Programmation d'algorithmes mutationnels pour la morphogenèse

Abdoulaye Sarr

TELECOM Bretagne

TELECOM
BRETAGNE

CENTRE EUROPÉEN
DE RÉALITÉ
VIRTUELLE

RAPPORT DE STAGE DE RECHERCHE

Programmation d'algorithmes mutationnels pour la morphogenèse

Auteur :
Abdoulaye SARR

Encadrant :
Alexandra FRONVILLE

1 juin 2012

Table des matières

Remerciements

Je tiens à remercier mon encadrant, Alexandra FRONVILLE pour m'avoir suffisamment consacré de son temps en soutien et conseils pendant toute la durée de ce stage. Je n'oublie pas sa générosité intellectuelle et son aménité.

Introduction

> *« La recherche n'est pas qu'une entreprise froide de déchiffrage et de manipulation de la réalité. Elle peut aussi nous révéler la richesse toujours nouvelle de nos mondes intérieurs. Elle peut être aussi source de surprises et d'émerveillements quand ce qu'elle cherche c'est elle-même, nous-mêmes. Quand comme le dit Borges, les chercheurs sont sans le savoir ce qu'ils cherchent. »*
>
> Jean-Claude AMEISEN

L'ensemble du processus d'apparition de forme, faisant intervenir des mécanismes et des lois, désigne la morphogénèse. Ce processus émergent, composé pour la plupart d'entités autonomes peut aussi bien être l'œuvre de la nature que celle de l'homme. Dans le premier cas, il se manifeste à travers des systèmes complexes qui émergent perpétuellement autour de nous : dunes de sables, géométrie des plantes, organismes vivants . . . Dans le second cas, ce processus peut sous entendre des structures physiques telles que l'organisation des villes mais aussi des architectures logicielles telles que les réseaux.

Que ce soit des structures inanimées ou des organismes vivants, tous ces processus sont des manifestations d'une auto-organisation collective visant à atteindre une forme déterminée [9]. Il apparait donc que faire de ces processus un objet d'étude à des fins d'analyse et d'explication ne semblait pas aisé. Ainsi, l'étude de la morphogénèse était longtemps jugée non-objective et comme relevant du domaine de l'expérience phénoménologique. Cependant, avec la nécessité croissante de compréhension, de prédiction et de contrôle des systèmes biologiques, la question de la forme a soulevé un grand intérêt pour la science.

Les systèmes complexes observables dans la nature sont composés d'un nombre important d'éléments qui interagissent localement pour produire un comportement global à une échelle supérieure. C'est le cas des formes crées par prolifération cellulaire lors de l'embryogenèse des organismes vivants. Au cours de ce processus de développement, certaines cellules de la forme ont pu se différencier. Un mécanisme par lequel elles se dotent de nouvelles propriétés pouvant être associées à des fonctions bien précises faisant évoluer la forme vers sa cible. Pourtant au stade initial de la formation d'un être vivant, toutes ses cellules ont le même génome, mais au cours du développement elles n'expriment pas toutes les mêmes gènes. Pourquoi la dynamique de ces cellules a-t-elle changé pour amener à une différenciation ? Quels sont les facteurs qui entrent en jeu ? Quel est le mécanisme par lequel elle se réalise ? Telles sont les questions qui sous-tendent toute la compréhension des mécanisme d'auto-organisation et d'auto-adaptation collective qui cible et atteint la forme bien guidée des organismes vivants.

Pour répondre à ces questions, il convient d'aller au-delà de la génétiques et

de la biologie moléculaire pour reconsidérer l'environnement des cellules. En effet, une propriété des systèmes complexes est qu'il n'existe aucune théorie qui permet de déduire a priori leur comportement global à partir de celui de leurs composants. L'environnement est déterminant dans la différenciation cellulaire [21]. Les cellules évoluent en tenant compte des feedback qu'elles reçoivent du milieu extérieur. En effet, les cellules de l'organisme évoluent et modifient celui-ci, qui à son tour rétro-agit sur les cellules.
Pour arriver à mieux comprendre les mécanismes qui sous-tendent la morphogénèse des organismes multicellulaires, il faut arriver à comprendre à travers ces trois niveaux d'organisation (le fonctionnement, le rôle, la structure) et les interactions des cellules qui mènent vers une forme stable. Comme nous le verrons, pendant longtemps certains travaux ont tenté de rendre compte du processus de morphogénèse. D'une part, en considéraient essentiellement la biologie moléculaire et la génétique. D'autre part, en représentant les cellules individuellement pour ensuite faire converger le comportement global du système vers des formes prédéfinies. Cependant, ces comportements globaux, parfois inattendus, sont le résultat des mécanismes sous-jacents à la dynamique cellulaire, laquelle demande une formalisation solide et adéquate. Toute la difficulté de la modélisation relève de cette complexité du système dynamique lors du passage de l'individuel au collectif, complexité dont l'étude mathématique et numérique est difficile à établir.
Ce stage de recherche s'est inscrit dans ce cadre là. Il s'est déroulé au CERV[1] dans l'équipe IHSEV[2] du laboratoire Lab-STICC[3].
La principale activité durant ce stage a été d'abord des modèles mathématiques pour la morphogénèse par une formalisation des mécanismes de la dynamique cellulaire. Cette formalisation étant basée sur l'analyse mutationnelle qui offre de nouvelles pistes permettant de représenter les formes par des ensembles qui se déplacent, se déforment, se multiplient et croissent. Ensuite, pour mieux étudier et comprendre l'influence de l'environnement sur la dynamique cellulaire, nous avons développé des algorithmes basés sur cette formalisation permettant d'explorer tout l'espace des génomes possibles d'une base de type cellulaire. L'exploration a pour but de vérifier les conditions dans lesquelles certains gènes sont activés par les cellules pour se différencier et voir si ces conditions ne sont valables que pour un certain nombre de génomes particuliers. Suivant la dynamique des cellules mis en jeu, si les gènes ne sont pas activés, on se propose, sous certaines conditions, de modifier de manière automatisé les propriétés du génome pour procéder à la vérification avec la nouvelle dynamique résultante. Enfin, nous avons simulé nos algorithmes avec un outil de simulation de prolifération

1. Centre Européen de Réalité Virtuelle
2. Interaction Humain Système et Environnement Virtuel
3. LABoratoire en Sciences et Techniques de l'Information, de la Communication et de la Connaissance

cellulaire[11] sur l'activation des gènes qui mènent à l'apparition du drapeau sénégalais.
Ce document soulève d'abord la complexité de certaines notions en biologie cellulaire avant de présenter quelques travaux de modélisation de la morphogénèse, notamment en mathématique et informatique (section 1). Ensuite, nous présenteront notre environnement technique durant ce stage (section 2). Dans une troisième partie, nous détaillerons nos travaux (section 3) et illustrerons les tests qui ont été menés avec l'outil de simulation. Nous terminerons ce rapport en discutant des enjeux qui découlent des nouvelles pistes de recherche (section 4.

1 Bibliographie sur les modèles pour la morphogénèse

Dans cette section, nous allons d'abord exposer les notions complexes en biologie cellulaire soulevées par les biologistes[23] comme étant des difficultés pour la modélisation de la morphogénèse 1.1. Ensuite, nous verrons en mathématiques des travaux en analyse fonctionnelle, les limites qu'ils ont posé et qui ont ouvert des pistes nouvelles 1.2. Nous aborderons en dernier lieu quelques modèles computationnels et des travaux en algorithmique ayant inspiré nos travaux durant ce stage de recherche1.3.

1.1 Morphogénèse animale : notions biologiques et complexité

Les études biologiques mettant en œuvre des procédures pour traquer les subdivisions cellulaires d'un embryon se confrontaient le plus souvent à des difficultés relevant du pas de temps limité. Ainsi, toutes les cellules ne pouvaient pas être prises en compte. Durant cette dernière décennie, les sciences biomédicales ont connu une nette révolution. Les avancées et innovations en biotechnologie, plus particulièrement en microscopie et en imagerie ont permis de disposer d'une plus grande quantité de données biologiques. Ces données ont permis notamment une description inédite en des détails précis de beaucoup de composants et structures d'organismes vivants. En 2007, C. Melani et coll[18] sont parvenus à obtenir des images en 3D, à temps réel, de divisions cellulaires d'un embryon de poisson zébre avec une microscopie confocale à balayage laser (voir figure 1). En 2008, M. Campana et coll.[6] ont conçu un logiciel pour la détection et la segmentation de membranes et de noyaux cellulaires observés à temps réel à partir d'un embryon de poisson-zèbre. En 2010, O. Nicolas et coll.[20] présente un outil pour la reconstruction du lignage cellulaire et du déploiement spatio-temporel 2D et 3D des 10 premiers cycles de division d'un embryon de poisson zébré. Avec toutes ces techniques nouvelles, il a été possible de collecter une grande quantité de données expérimentales sur les propriétés émergentes d'ordonnancement des cellules et surtout l'importance de la dynamique dans l'évolution des formes qu'il faudra désormais prendre en compte en biologie virtuelle. Au cours du développement embryonnaire des organismes multicellulaires, la morphogénèse et la différenciation cellulaire sont interdépendantes. L'émergence de nouvelles formes vivantes passe par le crible de l'ontogenèse [4] au sens où toute variation phénotypique [5] a ses fondements embryologiques. Et seules les variations ne compromettant pas la viabilité embryonnaire de l'organisme

4. Développement d'un individu depuis sa conception jusqu'à sa forme adulte définitive.
5. Variation dans un caractère observable de l'individu.

