adaptatif. L'auto reproduction et l'évolution par sélection naturelle sont alors les caractéristiques essentielles du vivant. Toutes ces définitions ne sont pas réellement mutuellement exclusives et, dans le domaine de la vie artificielle, on peut retenir un ensemble de huit propriétés proposées en 1992 par Farmer [10] :

1. Life is a pattern in space-time rather than a specific object.
2. Self-reproduction.
3. Information storage of a self-representation.
4. Metabolism.
5. Functional interactions with the environment.
6. Interdependence of parts.
7. Stability under perturbations.
8. Ability to evolve.

FIGURE 2.4: Machine autopoïétique (source : From autopoiesis to neurophenomenology : Francisco Varela's exploration of the biophysics of being)



2.2.4 L'autopoïèse

Un système autopoïétique (Figure 2.4) est un système se construisant lui-même. La caractéristique essentielle de la vie est alors de pouvoir construire sa propre organisation et de la maintenir lors l'apparition de perturbations extérieures :

un réseau de processus de production de composants qui régénèrent continuellement par leurs transformations et leurs interactions le réseau qui les a produits.

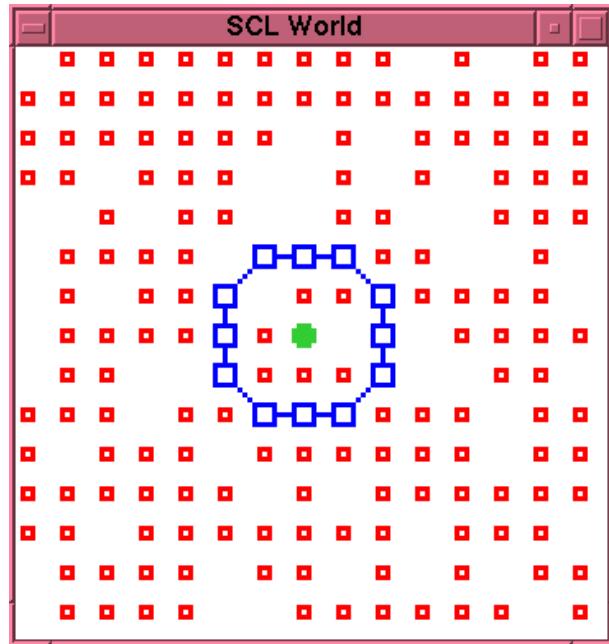
Ce concept découle d'un article de Francisco Varela [32]. Francisco Varela a proposé un automate cellulaire permettant de simuler l'autopoïèse (Figure 2.5) : Il comporte quatre états : vide, catalyseur substrat, les liens connectés et les liens isolés. Trois types de règles suffisent à déterminer le comportement de l'automate :

- Composition : si un catalyseur est en présence de deux substrats, un lien apparaît et le catalyseur est conservé.
- Concaténation : si des liens connectés sont en présence d'un lien isolé, celui-ci devient un lien connecté.
- Désintégration : dans certains cas, deux liens isolés, en présence l'un de l'autre, se désintègrent en deux substrats.

Il est à noter que Varela n'inclut pas la réplication dans sa définition de l'autopoïèse, et donc dans sa définition du vivant. Cette définition induit une vision des origines de la vie où le métabolisme apparaît avant la réplication.

D'autres chercheurs ont proposé des définitions très proches en cherchant à décrire un être vivant minimal. C'est, entre autre, le cas de Tibor Ganti avec son "chemoton model"⁴ qui insiste sur la nature nécessairement autocatalytique des réactions propres à ce système.

FIGURE 2.5: Automate cellulaire autopoïétique (source : 30 Years of Computational Autopoiesis : A Review - Barry McMullin)



⁴. caractérisé par des barrières chimiques, un moteur chimique et codage d'information chimique

2.3 Évolution artificielle

2.3.1 Fonction de “fitness” et Algorithme évolutionniste

Fonction de “fitness” En biologie, quand on parle de la “fitness” d’un organisme, on parle de son adaptation, de tout ce qui va augmenter l’efficacité avec laquelle il va pouvoir multiplier dans la durée des copies de lui même. Cette fitness, dans la nature, est le résultat d’interactions complexes entre l’organisme et son environnement, et il n’est généralement pas possible de la mathématiser. C’est néanmoins cette fitness des organismes que la sélection naturelle permet de maintenir, ou d’augmenter en dépit de changements nocifs de l’environnement. En s’inspirant de cette idée, il est possible de rajouter dans des simulations informatiques des “fonctions de fitness”. Le programme évalue alors les capacités des entités que l’on souhaite voir évoluer selon certains critères définis au préalable, et en tire une valeur numérique. Il est à noter qu’ici la fonction de fitness est explicite et aussi constante, tandis que dans la nature les critères qui vont déterminer la fitness d’un organisme évoluent en permanence.

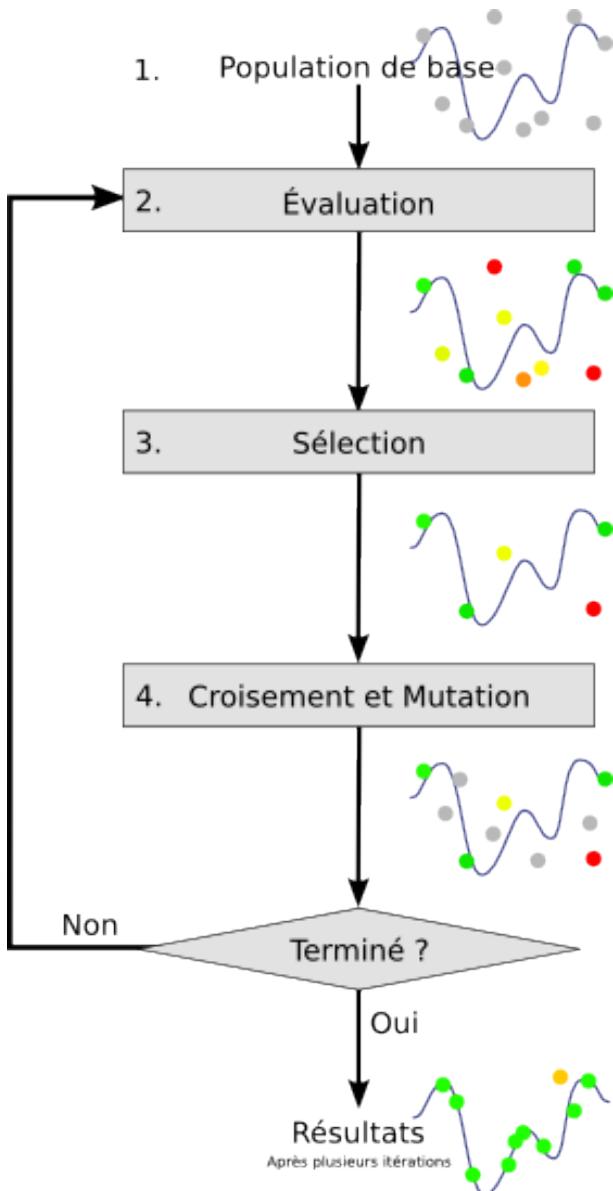
Algorithme génétique L’utilisation en informatique des algorithmes génétiques remonte aux années 1960⁵ et aux travaux de John Henry Holland⁵. L’idée était de réutiliser le principe de la sélection naturelle pour résoudre un problème. La méthode est particulièrement efficace et adaptée, lorsque qu’il existe différentes solutions de différentes qualités, que l’évaluation de la qualité d’une solution est simple, mais que l’on ne sait pas comment calculer directement la solution optimale ou que ce calcul est extrêmement coûteux en ressources. C’est alors un mécanisme d’optimisation qui peut s’avérer utile et pratique. Le principe est le suivant (Figure 2.6) :

- Générer aléatoirement un ensemble de “solutions” / “génotypes”
- Évaluer le résultat des différentes “solutions” associées grâce à l’utilisation d’une fonction de fitness
- Régénérer un ensemble de solutions à partir des meilleures. (Associer les meilleures solutions par paires les croiser et/ou ajouter des modifications aléatoires)
- Répéter l’opération avec différentes solutions filles jusqu’à l’obtention d’une solution satisfaisante.

La difficulté consiste souvent à trouver comment représenter les différentes solutions selon une forme adaptée à l’application de l’algorithme (permettant à la fois l’évaluation et les mutations). Il en existe de nombreuses versions (en conservant ou pas les “solutions parentes”, avec différentes méthodes de croisement, etc.) et cela montre, s’il en est besoin, l’efficacité du couple modification aléatoire/sélection de l’évolution en tant que méthode d’optimisation.

5. L’idée, elle, est beaucoup plus ancienne : elle vient de Walter Cannon et se retrouve même chez Alan Turing

FIGURE 2.6: Schéma expliquant le principe de fonctionnement des algorithmes génétiques (source de l'image : payo, NoJhan - Wikipedia)



Programmation génétique La programmation génétique se rapproche beaucoup des algorithmes génétiques, à ceci près qu'on est ici dans le cadre plus spécifique de la création automatique de programmes. Il faut donc utiliser un jeu d'instruction / un langage de programmation adapté qui supporte les modifications aléatoires. Il existe principalement deux catégories de programmation génétique :

- La programmation linéaire (développée par W. Banzhaf), dans laquelle l'on exécute une suite d'instructions organisées linéairement
- La programmation par arbre où chaque noeud de l'arbre va être un opérateur

2.3.2 Évolution ouverte

Nils A. Barricelli Les toutes premières expériences de simulation évolutionnistes furent effectuées en 1954 par Nils A. Barricelli.^[11] Elles consistaient à faire interagir des chiffres se déplaçant, et se copiant, selon des règles simples sur une matrice à une dimension (pouvant rappeler un automate cellulaire à une dimension). Après quelques itérations, le système faisait rapidement émerger des formes/configurations stables qui se maintenaient dans le temps.

Par la suite, dans les années 60, Nils A. Barricelli s'intéressa au choix d'une stratégie optimale à un jeu simple⁶. Barricelli fit évoluer à plusieurs reprises des populations de 10 000 agents, en incluant dans leur génotype non seulement leur stratégie mais aussi : le pourcentage de mutations, les chances de crossover, etc. Enfin la simulation était la première à utiliser, en plus des "one point crossover" (prendre linéairement une partie du génotype de l'un des parents, puis la suite chez l'autre parent pour constituer le génotype de

6. Chaque joueur tire une carte qui peut prendre deux valeurs différentes, puis mise une somme parmi trois sommes possibles, le joueur ayant misé la somme plus élevée remporte la mise, en cas d'égalité le joueur ayant tiré la carte avec la valeur la plus élevée remporte la mise.

l'enfant), des crossover dit “uniformes” (chaque gène de l'individu fils est choisi aléatoirement parmi les gènes de ses deux parents). L'expérience permet, en fin de simulation, d'obtenir jusqu'à 90% d'individus employant la stratégie optimale (celle-ci étant connue).

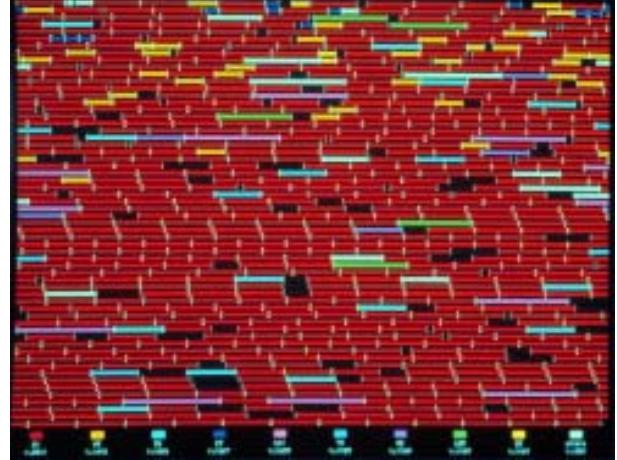
Tierra TiERRA est basée sur un “jeu” de programmeurs : CORE WARS. Inventé en 1984 par le mathématicien et philosophe Alexander Dewdney, Core Wars permet d'organiser des compétitions de programmes (écrit dans un langage particulier le REDCODE) dans un environnement virtuel appelé MARS (Memory Array Redcode Simulator). Le principe est le suivant : l'environnement virtuel est composé de 8000 cases mémoires, chaque instruction d'un programme occupe une case mémoire. A l'initialisation les deux programmes sont placés dans la mémoire et on exécute simultanément une instruction de chaque programme, puis les suivantes, ect. Le premier programme qui réussit par quelque moyen que ce soit à corrompre l'autre programme (rendre son exécution impossible) a gagné. Pratiquement, le jeu consiste à essayer de modifier les espaces mémoires dans lesquels sont stockés les instructions de l'autre programme, tout en déplaçant, réparant ou répliquant les siennes pour éviter leur destruction.

Quand, au début des années 1990, Tom Ray (biologiste des systèmes tropicaux) a conçu Tierra [24], il a repris le principe de Core War en le modifiant sur plusieurs points :

En partant d'un programme fonctionnel conçu en Redcode, ou dans un autre langage de programmations “classique”, le rapport “nombre de modifications du programme possibles dont résulte un programme opérant” divisé par “nombre de modifications du programme possibles” est voisin de zéro. Espérer obtenir de cette manière des réplicateurs fonctionnels serait du même ordre qu'espérer obtenir un livre passionnant en répartissant au hasard des lettres sur du papier. Tom Ray réduit donc le nombre d'instructions. Il les choisit bio-inspirées et de façon à ce qu'une mutation soit le plus possible susceptible de générer un programme effectif.

Tom Ray modifia la simulation de façon à ce que le nombre d'instructions allouées à chaque programme, à chaque cycle, augmente en fonction de la taille du programme. Enfin il mit en place un algorithme chargé d'éliminer les programmes les plus anciens et de générer des modifications aléatoires, lors de l'exécution des instructions.

FIGURE 2.7: Visualisation de Tierra (source : Tierra home page)



A l'initialisation, la mémoire contenait un unique programme autoréplicateur de 85 instructions.

De nombreux phénomènes obtenus par Tom Ray avec Tierra se rapprochent beaucoup de phénomènes obtenus en biologie : D'abord une créature autoréplicatrice de taille plus réduite (79 instruction) apparut. Ensuite, après une phase d'explosion de la diversité, une forme de parasite de seulement 45 instructions apparut. Celle-ci parasitait les autres créatures pour qu'elles la répliquent elle, au lieu d'elles-même. Une forme d'équilibre s'installa sur le mode proie prédateur. Par la suite certains programmes autoréplicateurs modifièrent leur code de façon à s'immuniser des virus d'une façon qui peut rappeler la manière dont les bactéries s'immunisent contre les bactériophages. Des hyperparasites et des hyper-hyperparasites apparurent. Enfin la sélection naturelle fit apparaître des créatures de 61 instructions partageant leurs instructions dans une forme de reproduction de groupe.

Par contre le processus n'aboutit pas à la création de programmes plus complexes (en terme de nombre d'instructions) que le programme de départ.

Tom Ray est aussi le promoteur du concept d'évolution ouverte en vie artificielle. Si Tierra semble "presque" générer une telle évolution, une étude de Marc Bedeau montra en 1997 que les écosystèmes similaires à Tierra aboutissent à des comportements cycliques[2], les éloignant de la richesse des écosystèmes réels et du qualitatif d'évolution ouverte/open ended evolution. La cause principale de ces comportements cycliques semble être la taille trop réduite de l'espace dans lequel s'effectue la simulation. Tierra a donc débouché sur Net Tierra, expérience consistant à faire fonctionner un réseau de simulateurs Tierra et ainsi augmenter l'espace de Tierra en multiplier les écosystèmes. Malheureusement le projet ne semble pas avoir abouti sur quelque chose de notable.

Polyworld (Larry Yaeger) Larry Yaeger a développé, à partir de 1993, avec POLYWORLD [34], un système qui se concentre presque exclusivement sur l'évolution du système nerveux dans un "complex rich ecology system". L'optique est aussi d'intégrer directement un certain nombre de fonctions du vivant tel qu'il s'est développé. L'environnement de Polyworld est un plateau (Polyworld utilise de l'OpenGL, mais les créatures dont il simule l'évolution ne profitent pas réellement de la troisième dimension - Figure 2.8). La simulation insiste particulièrement sur l'aspect énergétique : toute action, de même que la complexité du système nerveux, a un coût énergétique dans Polyworld. Des blocs de nourriture sont générés aléatoirement dans l'environnement et les créatures qui y évoluent peuvent utiliser un certain nombre de fonctions "basiques" (dans le sens où elles sont des possibilités offertes directement par le logiciel) tel que :

- pour les actions/sorties : se déplacer, manger de la nourriture, attaquer une autre créature, manger une créature morte, s'accoupler.
- pour les perceptions/entrées : voir/déetecter (les autres créatures/la nourriture/les éventuelles barrières),

le niveau d'énergie et une entrée aléatoire.

Le “génome” de chaque créature code la configuration initiale d'un réseau de neurones, sa rapidité, sa couleur, la fréquence des mutations dans ce génome. Le comportement de la créature évolue au cours de sa “vie”, l'apprentissage se fait par la règle de Hebb (renforcement du lien entre deux neurones, lorsqu'ils sont excités conjointement).

Polyworld a permis d'observer des phénomènes tels que l'émergence d'un système proie prédateur et une augmentation rapide de la complexité (en début de simulation) suivie d'une stabilisation.

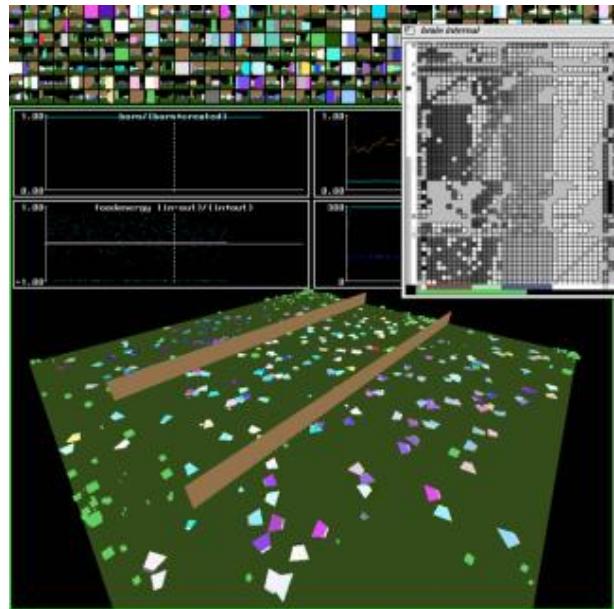
Le système Avida[1] fonctionne sur le même principe que Tierra avec néanmoins des différences très significatives : Le jeu d'instruction utilisé est proche de celui de Tierra (Redcode issu de Core Wars), mais contrairement à Tierra qui localisait des instructions et des pointeurs dans un monde à une dimension, ici, chaque réplicateur dispose d'un espace mémoire protégé lisible et modifiable uniquement par lui.

Les organismes exécutent leurs instructions en simultané, ce qui rend le programme aisément parallélisable. Chaque organisme dispose d'une plage de temps pour exécuter ses instructions. Cette plage de temps peut varier d'un organisme à l'autre, ce qui offre la possibilité de récompenser des comportements souhaités.

Les organismes sont localisés sur une grille torique en deux dimensions et ne peuvent effectuer que des actions locales. Ces actions locales sont principalement liées au mécanisme de réPLICATION. Les organismes disposent d'un âge et lorsqu'un organisme crée une copie (altérée) de lui-même, elle remplace l'organisme le plus vieux de son voisinage.

Avida est aisément adaptable à des usages particuliers, en spécifiant des bonus particulier ou en ajoutant d'autres formes d'interactions locales. C'est pourquoi, contrairement à Tierra, Avida est toujours utilisée de manière contemporaine pour effectuer des simulations de vie artificielle.

FIGURE 2.8: Interface de Polyworld (source Polyworld-Square, site de Virgil Griffith)



2.4 Simulations basées sur des automates cellulaires

2.4.1 Automates évolutifs

EvoCA Pour Tim Taylor, créateur d’EvoCA[31], la plupart des automates cellulaires et des écosystèmes, tel que Polyworld de Larry Yaeger évoqué précédemment, créent des règles de fonctionnement visant à obtenir un certain résultat et se concentrent trop sur l’autoréplication. Cette démarche les rend peu susceptibles d’expliquer l’émergence de la vie :

Much of this work is characterized by an emphasis on the computational capacities of the organisms. [...] Accompanying this perspective has been an (over-)emphasis on the process of self-reproduction, often to the exclusion of other important issues, such as the properties of the environment, and the representational relationship between organisms and environment. If nothing else, the poor evolvability of these systems demonstrates that the processes of self-reproduction with heritable mutation and selection, by themselves, are insufficient to explain the evolutionary origin of complexity.

Pour Tim Taylor il serait plus pertinent de considérer les organismes (le phénotype) comme partie intégrante de l’environnement et le génotype comme une modification locale des lois physiques :

This perspective, then, sees organisms as entities whose phenotypes are embedded within an environment viewed as a dynamical system, and whose genotypes interact with the environment by specifying constraints upon its dynamics, thereby generating the phenotypes. That is, the abiotic environment has its own dynamics and self-organizational properties; genotypes act to “sculpt” these pre-existing dynamics by supplying constraints. From this point of view, the most important distinction is not between organisms and their abiotic environment, but rather between the environment as a whole (including organism phenotypes) and organism genotypes. It is the relatively time-independent genotypes, by supplying local constraints to the dynamics of the environment that reify phenotypes as distinct entities within the environment.

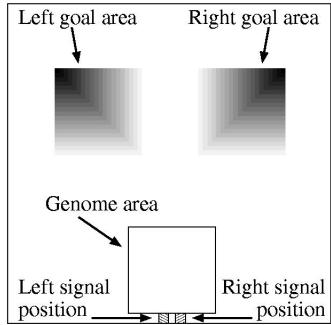
Le principe d’EvoCA est le suivant : sur une première grille en deux dimensions (une expérience similaire pourrait être menée avec un automate à trois dimensions) tourne un premier automate cellulaire (en l’occurrence le “jeu de la vie”). Dans une seconde grille de même taille, où chaque cellule de la première grille possède une cellule correspondante, se trouvent des “génotypes”.

Il existe ensuite deux versions d’EvoCA :

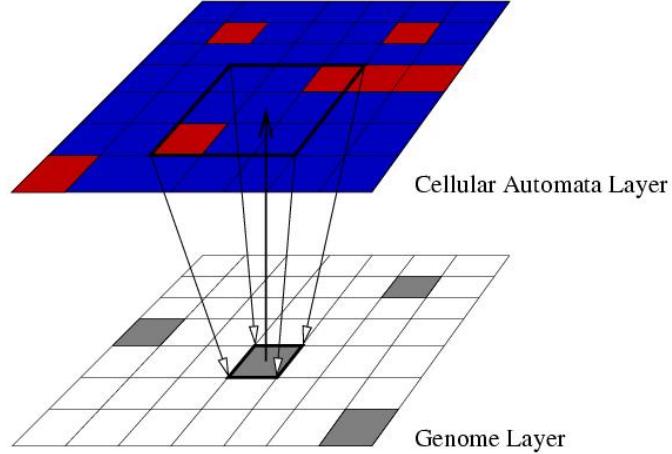
- Dans EvoCA-A (Figure 2.9a) ces “génotypes” ont la possibilité de modifier les états des cellules de la première grille dans une “zone d’influence” contenant la cellule symétrique de celle où ils se trouvent

FIGURE 2.9: EvoCA

(a) Evo CA-A (source : [31])



(b) Evo CA-B (Source : [31])



et certaines cellules situées aux alentours de celle-ci. Le génotype modifie prioritairement (par rapport aux règles “jeu de la vie” qui gouvernent la première grille) les états des cellules de la première grille, si celles-ci satisfont des conditions décrites dans le génotype (qui peuvent être de type temporel : “la cellule $A5$ passe à l'état s_1 quand $t = 5$ ” ou conditionnel en fonction des états des cellules de la zone d'influence du génotype). Tim Taylor crée alors une population de génotypes aléatoires et les fait évoluer selon un algorithme génétique. L'environnement dans lequel s'exécute la fonction de fitness est le suivant : On définit dans la seconde grille une position du génotype, et dans la première on définit deux zones cibles (toutes deux extérieures à la zone d'influence du génotype) et une zone de signal (en bordure de la zone d'influence du génotype). Les cellules hors de la zone de signal sont initialisées à l'état inactif (les cellules du jeu de la vie ont deux états, actifs et inactifs). En fonction de l'état des cellules dans la zone de signal, l'une des zones cibles sera définie comme zone à atteindre, tandis que l'autre zone sera définie comme zone à éviter. Ensuite la fonction de fitness consiste à évaluer si des cellules passent à l'état actif dans la zone à atteindre et restent à l'état inactif dans la zone à éviter, et ce, tour à tour avec les différents états de signal possibles et donc voir si le génotype arrive à “manipuler” les cellules sous son influence pour arriver à cet objectif. Les résultats furent positifs et Tim Taylor obtint des génotypes capables d'activer les cellules voulues.

- Dans EvoCA-B les génotypes se propagent et sont en concurrence. Par ailleurs ils ne peuvent agir que sur une seule cellule (la cellule symétrique dans la grille 1, à celle où ils se trouvent dans la grille 2) et uniquement en fonction en l'état de cette cellule et de ses huit cellules voisines. D'autre part, chaque génotype associe aussi des directions de propagation à des configurations dans la première

grille. Finalement, à chaque état et configuration du voisinage possible de la cellule symétrique dans la première grille, sont associés (de manière fixe) des pourcentages de chance de mort, propagation et mutation du génome.

Moshe Sipper Moshe Sipper s'est à plusieurs reprise intéressé aux automates cellulaires hétérogènes. Dans un premier temps, il a mené, au début des années 90 [28], différentes expériences autour d'automates hétérogènes à deux dimensions, où chaque cellule possédait un ensemble de règles propres, ou "génome", et où la propagation de ces génomes était déterminée par le succès ou l'échec des cellules. Succès qui, selon les simulations, pouvait consister à être dans le même état que la majorité des cellules environnantes, à gagner au jeu du prisonnier, etc. Le système s'est révélé efficace et quelle que soit la simulation, les cellules adoptaient assez rapidement un comportement proche du comportement considéré comme optimal. La rapidité de la convergence vers ce comportement étant fonction de la diversité génétique de la population initiale (la convergence étant plus rapide avec une population plus diverse). Dans le même article Sipper a aussi montré l'intérêt de tels automates pour la créations de structures complexes. Il montre que de telles structures peuvent résulter de la coopération de plusieurs génotypes et que cette coopération permet de réduire le nombre d'états nécessaires à la création d'une telle structure, par rapport à une structure équivalente dans un automate homogène.

Par la suite vers la fin des années 90 [30, 29] Moshe Sipper s'est intéressé au potentiel des automates hétérogènes à une dimension, en terme de résolution de problème (par exemple augmenter jusqu'à 100% la densité de cellules à l'état le plus présent ou générer un nombre aléatoire). L'idée consiste ici à coupler de tels automates à des algorithmes génétiques et à faire évoluer un automate (et non une population d'automates), le génome de chacune de ses cellules évoluant individuellement (par rapport à sa fitness propre et en échangeant du matériel génétique uniquement avec ses cellules voisines). Sipper montre grâce à cette simulation que les automates cellulaires hétérogènes coévoluent et arrivent à effectuer ces tâches pourtant complexes (Figure 2.10a).

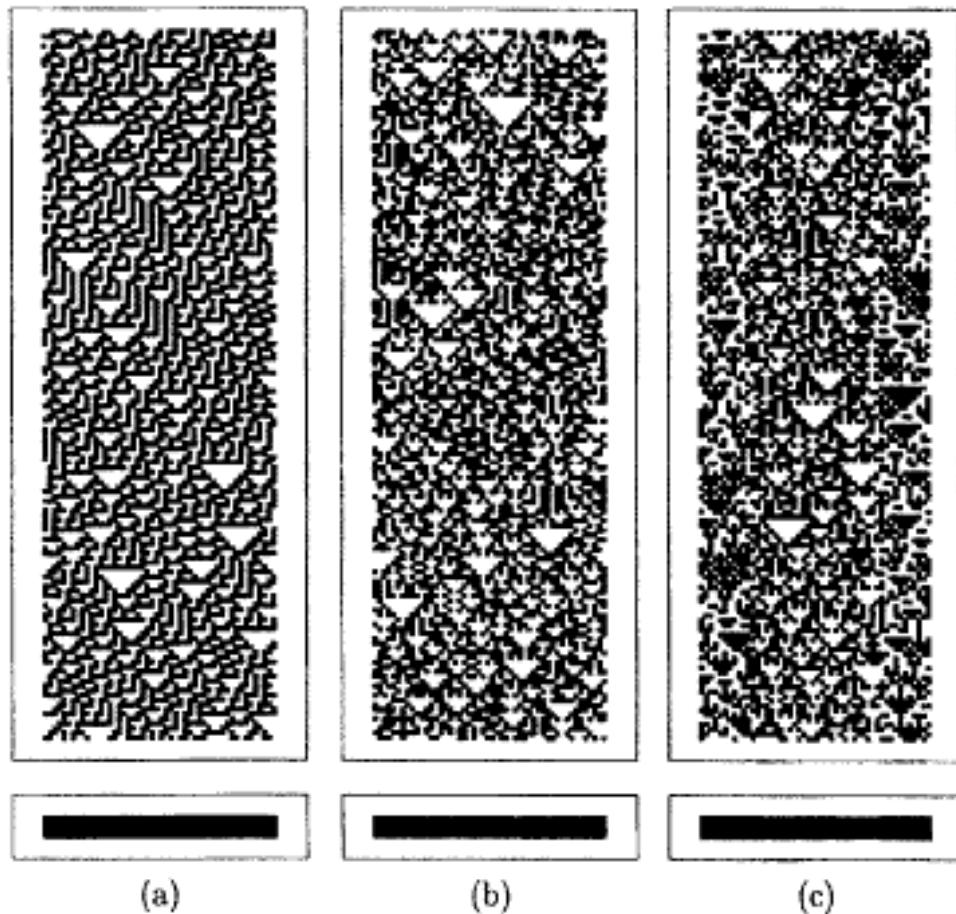
Évolution spatiale Le jeu du prisonnier⁷ est souvent utilisé pour montrer l'importance de dynamiques locales et pour étudier l'altruisme. Dirk Helbing [13] l'utilise pour étudier les effets de l'imitation et des déplacements. Pour cela il effectue quatre simulations (dans chacune d'elles la moitié des agents sont initialement altruistes, c'est à dire qu'ils optent toujours pour la non dénonciation, tandis que l'autre partie des agents

7. Jeu de la théorie des jeux où dans son implémentation la plus classique deux prisonniers ne pouvant pas communiquer entre eux doivent choisir si il dénonceront ou non leur complice sachant que :

- Si ils se dénoncent l'un l'autre ils écoperont d'une peine de prison moyenne (par exemple 5 ans)
- Si l'un d'entre eux dénonce l'autre sans que celui-ci fasse de même il sera libéré et l'autre écoperà de la peine maximale (par exemple 10 ans)
- Si ni l'un ni l'autre ne se dénonce ils écoperont tout deux de la peine minimale (par exemple 1 an)

FIGURE 2.10: Travaux de Moshe Sipper

(a) Génération de nombres aléatoires à partir d'un automate cellulaire à une dimension. les cellules de [a] utilisent une règle, celle de [b] deux et celles de [c] trois règles différentes. (source : [30])



(b) Des territoires se forment grâce à l'utilisation du critère de succès "agree". Source : [28]

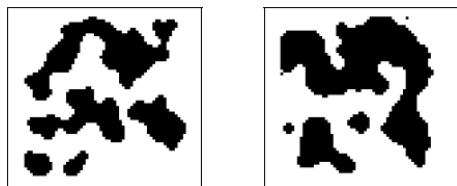
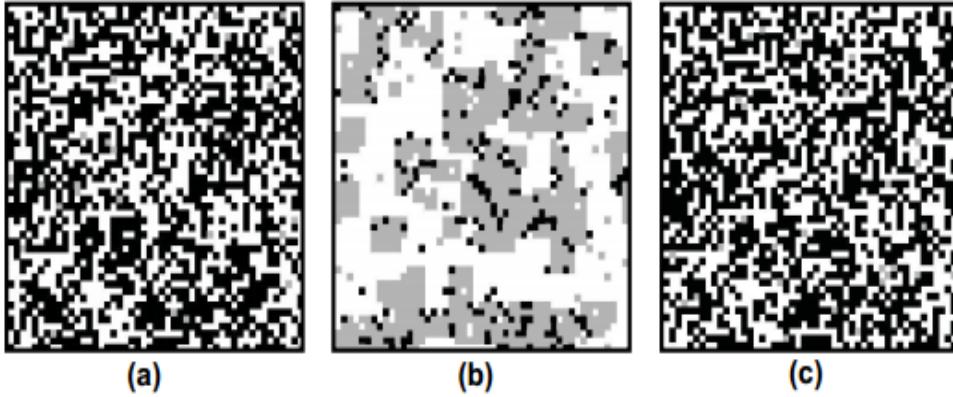


FIGURE 2.11: Simulation de jeu du prisonnier avec migration des agents, imitation et interaction large (a) Si le voisinage d'interaction est large et la distance de migration petite la coopération échoue. La raison est la suivante : quand tout le monde peut interagir avec tout le monde, il n'y a pas d'intérêt à migrer. (b) Quand le voisinage d'interaction est petit et la distance de migration importante, des groupes d'individus coopérants se forment et la coopération/l'altruisme est un succès. Enfin (c) quand la distance de migration est importante et le voisinage d'interaction est important, l'égoïsme l'emporte rapidement sur la coopération (Source : [13])



sont égoïstes, c'est à dire qu'ils optent systématiquement pour la dénonciation).