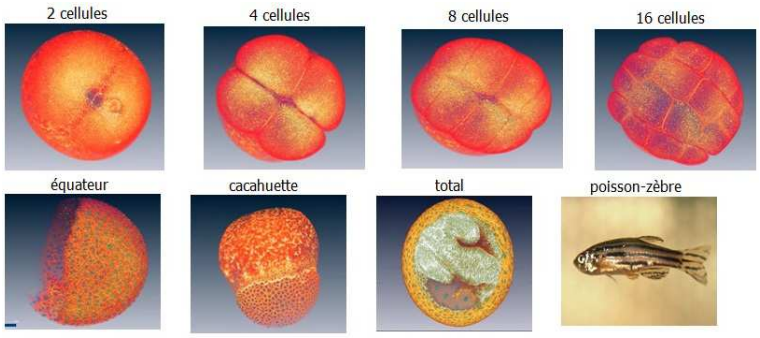

FIGURE 1 – Image vidéo réelle pour la reconstruction en 3D [18]

seront transmises au fil des générations. Par ailleurs, il faut noter que la question de la diversité cellulaire se pose avant même celle de l'acquisition de la forme. Lorsque l'embryon ne compte que quelques paires de cellules, on peut constater déjà une diversification du contenu biochimique voire de la morphologie des cellules embryonnaires. Cette diversification peut se générer de trois manières : première cellule hétérogène, grappes de cellules d'environnements différents, interactions moléculaires et génétiques. Cette acquisition de propriétés différentielles d'adhésion, de mobilité cellulaire et d'expression génétique conduit à l'émergence de patterns. Au regard de ce constat, si on sait repérer une cellule et la suivre pour a posteriori connaitre sa lignée, on peut aussi identifier à chaque instant les cellules qui ont le même futur. Ce dernier permet de délimiter les populations de cellules qui préfigurent les organes. Cependant, son individualisation spatio-temporelle n'est pas simple à délimiter. Comme l'a souligné R. Lewontin[8], la métaphore du développement n'est pas la mieux choisie pour désigner l'ontogenèse et les processus morphogénétiques qui sous-tendent la formation de l'organisme. En effet, la morphogénèse n'est pas un simple aboutissement d'un déploiement vers une forme prédéfinie mais sous-entend plutôt une succession de bouleversements qui révèlent de nouvelles frontières entre populations de cellules, de nouveaux compartiments qui se déforment et se transforment. Et jusque là, peu d'aspects de ces évènements sont connus. Par ailleurs, N. Peyrieras [23] indique que le paradigme dominant dans lequel évolue la biologie, en particulier la biologie du développement est caractérisé par le concept de « genetic switch »par lequel chaque état cellulaire est déterminé par une combinatoire unique de gènes « on »ou « off ». Alors que dès les années 1940-1950, C. H. Waddington distinguait déjà les facteurs épigénétique comme intervenant aussi dans la construction de la forme. Les aspects génétiques sont en effet la partie ' programmée ' de l'individu, tandis que la partie épigénétique regroupant toutes les composantes extérieures aux gènes (dynamique suivant l'environnement de l'organisme) est susceptible de donner toutes les formes possibles. Le cancer par exemple est une perturbation de type épigénétique.

Il se définit comme une prolifération anormalement importante de cellules au sein d'un tissu l'intérieur d'un tissu jusqu'à compromettre sa survie. Cependant, si les facteurs génétiques sont determinants, les facteurs épigénétiques eux sont réversibles. En ce sens, l'étude des mécanismes sous-jacents à la dynamique cellulaire dans le développement des formes ouvre des perspectives thérapeutiques pour le cancer. Pour ce faire N. Peyrieras prône une prise en compte des dynamiques spatio-temporelles à toutes les échelles. Elle pense que les comportements cellulaires ou morphodynamiques sont stéréotypés : la cellule vit et meurt. Alors qu'une cellule peut aussi se diviser, changer de forme, modifier ses interactions avec ses voisines, se mouvoir. Et ces comportements individuels peuvent faire émerger des comportements collectifs pouvant engendrer un pattern ou une forme. En outre, les mesures nécessaires à l'interprétation des mouvements et déformations cellulaires et donc des forces biomécaniques en jeu dans l'organisme sont peu disponibles. Ces forces se propagent à travers les tissus à longue distance et pourraient déterminer la coordination des mouvements de populations cellulaires. Cette forme de causalité, qu'on pourrait qualifier de « top-down »est peu prise en compte dans les descriptions de processus de morphogénèse. Par ailleurs, le modèle de tenségrité[6] propose que la morphogénèse des tissus dépende de l'interaction mécanique entre les cellules et la matrice extracellulaire qui met le tissu sous tension isométrique. Des remaniements locaux de la matrice extracellulaire peuvent alors conduire à un étirement de la matrice extracellulaire et des cellules qui y adhèrent.

1.2 Modèles mathématiques

Comme le dit G. Desmeulles[7], les mathématiciens n'ont pas attendu l'ère de l'informatique pour traiter la complexité à tel point que derrière tout outil informatique utilisé dans la compréhension d'un phénomène complexe, nous retrouvons des concepts mathématiques associés. Ainsi, il existe de nombreuses théories et modèles mathématiques pour expliquer les phénomènes biologiques. Parmi celles-ci, la modélisation de systèmes par équations différentielles. Elle comporte principalement deux avantages essentiels :

1. l'approche est formalisée. En effet, une équation mathématique est universellement compréhensible, des solutions analytiques peuvent être trouvées et si ce n'est pas le cas, des simulations numériques peuvent être effectuées,
2. un système d'équations différentielles permet de décrire l'évolution d'une population de cellules ou de nombreux types d'interactions entre plusieurs populations de cellules.

6. Faculté d'une structure à se stabiliser par le jeu des forces de tension et de compression qui s'y répartissent et s'y équilibrent.

La morphogénèse traite de l'évolution des formes, qui sont essentiellement des ensembles, au sens mathématique du terme. Leur évolution et leur analyse nécessitent donc une analyse intrinsèque au niveau des ensembles. Une tradition qui s'est maintenue jusque là consiste à forger des outils, en analyse fonctionnelle, représentant les formes par des fonctions[19] : fonction caractéristique, fonction d'appui, indicateur, jauge, distances tant usuelles que signées, fonction de niveau, etc. Cependant, les diverses techniques associées se sont basées sur des propriétés de fonctions, des applications, et des résultats en analyse et en géométrie. Alors que celles-ci exigent une régularité mathématique des ensembles que ne possèdent pas les formes biologiques, typiquement des propriétés de différentiabilité des bords [2]. Ces théories ne sont d'ailleurs pas conçues pour étudier des ensembles mais plutôt des fonctions.
La théorie de la viabilité, développée par J.P Aubin[1] offre des outils et des concepts pour contrôler un système dynamique dans un environnement fixe afin de le maintenir dans un ensemble de contraintes de viabilité. En effet, pour un système dynamique donné, discret ou continu, et un ensemble $\mathcal{K}$ défini par un système de contraintes indépendantes du temps, il n'est pas certain qu'à partir d'un point initial $x0$, il existe une solution du système dynamique qui reste toujours dans K. Si de tels points existent dans $\mathcal{K}$, le plus grand ensemble qu'ils définissent est appelé *noyau de viabilité* de $\mathcal{K}$. Le sous-ensemble de ce *noyau de viabilité* défini par les points initiaux $x0$ de $\mathcal{K}$ à partir desquels toutes les solutions issues de ces points ne sortent jamais de $\mathcal{K}$ est appelé *bassin de capture* (voir figure 2). Cette théorie se donne

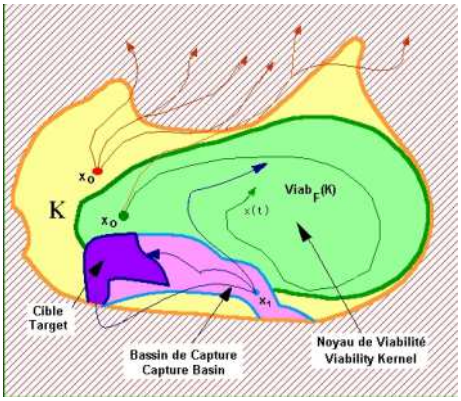

FIGURE 2 – évolutions viables, restant dans un environnement décrit par des *contraintes de viabilité*

comme objectif d'expliquer mathématiquement et numériquement les évolutions gouvernées par des *systèmes évolutionnaires* dans bien des domaines (économie, sciences cognitives, théorie des jeux, automatique, biologie, etc.). De tels systèmes ne sont pas déterministes, mais régissent sous incertitude

des évolutions soumises à des contraintes de viabilité (ou d'optimalité intertemporelle) et guident ces évolutions vers des cibles afin de les atteindre en temps fini. Il s'agit essentiellement de faire émerger les rétroactions sous-jacentes qui permettent de réguler le système et de trouver des mécanismes de sélection pour les mettre en œuvre. Ces évolutions peuvent concerner :

- des contraintes à s'adapter à un environnement
- une évolution sous incertitude contingente, tychastique ou stochastique,
- une co-évolution avec un environnement (viabilité mutationnelle et morphologique)...

Bien qu'offrant un cadre adéquat pour la modélisation de l'évolution de système dynamique, la théorie de la viabilité s'applique cependant dans le cadre de l'analyse univoque (application associant à un point, un autre point). Elle ne semble pas de ce fait appropriée pour étudier un système comme la dynamique cellulaire qui croît et se multiplie, relevant de ce fait de l'analyse multivoque (application associant à un point, un ensemble). D'où l'intérêt d'avoir développé de nouvelles pistes en définissant une notion de vitesse pour ensuite donner du sens à cette notion nouvelle d'équations morphologiques gouvernant l'évolution d'ensemble, leur déplacement, leur déformation, leur accroissement par extension du concept d'équations différentielle. L'analyse mutationnelle que nous verrons en détails dans la section des travaux menés en modélisation mathématique durant ce stage 3.1, est le fondement mathématique de cette piste nouvelle.

1.3 Modèles computationnels

Les domaines d'investigation que constituent les quatre disciplines suivantes : nanoscience, biotechnologie, technologie de l'information et science cognitive, résumées en NBIC [3], ont convergé pour offrir tous les composants de l'ingénierie des systèmes complexes appliquée à la biologie. Plusieurs modèles existent pour l'étude de la morphogénèse, nous allons présenter ici quelques uns basés sur les automates cellulaires. Ensuite, nous présenterons l'approche agent qui est adopté dans la conception de l'outil de simulation que nous avons utilisé durant ce stage.

1.3.1 Automates cellulaires

Les automates cellulaires ont été inventés par Stanislaw Ulam et John Von Neumann, à la fin des années 40. L'objectif de départ était l'étude de l'auto-reproduction. Ainsi, dans les années 60, Conway invente le plus célèbre de leurs ancêtres : le jeu de la vie. Ses utilisations principales étant de traiter les problèmes selon une approche ascendante, parallèle et en déterminant les comportements des entités élémentaires de façon locale. Le système évolue

sur un réseau maillé, le plus souvent une grille en deux dimensions à mailles carrées, comportant :

- N cellules identiques,
- un pattern de voisinage d'interaction,
- un ensemble fini d'états possibles,
- une règle de transition appliquée simultanément à chaque pas de simulation à toutes les cellules.

Les automates cellulaires sont synchrones et déterministes. Pourtant, aucune théorie ne permet de prévoir a priori leurs comportements complexes et parfois chaotiques. Plusieurs travaux ont été réalisés avec les automates cellulaires pour la modélisation du développement d'un organisme biologique. En 1999, H. De Garis [12] a proposé un modèle d'embryogenèse artificielle qui a été parmi les premiers systèmes qui ont exploité l'évolution d'automates cellulaires par algorithme génétique afin de produire des formes en 2D. Cependant, les structures cellulaires obtenues sont prédéfinies dans son modèle. L'auteur a envisagé d'ajouter dans le modèle des gènes activateurs qui vont doter les cellules de mouvement et permettre à un groupe de cellules de remplir certaines fonctions.
En 2001, A. F. M. Marée et coll. [14] présentent un cas d'amibes capables de se regrouper entre elles pour former un organisme multicellulaire lorsqu'elles n'ont plus de nourritures. Dans ce modèle, la seule différence entre les cellules est la force d'adhésion. En effet, les cellules de la tige adhèrent fortement entre elles mais peu avec celles qui les entourent. Cette différence induit la formation d'un tube qui donnera ensuite la tige de la fructification (voir figure 3). Dans ce modèle, on remarque que l'apparition de la forme est due uniquement aux forces avec lesquelles certaines cellules spécialisées adhèrent entre elles.

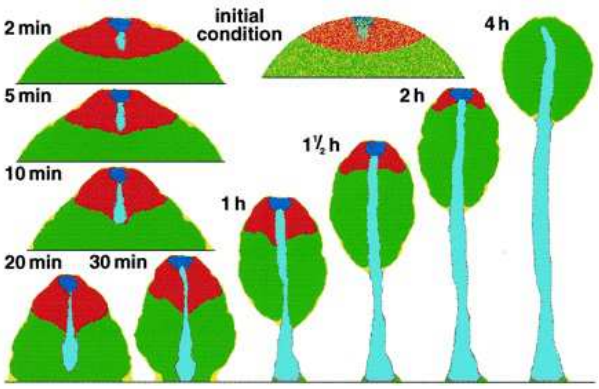

FIGURE 3 – sequence de simulation du processus d'évolution : `http://www.pnas.org`

En 2005, L. Forest développe dans une partie de sa thèse un modèle pour la prolifération et la différenciation cellulaire lors de la croissance radiale des conifères. Ce modèle met en évidence la prise en compte des règles de voisinages entre cellules dans la différenciation. Les cellules prennent trois états selon leur voisinage. Leur division se fait de manière aléatoire mais il avec une plus forte probabilité de reproduction lorsqu'elles ont un voisinage de même type.
En 2007, A. Chavoya Pena[22] a proposé un modèle de croissance artificiel qui soit capable de générer des structures cellulaires reposant sur des mécanismes d'auto-organisation et d'interaction avec un environnement artificiel. Ce modèle visait à comprendre comment des formes complexes peuvent apparaitre à partir d'un petit groupe de cellules initiales indifférenciées. Dans ce modèle, une série de gènes régulateurs codés au début de chaque génome constitue un Réseau de Régulation Artificiel (RRA) et détermine les règles d'évolution de l'automate. Ces gènes de régulation sont suivis par une série de gènes structurels, dont chacun peut générer une forme particulière simple telle qu'un carré ou une ligne (voir figure 4). Les différentes structures obtenues dépendent principalement du type de RRA. Un algorithme génétique est utilisé pour obtenir un RRA pouvant faire évoluer la table de transitions et le nombre d'itérations vers des formes prédéterminées. Démarrant l'algorithme avec une seule cellule active au milieu de la grille de l'automate, les cellules se reproduisent en suivant la table de transition définie par l'algorithme et pour autant d'itérations indiquées dans le champ de contrôle du chromosome. L'auteur souligne que ce modèle permet seulement de traiter des formes convexes. En outre, les cellules ne peuvent ni mourir, ni même se déplacer.

FIGURE 4 – Les cellules de l'automate forment une structure carrée à trois couleurs. Pour chacune des formes colorées, le réseau de régulation a déclenché un gène structural différent.

Dans un automate cellulaire, le temps et l'espace sont discrets. R. Doursat [10] propose lui, un modèle continu utilisant aussi des réseaux de gènes régulateurs comme dans le modèle de A. Chavoya Pena. Cependant, contrai-

rement à ce modèle là, dans le modèle de R. Doursat, l'ensemble des cellules est décrit sur un espace continu. Au regard des principes du développement multicellulaire comme inspiration des systèmes artificiels capables d'auto-organisation, le modèle comporte aussi des éléments de la biomécanique cellulaire. Des dynamiques de déformation topologique ou morphodynamique qui peut conférer une forme non triviale au système. Ainsi, des propriétés de la cellule sont pris en compte : taux de division cellulaire, forces d'adhésion et vitesse de migration. Ces propriétés dépendent du domaine d'expression du gène auquel appartient la cellule. Ce modèle a permis d'établir une dépendance fonctionnelle entre l'identité des cellules et leurs comportements mécaniques. Lors de la simulation du modèle, le niveau de concentration du gradient morphogénétique, relatif à la position de la cellule, indique à celle-ci la valeur d'expression de son gène lui conférant de ce fait ses nouvelles propriétés : apparition de nouvelles cellules par différenciation. Les cellules prolifèrent et s'auto-organisent selon leurs nouvelles propriétés (voir figure 5).

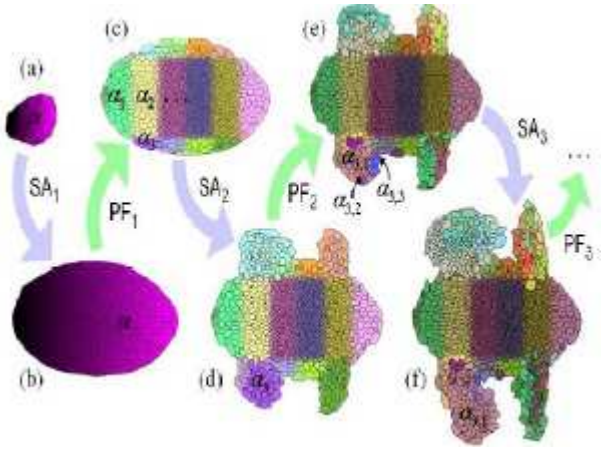

FIGURE 5 – Les deux étapes de simulation du modèle de R. Doursat [10] . Prolifération cellulaire et auto-assemblage :SA | Différenciation cellulaire et formation de patterns : PF

1.3.2 Systèmes multi-agents

Les systèmes multi-agents sont utilisés pour modéliser des systèmes complexes et offrent la possibilité de simuler un certain nombre de composants autonomes dans un environnement afin de déterminer la nature du phénomène étudié dans sa globalité sans contrôle centralisé[24, 9]. Pour J. Ferber, l'objectif est de donner naissance à des systèmes informatiques capables d'évoluer par interaction, adaptation et reproduction d'agents relativement autonomes et fonctionnant dans des univers physiquement distribués. Y.

Demazeau définit un système multi-agents comme étant composé de quatre concepts clefs que sont les Agents, l'Environnement, les Interactions et l'Organisation. C'est l'approche Vowels ou AEIO. Les composants ou agents, n'ont qu'une vision partielle de l'univers dans lequel ils évoluent. Chaque agent possède un cycle d'exécution pendant lequel il commence par percevoir son environnement à l'aide de capteurs, d'antennes ou de récepteurs. Ensuite, en fonction des informations provenant de l'environnement et en fonction de son état interne, il prend une ou plusieurs décisions. Chaque décision modifie l'état interne de l'agent, son comportement ou sa morphologie (voir figure 6). Un agent est dit réactif s'il ne possède pas, ou s'il possède de façon rudimentaire une représentation de son environnement. A contrario, un agent est dit cognitif s'il est capable de se représenter son environnement et d'en faire une carte pour planifier ses actions. Il est également possible, comme indiqué par G. Desmeulles [7], de définir les agents réactifs comme étant des automates cellulaires auxquels ont été ajoutées certaines propriétés tels le mouvement, l'asynchronisme et l'aléatoire par exemple, pour accroître les possibilités de modélisation.

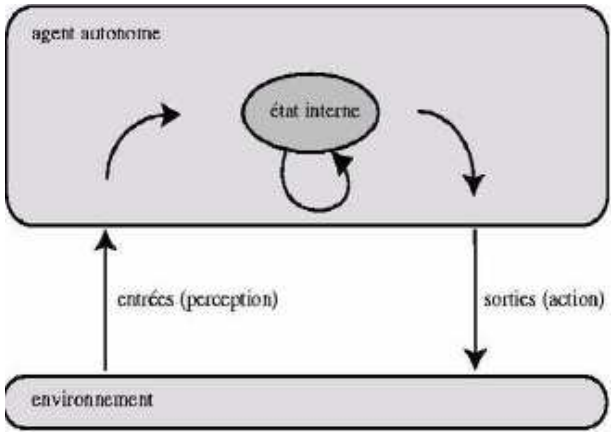

FIGURE 6 – Le système de perception-décision-action d'un agent dans un système multi-agent

De manière générale, les automates cellulaires et les systèmes multi-agents sont des paradigmes de prédilection pour les études du vaste domaine de la vie artificielle. Les systèmes multi-agents s'appuient sur des notions de robustesse, d'émergence, d'auto-organisation et d'adaptabilité reprenant ainsi les propriétés fondamentales des systèmes complexes observés dans la nature. Selon P. Ballet[4], les motivations en ce qui concerne la programmation multi-agents sont multiples. D'une part, tout système composé d'entités autonomes se modélise naturellement par une approche agent : le cas pour des systèmes multicellulaires. En effet, une cellule possède de nombreux récepteurs pour observer son environnement, sait prendre des décisions et détient

la possibilité d'agir sur son milieu. De plus, un système multi-agents peut facilement intégrer de nouveaux agents en son sein sans avoir à refondre le système. De même, la suppression d'un agent ne pose pas de complication ni du point de vue du concepteur, ni du point de vue du système. Cependant, cette simplicité et cette robustesse ne doivent pas cacher les difficultés de sa mise au point. Prévoir toutes les interactions possibles est souvent long mais indispensable pour déceler les éventuels problèmes d'implémentation. En plus, les simulations basées sur les systèmes multi-agents[5] peuvent poser des problèmes de convergence et de stabilité. En effet, comme le montre B. G. Lawson et S. Park [15], l'exécution des agents a un impact important sur les résultats de la simulation. Dans certains cas, un ordonnancement synchrone des exécutions des agents est nécessaire pour faire converger le système vers une forme et qui soit stable.

1.4 Algorithmes génétiques

Dans la recherche d'une solution de parcours des possibilités de génomes à partir d'une base de types cellulaires pour mieux comprendre l'influence de l'environnement dans l'évolution des formes et la différenciation cellulaire, nous avons étudié quelques algorithmes génétiques. Nous en présentons leurs idées principales, l'intérêt de leur approche par rapport à notre problématique et les limites de leur application.
Les principales caractéristiques d'un problème pour lequel, il est envisageable d'appliquer un algorithme génétique sont les suivantes :

- avoir un espace de recherche important,
- non existence d'algorithme déterministe adapté
- besoin d'arriver à une solution acceptable et assez rapide
- Solution proche d'un traitement naturel
- ...

Avant d'élaborer l'algorithme et de songer à la résolution du problème donné, il convient de respecter certains préalables[13] :

- Définir un espace de problème,
- Représenter l'expression génotypique (ou phénotypique) des solutions potentielle,
- mettre en place un mapping entre génotype et phénotype.
- Déterminer la manière d'exprimer la fonction de fitness pour être sûr de converger vers les solutions recherchées.