- Les agents jouent au jeu du prisonnier avec leurs voisins, ne se déplacent pas et imitent leur voisin qui a le plus de succès. Au terme de la simulation il n'y a plus que des individus égoïstes.
- Les agents jouent au jeu du prisonnier avec leurs voisins, ne changent pas de stratégie mais se déplacent vers leur voisin qui a le plus de succès en s'éloignant des moins efficaces. Au terme de la simulation des groupes se forment avec, au centre, les individus altruistes et en périphérie les individus égoïstes.
- On active à la fois l'imitation et le déplacement et au final c'est l'altruisme qui devient le caractère dominant.
- Enfin, on active à la fois l'imitation et le déplacement, mais on autorise les agents à interagir avec un nombre très important d'individus (en augmentant la taille de leur voisinage) et non pas seulement avec leur voisin immédiat. Au final, c'est à nouveau l'égoïsme qui l'emporte (Figure 2.11)

On peut relier cette expériences à trois éléments distincts :

- La mémétique : dans le géne égoïste[6] Richard Dawkins évoque dans un chapitre l'existance possible d'autres réplicateurs soumis à la sélection naturelle et donc à l'évolution : les mèmes. Ceux-ci seraient des éléments culturels (humains) transmis par imitation. Par la suite c'est sans doute Susan Blackmore qui a le mieux développé cette théorie[9] en s'intéressant entre autre à l'apparition de l'imitation chez l'humain, la mémétique comme moyen d'expliquer l'altruisme et le possible conflit entre pression génér-

tique et pression mémétique⁸. On retrouve ici une équivalence dans la mesure ou le mode de réPLICATION est l'imitation.

- La sélection de groupe : initialement proposée par Darwin, la sélection de groupe consiste à considérer que des caractères altruistes pourraient être sélectionnés en favorisant, entre plusieurs groupes relativement isolés mais appartenant à une même espèce, ceux d'entre eux comportant le plus grand nombre d'individus altruiste. Néanmoins pour celà il faut néanmoins que le gain en terme de croissance de la population des groupes “altruistes” permettent de compenser l’augmentation de la part d’individus égoïstes au sein de ces mêmes groupes. Ce qui est loin d’être évident. Les travaux de George Price[23] ce ne fut que bien après Darwin, en 1970, que finalement George Price proposa une première mathématisation de la sélection de groupe et expliqua dans quels cas, grâce à elle, des caractéristiques altruistes peuvent être sélectionnées. Par la suite, plus récemment, David Sloan Wilson a proposé une théorie de la sélection à plusieurs niveaux[33] (“Multilevel Selection Theory” MLS) : Gène > cellule > individus > groupe. Il faut alors voir le groupe comme un véhicule pour la réPLICATION des niveaux inférieurs (de même que Dawkins présente l’individus comme un véhicule pour les gènes). Dans cette expérience on retrouve une vision assez classique de la sélection de groupe la formation de groupes relativement isolés permettant le succès des caractéries altruistes.
- Outre ses liens avec la sélection de groupe et de la mémétique il faut souligner que cette expérience montre l’importance des interactions locales dans le succès ou l’échec d’une stratégie.

2.4.2 Automates cellulaires et structures autoréPLICATRICES

Zhijian Pan et James A. Reggia [21] Dans leur simulation Zhijian Pan et James A. Reggia démontrent la possibilité de créer des structures autoréPLICATRICES par utilisation de programmation génétique couplée à des automates cellulaires. Pour cela ils utilisent deux arbres (Figure 2.12) :

- S Tree : est un arbre permettant d’encoder une forme (en l’occurrence une structure que l’on va chercher à répliquer).
- R Tree : est un arbre permettant de coder des règles de transition pour un automate cellulaire.

Ce couple S Tree / R Tree permet d’utiliser efficacement la programmation génétique pour sélectionner des règles qui vont permettre la réPLICATION d’une structure initiale. Pour calculer la “fitness” de ces règles, on va analyser la configuration de la grille et évaluer la proximité des structures rencontrées avec la structure à répliquer codée par le “S Tree”. Il est à noter que cette méthode donne des mécanismes de réPLICATION qui semblent plus naturels et plus proches de la réPLICATION cellulaire, que des réPLICATEURS dans des automates

8. Les mêmes souffrent néanmoins toujours de ne pas avoir été initiallement clairement définis autrement que par une analogie par rapport aux gènes : Il n’existe pas d’équivalent de la distinction phénotype/génotype et la question de la nature d’un même reste posée (un même reste t'il un même en changeant de support par exemple?).

FIGURE 2.12: Principe de fonctionnement de la simulation de Zhijian Pan et James A. Reggia (a) le réplicateur comporte 7 “composants orientés” à $t = 0$, à $t = 2$ on retrouve deux exemplaires de la structure originelle. (b) le S-Tree correspondant à la structure à répliquer. (c) Un R-Tree obtenu par sélection qui sert de fonction de transition pour les cellules de l’automate. (Source : [21])

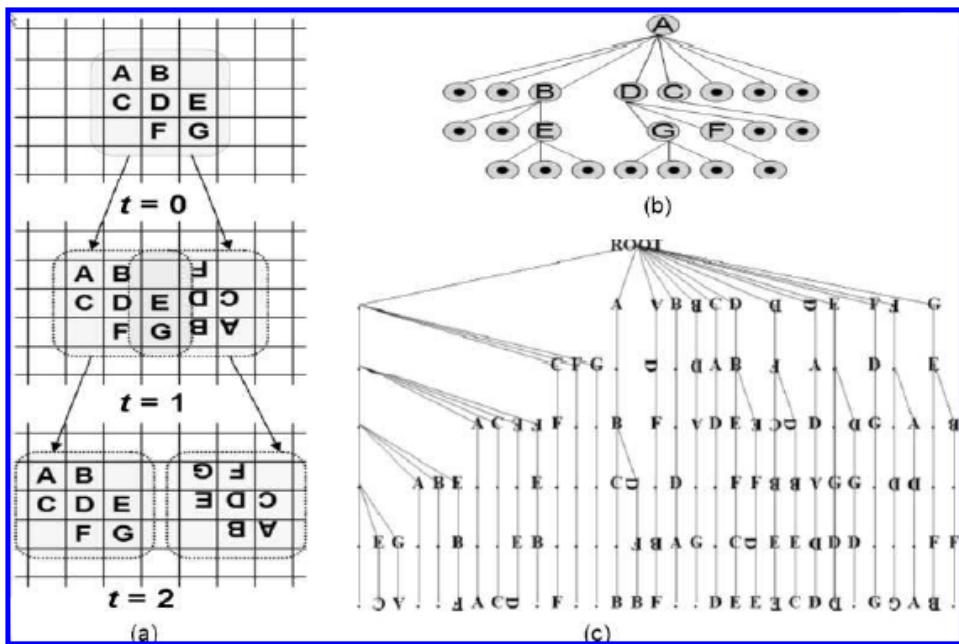
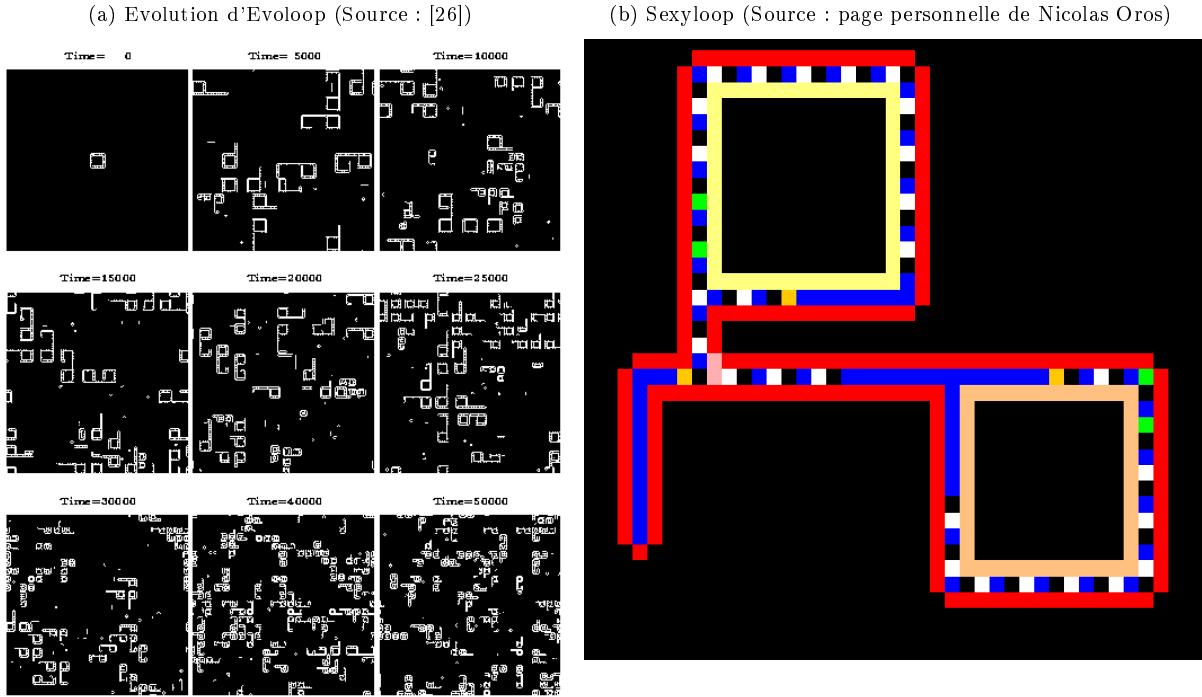


FIGURE 2.13: Boucles évolutives



cellulaires créés par l'homme, comme la boucle de Langton ou le réplicateur universel de John von Neumann.

Boucles évolutives Basées sur le principe de la boucle de Langton, les boucles “Structurally Dissolvable Self-Reproducing” (SDSR) rajoutent un nouvel état à la boucle de Langton, qui a pour but d'assurer la destruction de la boucle dans certains cas (manque de place, collision, boucle endommagée).

EvoLOOP [27] complexifie les règles SDSR de façon à obtenir une forme d'évolution : en cas de collision la structure de la boucle est modifiée, mais plus systématiquement détruite. Il est aussi à noter que l'univers est torique⁹. On obtient alors une évolution vers la simplicité : l'espace étant encombré par d'autres boucles autoréplicatrices, les plus petites maximisent leur chance de se reproduire dans l'espace limité.

Le dernier automate dérivé des boucles de Langton, SEXYLOOP [20] va encore plus loin et prétend proposer une forme de “reproduction sexuée” (Figure 2.13b) des boucles de Langton. En réalité, construit sur le principe d'EvoLoop, il lui ajoute un nouvel état (on passe donc à un automate à 10 états) et de nouvelles règles visant à faire en sorte que certaines collisions de boucles (au niveau des “coins”) donnent lieu à un transfert de caractéristiques. L'évolution de SexyLoop, tout comme celle d'EvoLoop, aboutit à des boucles de taille minimale, mais SexyLoop passe par une plus grande diversité de boucles (des boucles de taille plus importante que la boucle initiale apparaissent même, avant de finalement disparaître et de voir la simulation se stabiliser

9. le bord du haut de la grille est connecté avec le bord du bas de même pour les bords de droite et de gauche. Une représentation en 3D de l'automate serait une grille à la surface d'un anneau

sur des boucles de petite taille).

Chapitre 3

Comment simuler une évolution ouverte

Nous avons vu un certain nombre de tentatives de simuler une évolution ouverte, ainsi que d'autres préférant l'approche d'une évolution dirigée. Au vu de ces travaux et des caractéristiques de l'évolution, telle qu'elle existe au niveau du vivant, les difficultés pour recréer une évolution "ouverte" semblent être :

- De s'assurer que la simulation génère perpétuellement de la nouveauté (et ne recrée pas des génotypes identiques/très similaires, après des périodes d'extinction totale).
- De distinguer ces nouvelles structures "pertinentes" (qui vont être sélectionnées et se multiplier) du bruit que ne vont pas manquer de générer les mutations aléatoires. Nous verrons par la suite comment essayer de mesurer ces changements.
- De vérifier à terme qu'une telle simulation offre suffisamment de possibilités adaptatives pour, justement, ne pas se contenter de créer et recréer indéfiniment les mêmes génotypes.

Intéressons-nous aux moyens techniques de réaliser ces objectifs.

3.1 Sans fonction de fitness explicite

Souvent, en vie artificielle, des fonctions de fitness sont utilisées, et ce avec de très bon résultats en terme d'optimisation : on peut penser, entre autre, aux "virtual evolving creatures" de Karl Sims et aux algorithmes génétiques. Ces fonctions ont une utilité pratique et permettent de résoudre des problèmes bien identifiés. Par contre, elles aboutissent à des simulations où les entités considérées cessent généralement "d'évoluer" après un nombre de cycles plus ou moins important. Le propre d'une évolution "ouverte" est de ne pas utiliser de tels mécanismes. Dans le monde du vivant, il n'y a, bien sûr, pas de fonction de "fitness" explicite et celle-ci résulte des interactions complexes entre les différents réplicateurs et leur environnement. L'objectif ici est de tenter de s'approcher d'une telle forme d'évolution.

3.2 Un environnement complexe

La sélection naturelle est un processus qui aboutit à l'optimisation des entités étudiées en fonction de leur environnement. Il faut donc que l'environnement soit suffisamment riche pour qu'on puisse s'y adapter. Si l'environnement reste stable, se pose la question soulevée par Stephen Jay Gould des optima locaux : c'est à dire de solutions qui ne sont pas nécessairement les meilleures possibles, pour un environnement donné, mais qui sont suffisamment génétiquement éloignées des meilleures solutions, pour qu'il soit hautement improbable qu'une solution meilleure puisse apparaître, si tous les réplicateurs sont suffisamment génétiquement proches de la solution "localement optimale". Cette situation bloque alors le processus évolutif, jusqu'à l'apparition d'une mutation importante et rare et/ou un changement d'environnement permettant de sélectionner des solutions mieux adaptées à cette nouvelle situation. Sans un environnement et des possibilités complexes, on est finalement assez proche de l'utilisation d'une fonction de "fitness" (implicite mais fixe). L'interaction avec les autres réplicateurs fait bien sûr partie de la définition de l'environnement.

3.3 Interactions locales

Idéalement, il semble aussi souhaitable que l'environnement ne soit pas homogène, qu'il existe des spécificités "locales" au sein de la simulation et ce, que celles-ci soient le résultat de la répartition des différents génotypes/phénotypes, comme expliqué précédemment, ou que ces variations environnementales découlent directement de spécificités locales de l'environnement. Un tel système permet la présence d'une variété d'écosystèmes.

Pour que les réplicateurs fassent partie de l'environnement, il est nécessaire qu'ils aient la possibilité d'interagir localement entre eux (directement et indirectement). Par ailleurs, ces interactions locales offrent la possibilité de formation de groupes éloignés géographiquement, mais ayant un code génétique proche. Si, par la suite, on ajoute à cette simulation des mécanismes d'échanges génétiques/reproduction sexuée, ces groupes pourront jouer un rôle dans la spéciation et permettront peut-être aussi de tester des mécanismes tels que la "sélection de groupe" décrite par les équations de George Price[23].

3.4 Nombre important d'agents/de réplicateurs

Bien sûr, pour satisfaire ces différents critères, il semble nécessaire qu'un nombre le plus important possible de réplicateurs interagissent. Il est donc nécessaire de chercher à limiter le coût en terme de mémoire/processeur de ces réplicateurs, sans pour autant trop limiter leur capacité d'interaction. Mais ce problème est caractéristique de l'ensemble des simulations évolutionnistes.

3.5 Possibilité de comportements complexes

Enfin, la simulation doit potentiellement permettre l'émergence de comportements complexes tels que ceux présents chez les êtres vivants :

- La transmission d'information
- La compétition et donc la destruction d'autres réplicateurs concurrents
- La création de systèmes autopoïétiques

Ainsi que les différentes caractéristiques proposées, entre autres par Farmer[10].

3.6 Bilan

Notre objectif principal, ici, est de trouver comment maintenir sur de longues durées, ou indéfiniment, de la variation et de la diversité phénotypique (et génotypique) dans des automates cellulaires hétérogènes. Nous avons choisi de réutiliser des automates cellulaires, car ils ont prouvé qu'ils pouvaient être utilisés pour recréer des comportement complexes typiques de la vie (système autopoïétique, réplication, structures complexes, communication entre entités, ect.), comme le montrent les résultats des travaux mentionnés dans les sections 2.2.1 et 2.4. Mais, dans l'optique de simuler une “open-ended evolution” les automates cellulaires classiques (homogènes) ont plusieurs désavantages :

- Bien souvent les règles des automates cellulaires (comme par exemple celles utilisées pour les boucles de Byl ou de Langton) sont choisies pour obtenir un comportement particulier (de la réplication le plus souvent, parfois des choses beaucoup plus “étranges” tel un lecteur de bandes). La conception d'un univers dans le but d'obtenir un comportement spécifique, puis l'attente d'émergence de comportements différents, semblent contradictoires. Le succès du jeu de la vie est justement de voir de la complexité apparaître, alors que les règles choisies initialement ne permettaient pas de le prévoir.
- Une autre question consiste à savoir comment introduire de la variation dans un automate cellulaire (déterministe). Dans les automates cellulaires classiques (comme par exemple la boucle de Langton et ses descendants), la variation est le résultat de phénomènes de contingence. Intéressons-nous aux autres possibilités qu'offrent les automates cellulaires. “L'aléatoire” peut intervenir classiquement, dans un automate cellulaire, à deux niveaux, l'un et l'autre assez éloignés de la variation telle qu'elle existe au niveau génétique :
 - à l'initialisation (en choisissant aléatoirement des états pour les différentes cellules de l'automate)
 - ou en adoptant des règles probabilistes (si les conditions sont remplies la cellule a $X\%$ de chance de passer à l'état s_a , $Y\%$ de chance de passer à l'état s_b , etc.).

- Il est aussi assez compliqué d'obtenir une compétition entre plusieurs réplicateurs avec les automates cellulaires traditionnels. Il semble impossible par exemple de mettre en compétition des boucles de Byl et des boucles de Langton dans une même simulation et d'obtenir des copies améliorées par cette compétition. Tout au plus, on met en concurrence différentes variations d'un même réplicateurs avec les différentes versions des boucles SDSR.

Chapitre 4

Le modèle

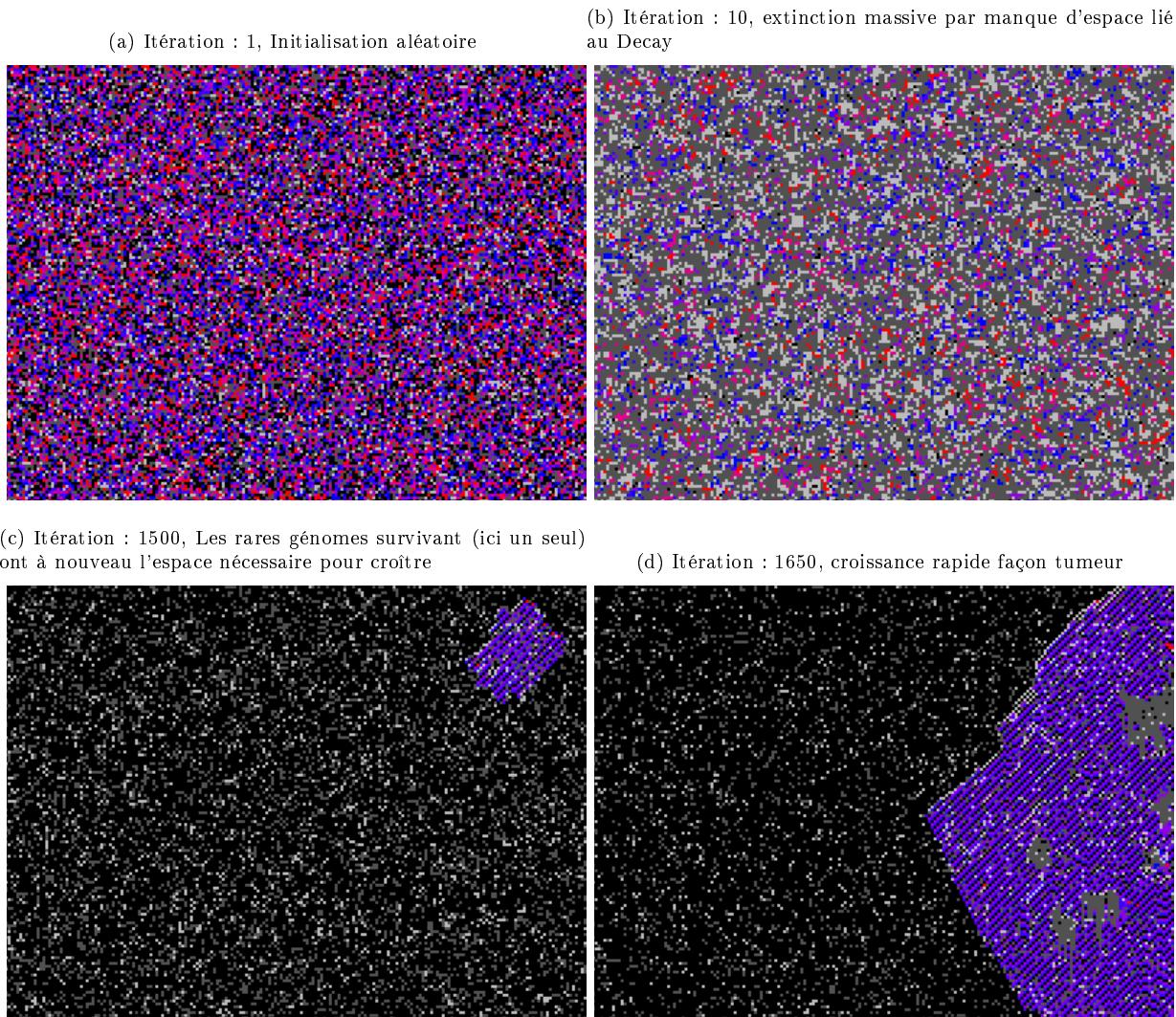
4.1 Présentation rapide

- Tout comme dans les automates cellulaires classiques :
 - Des cellules sont réparties sur une matrice à deux dimensions
 - Chaque cellule a un état
 - Ces cellules changent d'état de façon discrète en fonction de l'état des cellules voisines
- Contrairement aux automates classiques les règles de transitions d'état ne sont pas les mêmes pour chaque cellule :
 - Certaines cellules possèdent un “génotype” qui détermine leur règles de transition
 - Les génotypes sont susceptibles de subir des mutations et de se propager vers des cellules voisines
 - Des cellules spéciales “quiescent” ne possèdent pas de génomes et restent à cet état à moins qu’un génotype se propage vers elle
 - Les cellules possédant un génotype vieillissent et passent à un état particulier “decay” au bout d’un nombre d’itération prédéfini
 - Les cellules à l’état “decay” ne peuvent ni changer d’état ni acquérir de génotype mais retournent à l’état “quiescent” au bout d’un nombre d’itération prédéfini.

4.2 Le concept détaillé

Comme pour les automates cellulaires à deux dimensions classiques, nous utilisons une matrice (à deux dimensions) composée de cellules pouvant avoir différents états. Néanmoins, nous avons choisi d’opter pour des automates hétérogènes où il n’existe pas de règles de transition communes à l’ensemble des cellules, mais

FIGURE 4.1: Exemple de visualisation d'une simulation ($A_{limite} = 7, T_{decay} = 1850$ et $T_{muta} = 1/5000$) à différentes itérations. Ici, et par la suite les cellules “quiescent” à l'état s_q sont représentées par des cases noires, les cellules en “decay” à l'état s_d sont représentées par des cases grises (gris clair pour les cellules en “decay” naturel $s_{d\text{naturel}}$ et gris foncé pour les cellules en “decay” volontaire $s_{d\text{volontaire}}$), enfin les cellules vivantes sont représentées par des cases de couleurs (de rouge à bleu en fonction de leur cinq états possibles)



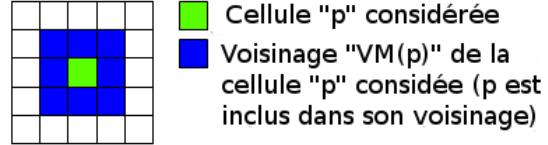
où chacune d'elle possède un système de fonction de transition spécifique (souvent appelé génotype par la suite). Il est à noter que l'on change ici d'analogie : les règles de transition d'un état à un autre ne sont plus ici l'équivalent des règles physiques du monde, mais l'analogie du code génétique de la cellule (la plupart du temps).

- Par ailleurs, pour qu'il y ait compétition, nous avons choisi d'ajouter (à la manière de la seconde version d'EvoCA) un mécanisme de propagation de ces règles, de façon à ce qu'une compétition entre réplicateurs puisse apparaître.
- Nous avons introduit de la variation au niveau de ces règles/génotypes, ainsi que dans le mécanisme de propagation des règles.
- Enfin nous avons choisi d'ajouter un mécanisme d'âge maximum A_{max} et de cellules mortes/en décomposition qui nous permet, entre autres, de grandement simplifier la lisibilité de la simulation.

Les automates cellulaires hétérogènes présentés ici sont un ensemble de cellules p dans un univers I . L'univers est une matrice rectangulaire torique de dimension $L \times l$. Dans les simulations présentées ici $L = 800$ et $l = 600$ (Figure 4.1). Chaque cellule à état $f(p)$ appartient à un alphabet fini Σ : $f(p) \in \Sigma$.

Le voisinage utilisé pour les fonctions de transition sera le voisinage de Moore, $VM(p)$ (Figure 4.2) sera le voisinage de la cellule p et on appellera $VM(p, t)$ le voisinage de la cellule p au temps t . On parlera aussi par la suite de $VM_{viv}(p, t)$ nombre de cellules vivantes¹ dans le voisinage $VM(p, t)$ de la cellule p au temps t .

FIGURE 4.2: Voisinage $VM(p)$ de la cellule p



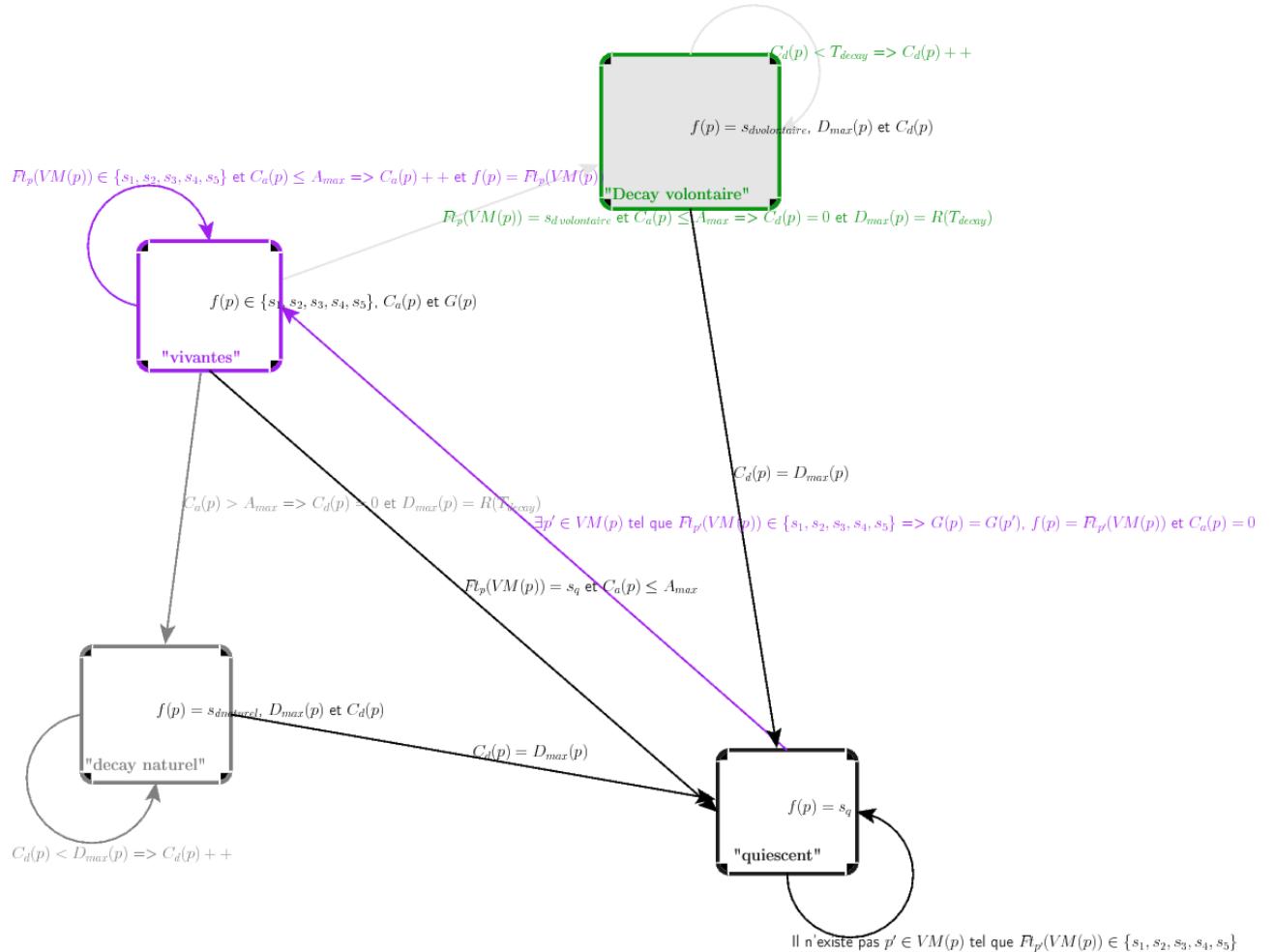
4.2.1 Les cellules normales / “vivantes”

Seules les cellules p à l'un des états “vivants” disposent d'un génotype $G(p)$ (Figure 4.3) :

- Elles peuvent prendre un état parmi les différents états possibles hors états spéciaux (5 états dans les simulations).
- Elles disposent d'un compteur C_a qui s'incrémente à chaque itération tant que la cellule reste dans l'un des état “vivants”.
- Elles disposent d'un génotype $G(p)$ qui est utilisé pour déterminer par une fonction de transition $Ft_p(VM(p))$ son prochain état, en fonction de l'état de ses voisines (dans le sens du voisinage de Moore $VM(p)$ - Figure 4.2) à condition que le compte à rebours A ne dépasse pas l'âge maximum A_{max}

¹. Une cellule p étant déclarée vivante ou non vivante en fonction de son état $f(p)$. On verra par la suite plus précisément dans quels états une cellule est déclarée vivante.

FIGURE 4.3: Diagramme d'état de la simulation pour une cellule p



autorisé, auquel cas la cellule change d'état et devient une cellule en décomposition (à l'état "Decay" s_d).

- A chaque itération, ces cellules peuvent voir leur génotype modifié. Si le génotype d'une cellule voisine (dans le sens du voisinage de von Neumann $VN(p)$ cette fois - Figure 4.4), permet de résoudre leur situation,² alors la cellule normale a une chance d'adopter le génotype de sa voisine. Si plusieurs voisines satisfont ces critères, le génotype adopté est choisi aléatoirement parmi les différents génotypes satisfaisant ce critère et le génotype courant de la cellule.
- Les cellules à l'un des états "normaux" (appelées par la suite cellules vivantes) sont représentées dans la simulation par des cases de différentes couleurs (en fonction de leur état) qui vont du rouge au bleu dans les simulations finales et du jaune au rouge dans les simulations exploratoires.

4.2.2 Cellules en décomposition / Decay

L'état "Decay" s_d est un état particulier. Les cellules dans cet état ont des caractéristiques notables (Figure 4.3) :

- Elles n'ont pas de génotype. Elles ne peuvent pas recevoir de génotype d'une cellule voisine.
- Elles disposent d'un compteur C_d qui s'incrémente à chaque itération.
- Elles restent au même état (dit de "Decay"), tant que leur compteur est inférieur au temps de "décomposition" $D_{max}(p)$ dont la valeur moyenne est T_{decay} . Une fois atteint ce temps, la cellule change d'état et passe à l'état neutre. Par ailleurs, il existe deux sous-types de "décomposition" : la décomposition volontaire $s_{decay, volontaire}$ (pour les cellules qui sont passées à l'état s_d avant que leur compteur dépasse l'âge maximum A_{max}) et la décomposition $s_{decay, naturelle}$ naturelle (pour les cellules qui sont passées à l'état "Decay" s_d après que leur compteur a dépassé la limite permise). Ces sous-types sont transparents pour les cellules (elles ne font pas la différence entre ces deux sous-états), mais ce système est utile pour voir si le Decay joue un rôle dans la compétition entre cellules.
- Les cellules en décomposition sont représentées dans la simulation par des cases grises (gris clair pour le "Decay" volontaire et gris foncé pour le "Decay" naturel).

4.2.3 Cellules "Quiescent"

L'état "quiescent" s_q est un état particulier. Les cellules, dans cet état, ont des caractéristiques notables (Figure 4.3) :

- Elles n'ont pas de génotype

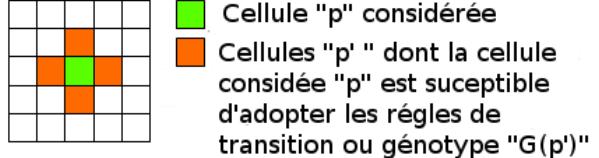
2. Un génotype résout la situation d'une cellule si, pour l'état et le voisinage de la cellule considérée il donne un état suivant qui ne soit ni "quiescent" ni "Decay".