I. Harvey présente en 2009, un algorithme génétique de type microbien où le transfert génétique ne se fait pas par tout individu d'une génération vers un autre individu de la génération suivante comme c'est le cas avec les algorithmes génétiques bactériens. Le choix de transfert génétique a lieu à l'issue d'une compétition entre deux individus d'une même génération. Le gagnant peut ainsi transférer son génotype au perdant qui va de ce fait être mieux

adapté pour faire parti de la génération suivante. Son algorithme peut être implémentée de deux manières :

1. Single role
   - choisir un paire au hasard dans la génération actuelle
   - les recombiner puis les muter pour produire un nouvel individu
   - évaluer les deux individus initialement choisis
   - Conserver plus adapté : le gagnant
   - Remplacer le moins adapté (perdant) par le nouvel individu créé dans la génération suivante
2. Dual role (voir figure 7)
   - choisir un paire au hasard dans la génération actuelle
   - évaluer les deux individus
   - copier un segment (de 0 à 100%) du génotype du gagnant
   - coller la copie au segment correspondant dans le génotype du perdant
   - muter le perdant avec son nouveau génotype pour constituer un nouvel individu
   - remplacer le perdant par le nouvel individu créé dans la génération suivante

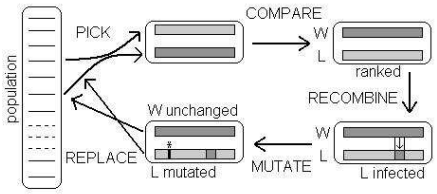

FIGURE 7 – Procédure de l'algorithme génétique microbien en dual role | *Source : [13]*

L'avantage de cet algorithme est sa simplicité et son applicabilité à un grand nombre de problèmes. En plus,il offre la possibilité d'évaluer chaque individu dans un processeur différent en cas d'accès à des clusters de processeurs tournant en parallèle. Dans le cas de l'étude de la prolifération cellulaire, il peut être appliqué si l'objectif est de contraindre la dynamique cellulaire à respecter un certain nombre de contrainte qui lui permettront d'atteindre une forme donnée. Cependant, dans le cas de notre sujet de recherche, l'efficacité est toujours acquise par une dynamique de cellules et non par les cellules prises individuellement en tant qu'agent.

2 Environnement technique

Dans cette section, nous présentons la bibliothèque `AReVi` (Atelier de Réalité Virtuelle)qui a servi à développer l'outil de simulation que nous avons utilisé durant ce stage et dont l'architecture et les fonctionnalités vont être détaillées en seconde partie.

2.1 Atelier de Réalité Virtuelle : AReVi

2.1.1 Mécanismes centraux

L'intégralité des algorithmes développés durant ce stage se sont réalisés avec le langage de programmation C++ en utilisant la bibliothèque `AReVi`. Le `C++` est un langage orienté objet, bas niveau (proche de la machine) et fortement typé. Cependant, les performances obtenues sont au prix de la gestion de la mémoire déléguée au programmeur. En ce sens, `AReVi` revêt un intérêt particulier dans la mesure où il propose des mécanismes de ramasse-miettes pour décharger l'utilisateur de cette tâche. `AReVi` est une bibliothèque de simulation d'entités autonomes et de rendu `3D` développée au CERV par F. Harrouet. Elle utilise le mécanisme de `template` afin de fournir des types qui se substituent complètement aux pointeurs mais s'utilisant de la même manière. Leur principal rôle est de mettre en œuvre un système de ramasse-miettes qui soit à la fois automatique et contrôlable. Une séquence est respectée dans la destruction des objets pour éviter des effets de bord. Les types fournis se décomposent en trois catégories :

- représentant des substituts de pointeurs, permettant entre autre de bénéficier d'un ramasse-miettes : `ArRef<T>`, `ArConstRef<T>`, `ArPtr<T>`, `ArConstPtr<T>`
- une hiérarchie de classe dont l'ancêtre commun est `ArObject`. Ces instances sont manipulées par des `ArRef<T>`, `ArConstRef<T>`, `ArPtr<T>`, `ArConstPtr<T>`
- le singleton `ArSystem` initialisant l'application et l'exécution de la boucle de simulation

En fonction de son programme, il est possible de choisir les objets qui sont susceptibles d'être détruits automatiquement par le ramasse-miettes (transitoires) et ceux qui ne le sont pas (non-transitoires).

2.1.2 Hiérarchie

Toutes les classes de la bibliothèque `AReVi` et toutes les classes d'une application utilisant ses services, hérite d'une classe : `ArObject`. La définition des classes comporte des similarités avec ce qui se fait en C++. Il existe toutefois des règles à respecter dont quelques unes :

- les objets de type `ArObject` ou d'un type dérivé ne peuvent être créés que par allocation dynamique et ne sont manipulés qu'à travers des pseudo-pointeurs.
- Une macro `ARCLASS` figurant dans la partie publique et reprenant le nom de la classe est nécessaire pour mettre en place un certain nombre de mécanismes (comme interdire l'usage du constructeur par défaut ou par recopie et de l'opérateur d'affectation ou encore définir les méthodes statiques `nullRef`, `nullPtr`, `thisRef`, `thisPtr`)
- l'utilisation d'une macro nommée `ARCLASSDEF` est indispensable pour réification de la classe implémentée. Si elle est une classe abstraite, ce sera `ARCLASSNOVOIDDEF`.
- l'héritage multiple est volontairement prohibé dans la hiérarchie `ArObject`.

Toutes les instances d'`ArObject` disposent de leur propre boîte à messages et peuvent en consulter le contenu. La communication est asynchrone car le fait d'envoyer un message à un objet ne provoque pas nécessairement son traitement immédiat (contrairement aux appels de méthodes). Les objets disposent d'une certaine autonomie de décision, ils peuvent choisir de traiter ou non le message reçu, dans un délai variable et selon des critères changeants. Cette démarche évoque la notion d'acteur ou d'agent.

2.1.3 Boucle de simulation

La classe `ArSystem` ne s'inscrit pas dans la hiérarchie des `ArObject`. Elle sert principalement à initialiser l'application et à effectuer une boucle de simulation qui active les différents traitements tout en assurant la mise à jour de l'affichage et la réaction aux interventions de l'utilisateur. Cette instance doit être unique et se fait dans le programme principal de l'application. Le constructeur attend les arguments de la ligne de commande afin de les mémoriser pour pouvoir les restituer plus tard dans l'application. Il est aussi possible d'effectuer ces mêmes opérations d'initialisation et de destruction à l'aide de méthode statique : utile pour embarquer `AReVi` dans une autre application. Le déroulement normal d'une application `AReVi` se décompose généralement en deux phases :

- l'initialisation de la simulation par la création d'un ensemble d'objets dotés d'une autonomie d'exécution.
- la simulation proprement dite consistant à laisser vivre ces objets

Pendant toute la durée de la simulation, `AReVi` doit se charger de la mise à jour des vues `3D` et de la détection des actions de l'utilisateur. L'activation des entités autonomes est assurée par un ordonnanceur qui gère des activités. Il est responsable de l'avancement du temps et doit donc exécuter les activités lorsque leur date de déclenchement respective est atteinte. Il est possible de :

- répartir les activités en priorité (1, 0, -1), pour des raisons utilitaires

- provoquer un traitement unique avec une activité ou un traitement répétitif,
- créer plusieurs activités pour un même objet qui seront toutes indépendantes,
- modifier le délai de déclenchement d'une activité,
- redéfinir la priorité d'une activité,
- affecter une période nulle à une activité pour qu'elle soit déclenchée aussi souvent que possible (à chaque cycle de l'ordonnanceur).

2.1.4 Construction et installation

La construction d'`AReVi` repose exactement sur les mêmes principes que la construction d'une application ou d'une bibliothèque utilisant `AReVi`. Pour des raisons de portabilité, les commandes d'`AReVi` ne prennent aucune option qui soit spécifique à un environnement. `AReVi` est principalement constitué d'une bibliothèque contenant la très grande partie des services proposés. Cette bibliothèque est accompagnée de `plugins` fournissant des services annexes. Les étapes de son utilisation sont les suivantes :

- se place dans le répertoire `AReVi`,
- générer les fichiers `makefile` pour la bibliothèque `AReVi` et les `plugins`,
- compiler la bibliothèque `AReVi` et les `plugins`.

Concernant le rendu `3D`, `AReVi` utilise `OpenGL` dans sa version 1.1 pour limiter la dépendance à des fonctionnalités `3D` avancées. Il existe toutefois des mécanismes d'extension pour bénéficier de ces fonctionnalités lorsqu'elles sont disponibles.

2.1.5 Développement avec AReVi

La commande `arevi-config` affiche toutes les options de commande possibles pour obtenir des informations d'ordre général sur `AReVI`. Pour instrumenter le code de son programme avec `AReVi`, on peut utiliser un outil dont le fonctionnement se base sur `doxygen`[7].
Sources : `http://www.enib.fr/~harrouet/arevi.html`

2.2 Outil de simulation : DynCell

J. Echasserieau[11], lors de son stage au CERV, a utilisé la plateforme `AReVi` pour mettre en place une application permettant de simuler des entités sphériques se multipliant sur une surface pouvant prendre plusieurs formes : DynCell. L'objectif était de simuler la prolifération des cellules soumises à

7. logiciel libre permettant de créer de la documentation à partir du code source d'un programme

des contraintes spatiales à travers les différentes configurations possibles de l'environnement. Nous allons présenter l'architecture de DynCell avant de décrire ses fonctionnalités.

2.2.1 Architecture

L'architecture du programme est basée sur la notion de forme. Toutes les classes de l'application héritent d'une classe mère générique intitulée `Form` (voir figure 8). Chaque objet instancié dans la simulation est par définition, une forme. Cette classe principale possèdent deux classes filles : `Environment` et `Cell`, qui sont les deux types d'objets instanciés pour effectuer une simulation.

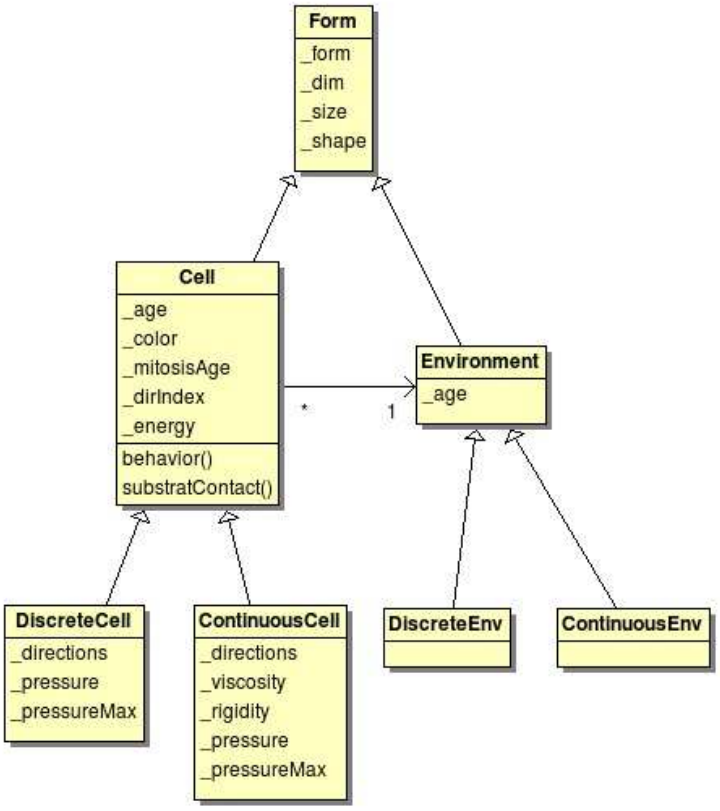

FIGURE 8 – Diagramme des classes de DynCell, l'architecture du programme est basée sur la classe générique Form | *Sources [11]*

L'exécution d'une simulation instancie toujours un objet `environnement` et des objets `cellules` évoluant dans cet environnement. La classe `Cell` est chargée de représenter des cellules. Un objet `Cell` est défini par son age,

une couleur, une vitesse de division et une quantité d'énergie. Sa vitesse de division correspond au pas de temps pour lequel la cellule peut faire une mitose. Lors d'une mitose,un autre objet `Cell` est créé à côté de la cellule mère qui a déclenchée la division (voir figure 9). Un environnement est quant à lui un objet `Form` qui a une durée de vie. Il sert délimiter l'espace dans lequel la population de cellules évolue ou encore à exercer des forces extérieures sur elles.

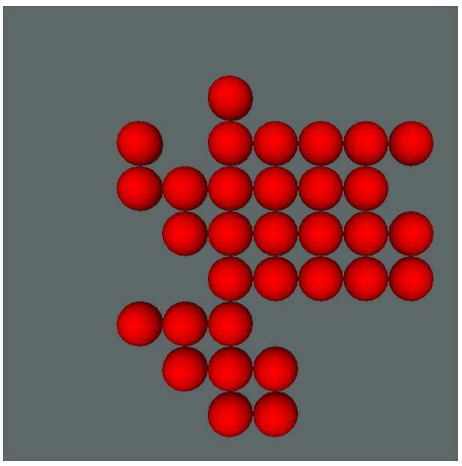

FIGURE 9 – Simulation en environnement discret avec DynCell | *Sources [11]*

2.2.2 Fonctionnalités

Avec DynCell, différents comportements peuvent être observés parmi les modèles virtuels obtenus suivant le paramétrage choisi. Pour mieux apprécier et comprendre les mécanismes qui entrent en jeu lors de la morphogénèse, il faudra toutefois garder une certaine flexibilité dans le paramétrage des cellules. Le programme possède deux modes de simulation : un mode stochastique et un mode sous contrôle. Les cellules représentées à l'écran sous forme de sphères peuvent proliférer dans un environnement discret ou continu. Dans le second cas, le mouvement des cellules est plus finement décrit. La distance entre cellules qui définit la force d'attraction et de répulsion entre elles est variable. Une interface graphique (voir figure 10) permet de choisir les paramètres de simulation, de procéder à une modification dynamique de ces paramètres et de choisir les mécanismes cellulaires (tel que l'apoptose) à activer ou désactiver durant la simulation. D'autres options telles que la présentation en `2D` ou `3D`, la taille et la forme de l'environnement ou des cellules peuvent être définies et ajustées. Dans le cas d'une simulation continue, chaque cellule peut percevoir ses voisins à travers un rayon d'attraction et évaluer les contraintes spatiales sous forme de tension.

Les contraintes sont cruciales à l'évolution de la cellule, trop fortes, elles empêchent à la cellule de se diviser et donc d'être viable. Un paramètre de contrainte maximale permet de définir un seuil sous lequel la cellule reste viable. Pour tenir compte de l'influence de l'environnement, un paramètre définit le nombre maximal de cellules qu'une cellule peut pousser en se divisant. Et lorsque la tension de la cellule est plus forte que le seuil maximal, la cellule ne peut plus se diviser. Deux modes de division sont considérés : soit la cellule choisit de se diviser dans la direction où la tension est moins intense, soit la direction de la division est prédéfinie. Il est aussi possible d'attribuer une certaine quantité d'énergie à chaque cellule. Une partie de cette énergie étant utilisée pour maintenir sa structure et assurer sa croissance et l'autre partie pour la maturation et la reproduction. La cellule meurt lorsque le niveau d'énergie vient à être trop bas. Elle peut cependant en recouvrer lorsqu'elle est en contact avec une partie appropriée de l'environnement.

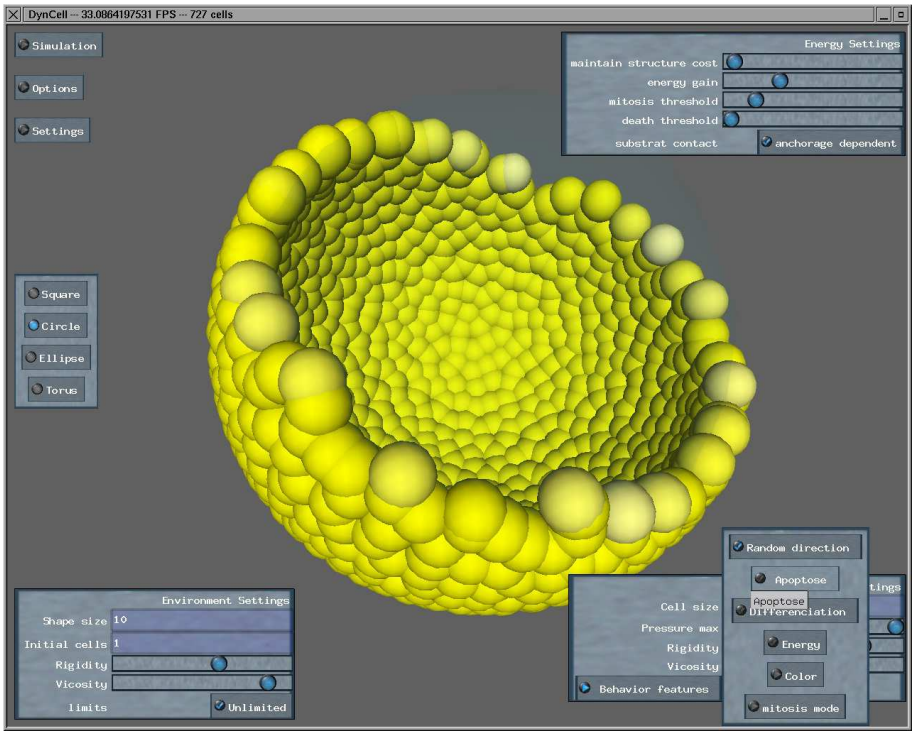

FIGURE 10 – Interface graphique de DynCell | *Sources [11]*

3 Travaux

Dans cette section, nous allons exposer spécifiquement nos travaux durant ce stage. Nous présentons d'abord notre modèle mathématique pour la morphogènes basée sur l'analyse mutationnelle 3.1. Ensuite nous allons décrire les étapes de l'implémentation des algorithmes développés pour l'exploration de génomes 3.2. Enfin, nous illustrerons ces algorithmes sur des exemples avec l'outil de simulation DynCell 3.3.

3.1 Modélisation mathématique

Les formes sont des ensembles du point de vue mathématique, des ensembles qui se déplacent, se déforment, se multiplient et croissent (voir figure 11).

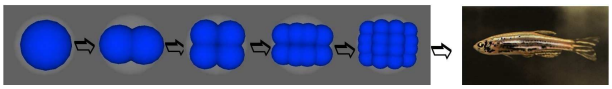

FIGURE 11 – Analyse multivoque pour la modélisation d'une cellule qui se déplace et se multiplie

En réalité, lors d'une évolution infinitésimale de la forme, chaque élément peut non seulement se déplacer en un autre point de l'espace mais éventuellement dilaté si d'autres éléments le succèdent. C'est ainsi que se justifie le développement de l'analyse multivoque, associant à tout élément d'un espace de départ un sous-ensemble de l'espace d'arrivée. Elle vient enrichir la panoplie d'outils mathématiques pour la modélisation de la morphogénèse, déjà développés en analyse fonctionnelle.En effet, les équations morphologiques, cas particulier des équations mutationnelles, ont des propriétés similaires à celles des équations différentielles (Peano theorem, Cauchy-Lipschitz, Nagumo)[16]. Elles gouvernent l'évolution d'ensembles de la même manière que les équations différentielles gouvernent l'évolution de vecteurs. Ces évolutions d'ensembles doivent respecter des contraintes de confinement ou des contraintes dites géométriques[2]. Elles doivent atteindre des cibles en temps finis et elles doivent également co-évoluer avec leurs éléments qui obéissent individuellement à d'autres dynamiques. De ce fait, il faut adopter la stratégie inverse de celle de l'analyse fonctionnelle. C'est-à-dire considérer les fonctions comme des ensembles en les caractérisant par leur graphe. Ensuite étudier dans cette perspective :

- les opérations sur les ensembles,
- les évolutions d'ensembles,
- les hyperespaces : espaces de sous-ensembles d'un espace donné,
- les correspondances ou applications multivoques,

– les applications de forme : applications associant à un point un ensemble,
– et celles qui associent à un ensemble un autre ensemble.

Dans le cas des organismes multicellulaires qui évoluent, un autre phénomène important est la co-évolution : les cellules évoluent avec l'organisme et avec l'environnement de l'organisme. Ces trois niveaux doivent être capables de réagir aux évènements de façon appropriée pour que le système soit viable : c'est le concept de co-viabilité. Les systèmes évolutionnaires gouvernent l'évolution des états et des environnements et dépendent en même temps de l'état et de l'environnement. Et dans un tel système, pour aboutir à des solutions, il faut y adapter le théorème de la viabilité. C'est-à-dire vérifier qu'il existe, au moins une évolution co-viable de l'état et de l'environnement de chaque paire état/environnement. L'ensemble des conditions pour lesquelles une telle solution existe définit le noyau de viabilité du système dans notre cas.
Nous allons donc formaliser dans un espace métrique la dynamique des cellules pour trouver les conditions (décisions, états) dans lesquelles les contraintes opérationnelles (comme celles induites par le tissu ou la consommation des ressources) sont toujours satisfaites et par conséquent dans lesquelles le système est viable et maintient son état en se renouvelant.

Durant le développement embryonnaire, le confinement est imposé par la cohésion des tissus et la présence d'une enveloppe telle que la couche épithéliale qui recouvre l'embryon. Ainsi, il y a une co-évolution de la membrane cellulaire et de la dynamique de chaque cellule. La forme du confinement ainsi constituée ne peut évoluer qu'en respectant les contraintes que nous allons étudier en utilisant l'analyse morphologique.