- Elles peuvent par contre recevoir des génotypes de la part de cellules voisines : A chaque itération, ces cellules sont susceptibles d'adopter un génotype. Si le génotype d'une cellule voisine (dans le sens du voisinage de von Neumann $VN(p)$ - Figure 4.4), permet de résoudre³ leur situation, alors la cellule neutre a une chance d'adopter le génome de sa voisine. Si plusieurs voisines satisfont ces critères, le génotype adopté est choisi aléatoirement. La cellule prend ensuite l'état suivant correspondant à son voisinage dans son nouveau génotype.
- Les cellules “quiescent” sont représentées dans la simulation par des cases noires.

4.2.4 Notation

Par la suite, nous considérerons que les cellules peuvent prendre différents états inclus dans un alphabet $\Sigma = \{s_1, \dots, s_{|\Sigma|-2}, s_d, s_q\}$ avec s_d correspondant aux cellules à l'état “Decay” et s_q correspondant aux cellules à l'état quiescent. Parfois, on distinguera aussi les cellules à l'état s_d qui résultent d'un dépassement de l'âge limite $s_{d\text{ naturel}}$, de celles qui résultent de la fonction de transition générée par le génotype de la cellule $s_{d\text{ volontaire}}$.

FIGURE 4.4: Voisinage $VN(p)$ dont la cellule p est susceptible d'adopter le génotype



4.3 Paramètres initiaux

A l'initialisation de la simulation, nous choisissons une taille pour la grille (hauteur largeur), nous avons choisi d'opter pour une matrice torique (comme celle utilisée dans Avida et dans les différentes versions des boucles SDSR), ce qui permet d'éviter les problèmes d'effets de bords. Nous remplissons la grille avec un certain pourcentage de cellules non “quiescent” réparties entre les différents états non “quiescent” possibles (y compris l'état “Decay”). Après l'expérimentation informelle, nous avons choisi d'opter pour un pourcentage de 75% de cellules non neutres (ce qui semble légèrement en-dessous mais proche de la part de l'ensemble des cellules qu'elles représenteront par la suite de la simulation). Ensuite, chaque cellule dans un état non spécial (non neutre non “Decay”) est associée à un génotype généré aléatoirement.

De même, on fixe aussi, lors de l'initialisation, une durée pour la simulation en itérations (durée qu'il sera possible d'ajuster en cours de simulation).

^{3.} voir note 2

4.3.1 Temps de décomposition

Le paramètre T_{decay} correspond au nombre d’itérations moyen nécessaire à une cellule à l’état s_d pour passer à l’état quiescent. Il est à noter que ce nombre est le nombre moyen d’itérations et non le nombre précis d’itérations qui sera nécessaire. Quand une cellule passe à l’état “Decay” une valeur $D_{max}(p)$ est choisie aléatoirement entre $\frac{T_{decay}}{2}$ et $\frac{3 \cdot T_{decay}}{2}$ par une fonction $R(T_{decay})$, c’est ce nombre d’itérations qui lui sera nécessaire pour passer à l’état neutre (le choix de ne pas opter pour un temps de Decay fixe a été motivé par la volonté d’éviter de voir apparaître des mécanismes d’expansions cycliques calées sur T_{decay} . On verra par la suite que l’on voit néanmoins de tels cycles fréquemment apparaître en début de simulation).

Il y avait différentes façon d’utiliser le principe du “Decay”, au terme de la phase d’expérimentation informelle, nous avons choisi d’opter pour des valeurs très élevées pour différentes raisons :

- Éviter la possibilité d’existence de génotypes optimaux indépendamment de l’environnement.
- Faciliter la lisibilité de la simulation en maintenant un nombre élevé de cellules “non vivantes” (aux états “quiescent” s_q ou “Decay” s_d).
- Isoler des groupes de cellules pendant des périodes temporelles importantes pour favoriser leur différenciation génétique.

4.3.2 Âge limite

L’âge limite A_{limite} est un paramètre global (Figure 4.3). Par défaut⁴ $A_{max}(p, t) = A_{limite}$ avec $A_{max}(p, t)$ le nombre maximal d’itérations successives que la cellule p peut avoir été vivante⁵ à l’itération t (si elle a dépassé ce maximum elle “meurt”/passe immédiatement à l’état “Decay” $s_{d\ naturel}$).

Au terme de l’expérimentation informelle, nous avons déduit qu’un “âge limite” très bas (inférieur à 10 itérations) avait plusieurs avantages :

- Un comportement moins chaotique des cellules et l’apparition rapide de structures
- Une compétition entre phénotypes/génotypes plus évidente

Nous avons mené deux simulations de très longue durée (400 000 itérations) avec respectivement une espérance de vie de 4 itérations (c’est à dire moins que le nombre d’états possibles hors s_d et s_q) et de 7 itérations (plus que le nombre d’états possibles hors s_d et s_q).

Il nous a semblé que l’une comme l’autre permettent sur le long terme une diversification des phénotypes même si la diversité semble plus importante avec une espérance de vie de 7 et au contraire les extinctions massives plus importantes avec une espérance de vie de 4. Par la suite, nous avons choisi de poursuivre les simulations tests avec des cellules ayant un âge limite A_{limite} de 7 itérations.

4. hors cas particulier d’existence d’un bonus comme nous le verrons par la suite

5. ni à l’état “quiescent” ni à l’état “Decay”

4.3.3 Nombre d'états possibles

C'est le nombre d'états qu'il est possible d'utiliser dans la simulation sans y inclure les états "spéciaux" comme l'état neutre et "Decay". Nous avons choisi arbitrairement de fixer ce nombre à cinq. Plus d'états signifierait que des structures plus compactes pourraient contenir autant d'informations/avoir le même niveau de complexité que des structures plus larges utilisant moins d'états. Mais utiliser davantage d'états signifie aussi ralentir la simulation.

4.3.4 Taux de propagation

Le taux de propagation est le pourcentage de chance qu'une cellule "quiescent" ou "vivante" lance le processus d'adoption d'un nouveau génome⁶.

Les cellules "vivantes" et les cellules "quiescent" peuvent avoir des taux de propagation différents, néanmoins nous avons choisi ici d'opter pour un taux de propagation de 100% dans les deux cas. Cela semble être le mécanisme le plus intéressant, car il permet aux cellules considérées d'avoir les comportements les plus complexes possibles (en l'absence de génotypes concurrents résolvant⁷ les mêmes situations que lui, un géotype peut se comporter ici exactement comme un automate cellulaire classique). On pourrait théoriquement initialiser l'automate avec une boucle de Byl en réplicateur initial (dans un environnement de cellules quiescent) possédant les règles adéquates et la voir se multiplier exactement comme le ferait l'automate classique avec pour seule limitation les modifications dues aux mutations.

4.3.5 Mutations

Après l'expérimentation informelle, nous avons conclu que la meilleure répartition des mutations était un taux de mutation $T_{muta} = 0,0002$ (à chaque itération chaque cellule a une chance sur 5000 de subir une mutation). Par ailleurs, grâce au codage du géotype par programmation génétique (comme nous le verrons au chapitre 4.3.3), il est possible de graduer les mutations. Au terme de l'expérimentation informelle, nous avons choisi de les répartir ainsi :

- 80% de petites mutations d'amplitude 0,05
- 19,5% de mutations moyennes d'amplitude 0,1
- 0,5% de mutations importantes d'amplitude 0,5

La forme exacte que prennent ces mutations et la signification que prend leur amplitude est précisé dans la section suivante (voir 4.4.3).

6. comme vu précédemment ce processus ne signifie pas forcément qu'une cellule vivante perdra son génome

7. voir note 2

On aurait aussi vraisemblablement pu opter pour une répartition de l'amplitude des mutations selon une courbe gaussienne.

4.4 Différents moyens de représenter les génotypes des cellules

Le rôle des génotypes est de pouvoir donner un état final pour chaque combinaison état/voisinage possible (y compris pour une cellule à l'état “quiescent” s_q puisque ce sera utilisé pour la propagation des génotypes et à l'exclusion de l'état “Decay” s_d puisqu'une cellule à l'état s_d n'a jamais de génotype)

4.4.1 Tableaux

La manière la plus simple consiste à les représenter par un tableau comportant autant d'entrées que de configurations du voisinage possibles, puis faire correspondre chaque configuration à un état final. Et d'associer chaque cellule à un tableau de règles.

Nous avons effectué les premières expérimentations informelles en nous basant sur ce système. S'il a l'avantage de permettre des simulations assez rapides (très rapides pour ce qui est de la fonction de transition, moins rapides pour ce qui est des copies, à moins de résérer en permanence l'espace mémoire nécessaire pour chaque cellule qu'elle soit vivante ou non), il a malheureusement aussi beaucoup d'inconvénients :

- Un tel système nécessite beaucoup d'espace mémoire (proportionnel au nombre d'états et à la taille de la grille et ne permettant pas d'effectuer la simulation avec 7 états et grille de 120 000 cellules)
- le système est assez peu adapté aux modifications inspirées par la génétique.

Pour toutes ces raisons, nous avons choisi de ne pas l'utiliser au terme de la phase exploratoire.

4.4.2 Arbres

Nous avons aussi essayé de représenter les génotypes cellulaires par des R-Tree tels que ceux décrits par Zhijian Pan et James A. Reggia [21]. Ce système a l'avantage de nécessiter beaucoup moins de mémoire et d'avoir prouvé son évolvabilité. Malheureusement, il a un autre inconvénient : il est beaucoup plus coûteux en terme de temps de calcul que le système précédent. A tel point qu'il s'est révélé, lui aussi, inadapté dans le cadre de notre simulation.

4.4.3 Représentation issue de la “Linear genetic programing” (LGP)

Nous utilisons une forme modifiée de programmation génétique (PG) adaptée à la représentation des règles de transition d'un automate cellulaire. Le modèle utilisé est issue de la programmation génétique

linéaire (LGP)[4] et notre version est appelée CA-LGP. Il est à noter que la simulation n'étant pas dirigée nous n'utilisons pas de fonction de fitness.

L'idée générale de la LGP est d'utiliser une suite de registres et d'opérations sur ces registres de manière linéaire. Au départ les entrées du programme (ici les états des différentes cellules du voisinage) sont copiées dans des registres initiaux. Ensuite des registres additionnels, contenant des valeurs constantes, sont listés. Puis, une série d'opérations sont effectués sur ces registres de façon linéaire. Enfin, la valeur du premier registre est renvoyée en sortie du programme.

Par exemple en pseudo java code le programme suivant est un programme généré par LGP qui renvoie la distance entre les points aux coordonnées (x1,y1) et (x2,y2) :

```
double LGP(double x1, double y1, double x2, double y2) {  
    double R1 = x1;  
    double R2 = y1;  
    double R3 = x2;  
    double R4 = y2;  
    double R5 = 0.236194;  
    R5 = R1 - R3;  
    R1 = R5 * R5;  
    R5 = R2 - R4;  
    R2 = R5 * R5;  
    R1 = sqrt(R1);  
    return R1;  
}
```

La LGP est une forme intéressante de programmation génétique :

- Il est facile de générer aléatoirement de tels programmes.
- Il est aussi simple de les faire muter.
- C'est une forme d'algorithme génétique naturellement modulaires (puisque les registres peuvent être réutilisés).
- L'exécution est rapide (puisque les programmes peuvent être directement convertis en langage machine).
- Et il supporte la présence de matériel "génétique" "non-codant"/ "neutre" susceptible d'augmenter l'évolvabilité [36]

Néanmoins, nous devons ici modifier le modèle des LGP pour générer quelque chose capable de représenter les règles d'un automate cellulaire. C'est à dire accepter en entrée le voisinage d'une cellule, et renvoyer en sortie un état parmi les états possibles.

Pour cela nous définissons ainsi CA-LGP (Figure 4.5) :

- une liste de N registres (N correspond au nombre de cellules comprises dans le voisinage + 1, ici 9 puisque nous utilisons le voisinage de Moore VM), ces registres prenant à l'initialisation les valeurs des états des différentes cellules du voisinage (voisinage étendu à la cellule considérée elle-même).

- Une liste de $|\Sigma|$ registres d'état initialisés génétiquement ou $|\Sigma|$ correspond au nombre d'états que les cellules peuvent prendre (c'est à dire : les différents états "vivants", s_q , et $s_{d\text{ volontaire}}$.

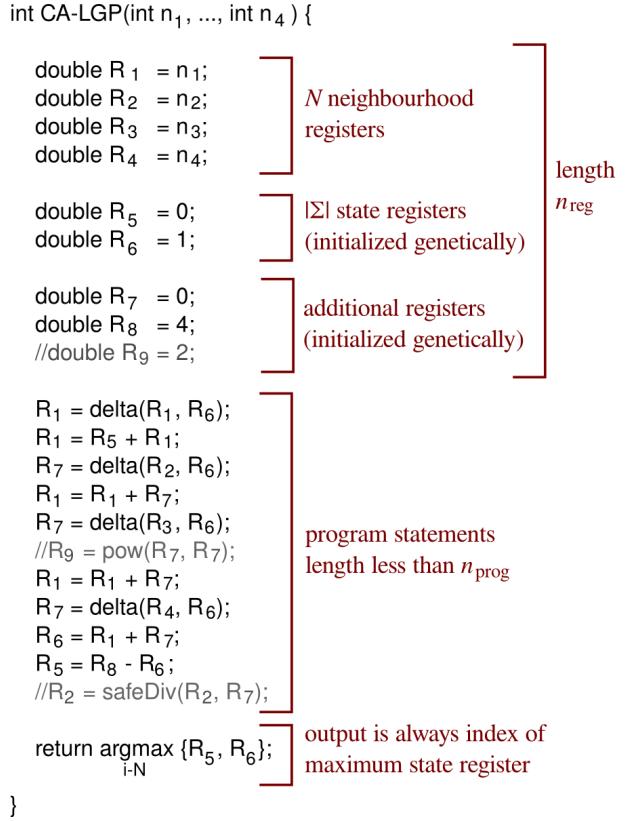
$s_{d\text{ naturel}}$ ne pouvant être adoptés qu'après dépassement de l'âge maximum $A_{max}(p, t)$.

- Une liste de registres additionnels aussi initialisés génétiquement.
- Une liste de n (n étant variable génétiquement) opérations sous la forme suivante : $RX = O(RY, RZ)$, où X, Y et Z sont les indices des registres précédemment définis et où O est un opérateur de la liste des fonctions possibles.
- La valeur retour renvoyée est l'état associé au registre d'état ayant la valeur la plus élevée au terme des opérations.

Voici un exemple d'un tel programme (Figure 4.5) pour un automate cellulaire utilisant le voisinage de von Neumann $VN(p)$ et avec deux états possibles (1 et 2). Le programme commence par calculer le nombre de cellules à l'état 1, puis le nombre de cellules à l'état 2 et renvoie finalement l'état qui est le plus fréquent.

Un tel programme peut être initialisé aléatoirement en :

FIGURE 4.5: Schéma d'un génome CA-LGP, Le code neutre est représenté ici en gris.



- choisissant le nombre de registres additionnels.
- Spécifiant des valeurs pour les registres additionnels et les registres d'états.
- Spécifiant chaque opération, en choisissant quatre valeurs pour X , Y , Z et O où X , Y et Z sont choisies aléatoirement et uniformément parmi tous les registres disponibles et où O est choisi aléatoirement et uniformément, parmi toutes les opérations possibles de la liste de fonctions.

Le processus de mutation, lui, est lancé avec un facteur de mutation p_{mutate} qui détermine l'amplitude de la mutation. Il consiste en deux sous processus.

- La mutation des registres : Chaque registre a une chance p_{mutate} de voir sa valeur d'initialisation modifiée d'une valeur V choisie entre -3 et 3 selon une fonction gaussienne.
- La mutation des opérateurs : Une des trois opérations équiprobables suivantes sera aléatoirement choisie.
 - Supprimer aléatoirement une opération.
 - Rajouter aléatoirement une nouvelle opération (générée aléatoirement) à un endroit du programme choisi au hasard (si le nombre maximum d'opération n'a pas été atteint).
 - Chaque opération à une chance p_{mutate} de subir une modification (qui consiste à voir un de ses registres ou changer l'opération effectuée).

On considère généralement que prendre le nombre le plus important d'opérateurs possible est le meilleur choix pour la programmation LGP [15]. Nous utilisons donc les opérateurs suivants. :

nom	description	action sur les sorties (x,y)
ABS	valeur absolue	$ x $
ADD		$x + y$
DELTA	delta de Kronecker Δ	1, si $ x - y < 1/10000$, sinon : 0
DISTANCE		$ x - y $
INV		$1 - x$
INV2		$SAFEDIV(1, x)$
MAGPLUS		$ x + y $
MAX		$max x, y$
MIN		$min x, y$
SAFEDIV	version sécurisée de la division	x/y , si $ y < 1/10000$, sinon : 1
SAFEPOW	version sécurisée de l'exponentielle	x^y , si défini, 1 sinon
THRESH		1, si $:x > y$, sinon : 0
TIMES		$x \cdot y$
ZERO	Test si vaut zéro	1, si $: x < 1/10000$, sinon : 0

4.5 Les éléments rejetés à l'issues de la phase exploratoire

4.5.1 Bonus

Le principe des bonus d'espérance de vie est de récompenser des comportements utiles. Nous avons, lors des phases exploratoires, essayé d'utiliser ce principe en récompensant les groupes denses de cellules avec deux types de bonus.

- Des bonus cumulatifs : à chaque itération, on a $Gain(p, t) = VM_{vivante}(p, t) - VM_{vivante}(p, t - 1)$ avec $VM_{vivante}(p, t)$ le nombre de cellules vivantes dans le voisinage⁸ de la cellule p au temps t . Si $Gain(p, t) > 0$ le bonus cumulé $B_{cumulé}(p, t)$ de la cellule augmente de B_{aug} pour chaque nouvelle⁹ voisine vivante : $B_{cumulé}(p, t) = B_{cumulé}(p, t - 1) + B_{aug} \cdot Gain(p, t)$. Si $Gain(p, t) < 0$ le bonus cumulé $B_{cumulé}(p, t)$ diminue de B_{dim} pour chaque nouvelle voisine non vivante¹⁰ : $B_{cumulé}(p, t) = B_{cumulé}(p, t - 1) - B_{dim} \cdot Gain(p, t)$. Le bonus $B_{cumulé}(p, t)$ peut être négatif. Il est à noter qu'il est conseillé de choisir des valeurs telles que $B_{aug} < B_{dim}$ si l'on ne veut pas favoriser des comportements oscillatoires et voir des cellules avec une espérance de vie potentiellement infinie. $A_{max}(p, t) = A_{limite} + B_{cumulé}(p, t)$.
- Des bonus non cumulatifs fonctions du nombre de voisines vivantes (la cellule p voit son âge limite affecté par un modificateur $B_{fixe}(VM_{vivante}(p))$ qui est fonction de son nombre $VN_{vivante}(p)$ de voisines vivantes) : $A_{max}(p, t) = A_{limite} + B_{fixe}(VM_{vivante}(p, t))$. Il est possible avec de tels bonus d'essayer de favoriser des comportement proche du Jeu de la Vie en récompensant fortement la présence 4 et 3 voisines vivantes.

Les résultats, lors de la phase exploratoire, n'ont pas été probants (même si des indices laissent supposer que les bonus cumulatifs, nos valeurs de test ayant été $B_{aug} = 3$ et $B_{dim} = 4$, favorisent la mobilité des génotypes. Il est de plus possible que le fait d'avoir une espérance de vie variable soit un désavantage. Par ailleurs il est aussi probable que, lors de cette phase exploratoire, nous n'ayons pas poursuivi suffisamment longtemps ces simulations pour que les bonus aient le temps d'avoir un effet significatif (Peu de simulations exploratoires avec bonus ayant été poursuivies au-delà de 10 000 itérations). Une autre voie notable qui n'a pas été exploitée ici aurait été de faire varier les chances d'adoption des différents génotypes en cas de possibilités multiples lors de la propagation plutôt que de les fixer équiprobales.

8. de Moore

9. par rapport à ce même nombre lors de l'itération précédente

10. à l'état "Decay" ou "quiescent"

4.5.2 Maturité

La maturité consiste à interdire aux cellules de répandre/transmettre leur génotype (que ce soit vers des cellules vivantes ou “quiescent”), si elles ne sont pas vivantes depuis un nombre d’itérations successives supérieur ou égal à A_{mat} . Les effets de ce système étaient les suivants :

- Éviter l’oscillation
- Forte diminution des chances de survie des génomes initiaux générés aléatoirement

Nous avons choisi d’écartier ce paramètre, car il semblait difficile de faire survivre des cellules en utilisant simultanément le principe de la maturité et de la dégénérescence (“Decay”). Or la dégénérescence semblait avoir des propriétés plus intéressantes.

4.6 Analogie biologique

On peut considérer que les êtres vivants qui partagent le plus de caractéristiques avec nos cellules sont sans doute les bactéries et les végétaux :

- tous deux peuvent utiliser des formes de reproductions non sexuée.
- les végétaux sont essentiellement statiques.
- Par ailleurs, visuellement, la simulation évoque la croissance de colonies de bactéries ou de végétaux.

Néanmoins, l’analogie n’est pas parfaite dans le sens où l’un comme l’autre possède des mécanisme d’échange génétique (que ce soit par la reproduction sexuée pour les végétaux et par l’échange de matériel génétique pour les bactéries). On peut aussi rajouter que les bactéries sont mobiles et évoluent dans des milieux qui génèrent des déplacements, ce qui modifie grandement les interactions locales.

Chapitre 5

Analyses des résultats

5.1 Synthèse des résultats

Dans le cadre de cette expérimentation, nous avons effectué 6 séries de simulations :

- Une série de (9+1¹) simulations d'automates hétérogènes initialisés avec les paramètres suivants :
 $A_{limite} = 7$, $T_{decay} = 1850$, $T_{muta} = 1/5000$, pendant 75000 itérations.
- Une série de (7+1) simulations d'un automate hétérogène à règles locales sans mutations avec les paramètres suivants $A_{limite} = 7$, $T_{decay} = 1850$, $T_{muta} = 0$, pendant 75 000 itérations (et une simulation supplémentaire s'arrêtant à 35 000 itérations).
- Une série de 4 simulations d'un automate hétérogène à règles locales sans mutations avec les paramètres suivants $A_{limite} = 3$, $T_{decay} = 1850$, $T_{muta} = 0$, pendant 35 000 itérations.
- Une série de (3+1) simulations d'un automate hétérogène sans “Decay” avec $A_{limite} = 7$, $T_{decay} = 0$, $T_{muta} = 1/5000$ à 75 000 itérations.
- Une série de (5+4) simulations du jeu de la vie initialisées aléatoirement, pendant 75 000 itérations.
- Enfin une série de (13+1) simulations d'automates cellulaires classiques avec des règles de transition homogènes et définies à l'initiation aléatoirement avec le même algorithme que pour le reste de la simulation, pendant 75 000 itérations.

Nous aurions souhaité pouvoir effectuer davantages de tests mais malheureusement le temps imparti ne nous en a pas laissé la possibilité. Il est à noter que les premières séries ont, en plus des simulations mentionnées, abouti à un certain nombre de fois (21 fois pour la simulation avec désactivation des mutations, et 29 pour la simulation normale), à la disparition totale des cellules vivantes² avant la 1 000ème itérations. Nous avons

1. Les “+” indiquent les simulations de durées significativement supérieure à la durée précisée par la suite
2. non “quiescent” et non “Decay”

choisi d'exclure ces simulations des résultats (il semble évident que l'on pourrait réduire ce nombre d'échecs en augmentant la taille de l'espace de simulation). Par ailleurs les mesures étant prises toutes les 150 itérations quant on parle d'un nombre d'itération qui ne soit pas multiple de 150 on prend le multiple inférieur le plus proches ($75000 \Rightarrow 75000$ mais $35000 \Rightarrow 34950$).

Nous avons aussi effectué des simulations de durée plus importante listées dans le détail ici (et signalée précédemment par des “+”):

- Un automate homogène à 200 000 itérations
- Un automate hétérogène avec $A_{limite} = 7$, $T_{decay} = 1850$, $T_{muta} = 1/5000$ à 400 000 itérations.
- Un automate hétérogène avec $A_{limite} = 4$, $T_{decay} = 1850$, $T_{muta} = 1/5000$ à 400 000 itérations (deux valeurs aberrantes dues à un bug ont été enlevées des résultats de cette simulation)
- Un automate hétérogène sans mutations avec $A_{limite} = 7$, $T_{decay} = 1850$, $T_{muta} = 0$ à 400 000 itérations.
- 4 simulation du jeu de la vie à 400 000 itérations.
- Une simulation d'un automate hétérogène sans “Decay” avec $A_{limite} = 7$, $T_{decay} = 0$, $T_{muta} = 1/5000$ à 200 000 itérations.

Pour toutes ces simulations nous avons pris un ensemble d'état Σ tel que $\Sigma = \{s_q, s_1, s_2, s_3, s_4, s_5, s_d\}$ et donc $|\Sigma| = 7$.

Pour chacune de ces simulations nous prenons des mesures à chaque itération jusqu'à la 20ème itération, puis toutes les 150 itérations. Pour certaines d'entre elles, nous prenons aussi des mesures toutes les itérations, lors des 20 dernières itérations.

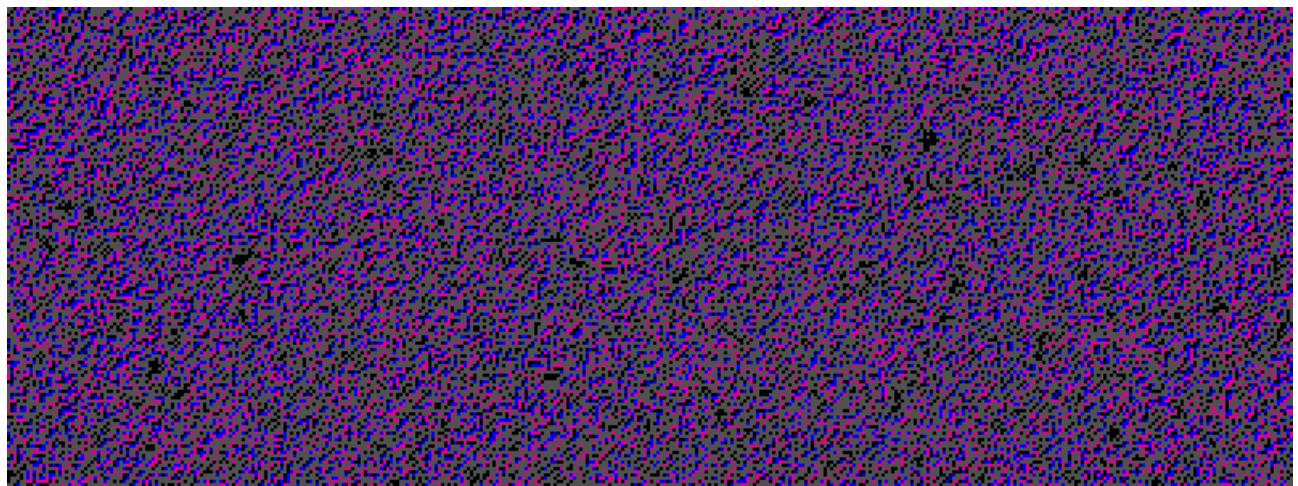
5.1.1 Paramètres importants

Le système de Decay/décomposition a été utile pour au moins deux choses :

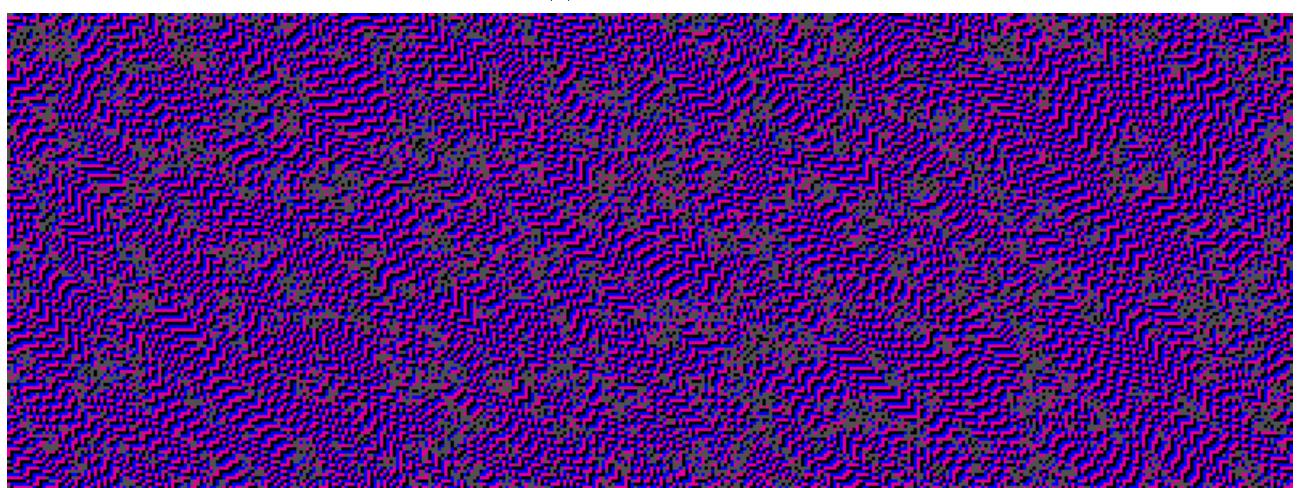
- Il permet de rendre la simulation plus aisément lisible visuellement (il permet de maintenir un nombre important de cellules “non vivantes” (s_q ou s_d), malgré la pression de sélection qui pousse à une augmentation du nombre de cellules “vivantes”).
- Mais il permet aussi de limiter l'efficacité d'un génotype “parfait” de type “quel que soit mon environnement je renvoie A (avec $A \neq s_d$ et $A \neq s_q$) comme état suivant”. Une telle règle aboutit, avec la combinaison Decay + Age limite, à l'extinction rapide du génotype qui l'utilisera par saturation de l'espace puis mort de “vieillesse”.
- Enfin on constate, sur cette simulation sans mutation (Figure 5.1), qu'il semble favoriser la sélection de génotypes mettant un temps important à aboutir à un phénotype stable (attracteur) ou (dans d'autres simulations) qui ne se stabilise pas.

FIGURE 5.1: Développement d'un environnement sur un temps long à partir d'un phénotype ($A_{limite} = 7, T_{muta} = 0$)

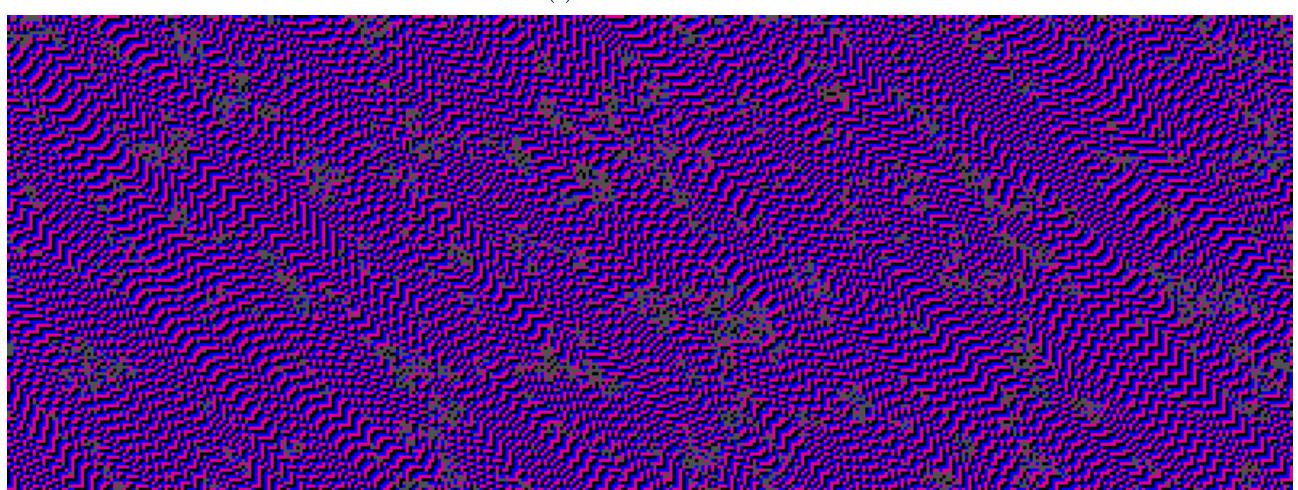
(a) Itération : 4350



(b) Itération : 25950



(c) Itération : 34650



L'espérance de vie, offre des possibilités en termes de récompenses de comportements souhaités. Même si cela n'a pas été utilisé lors des simulations retenues, le principe a un peu été exploré lors des simulations informelles avec le système des bonus. Par ailleurs, ce système permet aussi de fixer une espérance de vie assez basse, comme cela a été fait ici avec une espérance de vie de 7 (et même 4) itérations, et cette contrainte semble aboutir à des comportements moins chaotiques. C'est du moins ce qui nous a semblé lors des tests exploratoires, mais cela mériterait sûrement une étude plus approfondie. Bien sûr le système d'âge limite est par ailleurs indispensable pour mettre en place le système de "Decay".

Le taux de mutation a semblé avoir, durant les tests exploratoires, et même si nous n'avons pas effectué de mesures précises, un effet très important sur le type d'évolution que nous obtenons (quelque chose de très chaotique, quand on utilise un taux de mutation élevé, contre quelque chose de trop statique restant facilement dans des optimum locaux avec un taux de mutations trop bas). Au final la meilleure solution a semblé être d'opter pour un taux de mutations assez bas (à chaque itération, une cellule a une chance sur 5000 de subir une mutation) et surtout d'alterner entre différents types de mutation (forte, moyenne, et petite) pour obtenir des variations soutenues dans le temps.

5.1.2 Émergence de structures à partir de génotypes aléatoires

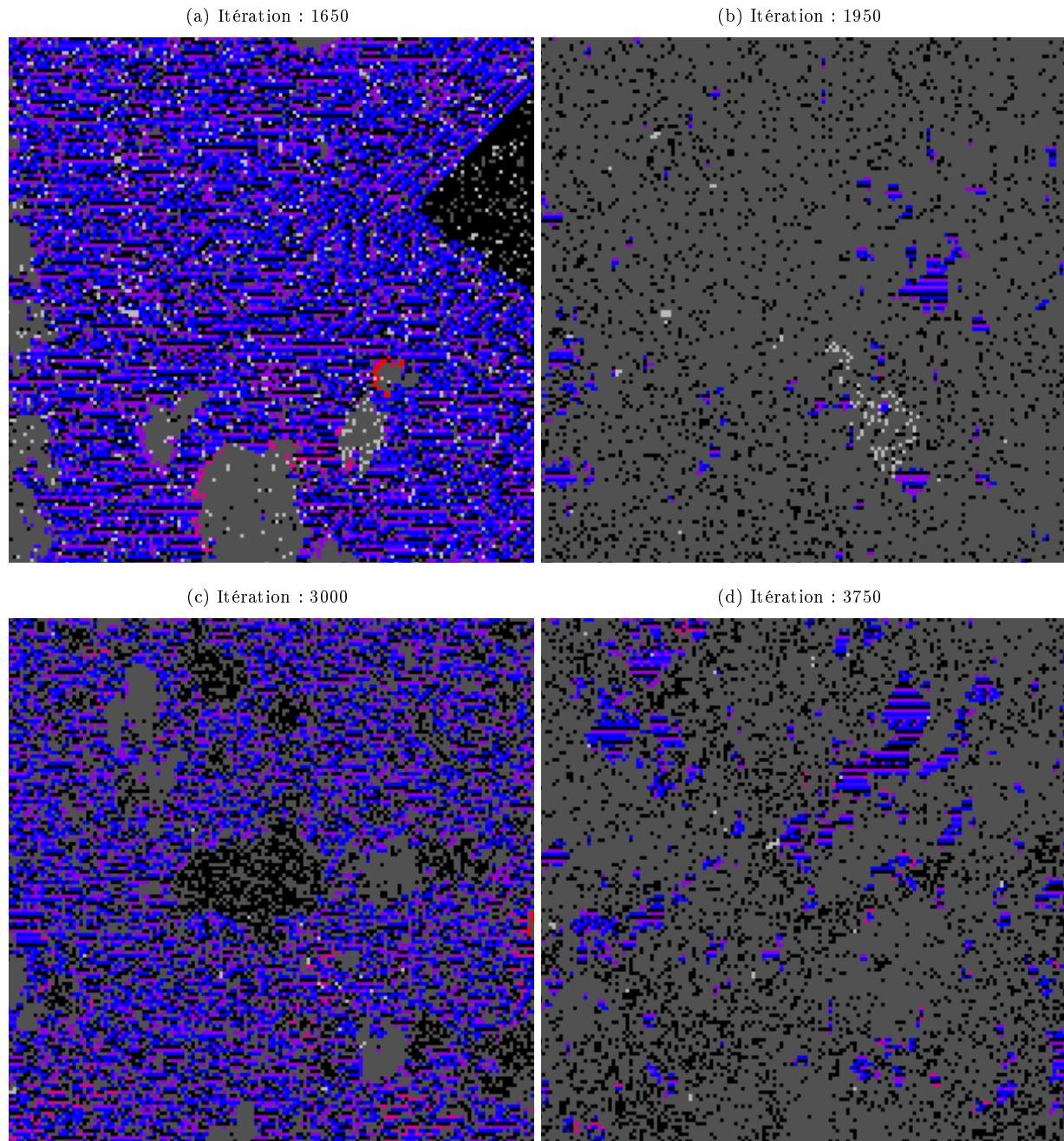
Il est à noter qu'une proportion importante des simulations n'aboutit pas (c'est à dire qu'aucune cellule vivante ne subsiste après 1 000 itérations). Pour les simulations qui aboutissent à des résultats non triviaux³, seulement un petit nombre des génotypes de départ survit après les 50 premières itérations . Ensuite, après plusieurs centaines de cycles, quelques génotypes commencent à occuper rapidement l'espace dans une phase évoquant l'expansion d'une tumeur. Ce, ou ces, génotype survivants étendent progressivement une structure/configuration dans l'espace. (Figure 4.1)

Généralement cette phase d'expansion est suivie de plusieurs phases d'extinction/expansion, (Figure 5.2) sûrement liées à la durée importante du Decay et au caractère simultané de la mort des génotypes générés aléatoirement initialement.

Enfin, comme nous le verrons plus loin, assez rapidement, une diversification et une compétition entre génotypes semble s'installer. Il est à noter que les premières structures à émerger semblent être le résultat d'une première sélection (générée par les principes de fonctionnement de ces automates hétérogènes : Decay, âge limite et système de propagation) sur la diversité des génotypes générés aléatoirement, lors de l'initialisation. Car des structures similaires apparaissent aussi dans les simulations sans mutations (Figure 5.3). Par contre, dans les simulations sans mutations, on aboutit souvent rapidement (mais comme nous l'avons vu

3. On entend par là, les simulations pour lesquelles il reste des cellules vivantes après la millième itération. Ces simulations représentent environ un tiers du total.

FIGURE 5.2: Exemple de phase d'expansion / extinction de début de simulation



pas systématiquement) à l'homogénéisation de la matrice avec l'extension d'une configuration stable dans le temps à l'ensemble de l'espace (attracteur), tandis que les simulations avec variations/mutations semblent empêcher une stabilisation définitive.

5.1.3 Différencier les phénotypes

Nous pouvons aisément remarquer différents phénotypes que ce soit par :

- l'utilisation d'un état particulier (Figures 5.4a et 5.4b).
- la présences de différentes configurations locales répétées (Figures 5.5a, 5.5b et 5.4b)

On peut aisément imaginer que cela mène à différents environnements et donc, potentiellement, à différentes formes d'écosystèmes.

5.1.4 Interactions locales entre phénotypes

On peut voir des phénomènes apparaître dans la zone de contact entre deux génotypes, dans cette simulation sans mutations, seules deux cellules (donc deux génotypes parmi les génotypes initiaux vu qu'elles sont très éloignées géographiquement) ont survécu à l'extinction initiale, ce que l'on peut voir à l'itération 450 (Figure 5.6a). Nous pouvons constater que le phénotype de l'un des génotypes est modifié au contact de l'autre géotype/phénotype (on distingue 3 phénotypes aux itérations 1650 et 1800 (Figure 5.7a et 5.7b) alors que la situation à l'itération 1350 (Figure 5.6b) ne montre que deux phénotypes, or ce troisième géotype ne peut être dû à une mutation (adaptative) vu que cette simulation n'incluait pas de mutation. Enfin on remarque qu'après la stabilisation (Figure 5.7c) deux phénotypes (très proches l'un de l'autre, diagonale droite et gauche) subsistent, bien qu'ils partagent sans doute le même génotypes. Le deuxième phénotype est essentiellement présent dans les zones anciennement occupées par les cellules à l'état s_d (Decay) et le phénotype violet. On peut en conclure que l'environnement semble influer sur l'expression d'un génotype dans notre simulation.

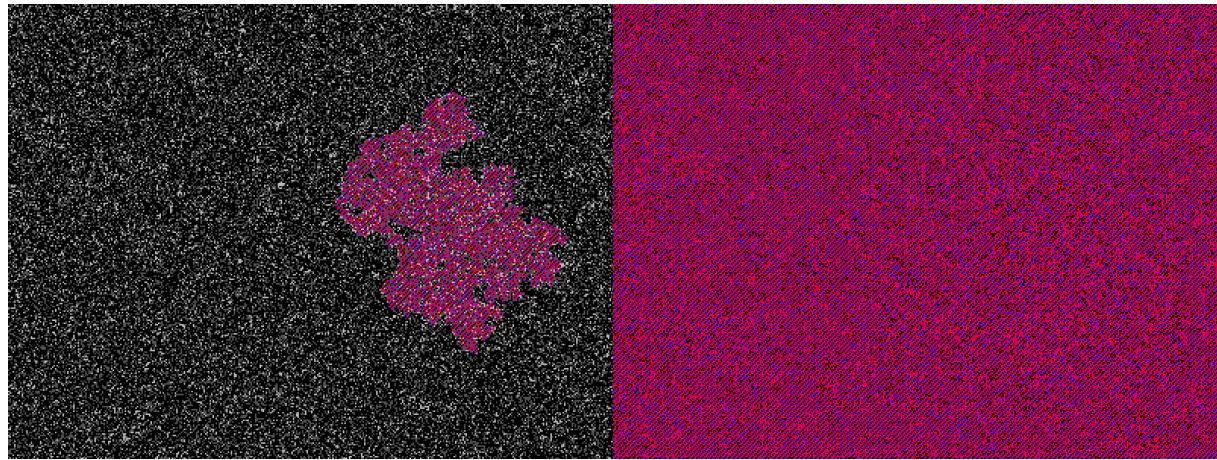
De même d'autres simulations sans mutations semblent montrer que deux phénotypes (Figure 5.8a), et donc sans doutes deux génotypes, peuvent subsister simultanément sur le long terme (Figure 5.9c) dans notre simulation. Ce qui semble suggérer que des mécanisme de symbioses sont possible dans une telle simulation.

5.1.5 Comportement typique du vivant

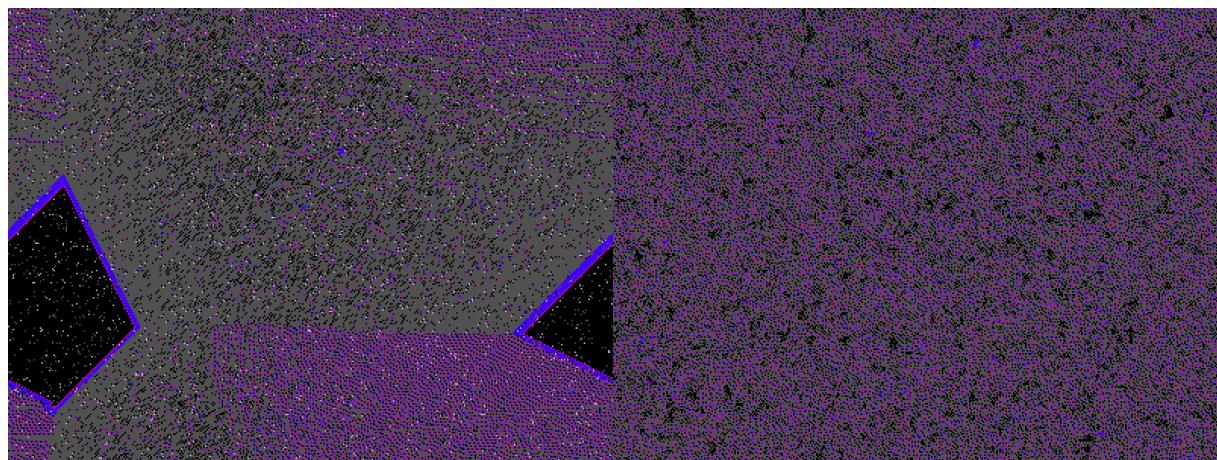
Les phases de croissances que nous pouvons observer lors des premières itérations de la simulation, la stabilité de certaines structures, de même que les irrégularités des configurations que l'on voit émerger, nous évoquent visuellement des caractéristiques que l'on retrouve aussi dans le monde du vivant.

FIGURE 5.3: Phase de croissance et forme stabilisée (attracteur) de différentes simulations sans mutations.

(a)



(b)



(c)

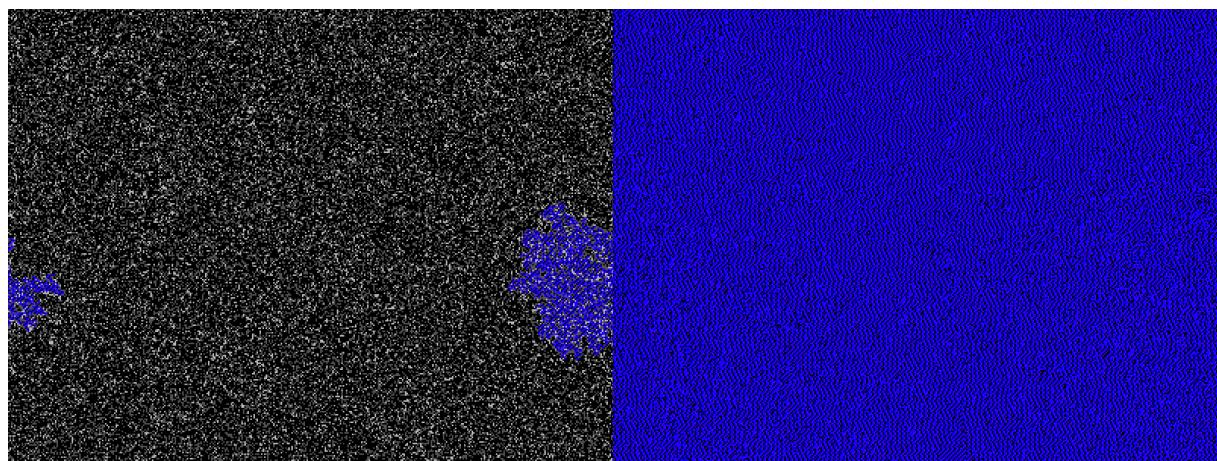
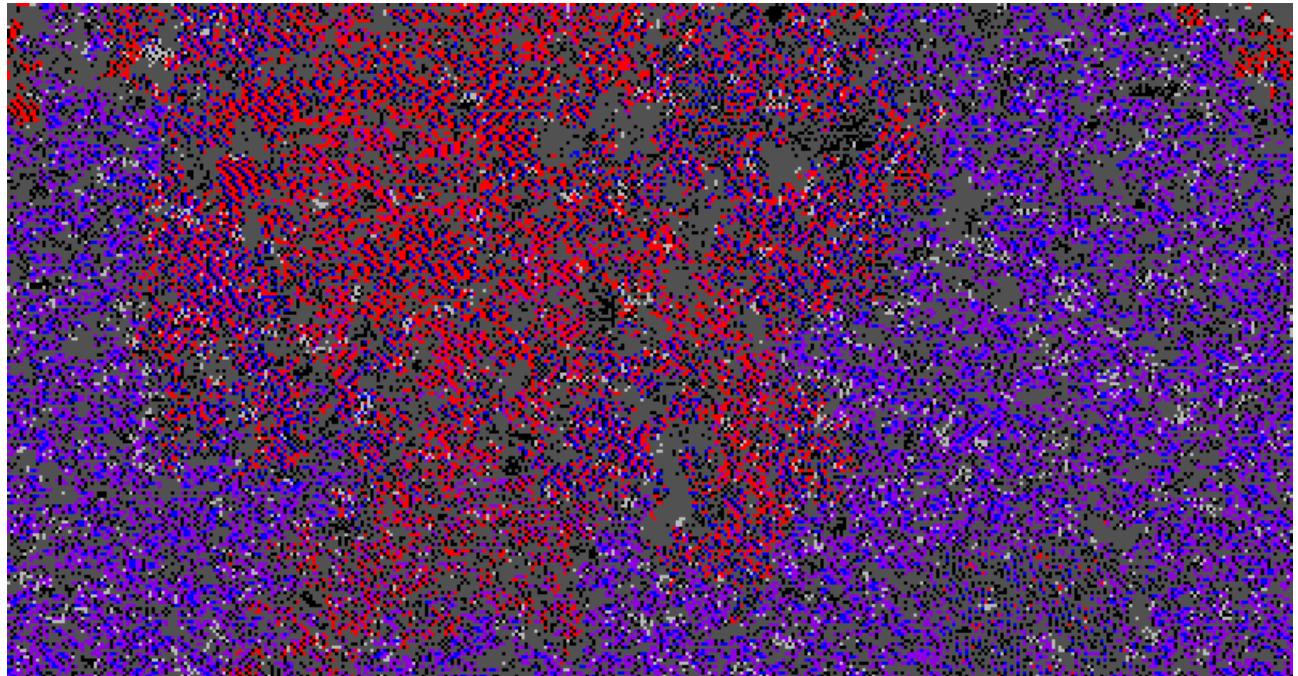


FIGURE 5.4: Diversité phénotypique (Disparité au niveau des états)

(a) Disparité des états (ici de façon évidente on identifie facilement des phénotypes en fonction de l'état des cellules)



(b) Disparité des structures et des états (les groupes de cellules rouge en haut à droite ne sont ni organisés spatialement similairement ni aux mêmes états que les autres cellules)

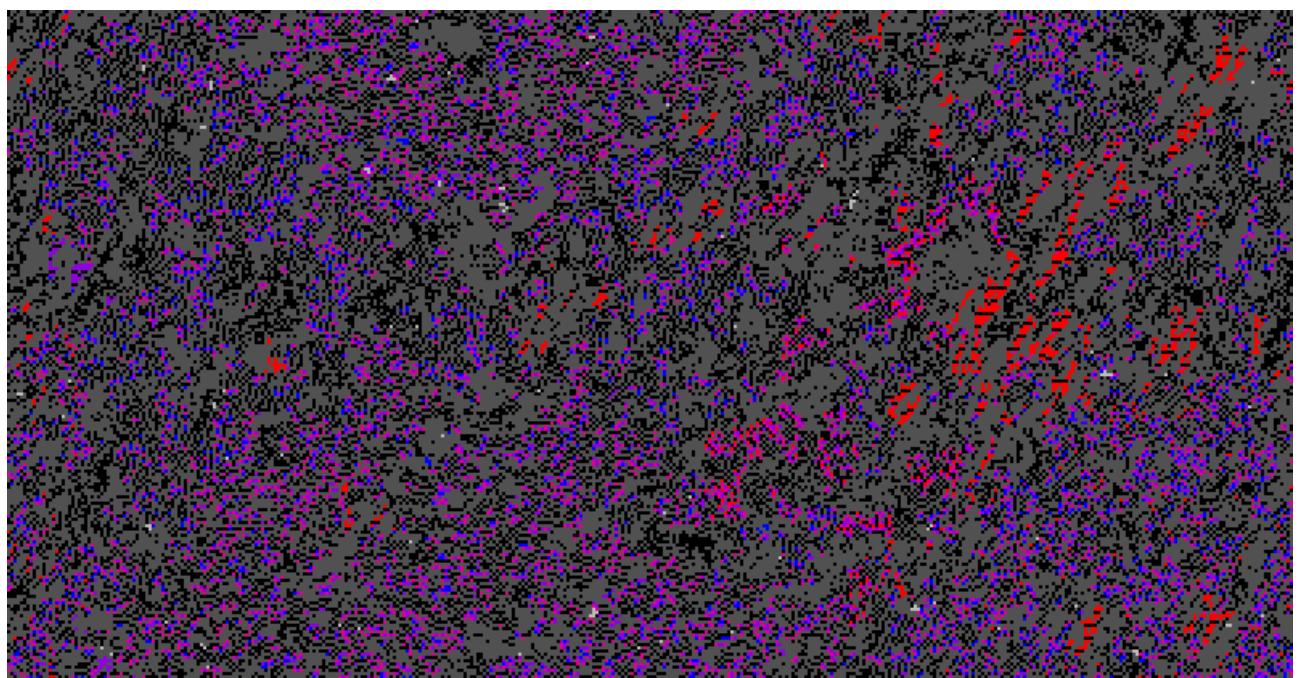
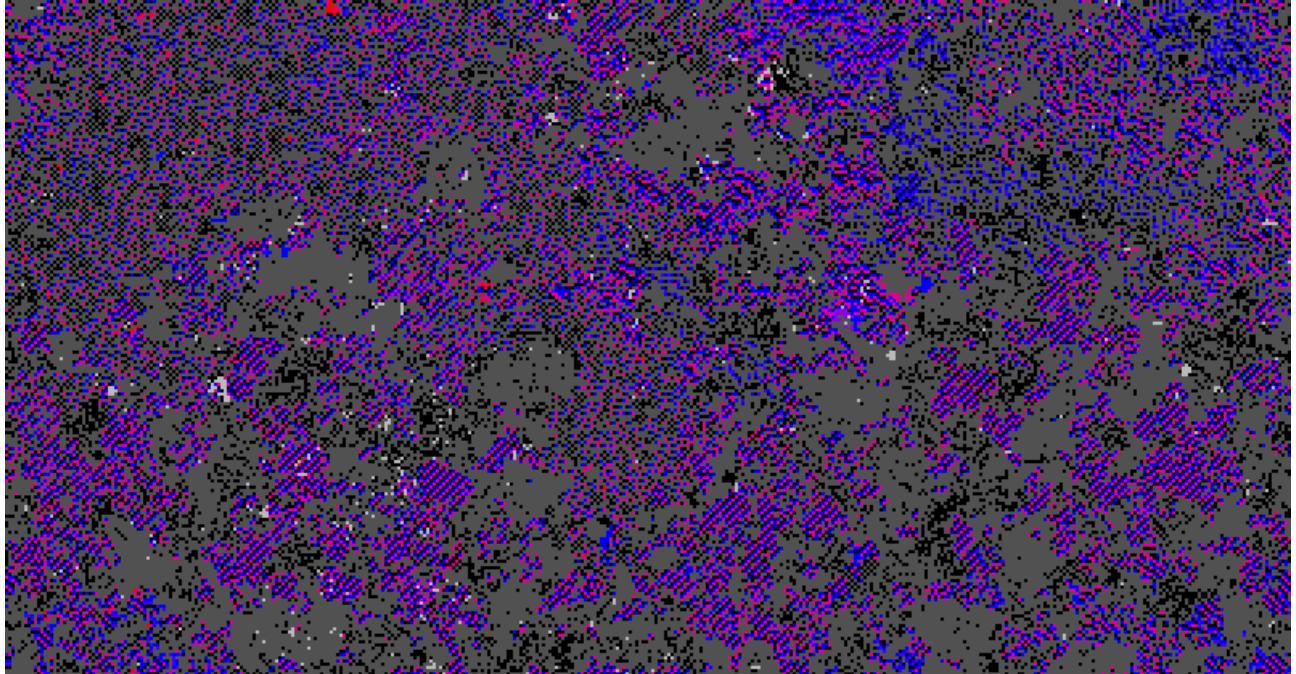


FIGURE 5.5: Diversité phénotypique (Disparité au niveau des configurations locales)

(a) Disparité des structures (on retrouve des zones de peuplement denses où les disparités d'état font apparaître des diagonales, tandis que d'autres zones sont moins denses et ne font pas apparaître de diagonales)



(b) Disparité des structures (on peut observer des groupes de cellules avec différentes organisations spatiales : tantôt des bandes horizontales, tantôt des agglomérations de points)

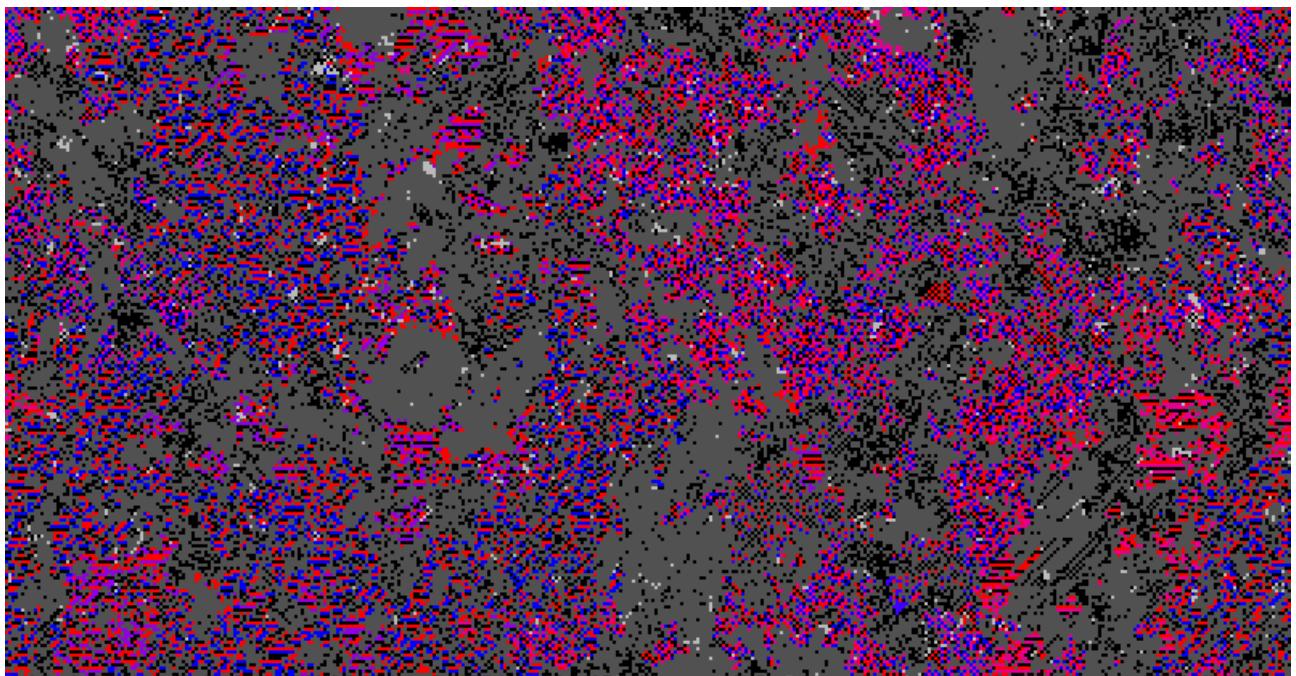
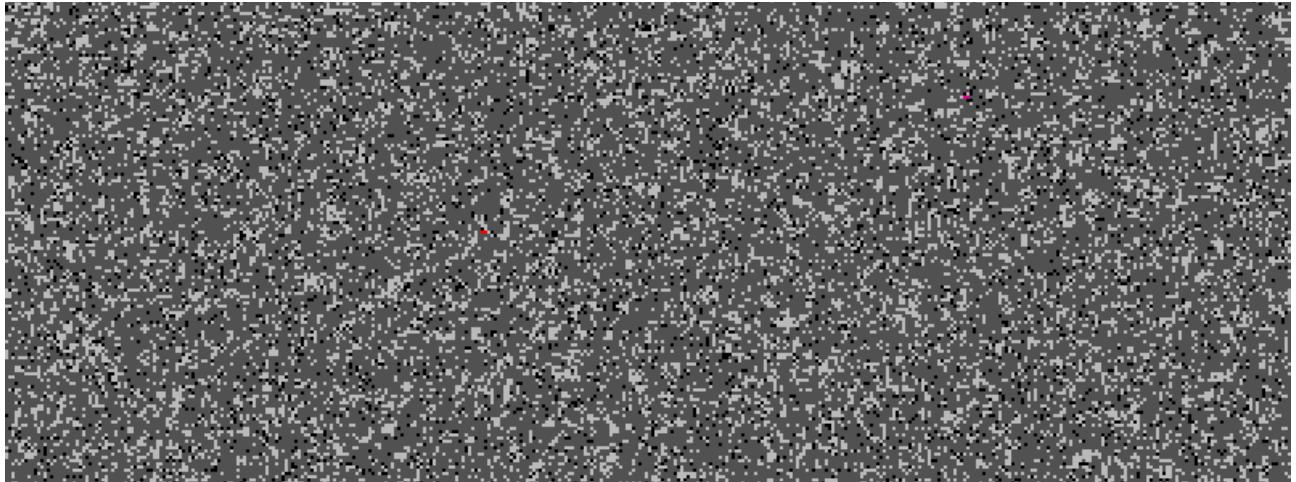
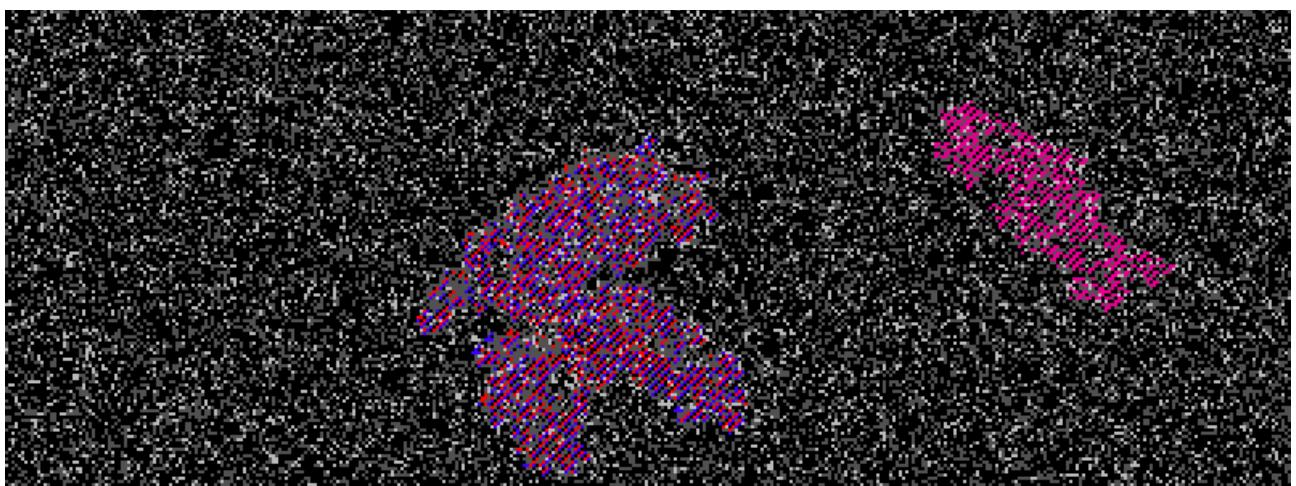


FIGURE 5.6: Développement différent d'un phénotype

(a) Itération 450



(b) Itération 1350



(c) Itération 1500

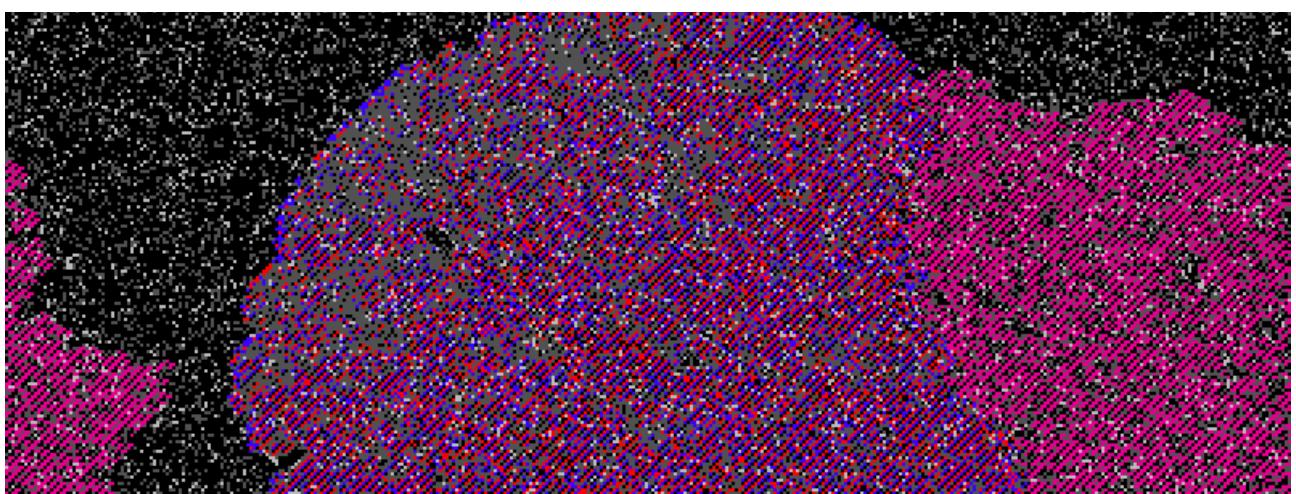
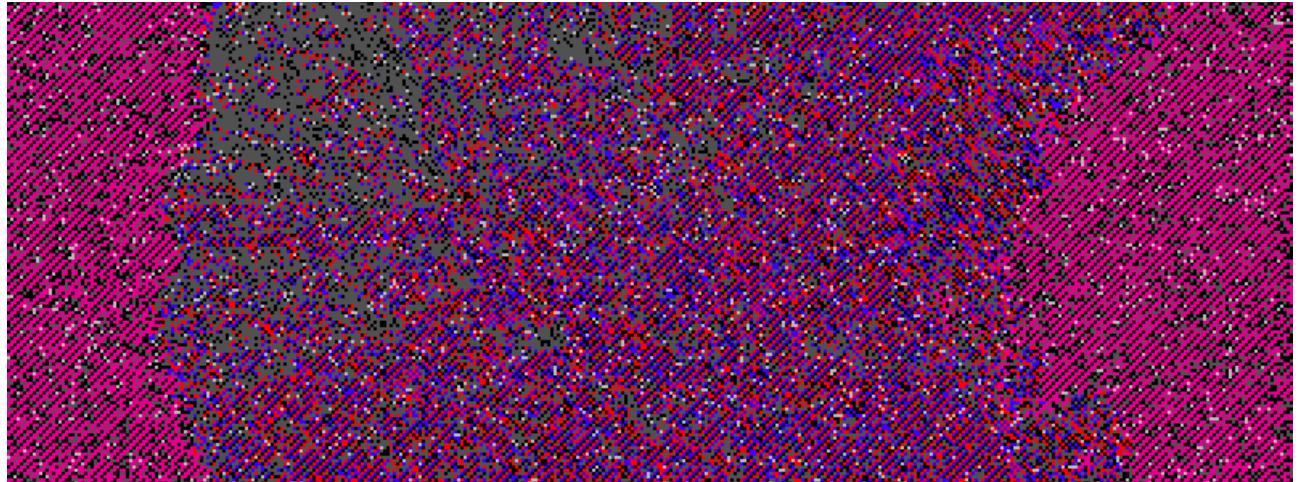
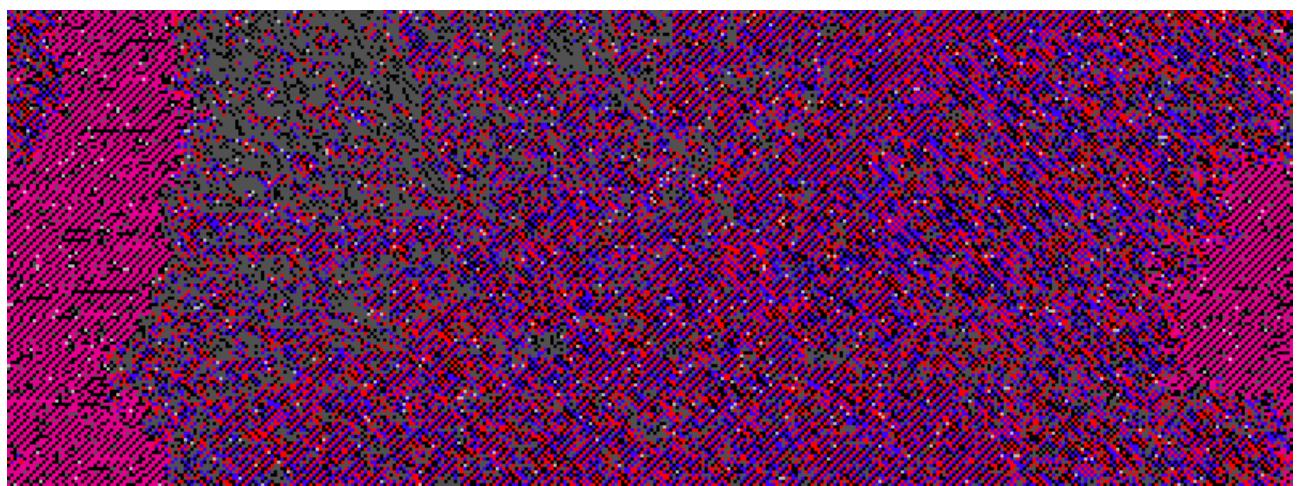