Soit M l'ensemble contenant les cellules, cet ensemble est inclus dans le complément de l'ensemble que constitue vitellus [8].

$K \subset R^3$ représente le tissu cellulaire, les cellules étant désignées par $x \in K \subset R^3$.

En restreignant la morphogénèse dans le plan,

$$\mathcal{D} := \{(1,0),(-1.0)(0,1),(0,-1)\}$$

autrement

$$\mathcal{D} := \{1,-1,2,-2\}$$

désigne l'ensemble des 4 directions du plan et

$$\overline{\mathcal{D}} := \mathcal{D} \cup \{(0,0)\} \cup \emptyset$$

8. Constitue les réserves énergétiques utilisées par les embryons durant le développement embryonnaire.

indiquent les 6 directions étendues.
Concernant la morphogénèse dans l'espace R^3,

$$\mathcal{D} := \{(1,0,0),(-1,0,0),(0,1,0),(0,-1,0),(0,0,1),\\ (0,0,-1)\}$$

autrement,

$$\mathcal{D} := \{1,-1,2,-2,3,-3\}$$

indiquent l'ensemble des 6 directions et $\overline{\mathcal{D}} := \mathcal{D} \cup \{(0,0,0)\} \cup \emptyset$ les 8 directions étendues.

On pose $A + \emptyset = \emptyset$ en algèbre max-plus algebra pour les opérations $\cup$ et $+$.

Et on note

$$\Xi_M(K,x) := \{u \in \mathcal{D} \text{ such that } x + u \in \{x\} \cup (M \setminus K)\}$$

et

$$R_M(K,x) := \Xi_M(K,x) \times \Xi_M(K,x).$$

Ainsi, on peut introduire la correspondance suivante :

$$\Psi(x,u,v) := \{x+u\} \cup \{x+v\}_{(u,v)\in R_M(K,x)}.$$

La dynamique morphologique Φ_M est ainsi définie par :

$$\Phi_M(K) := \bigcup_{x\in K} \bigcup_{(u,v)\in R_M(K,x)} \Psi(x,u,v) \tag{1}$$

Et la dynamique morphologique discrète par : $K_{n+1} = \Phi_M(K_n)$.

Ce qui nous permet de modéliser les différents cas de comportements cellulaires :

1. *apoptose*, obtenue en prenant $(\emptyset,\emptyset) \in R_M(K,x)$ puisque $\Psi(x,\emptyset,\emptyset) := \emptyset \cup \emptyset = \emptyset$
2. *migration* en prenant $u \in \mathcal{D}$ et $v = \emptyset$ ou $u = \emptyset$ et $v \in \mathcal{D}$ ou encore $u = v$
3. *inertie*, qui peut être définie comme une migration en prenant u et v égal à $(0,0,0)$
4. *mitose* en prenant $u := (0,0,0)$ et $v \in \Xi_M(K,x)$ (ou l'inverse)
5. *mitose et migration* en prenant $u \in \Xi_M(K,x)$ and $v \in \Xi_M(K,x)$

On peut à présent introduire une relation d'équivalence sur les directions

$$u \equiv_x v \text{ if and only if } x + u = x + v$$

On pose μ et ν des représentant. Remarquons, par construction, que pour chaque pair (μ, ν) la classe d'équivalence, pour tout $u \in \mu$ et $v \in \nu$, $\Psi(x, \mu, \nu) = \Psi(x, u, v)$ ne dépend pas du choix des directions appartenant aux classes d'équivalence. Car deux cellules ne peuvent pas occuper la même position et ne choisiront qu'une direction au plus dans chaque classe.

La correspondance de régulation est définie par le quotient :

$$\Theta_M(K, x) \ := \ R_M(K, x)/ \equiv_x \tag{2}$$

La dynamique morphologique Φ_M étant toujours définie par

$$\begin{aligned} \Phi_M(K) := & \bigcup_{x \in K} \bigcup_{(\mu,\nu)) \in \Theta_M(K,x)} \Psi(x, \mu, \nu) \\ = & \bigcup_{x \in K} \bigcup_{(u,v) \in \in R_M(K,x)} \Psi(x, u, v) \end{aligned} \tag{3}$$

Dans le cas d'une dynamique discrète, elle est définie par la séquence de contrôle (u_n, v_n) associée à K_n pour définir $K_{(n+1)}$.
Un exemple de codage des deux premières divisions cellulaires avec ce modèle est présenté ci-dessous :

$\forall x \in K_1 = \{(0, 0, 0)\}$, le premier choix de direction $U(1, x) = U(1) = [1, -1, 2, -2, 3, -3, 0]$ sera utilisé lors de la première division, comme le montre la figure 12 :

FIGURE 12 – $U(1, x) = U(1) = [1, -1, 2, -2, 3, -3, 0]$ signifie que le premier axe de division est $x - axis$ et la direction est $+1$

$\forall x \in K_2 = \{(0, 0, 0), (1, 0, 0)\}$, le deuxième choix de division

$$U(2, x) = U(2) = [2, -2, 1, -1, 3, -3, 0]$$

sera utilisé lors de la deuxième division, voir figure 13 : Ce codage va nous permettre de constituer notre métaphore de gènes.

FIGURE 13 – $U(2,x) = U(2) = [2,-2,1,-1,3,-3,0]$ signifie que le second axe de division est $y-axis$ et la direction est $+2$

3.2 Algorithmes d'exploration de génomes

L'idée des algorithmes que nous avons développés à des fins d'exploration de génomes s'est beaucoup inspiré du concept que Waddington a présenté en 1940 sous le nom de paysage épigénétique[25]. Toutes les trajectoires possibles à travers ce paysage (dont les obstacles sont analogues aux contraintes de la forme dans l'environnement) représentent les formes possibles à partir d'un même génome mais soumis à différentes conditions où des gènes particuliers seulement du génome seront activés en fonction de la contrainte (voir figure 14). D'où intervient la différenciation qui profère un caractère particulier à la forme à cet instant. Le langage de programmation utilisée est le

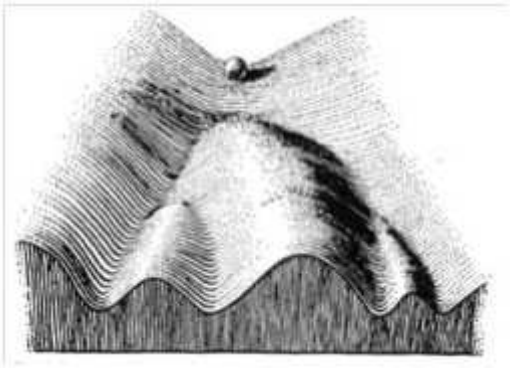

FIGURE 14 – Représentation d'un paysage épigénétique selon C.H. Waddington en 1957. Le chemin suivi par la balle correspond à l'histoire d'une partie du développement d'un organisme | *Sources : Journal of Cell Sciences, http://jcs.biologists.org/*

`C++` et le développement est entièrement basée sur la bibliothèque AReVi. Donc nous avons dû prendre en main tous les concepts de développement imposés par cette bibliothèque pour pouvoir bénéficier de ses services. Nous avons principalement deux classes qui héritent de `ArObject` :

1. Une première classe `Genome` qui représente un génome et qui nous per-

mettra de gérer tous les traitements qu'il est possible d'effectuer sur un génome. Telle que nous l'avons conçu, un génome est composé d'un ou plusieurs gènes qui seront activés ou non durant la simulation du génome en fonction de la dynamique résultante. Cette classe comporte aussi certaines méthodes permettant de gérer la lecture de gènes à partir d'un fichier ou l'écriture d'un génome sur un fichier.

2. La seconde classe `GenerateGenome` permet de construire toute l'activité de l'exploration, son initialisation, son lancement et son arrêt. Il contient entre autres des méthodes d'initialisation des paramètres de l'exploration, des appels de méthodes sur des génomes, des méthodes permettant de gérer un ensemble de génomes et des méthodes permettant d'enregistrer les résultats de l'exploration.

Pour respecter le vaste choix de paramétrage de DynCell, nous avons créé un événement-clavier permettant de lancer l'exploration. Les principes de l'exploration sont décrits dans ce qui suit

3.2.1 Construction de la base de génomes

A partir d'un fichier xml, on charge dans un vecteur tous les gènes contenus dans le fichier avec leurs attributs. A partir de ce vecteur, on crée toutes les combinaisons possibles de génomes qui vont constituer une base de génomes à explorer durant la simulation. Ensuite on le conserve dans un fichier. Le nombre de gènes que doit contenir les génomes est défini par celui que comporte la forme ciblée.

3.2.2 Algorithmes : initialisation

Lors de l'initialisation, on définit les paramètres de la forme ciblée tels que l'ordre de multiplicité de l'activation de ses gènes, l'âge de maturation de la forme et le nombre de cellules dont elle est composée. Cette forme cible a été obtenu en simulant un génome connu et les gènes activés lors de la simulation. Chaque définition de choix de paramètres tels que l'ordre de multiplicité donne un nombre de cellules bien déterminée. L'exploration peut servir à vérifier deux choses :

1. N'existe -t-il que ce génome pour arriver à cette forme caractérisée par la correspondance de chaque choix de définition de paramètres à un nombre de cellules données ?
2. Existe -t-il d'autres formes comportant le même nombre de cellules que la forme ciblée et le même nombre de gènes activés dont le génome à l'origine est différent ?

3.2.3 Algorithmes : activité

Pour répondre à la première question, on explore tous les génomes de la base. Chaque génome est simulé dans le but de vérifier s'il arrive à la forme ciblée, c'est à dire atteindre le même nombre de cellules étant dotée des mêmes choix de définitions de paramètres. Si le génome courant vérifie cette contrainte, il est répertorié dans une base qui recueille de tels génome. On envoie un message au console pour informer des gènes ayant été activés dans le génome, son indice dans la base et la définition de choix de paramètre ayant donné la correspondance. En dehors d'une correspondance, il peut y avoir deux cas :

1. que le nombre de cellules excède celui du choix de paramètre courant. Alors, il est admis que le génome n'est résolument pas correspondant. L'algorithme poursuit son exploration avec le génome suivant dans la base.
2. que le nombre de cellules n'atteigne pas celui du choix de paramètre courant. Alors, il est possible qu'il y ait correspondance en redéfinissant un autre choix de paramètre. L'algorithme recharge le même génome en modifiant ces paramètres. Si lors de la deuxième simulation, le nombre de cellules correspondant à ce nouveau choix est strictement atteint, il est répertorié. Par contre, si le nombre de cellules excède, l'algorithme poursuit son exploration avec le génome suivant dans la base. Si malgré la deuxième simulation, le nombre de cellule correspondant n'est toujours pas atteint, on redéfinit une troisième fois le choix de paramètres. L'algorithme recharge encore le même génome avec le même processus de décision.

Un nombre maximum de redéfinition de choix est admis. Au-delà de ce nombre, le génome n'est plus rechargé par l'algorithme et est considéré comme n'étant résolument pas correspondant. Et en ce moment seulement, l'algorithme poursuit son exploration de la base en passant au génome suivant.
Quand l'algorithme a fini d'explorer tous les génomes de la base, on arrête l'activité et on enregistre les génomes correspondants dans un fichier. Pour répondre à la deuxième question, on explore tous les génomes. Chaque génome sera simulé avec tous les choix de paramètres possibles. Et dans ce cas de simulation, la vérification consiste à tester s'il existe un génome dans la base qui atteint le même nombre de cellules que la forme ciblée avec un des choix de paramètres possibles. Si on retrouve un tel génome, on le répertorie. Dans le cas où le nombre de cellules est différent pour tous choix possibles de paramètres, l'algorithme poursuit la simulation avec le génome suivant dans la base.

3.3 Tests

Nous avons tenté de répondre à notre première question en appliquant l'algorithme d'exploration sur une forme particulière dont le génome est composé de trois gènes : 1-2-3. Comme correspondance entre choix de paramètres respectivement pour les gènes 1, 2 et 3 et nombres de cellules atteint, nous avons :

- multiplicité 3-6-3 → 70
- multiplicité 4-8-4 → 117
- ...
- multiplicité 10-20-10 → 651

Lorsque ce génome est simulé, tous ses gènes sont activés. La structure obtenue est composée de 3 couleurs formant le *frenchflag* introduit par L. Wolpert (1969). Donc la forme cible dans nos tests était le drapeau sénégalais (voir figure 15) en débutant par la recherche de correspondance entre *choix de multiplicité 3-6-3 et nombre de cellules 70* et limitant le nombre maximum de modification de paramètres pour un même génome à 3.

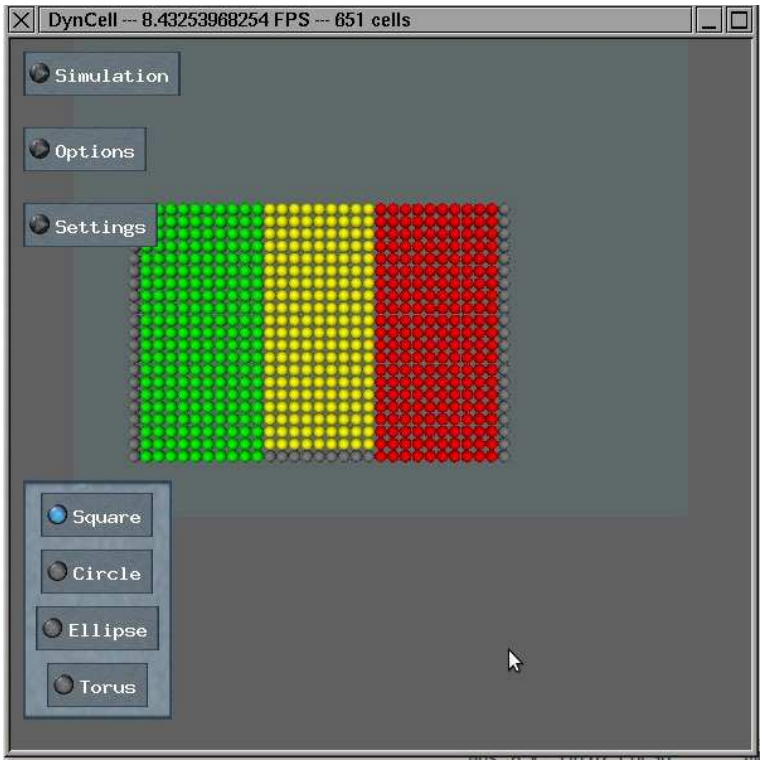

FIGURE 15 – Forme cible à atteindre par les génomes de la base explorés par l'algorithme

Nous avons paramétré l'algorithme pour que l'espace d'exploration soit moins important que tout l'espace des génomes possibles. Car, en ayant tenté de vérifier tous les génomes, nous nous sommes confronté à des problèmes de mémoire. En effet, avec 24 génomes, l'algorithme effectue le parcours et les tests en 3 minutes. Or tout l'espace compte 262.114 génomes. Un petite règle de trois nous donne en moyenne un coût en temps de 32768 minutes, environ 22 jours, 18 heures et 8 minutes. Les tests effectués sur 24 génomes exploré en occultant le génome permettant d'obtenir le drapeau sénégalais ont donné les résultats suivants :

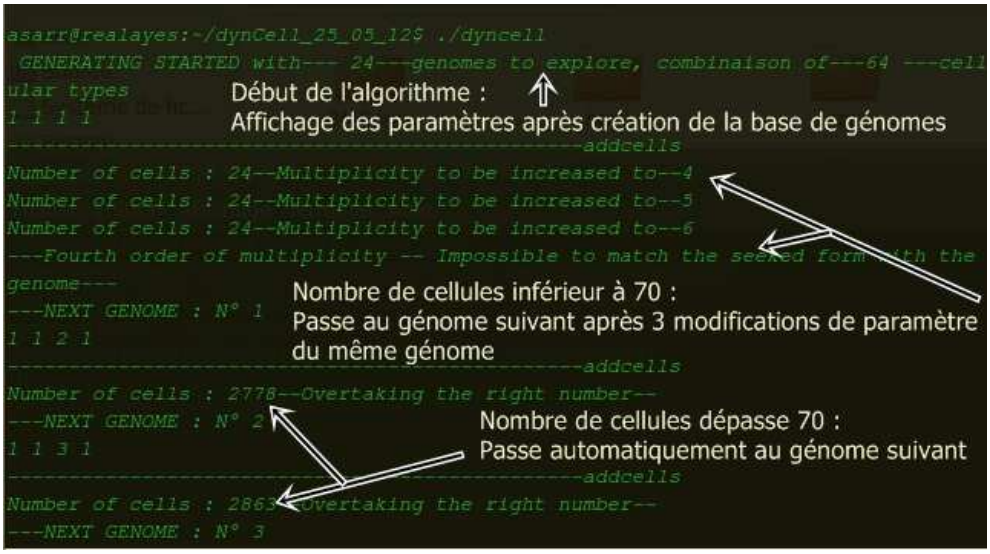

FIGURE 16 – Début de l'exploration de la base de génomes

Ces algorithmes sont testés sur un ordinateur de processeur Intel(R) core(TM) duo CPU T5750 2 GHz - 2 GHz. On pourrait envisager de les implémenter en parallèle sur un processeur multicoeurs pour pallier la montée en charge.