FIGURE 5.7: Développement différent d'un phénotype

(a) Itération 1650



(b) Itération 1800



(c) Itération 4050

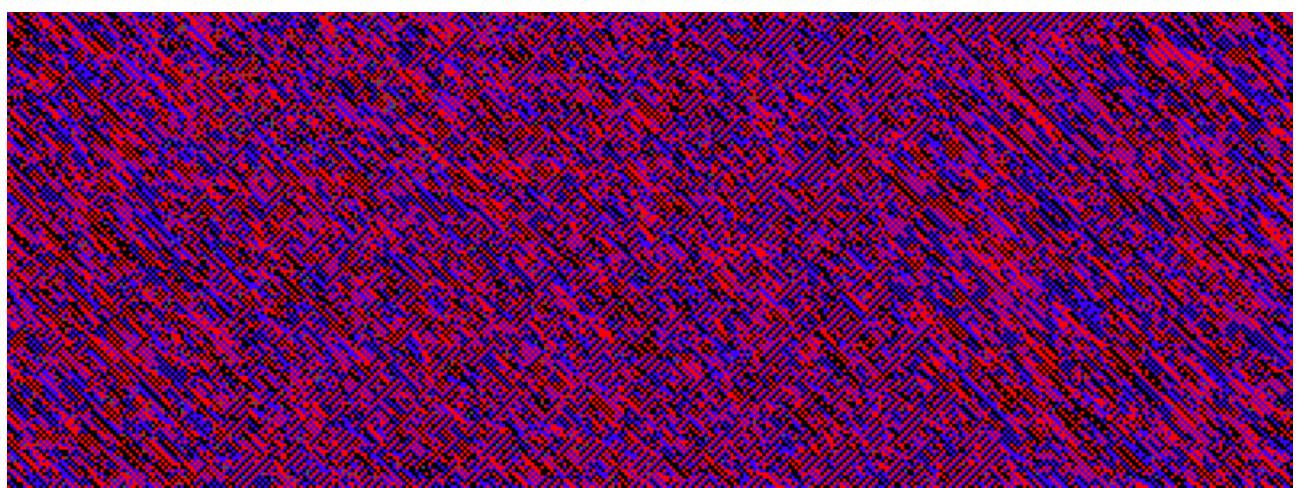
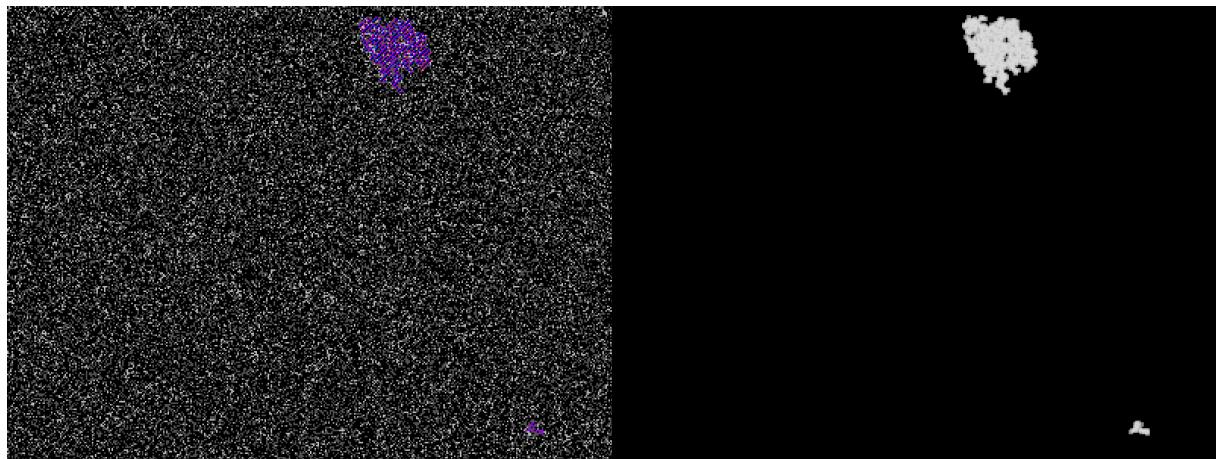
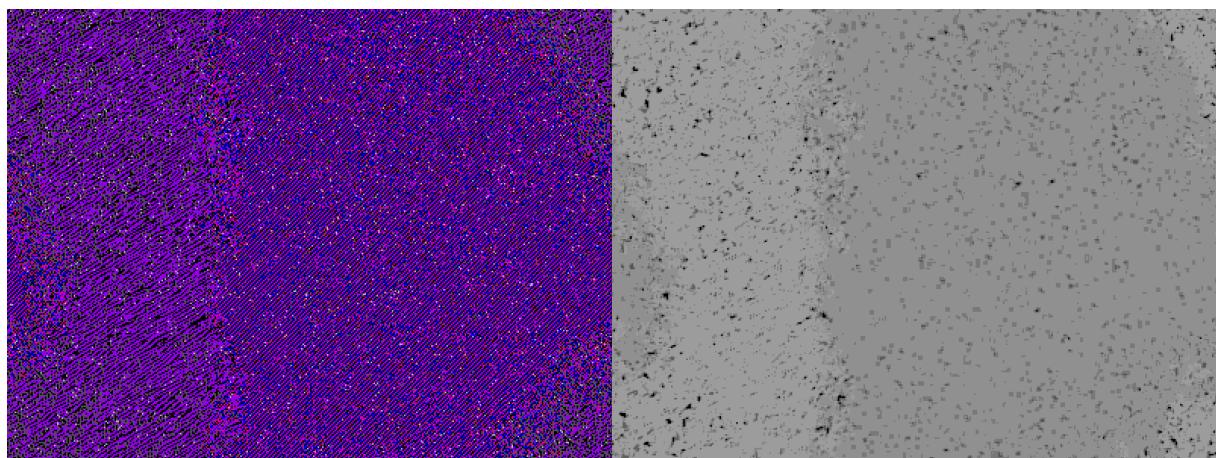


FIGURE 5.8: Diversité génotypique persistante sans mutation (représentation graphique et mesure locale ECF_{glob})

(a) Itération :1350



(b) Itération :1800



(c) Itération :3300

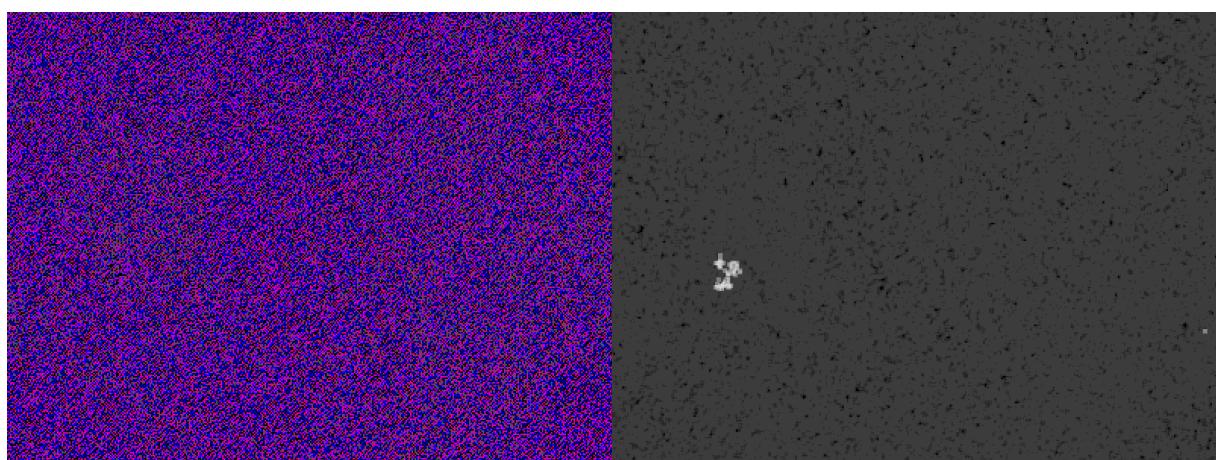
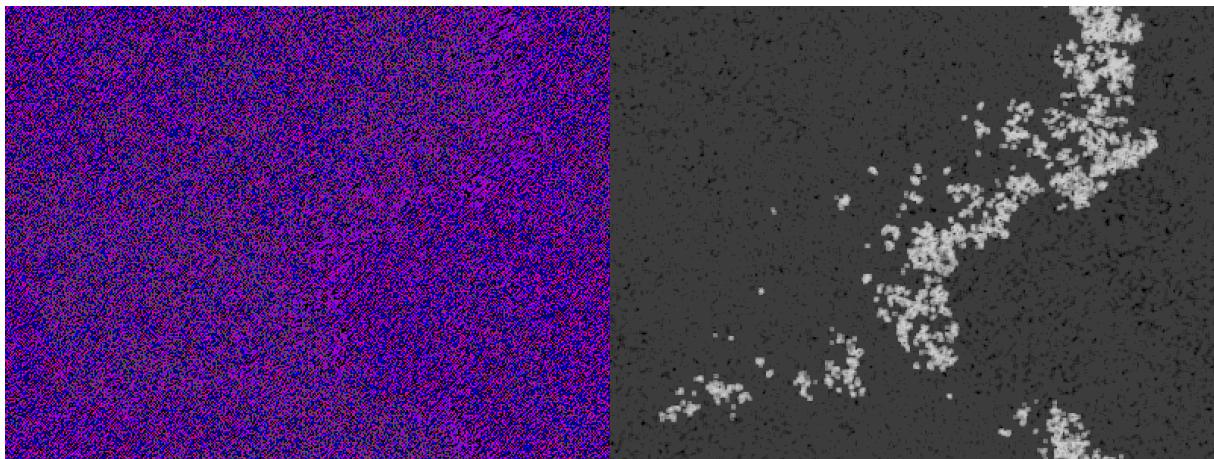
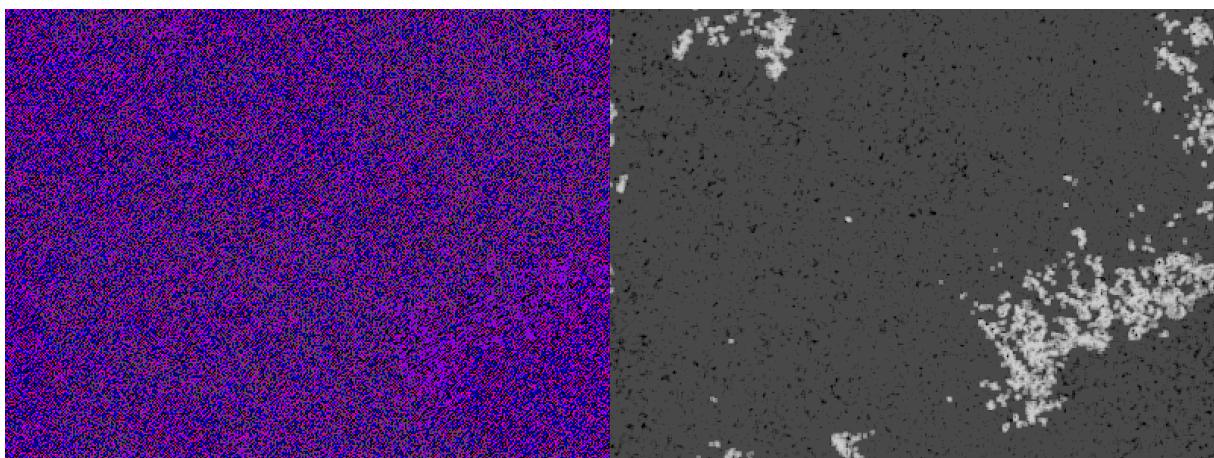


FIGURE 5.9: Diversité génotypique persistante sans mutation (représentation graphique et mesure locale ECF_{glob})

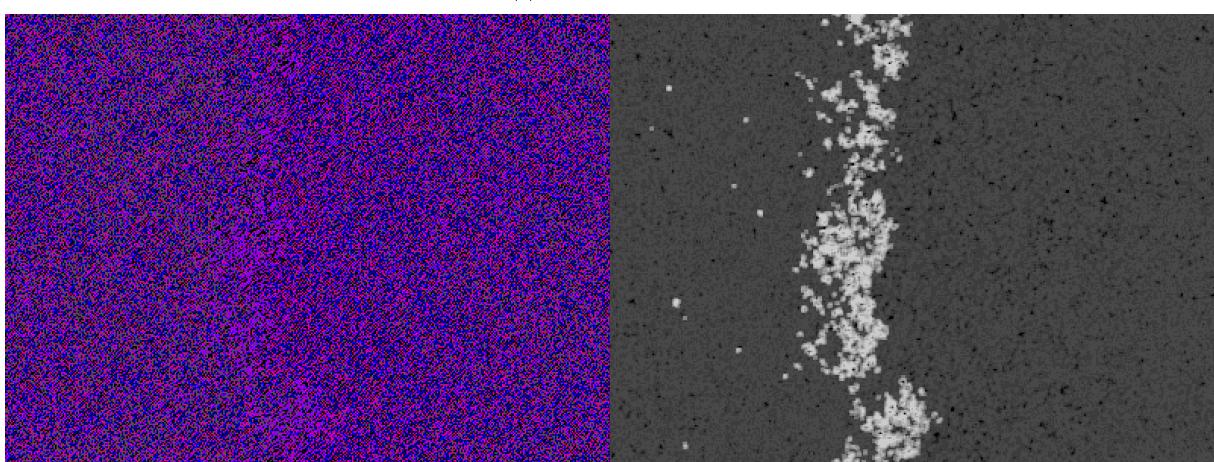
(a) Itération :10950



(b) Itération :11250



(c) Itération :72450



Et en effet on retrouve des comportements dynamiques complexe avec les trois caractéristiques (dynamique émergente : non prédictible, non mécanique et spontanée) Mauno Rönkkö caractérise ainsi les comportement typique du vivant :

We have examined the model of the ecosystem for interesting emergent dynamics with lifelike properties. In the ecosystem, basic emergence was found in the interaction dynamics of elements and organisms. We analyzed the dynamics of six nontrivial scenarios : formation of rivers and ponds, grass growing in rain, worms finding edible grass, a beetle jumping and correcting its orientation, beetles hunting worms, and the environment affecting the global dynamics. Each of these scenarios exhibited distinct emergent dynamics, and in each scenario the dynamics showed nonmechanistic, unpredictable, and sometimes even spontaneous characteristics. In this article, we considered these properties descriptive of lifelike behavior[25].

5.2 Compétition entre les phénotypes

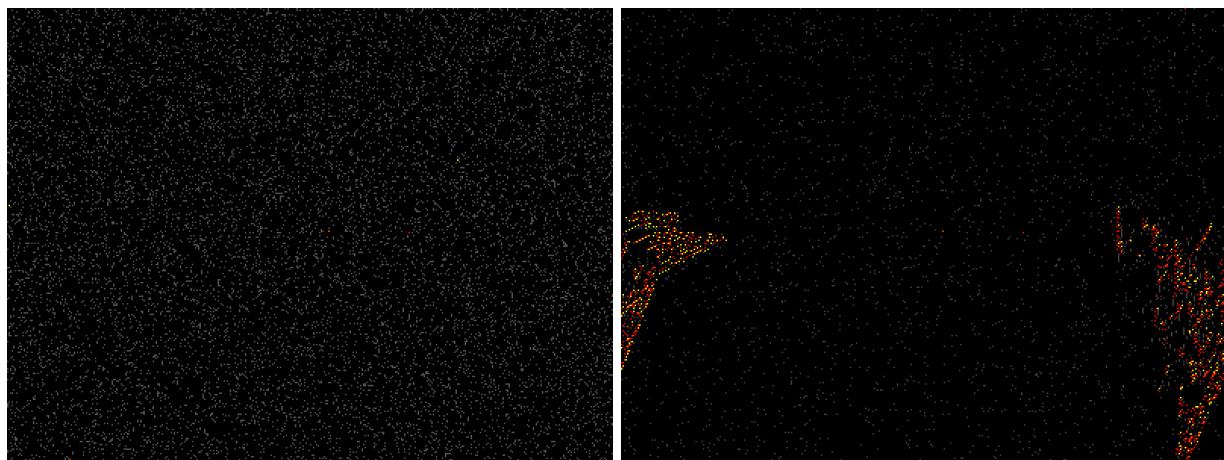
Pour démontrer définitivement qu'il existe une compétition entre les génotypes des cellules, il serait nécessaire de se livrer à une analyse de ces mêmes génotypes (surtout que, comme nous l'avons remarqué précédemment, il semble qu'un même génotype puisse donner lieu à expression de plusieurs phénotypes). Néanmoins en attendant une telle analyse, on peut remarquer différentes formes de compétitions au minimum au niveau phénotypique :

- Même si il n'y a pas de résultat définitivement concluant sur ce point (on peut remarquer que le Decay volontaire n'a d'intérêt que pour des cellules se trouvant confrontées à un génotype différent générant des configurations qu'elles sont incapables de résoudre dans les autres cas de compétition. En effet il semble plus intéressant de se laisser mourir). Certaines simulations préliminaires suggèrent, qu'à certains moments, le Decay est utilisé dans la compétition entre génotypes (Figure 5.10 et 5.11). Par ailleurs, si le Decay volontaire n'a pas d'intérêt évolutif, le nombre de cellules à l'état s_{decay} devrait logiquement être corrélé au nombre de cellules vivantes (dans un état qui ne soit ni s_d ni s_q), puisque qu'une cellule ne peut passer à l'état s_{decay} qu'en ayant été immédiatement précédemment vivante. Or on constate parfois des pics de s_{decay} qui commencent juste après une chute du nombre total de cellules vivantes (qui par ailleurs correspond généralement à un moment de compétition entre plusieurs phénotype/génotype - Figure 5.13). Ce qui peut suggérer que l'état s_{decay} peu jouer un rôle dans la compétition entre phénotypes. De plus, dans l'une des deux simulations de longue durée $A_{limite} = 7$, plusieurs phénotypes ont semblé générer des cellules à l'état s_{decay} (Figure 5.11). Il n'est néanmoins pas certain que cette caractéristique de ce phénotype ait été un réel avantage

FIGURE 5.10: Intérêt du “Decay” dans la compétition entre phénotypes/génotypes

(a) Itération : 900 (seuls quelques génotypes ont survécu à la phase d'extinction initiale)

(b) Itération : 900 (Deux génotypes sont en expansion rapide) le génotype de droite, à dominante rouge, génère des trainées le génotype de gauche, à dominante jaune, génère une structure plus dense.



(c) Itération : 1950 (L'expansion du génotype le plus dense semble bloquée par les cellules en Decay générées par son concurrent)

(d) Itération : 3300 (Seul le phénotype/génotype générant des cellules en “Decay” a survécu)

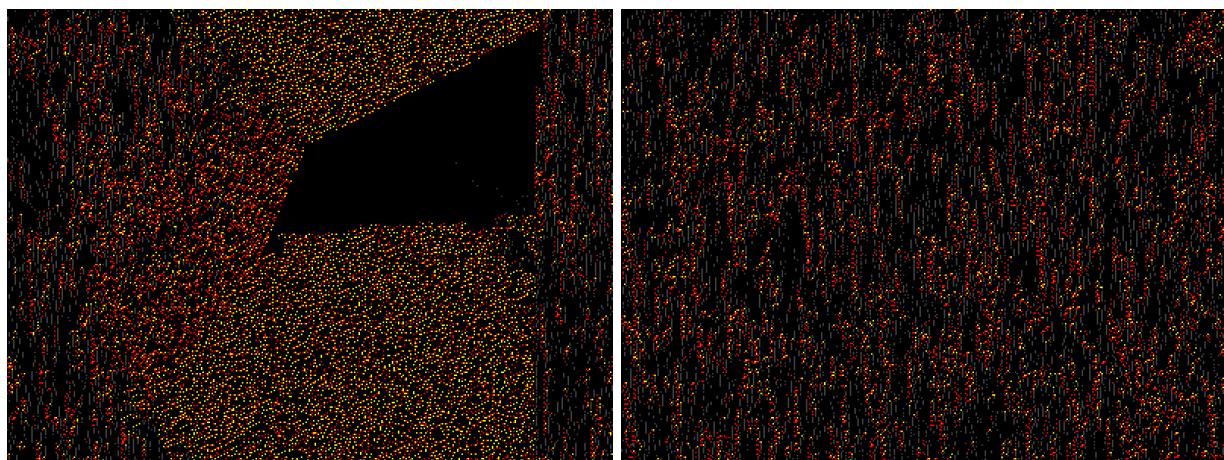
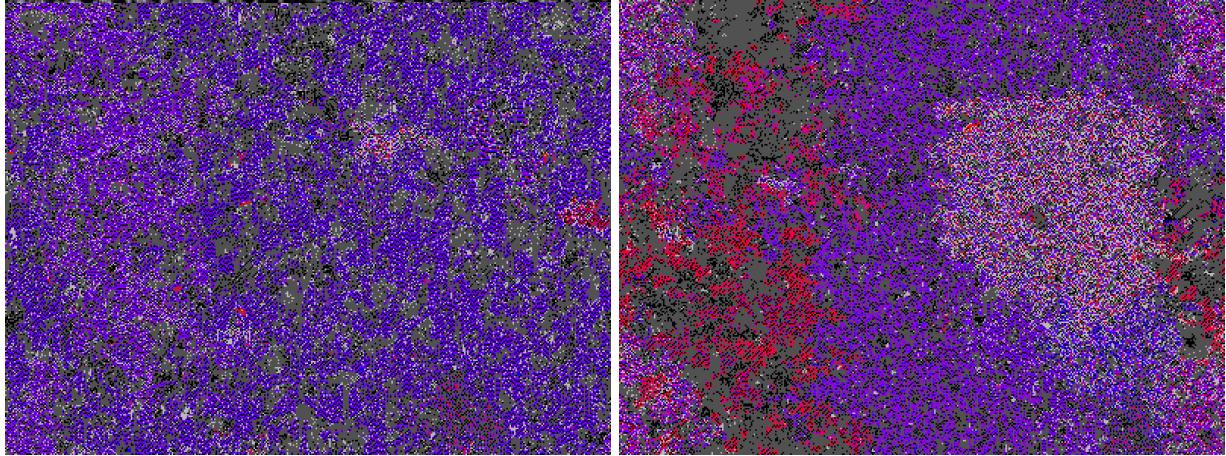
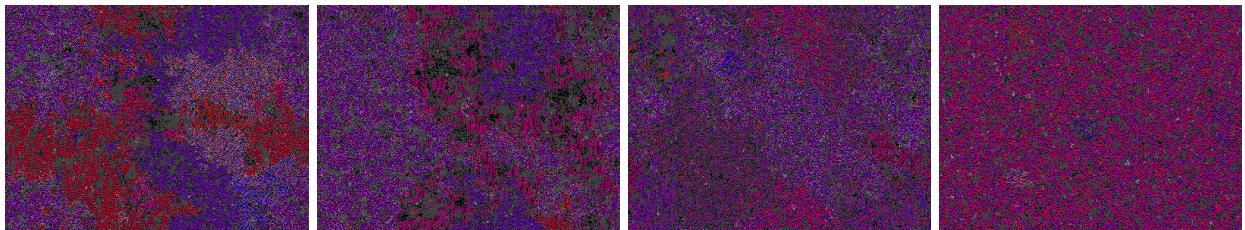


FIGURE 5.11: Utilisation du Decay dans la lutte entre phénotypes dans une des simulation de longue durée ($A_{limite} = 7$, $T_{decay} = 1850$ et $T_{muta} = 1/5000$)

(a) Des phénotypes comportant une densité importantes de cellules à l'état $s_{d\text{ volontaire}}$ peuvent parfois l'emporter sur des phénotypes en comportant moins. Difficile de savoir si le phénotype victorieux l'emporte grâce ou malgré sa capacité à utiliser le Decay volontaire (itérations 9500 et 20250)



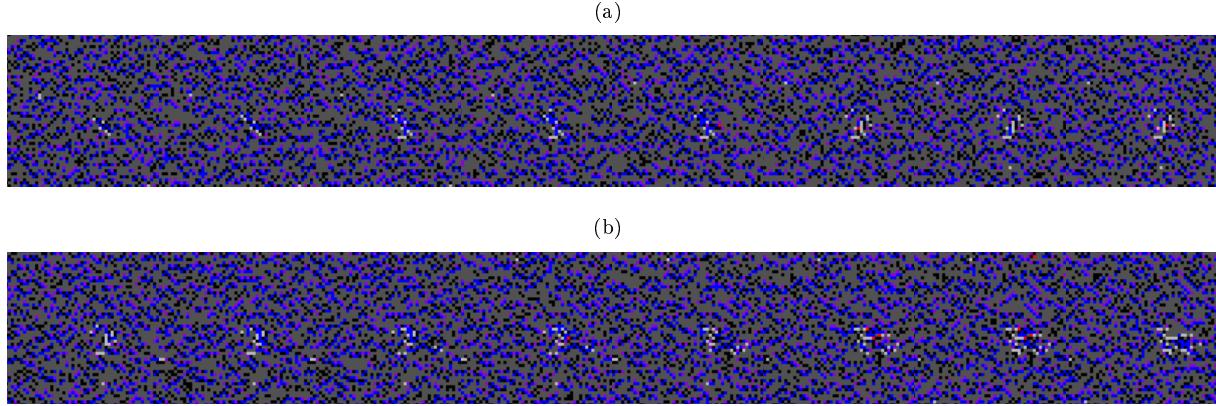
(b) Les phénotypes comportant un nombre important de cellules à l'état $s_{d\text{ volontaire}}$ ne disparaissent qu'après l'itération 150~000. Là encore il semble que le Decay volontaire ait été sélectionné comme méthode efficace dans la lute entre génotype même si l'on ne peut pas affirmer avec une certitudes totale qu'il n'est pas apparu par hasard et a mis longtemps à être éliminé par sélection naturelle malgré son caractère délétère. Cela semble néanmoins peu probable dans cet exemple puisque nombre de phénotype ne comportant pas de cellule à l'état s_d sont apparus et ont rencontré moins de succès que leur contemporain utilisant s_d (itération 26250, 40650, 101400 et 150150)



pour le génotype associé (on peut aussi imaginer que c'est un défaut qui justement explique à terme la disparition du couple phénotype/génotype et donc que ce phénotype a initialement survécu malgré cette caractéristique et non grâce à elle). Enfin toujours au sujet du Decay volontaire, il est à noter qu'il semble significativement plus présent lors des simulations avec $A_{limite} = 7$ qu'avec un $A_{limite} = 4$. Cela est peut être du au fait que plus l'espérance de vie est basse, plus la sélection est importante. Mais on peut aussi considérer que l'intérêt du Decay volontaire (dans les cas où il est intéressant) augmente quand l'efficacité de la stratégie qui consiste à simplement se laisser mourir de vieillesse diminue (or celle-ci est moins efficace sur de longues durées et toujours face à des génotypes concurrents générant des configurations de cellules non résolues)

- De même la densité en cellules vivantes, que permet un génotype, semble parfois aider à sa propagation (Figure 5.14)

FIGURE 5.12: Utilisation du Decay dans la lutte entre phénotypes : snapshot toutes les 150 itérations lors d'une simulation extraite de l'expérimentation informelle ($A_{limite} = 7$) il est à noter que si l'apparition de cellules aux états s_5 (rouge) et $s_{dvolontaire}$ (gris claire) semble corrélé il est difficile de savoir si le passage à s_d est utilisé par le phénotype bleu pour bloquer un phénotype concurrent où si c'est un défaut du phénotype émergent (contenant parfois des cellules à l'état s_5)



- Enfin, il que le plus souvent un phénotype prend le pas sur un autre grâce à l'utilisation d'un état, ou d'une combinaison d'états, qui crée un nouvel environnement inhospitalier pour le phénotype concurrent. (Figure 5.15)

5.3 Génération continue de nouveaux phénotypes

5.3.1 Système de mesure

Nous avons ensuite exploré l'évolution dans le temps de différentes mesures prises lors des simulations décrites en début de chapitre (et tout particulièrement les simulations longues ici) dans le but d'en identifier certaines permettant de mesurer l'évolution de la diversité phénotypique. Il est à noter que pour toutes les mesures suivantes les cellules non vivantes seront considérées comme étant au même état qu'elle soit à l'état $s_{dnaturel}$, $s_{dvolontaire}$ ou s_q :

- l'entropie Ent des différents états : $Ent = \frac{\sum_{i \in \Sigma} -\rho_i \cdot \log(\rho_i)}{\log(|\Sigma|)}$ avec $\rho_i = N_i/|I|$, N_i nombre de cellule à l'état i et $|I|$ nombre total de cellules
- leur écart type σ (par rapport à l'état le plus commun et en prenant comme différence le delta de Kronecker Δ) : $\sigma = \sqrt{\frac{\sum_{p \in I} \Delta(f(p), s_{med})}{|I|}} = \sqrt{\frac{|I| - N_{S_{med}}}{|I|}}$ avec S_{med} état le plus fréquent, $f(p)$ l'état de la cellule p et I l'ensemble des cellules de l'automate
- l'écart type des seuls états des cellules vivantes σ_{Viv} (sans prendre en compte donc les cellules à l'état neutre et celle à l'état Decay) $\sigma_{Viv} = \sqrt{\frac{\sum_{p \in I_{Viv}} \Delta(f(p), s_{medViv})}{|I_{Viv}|}} = \sqrt{\frac{|I_{Viv}| - N_{S_{medViv}}}{|I_{Viv}|}}$ avec I_{Viv} ensemble des cellules vivantes S_{medViv} état le plus fréquent hors s_d et s_q et $|I_{Viv}|$ nombre de cellules vivantes

FIGURE 5.13: Forte augmentation du nombre de cellules à l'état $s_{d\text{ volontaire}}$ après une chute de la population que l'on peut imaginer liée à la forte compétition due à l'apparition d'un nouveau génotype à ce moment ($A_{limite} = 4, T_{decay} = 1850$ et $T_{muta} = 1/5000$)

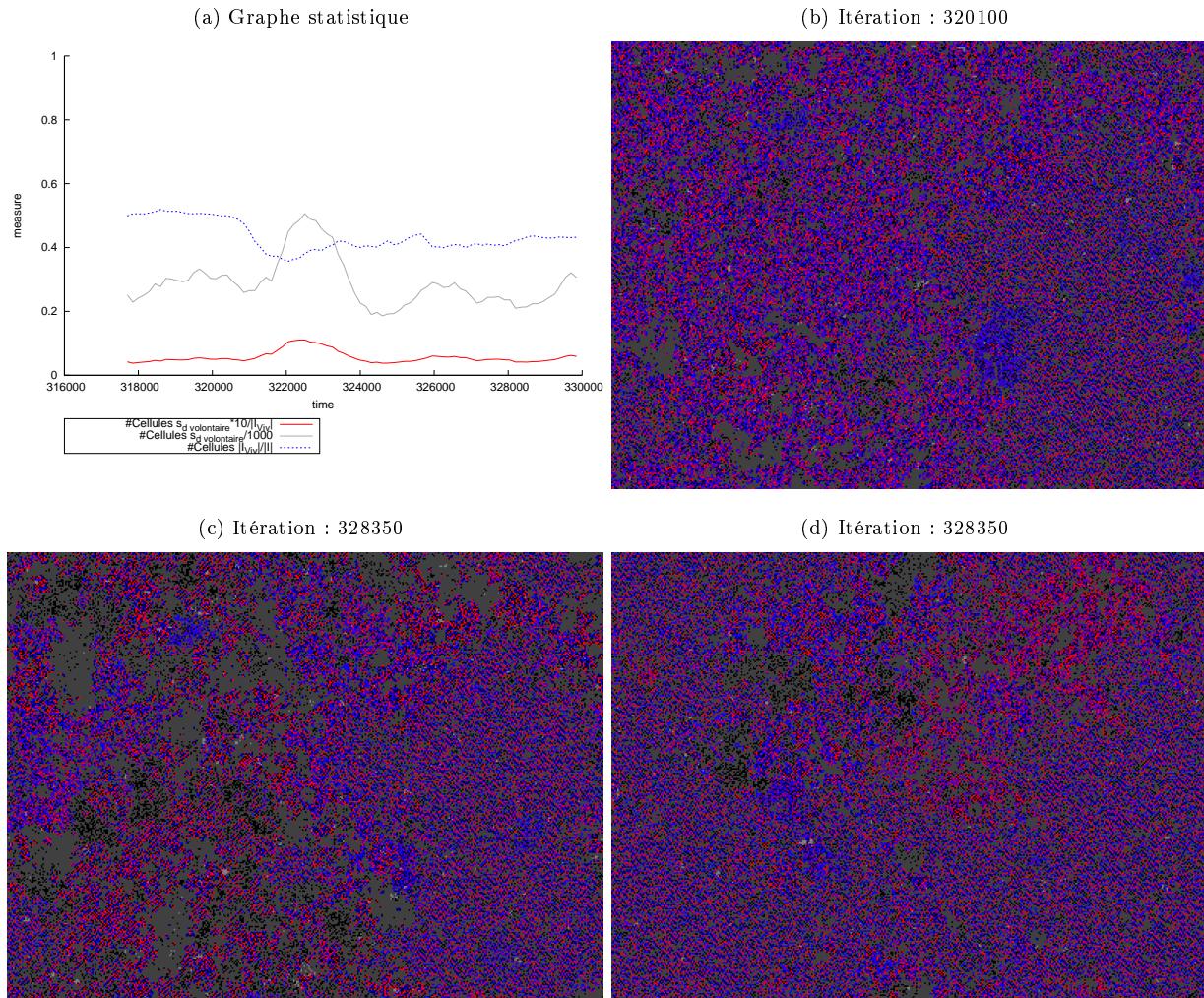


FIGURE 5.14: Le phénotype anciennement dominant peu dense est rapidement remplacé par un phénotype plus dense

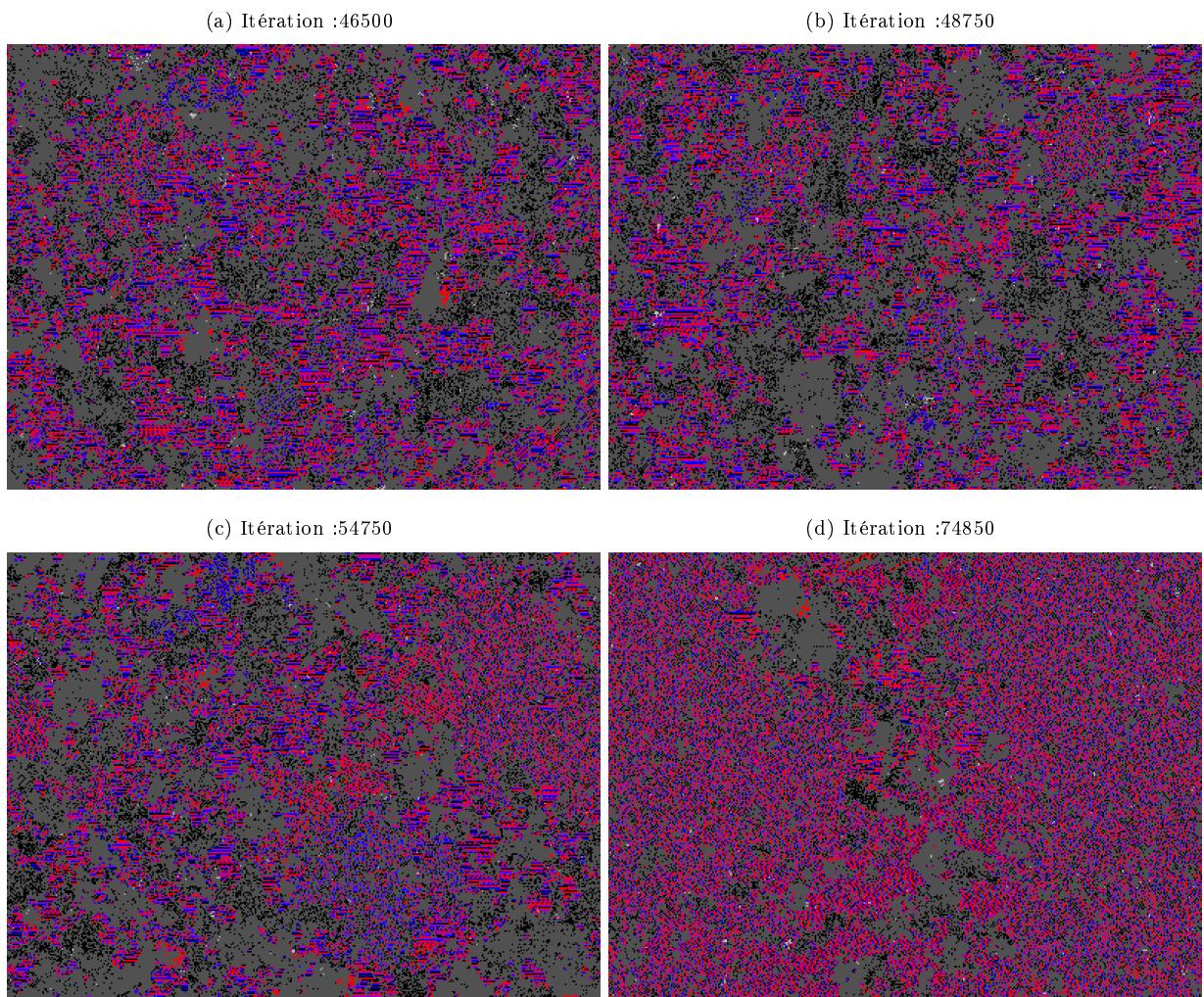
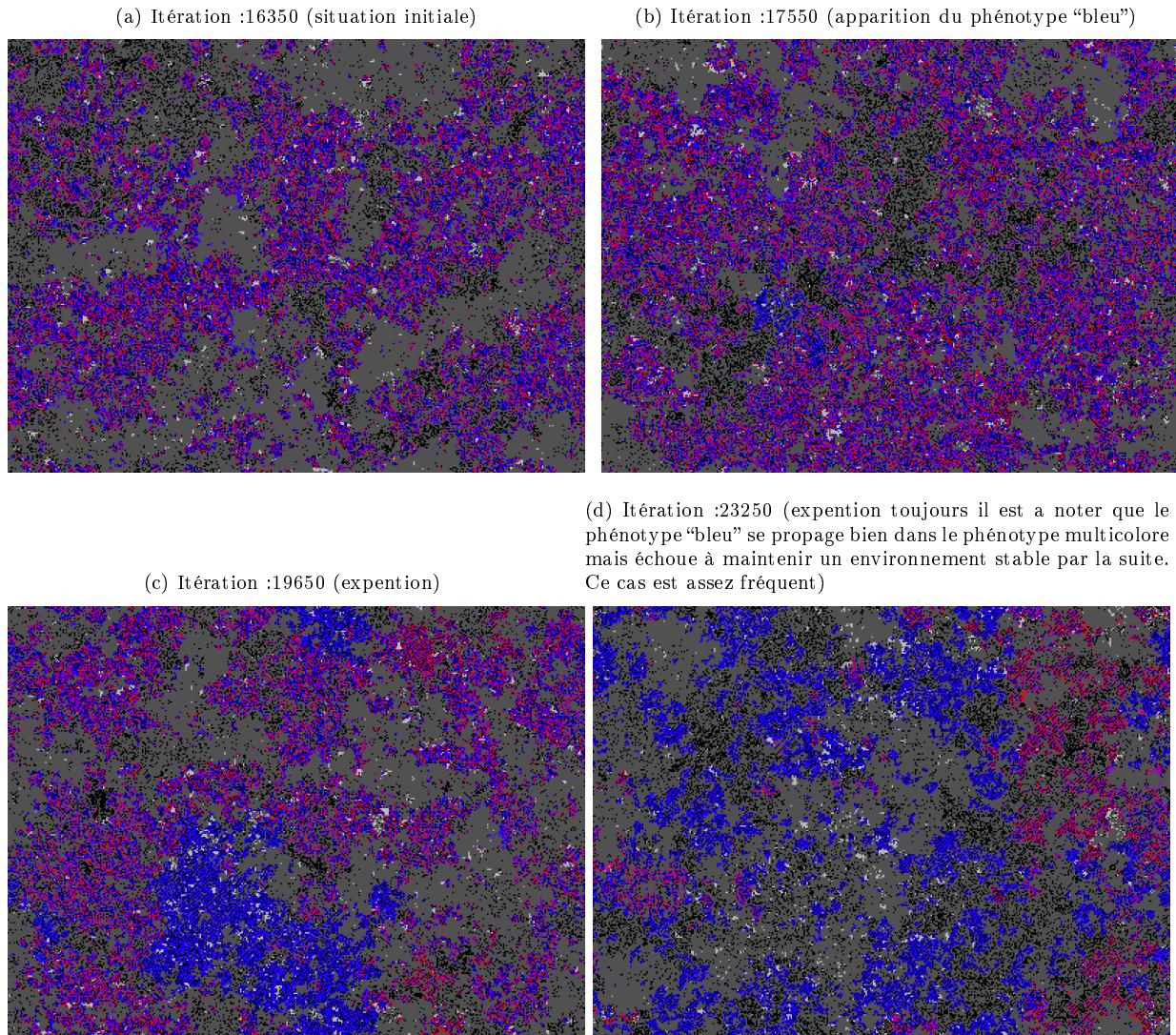


FIGURE 5.15: Le phénotype anciennement dominant “multicolore” semble ne pas pouvoir se propager dans l’environnement créé par le phénotype “bleu”



Nous avons ensuite attribué à chaque état i une valeur $\bar{\rho}_i$ qui soit fonction de la fréquence normalisée à laquelle on retrouve cet état : $\bar{\rho}_i = \frac{N_i - N_{min}}{N_{max} - N_{min}}$ avec N_{min} et N_{max} nombre de cellule à l'état le moins fréquent et nombre de cellule à l'état le plus fréquent. Avec presque la même formule, on peut aussi calculer une fréquence $\overline{\rho viv}_i = \frac{N_i - NViv_{min}}{NViv_{max} - NViv_{min}}$ en excluant les états “quiescent” s_q et “Decay” s_d et avec donc $NViv_{min}$ et $NViv_{max}$ nombre de cellule à l'état le moins fréquent et nombre de cellules à l'état le plus fréquent hors état s_q et s_d . Cette nouvelle valeur nous a permis de calculer :

- Un nouvel écart type : $\sigma_{Freq} = \sqrt{\frac{\sum_{p \in I} (\bar{\rho}_{f(p)} - \bar{\rho}_{s_{med}})^2}{|I|}} = \sqrt{\frac{\sum_{i \in \Sigma} N_i (\bar{\rho}_i - \bar{\rho}_{s_{med}})^2}{|\Sigma|}}$
- Un second écart type en ne prenant en compte que les cellules vivantes (pas s_d et s_q) : $\sigma_{FreqViv} = \sqrt{\frac{\sum_{p \in I_{Viv}} (\overline{\rho viv}_f(p) - \overline{\rho viv}_{s_{medViv}})^2}{|I_{Viv}|}} = \sqrt{\frac{\sum_{i \in \Sigma} N_i (\overline{\rho viv}_i - \overline{\rho viv}_{s_{medViv}})^2}{|I_{Viv}|}}$

Ensuite nous nous sommes intéressés à des mesures localisées (concernant une cellule et ses 8 voisines) nous avons utilisé trois mesures locales :

- L'écart type local $\sigma_{loc}(p)$ pour chaque cellule p : $\sigma_{loc}(p) = \sqrt{\frac{\sum_{p' \in VN(p)} \Delta(f(p'), s_{med})}{9}}$ avec p' cellules appartenant au voisinage de p
- Et la moyenne de la fréquence $\mu_{loc}(p)$ pour chaque cellule p : $\mu_{loc}(p) = \frac{\sum_{p' \in VN(p)} \rho_f(p')}{9}$
- L'écart type local $\sigma_{FreqLoc}(p)$ de la fréquence pour chaque cellule p : $\sigma_{FreqLoc}(p) = \sqrt{\frac{\sum_{p' \in VN(p)} (\rho_{f(p')} - \mu_{loc}(p))^2}{9}}$

Parfois nous multiplions ces mesures par une constante pour les rendre plus visibles sur les graphs.

A partir de ces trois données locales, il a été ensuite possible, à chaque itération, de calculer leur écart type sur l'ensemble des cellules pour avoir une idée de l'homogénéité de la répartition des états :

- $\sigma_{glob} = \sqrt{\frac{\sum_{p \in I} \sigma_{loc}(p) - \sigma_{moyen}}{|I|}}$ avec $\sigma_{moyen} = \frac{\sum_{p \in I} \sigma_{loc}(p)}{|I|}$
- $\sigma_{FreqGlob} = \sqrt{\frac{\sum_{p \in I} \sigma_{FreqLoc}(p) - \sigma_{FreqMoyen}}{|I|}}$ avec $\sigma_{FreqMoyen} = \frac{\sum_{p \in I} \sigma_{FreqLoc}(p)}{|I|}$
- $\mu_{glob} = \sqrt{\frac{\sum_{p \in I} \mu_{local}(p) - \mu_{moyen}}{|I|}}$ avec $\mu_{moyen} = \frac{\sum_{p \in I} \mu_{local}(p)}{|I|}$

5.3.2 Résultats

Ces données nous ont semblé pertinentes et varier brusquement dans des période d'intense compétition entre phénotypes.

En regardant l'évolution de l'ensemble de ces paramètres, il apparaît que leur variations sont corrélées à des modifications des phénotypes comme on peut le constater en comparant leur variation (Figure 5.16 et 5.17) avec les variations du nombre de cellules aux différents états possible (Figure 5.18). qu'après de fortes variations lors des premières itérations ces mesures sont très stables dans l'ensemble des simulations “tests” alors qu'elles varient fortement dans notre simulation. Les mesures locales semblent être un indicateur particulièrement pertinent (Figure 5.17). Il est à noter que σ_{glob} semble être corrélée au nombre de cellules en decay (elle détecte presque pas de variation dans la simulation sans decay) elle semble par conséquent moins

intéressante.

La plus sensible nous a parue être l'écart type de la variance locale de la fréquence $\sigma_{FreqGlob}$. C'est, de même que μ_{glob} , une approximation de la diversité phénotypique à un moment précis (Figure 5.19). A priori une mesure élevée signifie une diversité phénotypique importante. Et ce, même s'il peut exister de quelques biais, comme par exemple deux états présents à la même fréquence (qui ne seront pas distingués avec cette mesure) ou des configuration régulières organisées sur un nombre important de cellules (qui pourrons être interprété avec cette mesure par une diversité phénotypique plus élevée).

L'écart type des variations dans le temps de $\sigma_{FreqGlob}$ sur des cycles de 35 000, 75 000 où 200 000 itérations semble lui être une mesure intéressante des variation de la diversité phénotypique et donc nombre de bouleversements ayant eu lieu au cours de la simulation. Une valeur élevée après un nombre important d'itération signifie alors que l'automate ne se dirige pas vers un attracteur.

L'écart type de l'évolution dans le temps de la mesure de $\sigma_{FreqGlob}$ prise tous les 150 itérations est en moyenne de 1,12 après 75 000 itérations (Figure 5.1). Alors que la moyenne de cette même valeur prise sur un jeu de la vie est de 0.11. Elle est encore plus basse avec des automates classiques à règle unique générée aléatoirement . Dans ces deux cas et qu'on soit à l'itération 35000 ou 75000 on trouve que la différence est très significative ($p - value \ll 0.01$ avec le test de Welch).

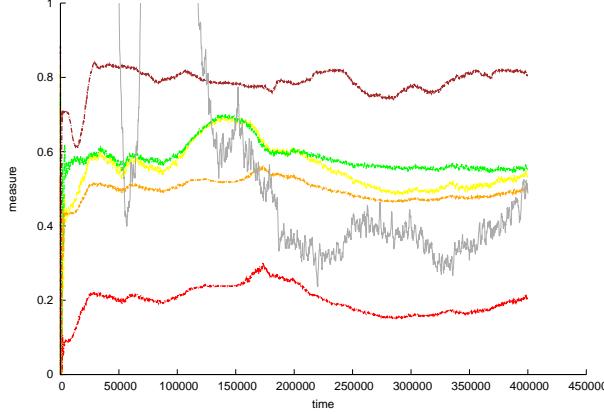
La différence de moyenne entre automate hétérogène avec ou sans mutation à l'itération 35 000 est un peu moins significative ($p - value = 0.011$ avec le test de Welch). Elle l'est beaucoup plus significative à l'itération 75 000 ($p - value \ll 0.01$ avec le test de Welch). Il semble que $\sigma_{FreqGlob}$ devienne rapidement stable après de fortes variation initiales ce qui semble cohérent avec les conséquences de l'absence de mutations. Et en effet la comparaisons des données aux itérations 75 000 et 35 000 ne montre qu'une faible présomption contre l'hypothèse nulle ($p - value = 0.053$ avec le test de Welch) pour la simulation avec mutations alors qu'elle montre une forte présomption contre l'hypothèse nulle ($p - value = 0.017$ avec le test de Welch) pour la simulation sans mutations.

Enfin pour ce qui est de la comparaison entre la simulation habituelle et une simulation sans decay la différence est très significative $p - value \ll 0.01$ dès la 35 000ième itération ($p - value \ll 0.01$ avec le test de Welch). Ce qui suggère bien que le decay joue un rôle dans l'évolutivité de l'automate. Au vu de ces résultats il semble donc que c'est bien le cumul du decay et des mutations qui permettent d'avoir une importante variation phénotypique.

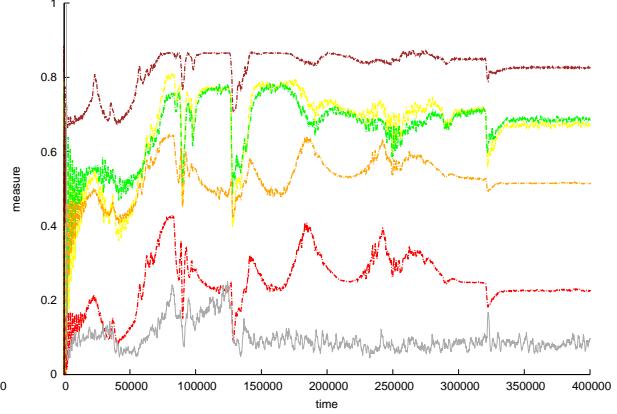
FIGURE 5.16: Mesures générales pour les 5 simulations sur au moins 200 000 itérations

Val. entropy : Ent — Val. EC des cel. vivantes : σ_{Viv} — Val. EC de la freq. des cel. vivantes : $\sigma_{FreqViv}$ —
 Val. ecart type : σ — Val. EC de la fréquence : σ_{Freq} — #Cellules s_d volontaire / 3000 —

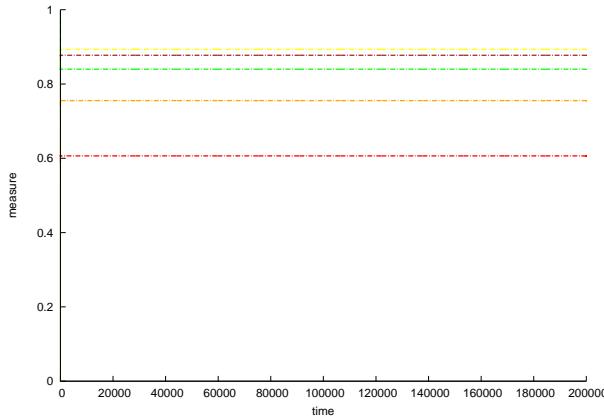
(a) Automate hétérogène ($A_{limite} = 7, T_{decay} = 1850$ et $T_{muta} = 1/5000$)



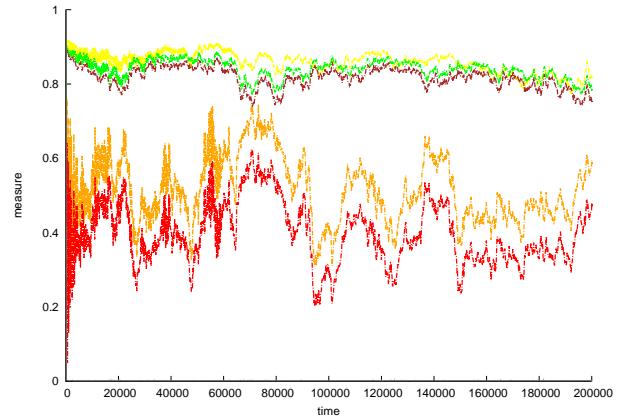
(b) Automate hétérogène ($A_{limite} = 4, T_{decay} = 1850$ et $T_{muta} = 1/5000$)



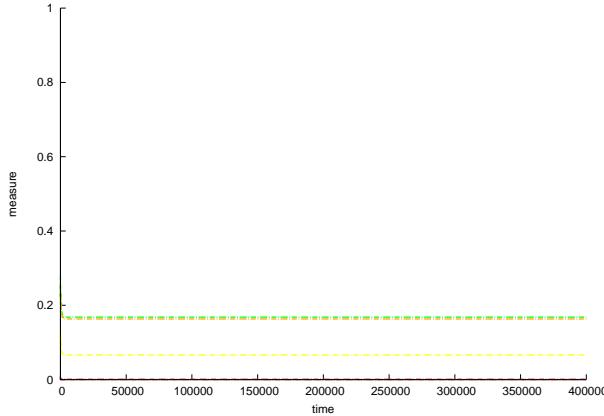
(c) Automate homogène aléatoire



(d) Automate hétérogène, sans “Decay” ($A_{limite} = 4, T_{decay} = 0$ et $T_{muta} = 1/5000$)



(e) Jeu de la vie



(f) Sans mutation ($A_{limite} = 4, T_{decay} = 1850$ et $T_{muta} = 0$)

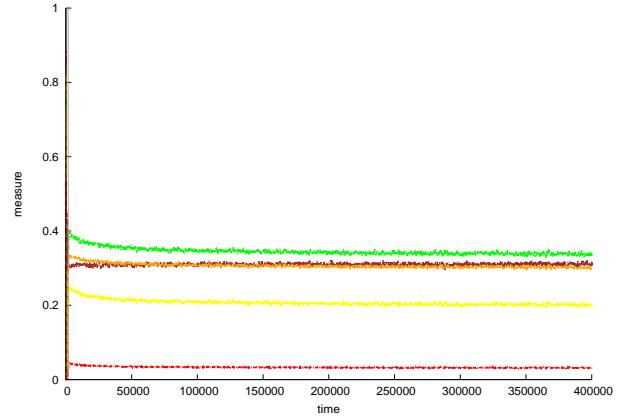
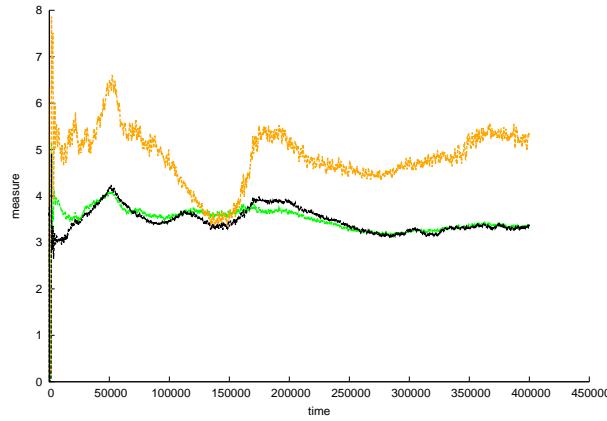


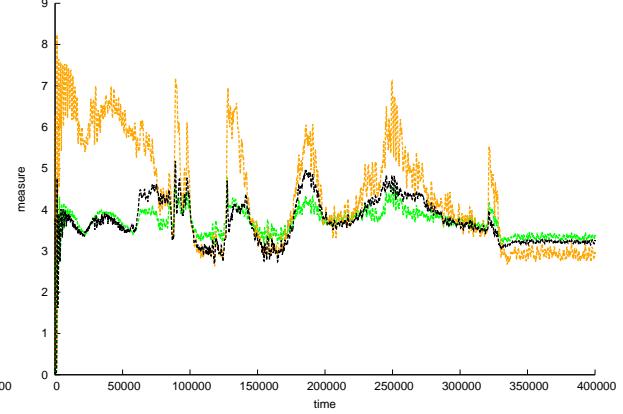
FIGURE 5.17: Variance des mesures locales pour les 5 simulations au moins 200 000 itérations

σ_{glob} --- σ_{FreqGlob} --- μ_{glob} ---

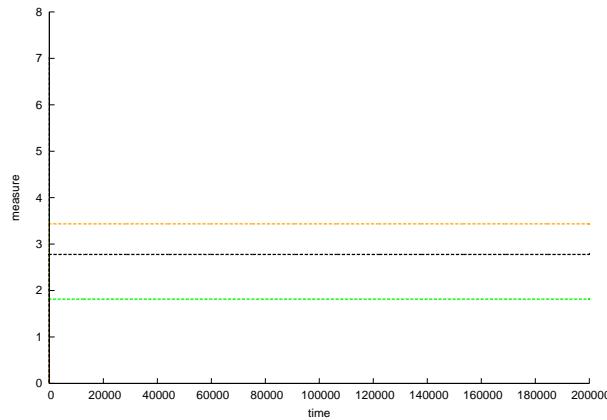
(a) Automate hétérogène ($A_{\text{limite}} = 7, T_{\text{decay}} = 1850$ et $T_{\text{muta}} = 1/5000$)



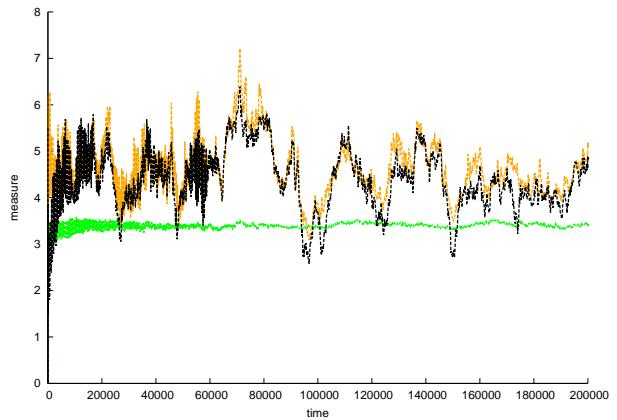
(b) Automate hétérogène ($A_{\text{limite}} = 4, T_{\text{decay}} = 1850$ et $T_{\text{muta}} = 1/5000$)



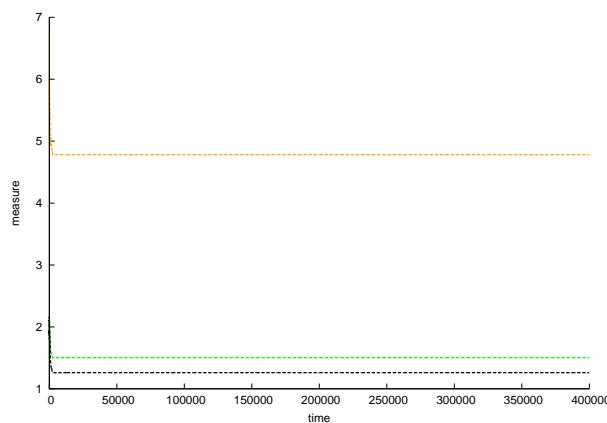
(c) Automate homogène aléatoire



(d) Automate hétérogène, sans "Decay" ($A_{\text{limite}} = 4, T_{\text{decay}} = 0$ et $T_{\text{muta}} = 1/5000$)



(e) Jeu de la vie



(f) Sans mutation ($A_{\text{limite}} = 4, T_{\text{decay}} = 1850$ et $T_{\text{muta}} = 0$)

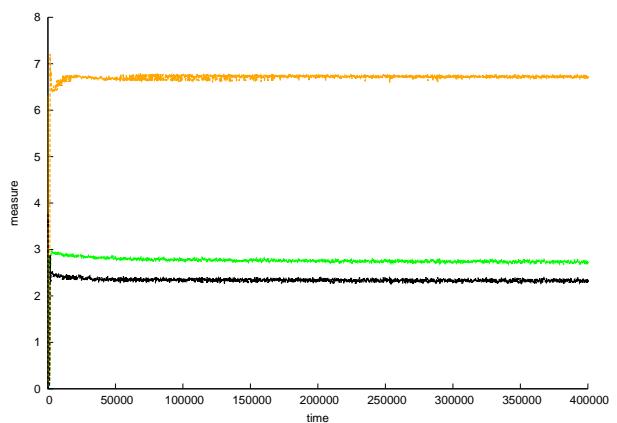
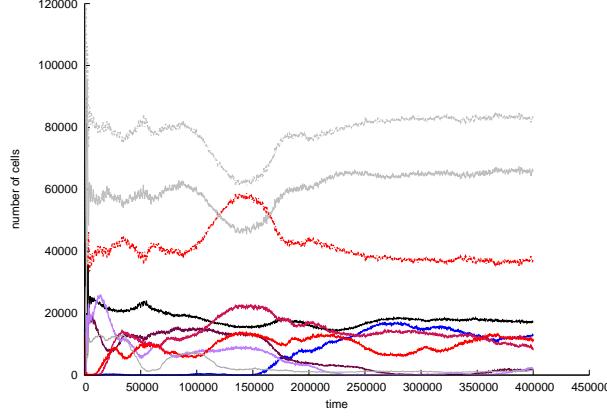


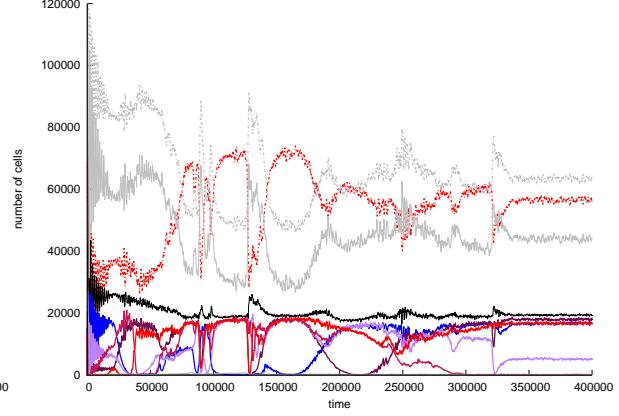
FIGURE 5.18: Variance du nombre de cellule à chaque état pour les 5 simulations au moins 200 000 itérations

#Cellules vivantes $\|_{Viv} (N_{total}(s_d \text{ et } s_q))$ dotted #Cellules s_1 solid #Cellules s_3 solid #Cellules s_5 solid #Cellules s_d volontaire solid #Cellules non vivantes (s_d et s_q) solid

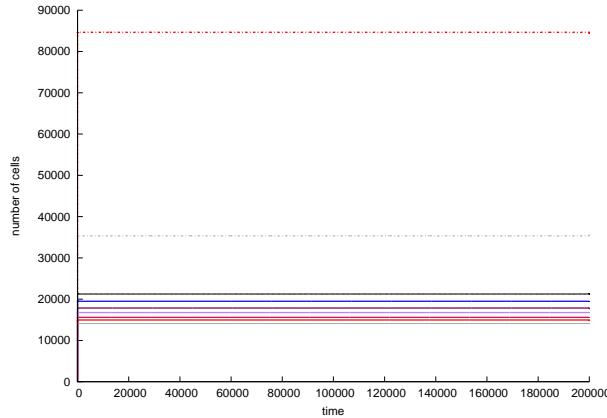
(a) Automate hétérogène, ($A_{limite} = 7, T_{decay} = 1850$ et $T_{muta} = 1/5000$)



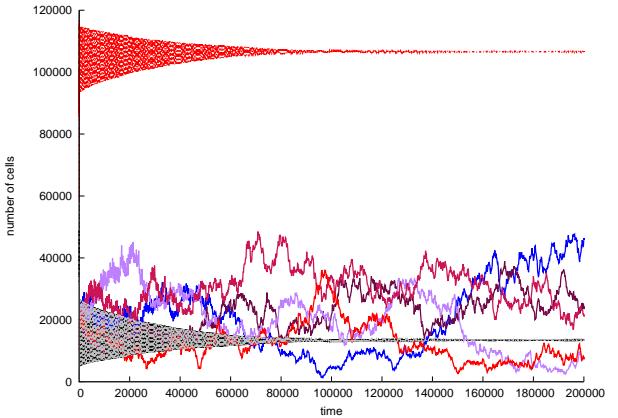
(b) Automate hétérogène ($A_{limite} = 4, T_{decay} = 1850$ et $T_{muta} = 1/5000$)



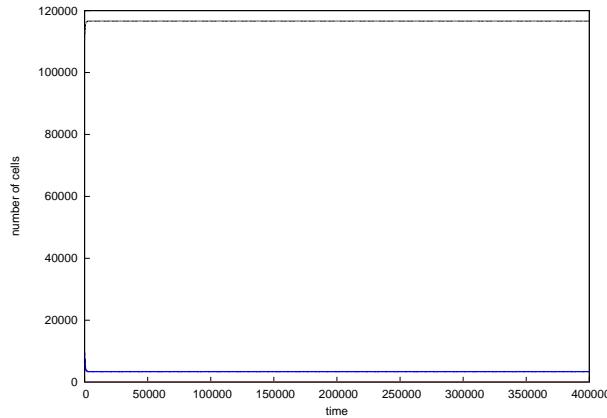
(c) Automate homogène aléatoire



(d) Automate hétérogène, sans “Decay” ($A_{limite} = 4, T_{decay} = 0$ et $T_{muta} = 1/5000$)



(e) Jeu de la vie



(f) Sans mutation ($A_{limite} = 4, T_{decay} = 1850$ et $T_{muta} = 0$)

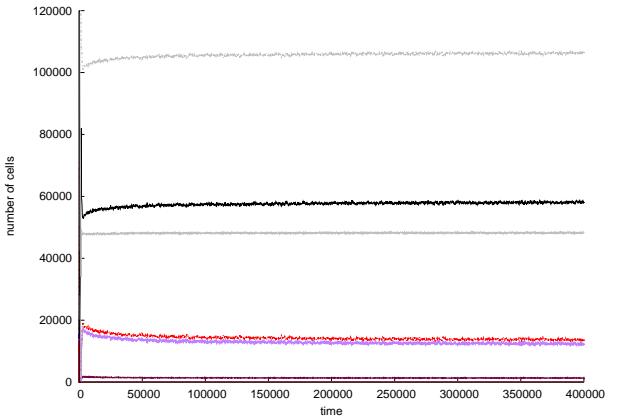


FIGURE 5.19: Evolution de l'écart types des mesures locales avec images ($A_{limite} = 7, T_{decay} = 1850$ et $T_{muta} = 1/5000$)

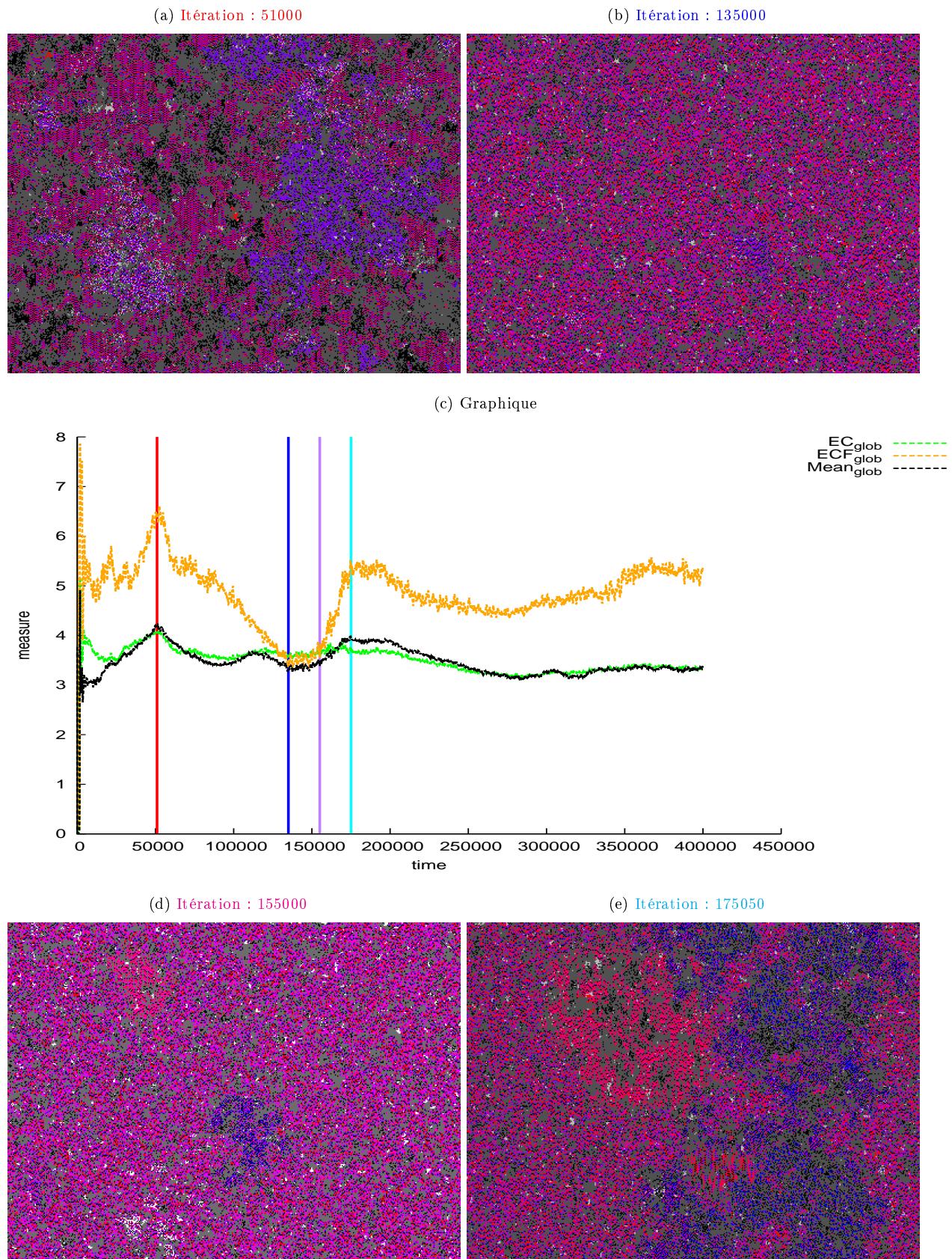


TABLE 5.1: Écart type moyen de la variation dans le temps de ECF_{glob}

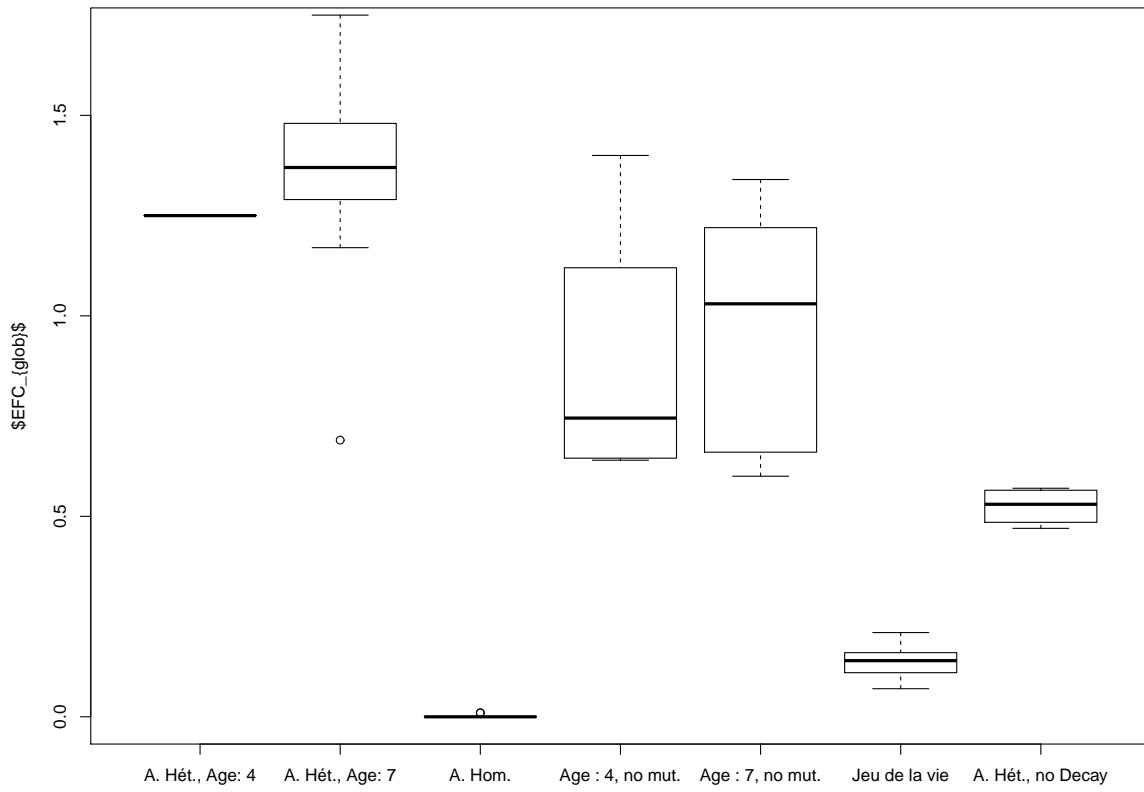
	Itération 35~000	Itération 75~000	Itération 200~000
Automate Hétérogènes 7 $A_{limite} = 7$, $T_{decay} = 1850$, $T_{muta} = 1/5000$	1.33	1.12	0.89
Automate Hétérogènes 4 $A_{limite} = 4$, $T_{decay} = 1850$, $T_{muta} = 1/5000$	1.25	0.95	1.4
Automate hétérogène sans Mutations 7 $A_{limite} = 7$, $T_{decay} = 1850$, $T_{muta} = 0$	0.96	0.63	
Automate hétérogène sans Mutations 3 $A_{limite} = 3$, $T_{decay} = 1850$, $T_{muta} = 0$	0.88		
Automate hétérogène sans Decay $A_{limite} = 7$, $T_{decay} = 0$, $T_{muta} = 1/5000$	0.53	0.75	0.62
Automate homogène aléatoire	0.0029	0.0029	0.0006
Jeu de la vie	0.13	0.10	0.067

 TABLE 5.2: Écart type de la variation dans le temps de ECF_{glob} pour les simulation longues

	Itération 200 000	Itération 300 000	Itération 400 000
Automate Hétérogènes 7 $A_{limite} = 7$, $T_{decay} = 1850$, $T_{muta} = 1/5000$	0.89	0.74	0.67
Automate Hétérogènes 4 $A_{limite} = 4$, $T_{decay} = 1850$, $T_{muta} = 1/5000$	1.4	1.24	1.29
Automate hétérogène sans Mutations 7 $A_{limite} = 7$, $T_{decay} = 1850$, $T_{muta} = 0$	0.57	0.44	0.53
Automate hétérogène sans Decay $A_{limite} = 7$, $T_{decay} = 0$, $T_{muta} = 1/5000$	0.62		
Jeu de la vie	0.067	0.054	0.047

FIGURE 5.20: Boîte à moustaches de l'écart type des variations dans le temps de $ECF_{glob}(1/2)$

(a) Itération 35 000



(b) Itération 75 000

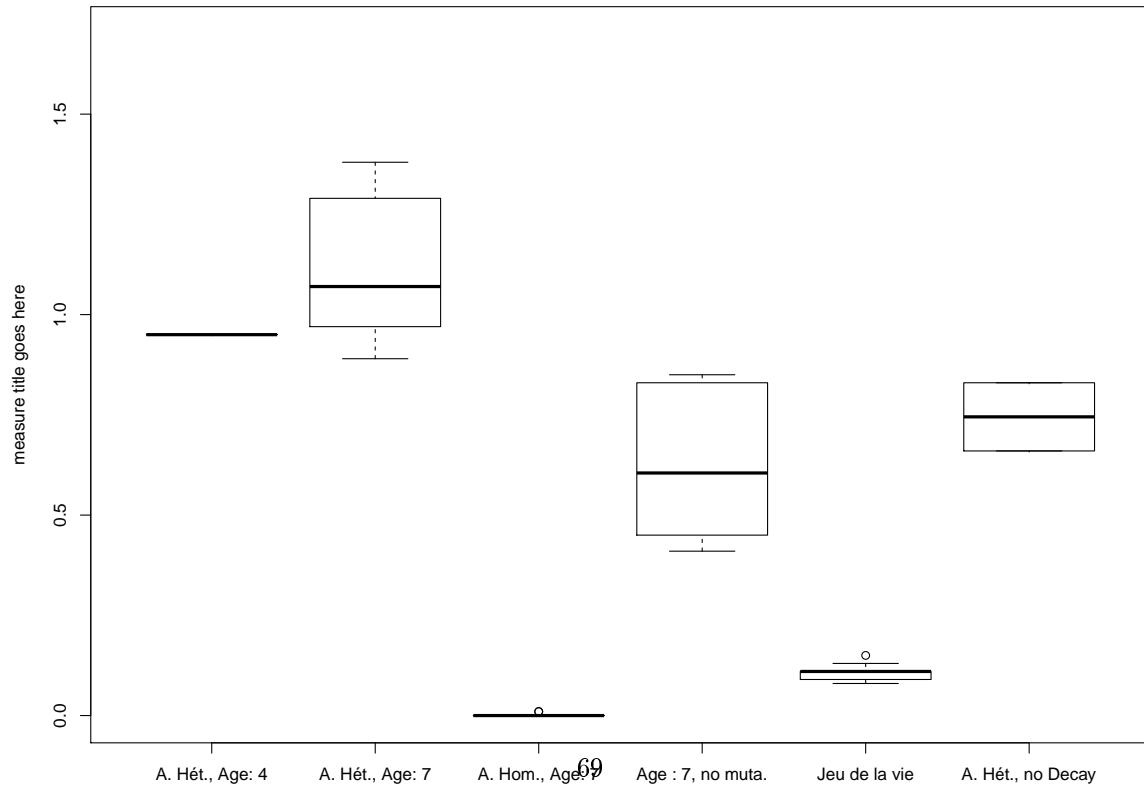


FIGURE 5.21: Boîte à moustaches de l'écart type des variations dans le temps de ECF_{glob} (2/2) Le manque de données limite fortement l'intérêt du graphique pour l'itération 400 000

(a) Itération 200 000

Model	Approximate Median (y)	Approximate Range (y)
A. Hét., Age: 4	~1.35	~0.95 - 1.40
A. Hét., Age: 7	~0.85	~0.80 - 0.90
A. Hom., Age: 7	~0.02	~0.00 - 0.05
Age : 7, no muta.	~0.50	~0.40 - 0.70
Jeu de la vie	~0.05	~0.02 - 0.08
A. Hét., no Decay	~0.60	~0.55 - 0.65

(b) Itération 400 000

Model	Approximate Median (y)	Approximate Range (y)
A. Hét., Age: 4	~1.25	~1.15 - 1.30
A. Hét., Age: 7	~0.65	~0.60 - 0.70
Age : 7, no muta.	~0.35	~0.25 - 0.40
Jeu de la vie	~0.05	~0.02 - 0.08

Chapitre 6

Potentiels développements futurs

6.1 Analyse des génotypes

Pour mesurer la diversité il est possible de chercher à mesurer la diversité phénotypique comme nous l'avons fait ici. Mais il serait intéressant d'opter pour une mesure plus fine (ici on ne fait pas la distinction entre deux phénotypes différents, si ils sont caractérisés par les même proportions de différents états). Enfin il serait très intéressant d'effectuer des mesures centrées sur les génotypes : par exemple, l'analyse des génomes en s'inspirant des travaux de Bedau[3]. Et plus simplement dresser la carte de la répartition des génotypes. Une modification de la simulation (stocker à part la liste des génomes, ainsi que la parenté des génomes survivant sous forme d'arbre) pourrait facilement permettre d'obtenir ce genre de données.

6.2 Rajouter de l'échange génétique

Pour obtenir une meilleures analogie biologique, il serait tout à fait possible de rajouter des mécanismes d'échange de matériel génétique (analogie des échange de gènes entre bactéries et de la reproduction sexuée chez les végétaux). Le système de CA-LGP permet d'ores et déjà de combiner deux génotype parents pour obtenir un génotype fils intermédiaire.

6.3 Comparer l'évolutivité des GP-CA aux autres méthodes pour coder le génotype des automates

Coder le génotype de nos automates hétérogènes semble, naïvement, plus efficace en terme d'évolutivité des automates (cela leur permet d'utiliser la puissance des fonctions mathématiques pour générer des symétries

par exemple contrairement aux systèmes par arbre ou tableau). Néanmoins, il serait intéressant d'obtenir une mesure plus précise de ces capacités évolutives, par exemple en utilisant un système d'évolution dirigée tel que celui décrit par Moshe Sipper [30] pour des automates à une dimension et en comparant les résultats obtenus en utilisant les différentes représentations de génotypes.

6.4 Modifier le voisinage

Il est à noter que pour le moment une cellule ne peut détecter un phénotype hostile en approche que sur ses diagonales (forcément pas encore au contact), alors que ce n'est pas un trajet obligatoire pour, justement, entrer au contact. Il serait intéressant de tester les effets d'un élargissement du voisinage par exemple passer à un voisinage de 12 cellules tel que celui-ci présenté en Figure 6.1. Les modifications en termes d'usage de $S_{d\text{volontaire}}$ seraient à observer particulièrement, un tel système en renforçant potentiellement l'intérêt.

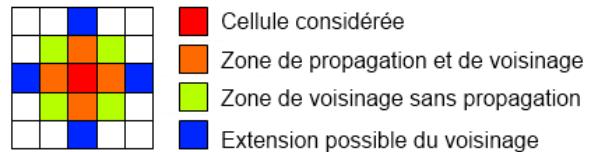
6.5 Des types de cellules spécifiques

Il est possible que le monde dans lequel évoluent nos réplicateurs ne soit pas suffisamment riche (qu'il faille enrichir l'équivalent des règles physiques du monde, à savoir le principe de fonctionnement de l'automate cellulaire). On peut par exemple imaginer des états associés à des caractéristiques particulières :

- un état cellulaire bloquant la réception de nouveaux génotypes (comme l'état Decay), mais possédant toujours un génome gouvernant ses changements d'état.
- Des murs permanents (des cellules à un état spécial et ne pouvant en aucun cas acquérir un génotype).
- Un état spécial réduisant l'espérance de vie des cellules aux alentours.
- Faire du Decay volontaire un état différent du Decay naturel (perceptible par les cellules).
- Des variations dans le voisinage utilisé en fonction de l'état de la cellule.

On pourrait par ailleurs associer ces nouveaux types cellulaires à des mesures de la complexité locale inspirée des travaux de Daniel W. McShea¹.

FIGURE 6.1: Extention potentielle du voisinage



1. Dans l'esprit des travaux de McShea il serait aussi intéressant de pouvoir comparer l'évolution de la complexité (si possible en comparant les résultat avec et sans pression de sélection naturelle). Une méthode pour réduire (supprimer ?) la pression de sélection serait d'utiliser un espace infini pour la simulation. Malheureusement c'est impossible pour des contraintes techniques évidente. Mais il serait possible de régulièrement sélectionner un petit groupe de cellule (au hasard) et réinitialiser le reste de la grille à s_q .

6.6 Explorer le développement

Il semble qu'un même génotype puisse laisser s'exprimer plusieurs phénotypes. Cet élément serait intéressant à explorer plus avant, car il est possible qu'utilisant ce principe une forme de développement apparaisse dans la simulation. De même que pour l'évolutivité, comme [30] il semble envisageable d'utiliser, pour mesurer l'aptitude au développement de notre système, des système d'évolution dirigée (algorithme génétique).

On pourrait aussi par exemple, dans une optique assez classique en développement artificiel, définir une structure à développer. Et en regroupant les génomes de même type (comme évoqué dans la partie 6.1) récompenser par un bonus au niveau de leur propagation (en cas de conflit) les génomes générant des cellules au bon état et au bon endroit. Les mécanismes d'age limite et de Decay nous permettront de nous assurer que la structure sera régulièrement altérée et nécessitera des réparations, par ailleurs si la structure est suffisamment complexe une telle simulation est suceptible de favoriser la coopération entre génomes[30, 29].

6.7 Des changements environnementaux

Il est aussi envisageable de générer régulièrement des variations environnementales (de manière localisée ou globale) par exemple en affectant un bonus à A_{max} , ou à la propagation (en cas de conflit) quand les cellules sont à un certain état. Il serait alors intéressant d'analyser les effets sur les génotypes des variations de la fréquences de telles modifications.

6.8 Questionner la nature de l'évolution

Il semble possible d'utiliser cette simulation pour étudier plus précisément les différentes visions de l'évolution. Il est à noter que certaines modifications seraient sans doute nécessaires. Par exemple pour Dawkins le réplicateur de base est le gène. Mais il différencie les gènes (unité de matériel génétique suffisamment petite pour être stable dans le temps et suffisamment grande pour avoir des effets phénotypiques) des cistrons (matériel génétique codant pour une protéine donnée). En reprenant cette idée, nous nous trouvons ici dans une situation où un organisme (véhicule) ne possède qu'un seul gène de manière évidente. Il serait néanmoins possible de modifier la simulation de façon à ce que chaque cellule soit porteuse de plusieurs gènes ce qui permettrait d'étudier des éventuels phénomènes de coopération entre gènes (le Green-beard effect [6])

De même il serait intéressant d'observer de manière plus précise l'apparition de phénomènes tels que les équilibres ponctués (voire essayer d'observer des phénomènes d'exaptation) décrit par Stephen Jay Gould.

On peut aussi imaginer réduire la pression de sélection (par exemple en travaillant sans Decay sur une grille extensible) et s'attacher à mesurer l'évolution de la complexité, telle que décrite par Daniel W. McShea

[19].

Cela demanderait une modification importante de la simulation, mais il semble aussi envisageable de chercher à étudier l'effet des interactions locales et pour cela effectuer une simulation en les désactivant (et en déterminant un voisinage aléatoire pour chaque cellule, à chaque itération).

6.9 Structures autoréPLICATRICES

Un des objectifs, qui semblerait intéressant avec cette simulation, serait d'arriver à obtenir des structures multicellulaires et autoréPLICATRICES faciles à distinguer de leur environnement. Pour cela trois voies semblent intéressantes à développer :

- Des expériences de longue durée avec des bonus (entre autres avec l'objectif d'obtenir des structures multicellulaires autoréPLICATRICES).
- Un système de connexion (Une façon d'avoir des outils pour récompenser des comportements que l'on cherche à favoriser tels que, par exemple, la réPLICATION. Pour récompenser les comportements utiles, on peut jouer sur l'espérance de vie, mais aussi les récompenser en leur donnant davantage de chance de transmettre leur génome, en cas de présence de compétition entre génomes).
- Tester des réPLICATEURS potentiels en désactivant les mutations et en les faisant évoluer dans un environnement de cellules "quiescent".

Chapitre 7

Conclusion

Au cours de ces travaux nous avons dressé un panorama des approches de l'évolution ouverte en vie artificielle et identifié certains mécanismes qui semblent nécessaires à la simulation d'une telle évolution. Par la suite nous avons proposé un modèle basé sur des automates cellulaires hétérogènes dont nous avons décrit le fonctionnement et analysé qualitativement les différentes caractéristiques. Au vu des mesures de la variabilité de la diversité phénotypique utilisées et après comparaison avec des contrôles (jeux de la vie, automates cellulaires homogènes) il semble que les automates cellulaires hétérogènes soient un outil adéquat dans la quête d'une évolution ouverte simulée.

Il serait néanmoins intéressant de poursuivre ces simulations sur des laps de temps plus importants pour voir si les problèmes classiques (stagnation, répétition¹) des simulations, cherchant à simuler une évolution ouverte, n'apparaissent pas par la suite. Il serait aussi utile de coupler ces simulations de très long terme à des mesures de la diversité génotypique. Néanmoins, on peut déjà remarquer avec les résultats actuels que l'évolution que l'on observe semble non linéaire, assez proche de ce que décrit la théorie des équilibres ponctués avec des périodes de relative stagnation entrecoupées de changements assez rapides. Nous obtenons des résultats très significatifs comparativement à des automates cellulaires homogène ou au jeu de la vie et nos résultats montrent aussi que le Decay et les mutations jouent un rôle dans cette évolution. Les deux axes de développement du système qui semblent les plus prometteurs sont sans doute de tester son évolutivité par de l'évolution dirigée (que ce soit via le développement artificiel ou les algorithmes génétiques) et la mise en place d'outils d'analyse des génomes des cellules.

1. On s'appuie principalement sur les graphiques pour affirmer qu'il n'y a pas d'oscillation en fin de simulation alors qu'on en observe souvent en début de simulation

Bibliographie

- [1] Chris Adami, C. Titus Brown, and W.K. Kellogg. Evolutionary learning in the 2d artificial life system "avida". In *Artificial Life IV*, pages 377–381. MIT Press, 1994.
- [2] Mark A. Bedau, Emile Snyder, C. Titus Brown, and Norman H. Packard. A comparison of evolutionary activity in artificial evolving systems and in the biosphere. In *Proceedings of the fourth European Conference on Artificial Life*, pages 125–134. MIT Press, 1997.
- [3] Mark A. Bedau, Emile Snyder, and Norman H. Packard. A Classification of Long-Term Evolutionary Dynamics. In Christoph Adami, Richard K. Belew, Hiroaki Kitano, and Charles Taylor, editors, *Artificial Life VI : Proc. of the Sixth Int. Conf. on Artificial Life*, pages 228–237, Cambridge, MA, 1998. The MIT Press.
- [4] M.F. Brameier and W. Banzhaf. *Linear Genetic Programming*. Genetic and Evolutionary Computation Series. Springer, 2007.
- [5] Matthew Cook. Universality in Elementary Cellular Automata. *Complex Systems*, 15(1) :1–40, 2004.
- [6] Richard Dawkins. *The Selfish Gene*. Oxford University Press, September 1990.
- [7] B. Durand and Zsuzsanna Roka. The game of life : universality revisited, 1998.
- [8] Jay Gould S. Eldredge N. Punctuated equilibria : an alternative to phyletic gradualism. *Models in Paleobiology*, pages 82–115, 1972.
- [9] Matthew Elton. Susan blackmore, the meme machine. 11(3), August 2001.
- [10] J. D. Farmer and Alletta Belin. Artificial Life : The Coming Evolution. volume 10, Redwood City, CA, 1992. Santa Fe Institute Studies in the Sciences of Complexity Proc., Addison-Wesley.
- [11] D B Fogel. Nils barricelli - artificial life, coevolution, self-adaptation. *IEEE Computational Intelligence Magazine*, 1(1) :41–45, 2006.
- [12] Stephen J. Gould and Elisabeth S. Vrba. Exaptation-A Missing Term in the Science of Form. *Paleobiology*, 8(1) :4–15, 1982.

- [13] Dirk Helbing, Wenjian Yu, and Heiko Rauhut. Self-organization and emergence in social systems : Modeling the coevolution of social environments and cooperative behavior. *The Journal of Mathematical Sociology*, 35(1-3) :177–208, 2011.
- [14] J.C. Heudin. *Les créatures artificielles : Des automates aux mondes virtuels*. Sciences (Ed. O. Jacob). O. Jacob, 2007.
- [15] (in submission) T. Kowaliw and R. Doursat. Bias-variance decomposition for genetic programming, in genetic programming and evolvable machines. 2012.
- [16] (Forthcoming) T. Kowaliw and W. Banzhaf. Mechanisms for complex systems engineering through artificial development. In R. Doursat, H. Sayama, and O. Michel, editors, *Morphogenetic Engineering : Toward Programmable Complex Systems*. Springer-Verlag, 2012.
- [17] Christopher Langton, Interdisciplinary Workshop on the Synthesis, and Simulation of Living Systems. *Artificial life : the proceedings of an Interdisciplinary Workshop on the Synthesis and Simulation of Living Systems, held September, 1987, in Los Alamos, New Mexico*. Addison-Wesley Pub. Co. Advanced Book Program, Redwood City Calif., 1989.
- [18] Christopher G Langton. Studying artificial life with cellular automata. *Physica D : Nonlinear Phenomena*, 22(1-3) :120–149, 1986.
- [19] D.W. McShea and R.N. Brandon. *Biology’s First Law : The Tendency for Diversity and Complexity to Increase in Evolutionary Systems*. University of Chicago Press, 2010.
- [20] N Oros and C L Nehaniv. Sexyloop : Self-reproduction, evolution and sex in cellular automata. *Artificial Life 2007 ALIFE 07 IEEE Symposium on*, pages 130–138, 2007.
- [21] Z. Pan and J.A. Reggia. Computational discovery of instructionless self-replicating structures in cellular automata. *Artificial Life*, 16 :1064–5462, 2010.
- [22] Umberto Pesavento. An implementation of von neumann’s self-reproducing machine. *Artificial Life*, 2 :337–354, 1996.
- [23] G. R. Price. Extension of covariance selection mathematics. *Annals of Human Genetics*, 35(4) :485–490, 1972.
- [24] Thomas S. Ray. Evolution, ecology and optimization of digital organisms. Technical report, Report 92-08-942 of the Santa Fe Institute, 1992.
- [25] M. Ronkko. An artificial ecosystem : emergent dynamics and lifelike properties. *Artificial Life*, 13(2) :159–187, 2007.
- [26] Hiroki Sayama. Spontaneous evolution of self-reproducing loops implemented on cellular automata : A preliminary report. *Proceedings of the Second International Conference on Complex System*, 1998.

- [27] Hiroki Sayama. A new structurally dissolvable self-reproducing loop evolving in a simple cellular automata space. *Artificial Life*, 5 :343–365, 1999.
- [28] M. Sipper. Non-uniform cellular automata : Evolution in rule space and formation of complex structures. In R.A. Brooks and P. Maes, editors, *Proceedings of the 4th International Workshop on the Synthesis and Simulation of Living Systems (Artificial Life IV)*, pages 394–399, 1994.
- [29] M. Sipper. Computing with cellular automata : Three cases for nonuniformity. *Phys. Rev. E*, 57 :3589–3592, Mar 1998.
- [30] M. Sipper and M. Tomassini. Computation in artificially evolved, non-uniform cellular automata. *Theoretical Computer Science*, 217(1) :81–98, 1999.
- [31] Timothy Taylor. Redrawing the boundary between organism and environment. In *Ninth International Conference on the Simulation and Synthesis of Living Systems (ALIFE 9)*, Feb 2004.
- [32] F G Varela, H R Maturana, and R Uribe. Autopoiesis : the organization of living systems, its characterization and a model. *Currents in Modern Biology*, 5(4) :187–196, 1974.
- [33] David S. Wilson and Elliot Sober. Reintroducing group selection to the human behavioral sciences. *The Behavioral and Brain Sciences*, 17 :585–654, 1994.
- [34] Larry Yaeger. Computational genetics, physiology, metabolism, neural systems, learning, vision, and behavior or polyworld : Life in a new context. In *Artificial Life III, Vol. XVII of SFI Studies in the Sciences of Complexity, Santa Fe Institute*, pages 263–298. Addison-Wesley, 1993.
- [35] A.R. Yinusa and C.L Nehaniv. Study of inheritable mutations in von neumann self-reproducing automata using the golly simulator. In *Artificial Life (ALIFE), 2011 IEEE Symposium on*, pages 211–217, 2011.
- [36] Tina Yu and Julian Miller. Neutrality and the evolvability of boolean function landscape. In *Genetic Programming, Proceedings of EuroGP 2001, volume 2038 of LNCS*, pages 204–217. Springer-Verlag, 2001.

Chapitre 8

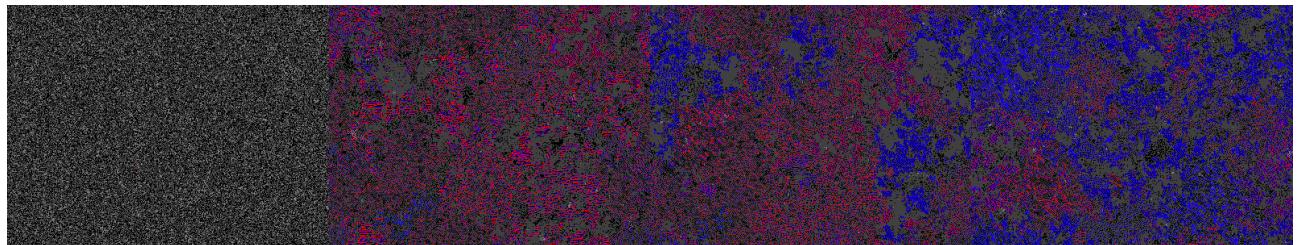
Annexes

FIGURE 8.1: Légende

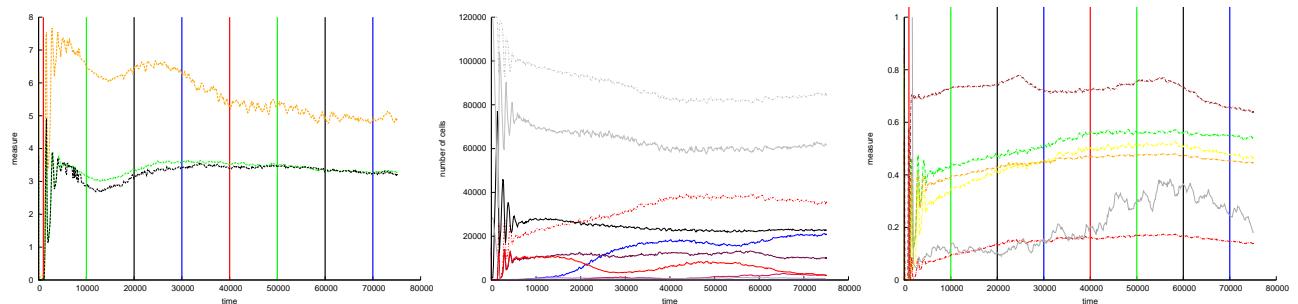
Val. entropy : Ent Val. EC des cel. vivantes : σ_{Viv} Val. EC de la freq. des cel. vivantes : $\sigma_{FreqViv}$
Val. ecart type : σ Val. EC de la fréquence : σ_{Freq} #Cellules s_d volontaire /3000
#Cellules vivantes $|Viv| (N_{total}(s_d \text{ et } s_q))$ #Cellules s_1 #Cellules s_3 #Cellules s_5 #Cellules s_d #Cellules s_d volontaire
#Cellules S_q #Cellules s_2 #Cellules s_4 #Cellules s_d #Cellules non vivantes (s_d et s_q)
 σ_{glob} $\sigma_{FreqGlob}$ μ_{glob}

FIGURE 8.2: Automate hétérogène sur 75 000 itérations ($A_{limite} = 7, T_{decay} = 1850$ et $T_{muta} = 1/5000$)

(a) itérations : **1000**, **10000**, **20000** et **30000**



(b) Graphiques



(c) itérations : **40000**, **50000**, **60000** et **70000**

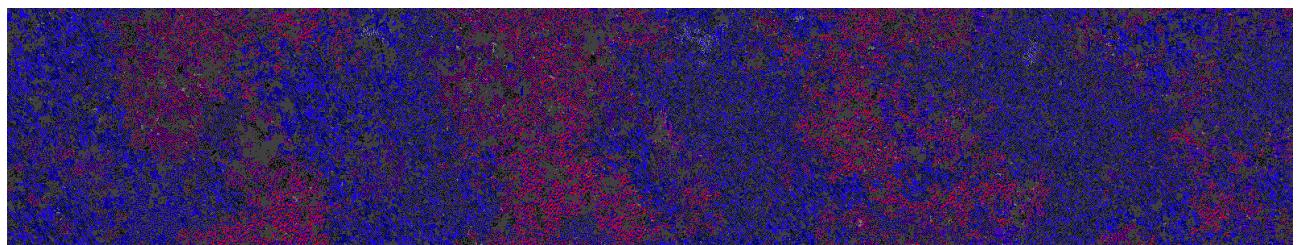
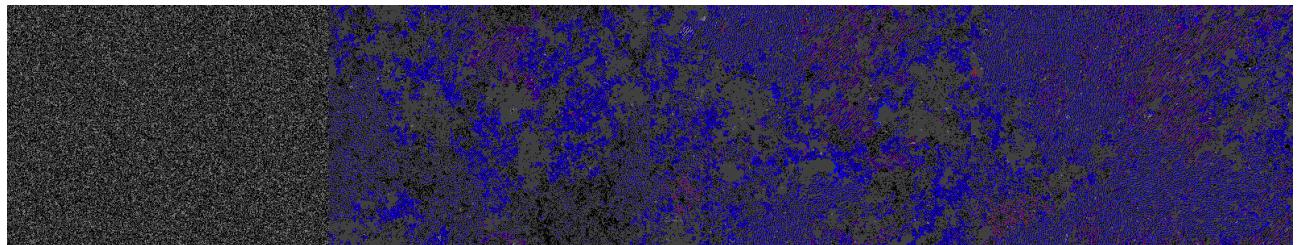
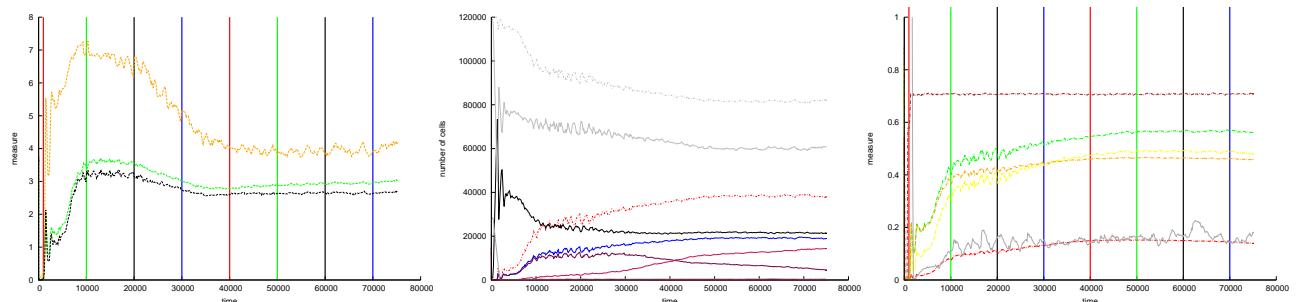


FIGURE 8.3: Automate hétérogène sur 75 000 itérations ($A_{limite} = 7, T_{decay} = 1850$ et $T_{muta} = 1/5000$)

(a) itérations : **1000**, **10000**, **20000** et **30000**



(b) Graphiques

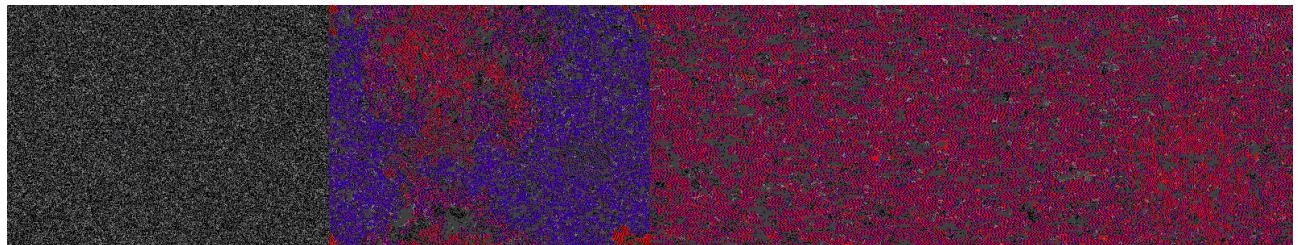


(c) itérations : **40000**, **50000**, **60000** et **70000**

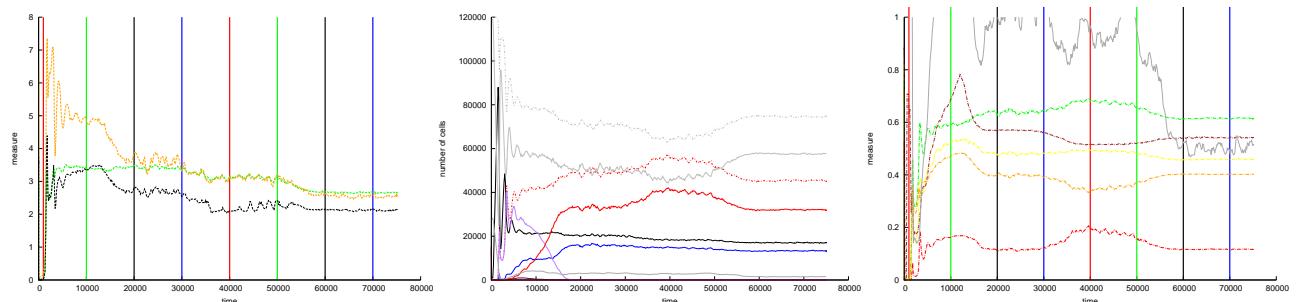


FIGURE 8.4: Automate hétérogène sur 75 000 itérations ($A_{limite} = 7, T_{decay} = 1850$ et $T_{muta} = 1/5000$)

(a) itérations : **1000**, **10000**, **20000** et **30000**



(b) Graphiques

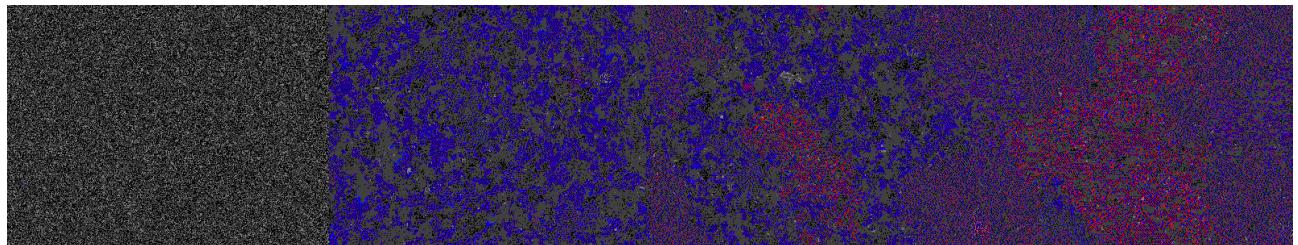


(c) itérations : **40000**, **50000**, **60000** et **70000**

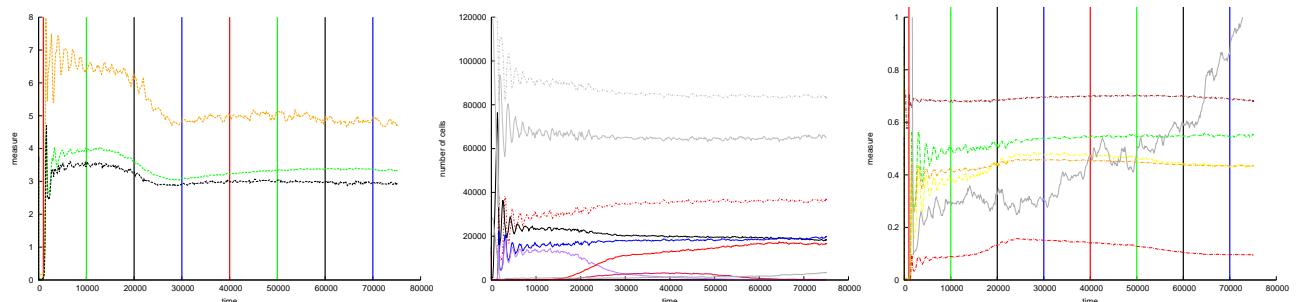


FIGURE 8.5: Automate hétérogène sur 75 000 itérations ($A_{limite} = 7, T_{decay} = 1850$ et $T_{muta} = 1/5000$)

(a) itérations : **1000**, **10000**, **20000** et **30000**



(b) Graphiques



(c) itérations : **40000**, **50000**, **60000** et **70000**

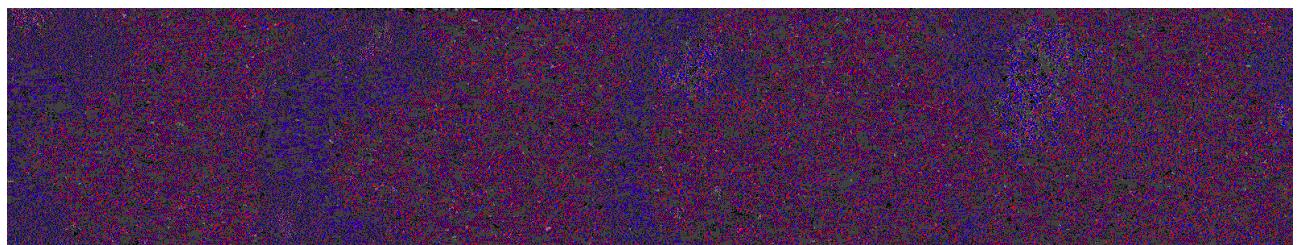
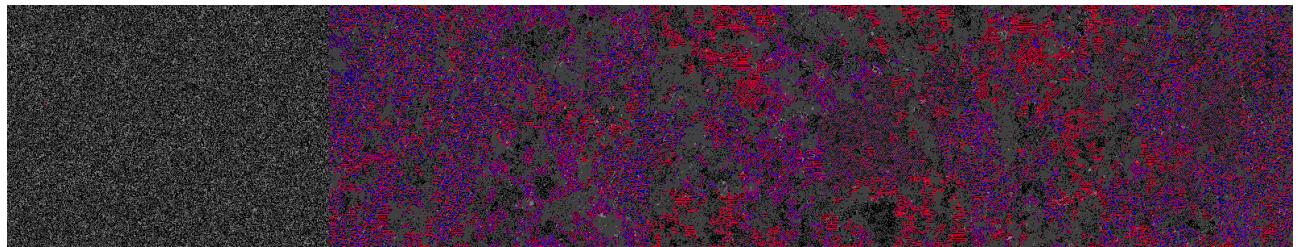
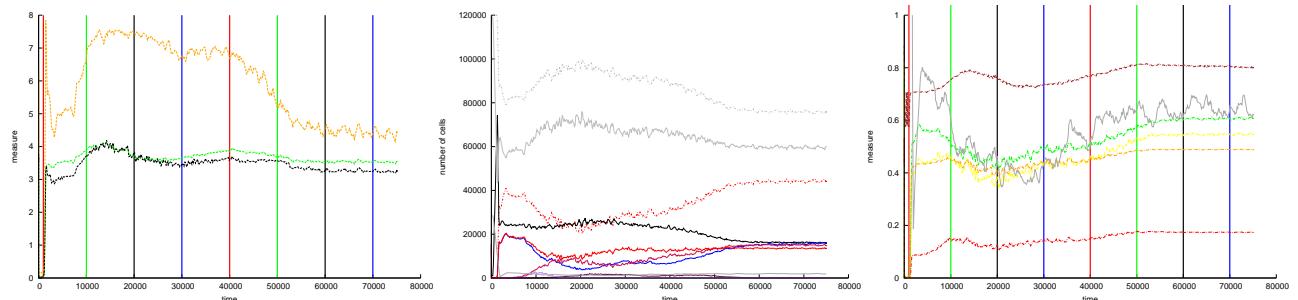


FIGURE 8.6: Automate hétérogène sur 75 000 itérations ($A_{limite} = 7, T_{decay} = 1850$ et $T_{muta} = 1/5000$)

(a) itérations : **1000**, **10000**, **20000** et **30000**



(b) Graphiques



(c) itérations : **40000**, **50000**, **60000** et **70000**

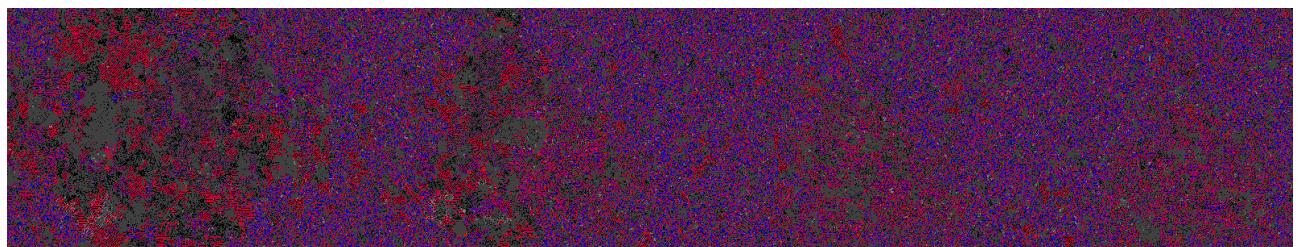
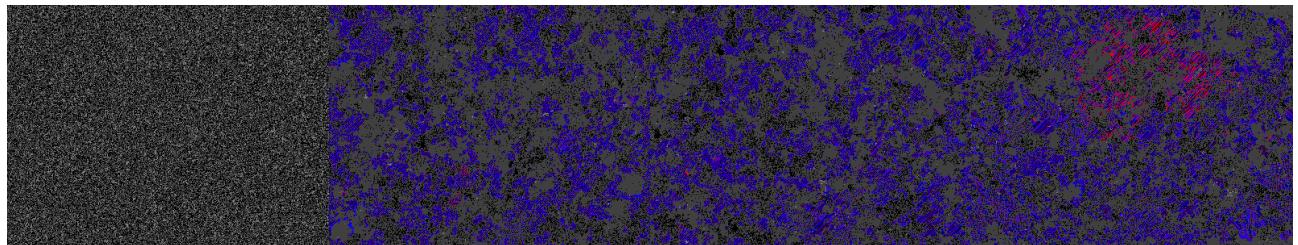
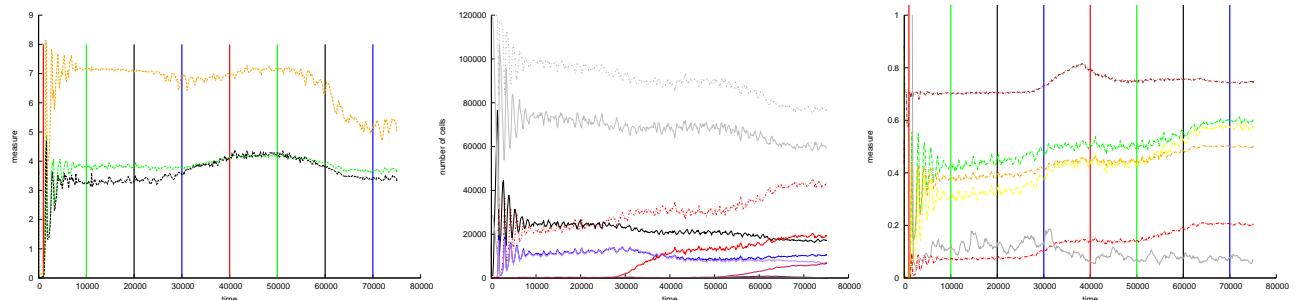


FIGURE 8.7: Automate hétérogène sur 75 000 itérations ($A_{limite} = 7, T_{decay} = 1850$ et $T_{muta} = 1/5000$)

(a) itérations : **1000**, **10000**, **20000** et **30000**



(b) Graphiques



(c) itérations : **40000**, **50000**, **60000** et **70000**

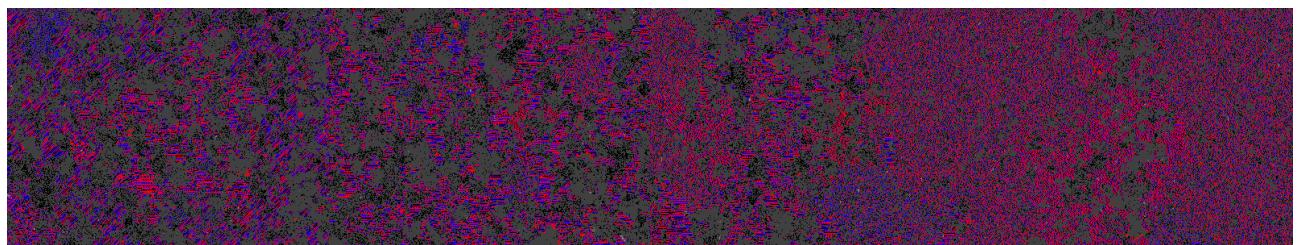
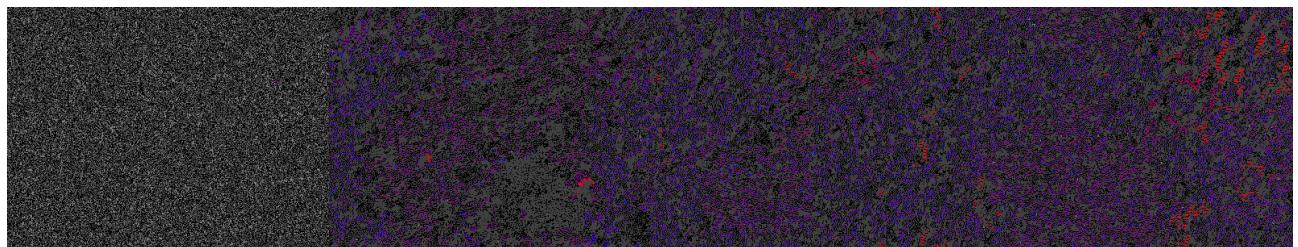
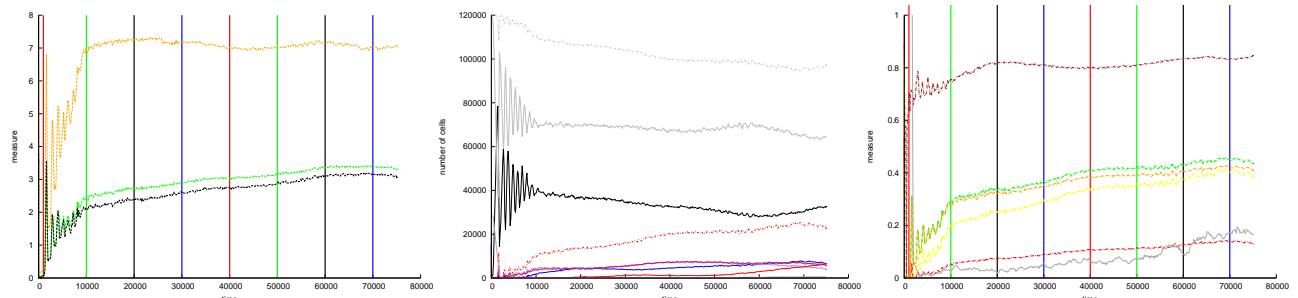


FIGURE 8.8: Automate hétérogène sur 75 000 itérations ($A_{limite} = 7, T_{decay} = 1850$ et $T_{muta} = 1/5000$)

(a) itérations : **1000**, **10000**, **20000** et **30000**



(b) Graphiques



(c) itérations : **40000**, **50000**, **60000** et **70000**

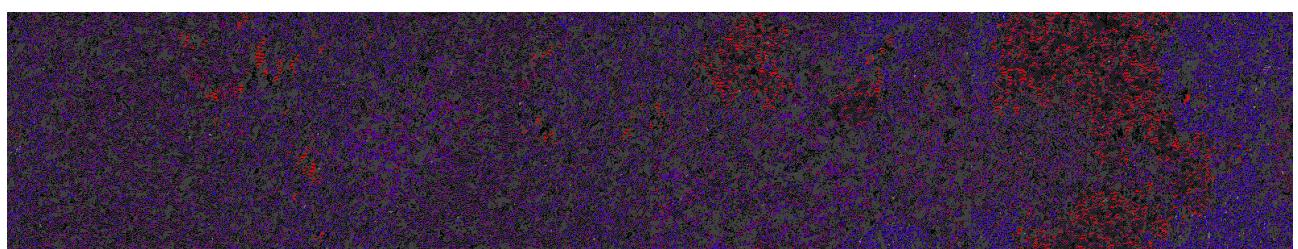
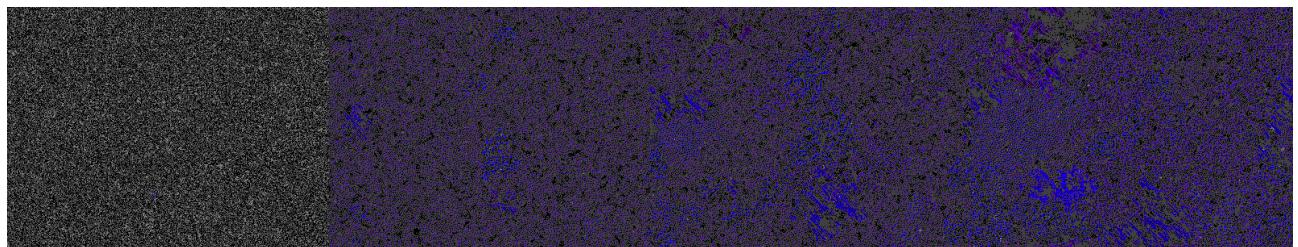
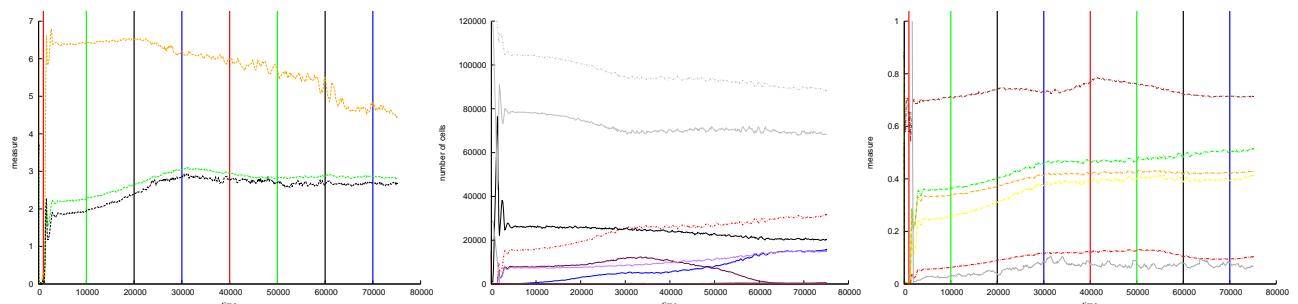


FIGURE 8.9: Automate hétérogène sur 75 000 itérations ($A_{limite} = 7, T_{decay} = 1850$ et $T_{muta} = 1/5000$)

(a) itérations : **1000**, **10000**, **20000** et **30000**



(b) Graphiques



(c) itérations : **40000**, **50000**, **60000** et **70000**

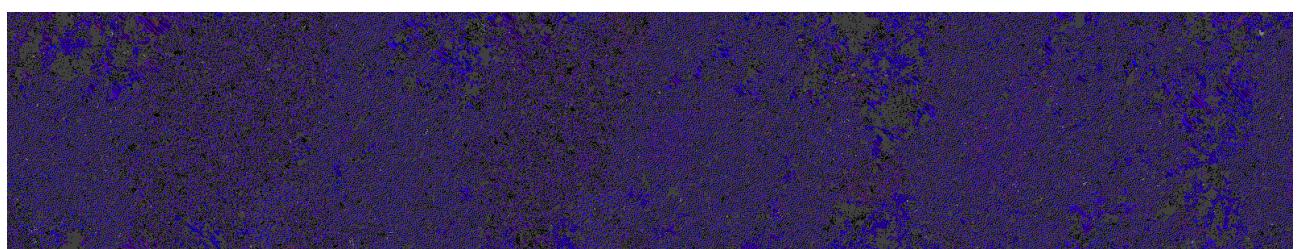
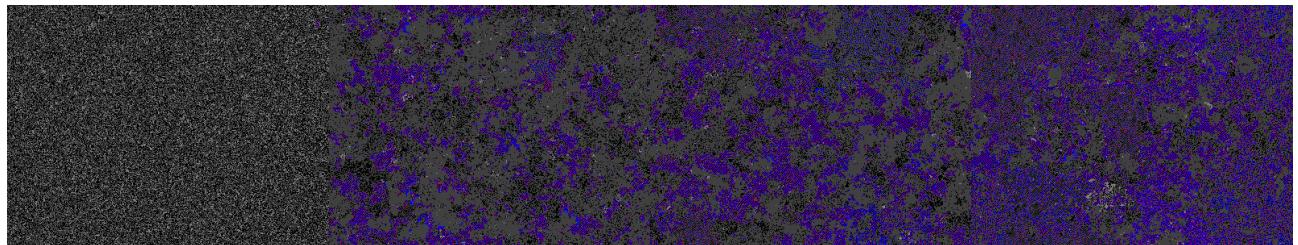
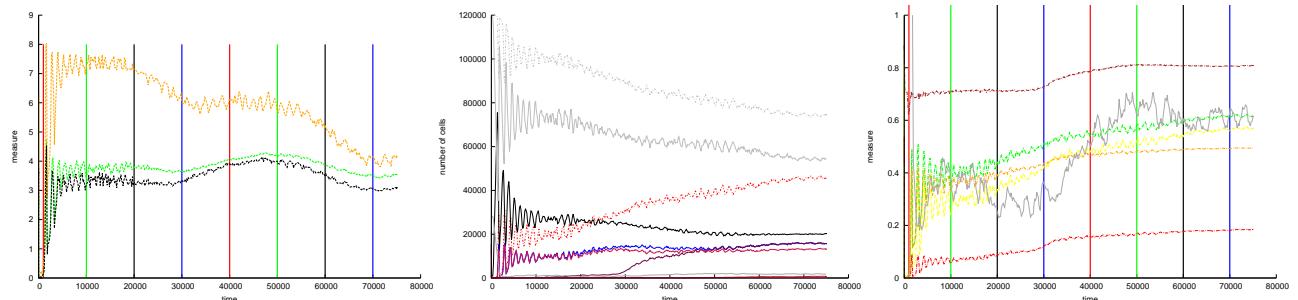


FIGURE 8.10: Automate hétérogène sur 75 000 itérations ($A_{limite} = 7$, $T_{decay} = 1850$ et $T_{muta} = 1/5000$)

(a) itérations : **1000**, **10000**, **20000** et **30000**



(b) Graphiques



(c) itérations : **40000**, **50000**, **60000** et **70000**

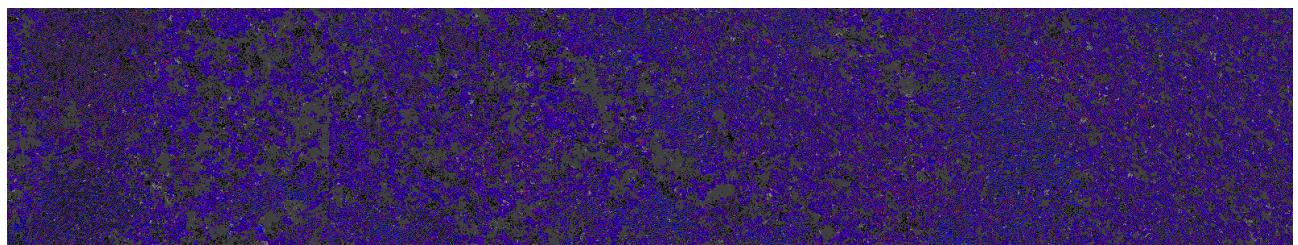
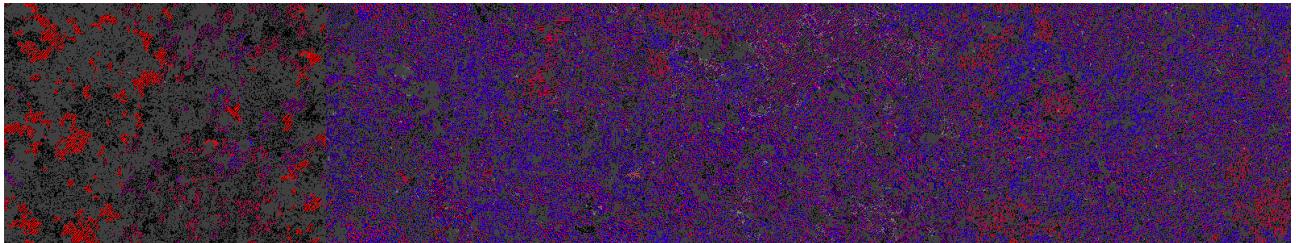
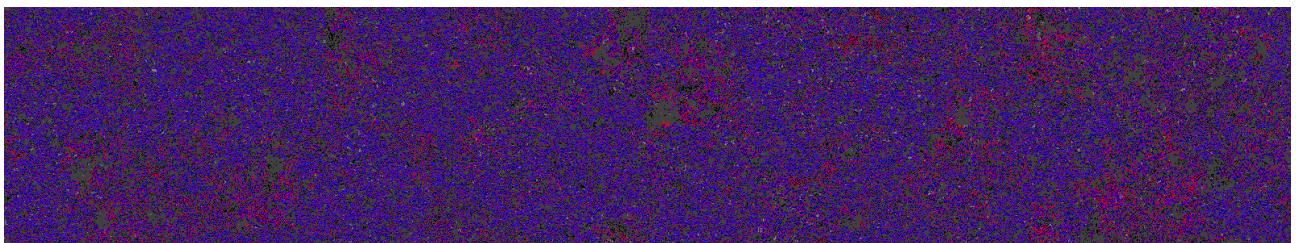


FIGURE 8.11: Automate hétérogène sur 1 000 000 itérations ($A_{limite} = 7$, $T_{decay} = 1850$ et $T_{muta} = 1/5000$)

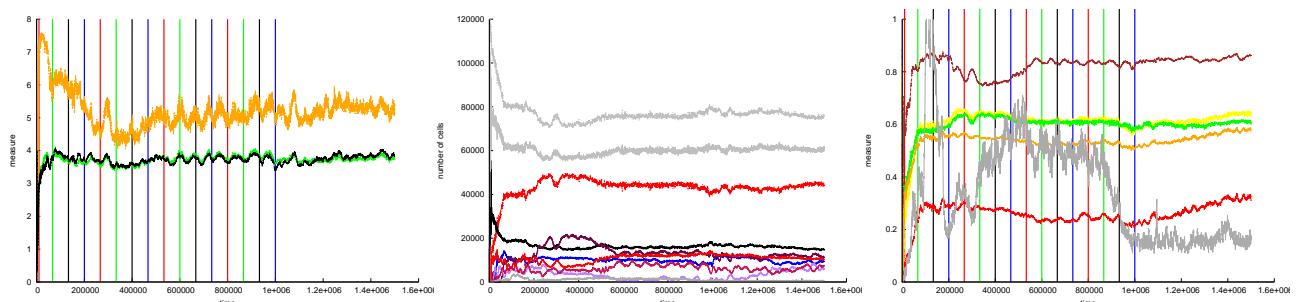
(a) itérations : **10000**, **66666**, **133333** et **200000**



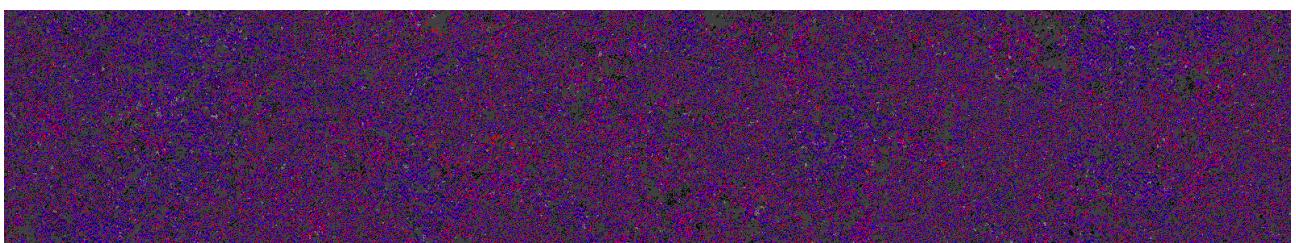
(b) itérations : **266666**, **333333**, **400000** et **466666**



(c) Graphiques



(d) itérations : **533333**, **600000**, **666666** et **733333**



(e) itérations : **800000**, **866666**, **933333** et **1000000**