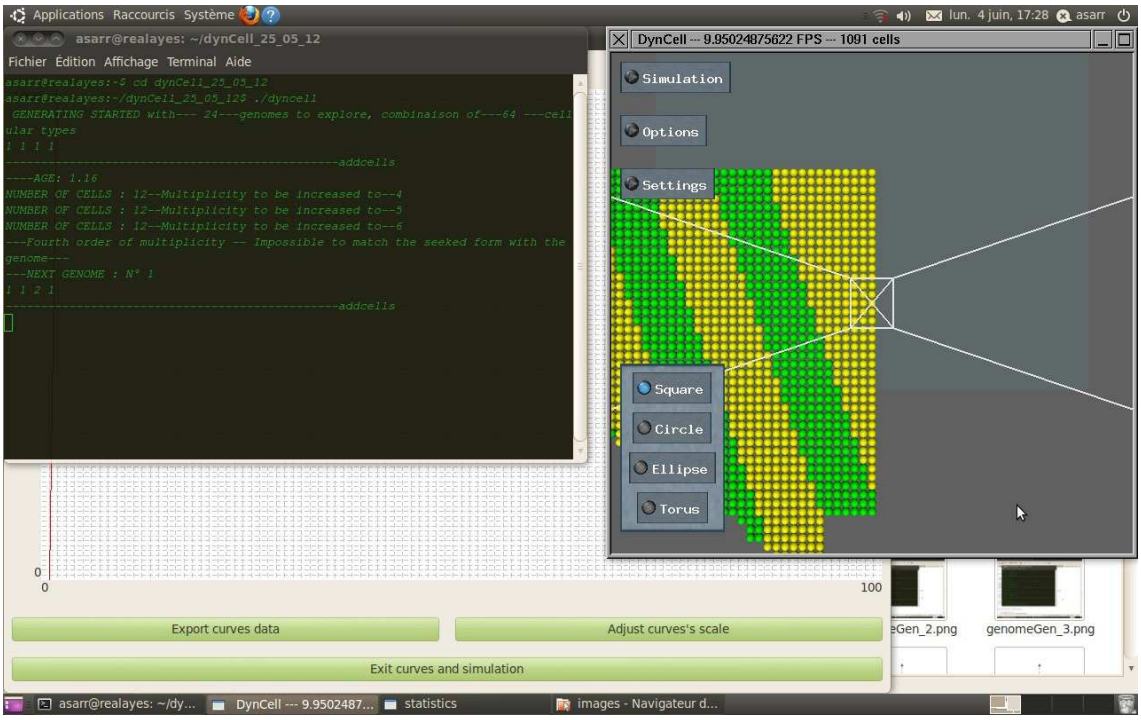

FIGURE 17 – Test du premier génome

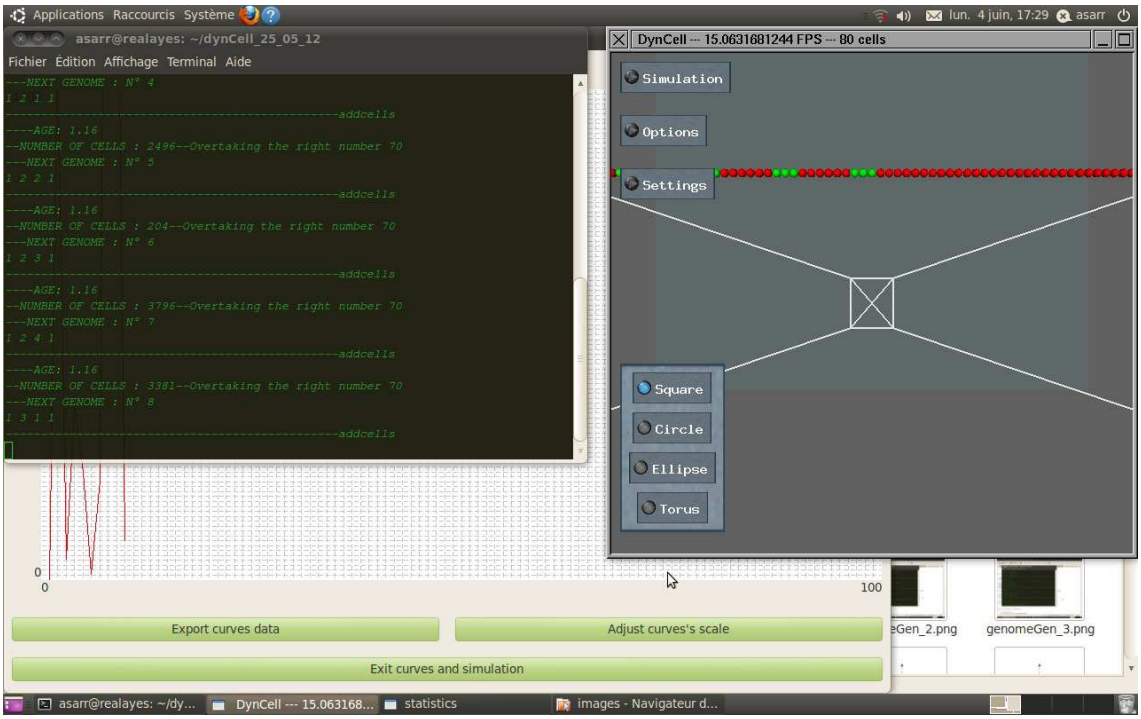

FIGURE 18 – Algorithme : Test du génome Numéro 8

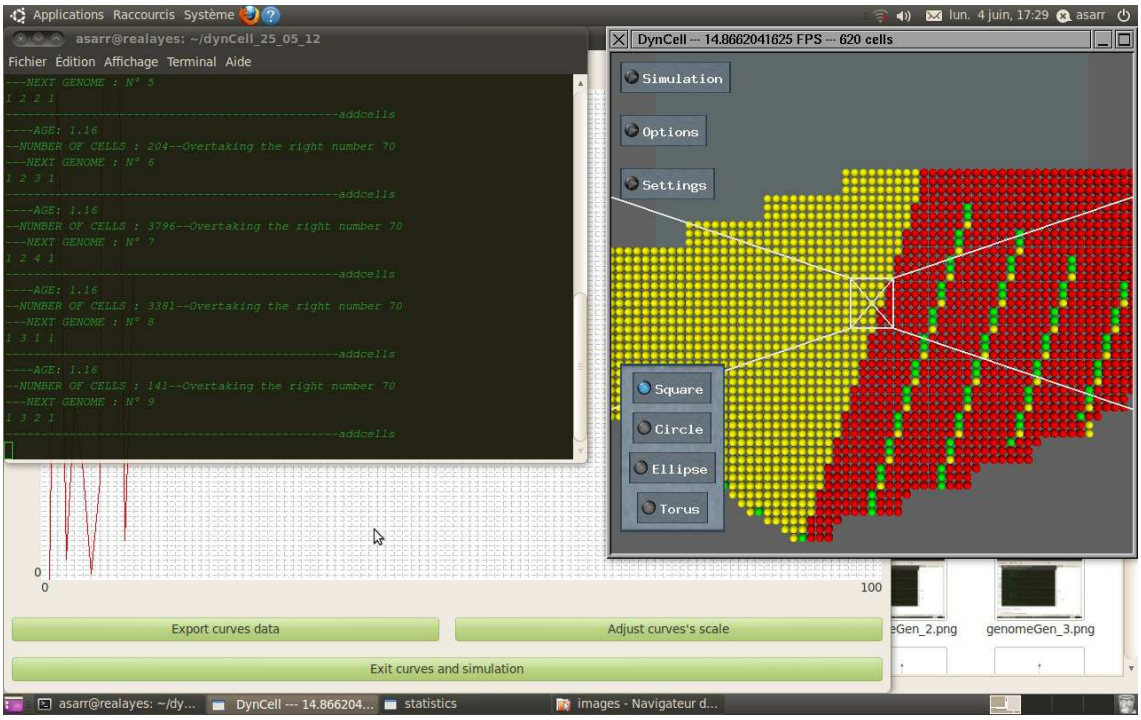

FIGURE 19 – Algorithme : Test du génome Numéro 9

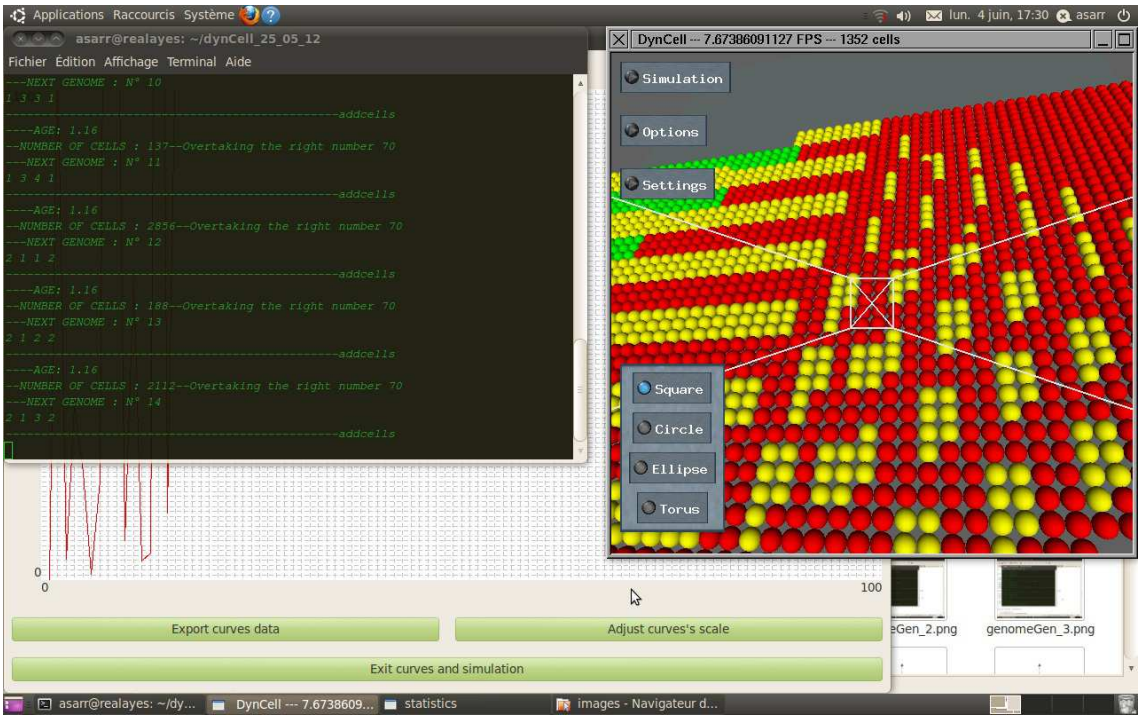

FIGURE 20 – Algorithme : Test du génome Numéro 14

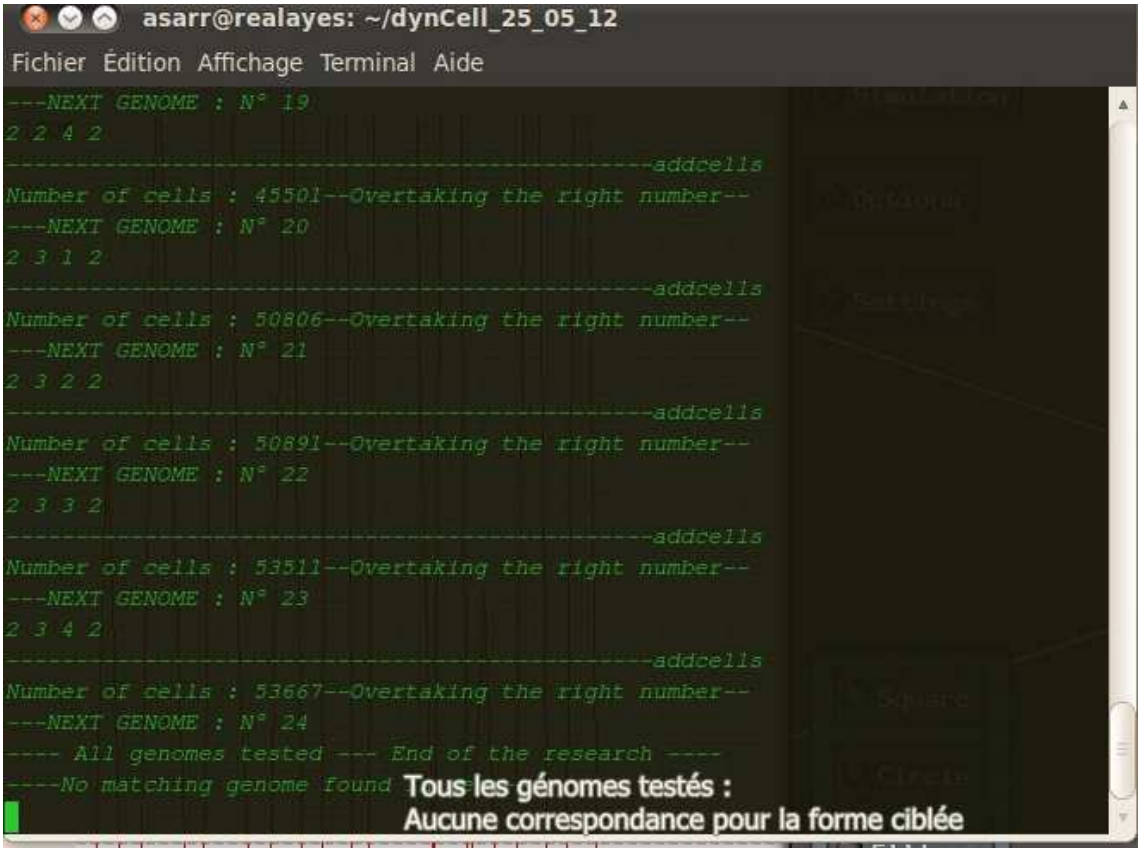

FIGURE 21 – Algorithme : Fin

4 Perspectives

4.1 Modélisation mathématique

Afin d'envisager décrire toutes les formes de dynamiques, J. P. Aubin,A. Désilles et A. Fronville proposent de généraliser le modèle mathématique actuel. En effet, dans ce modèle, lorsqu'une cellule déclenche une mitose, on crée juste à côté d'elle un autre objet cellule. Ainsi, la cellule mère ne met en oeuvre aucun autre mécanisme cellulaire. Or, dans ce modèle la cellule mère peut à chaque pas de temps :

- rester à un état stationnaire comme dans le modèle actuel*
- migrer vers un autre point
- ou encore mourir

Cette formalisation permet d'atteindre toutes les formes possibles et d'utiliser tout l'arsenal des méthodes viabilistes (noyau de viabilité, bassin de capture, trajectoires lourdes ...) lorsqu'il sera implémenté dans le cadre des équations mutationnelles. Concevoir un modèle de la sorte pose de nouveaux défis en mathématique pour le codage et surtout en informatique. Car en toutes les dynamiques possibles pour chaque cellule à, chaque pas de temps offre beaucoup trop de possibilités dont il faudra implémenter des algorithmes et réussir à les faire tourner. Cependant, le défis nous semble enhardissant à partir du moment où cette approche nous apprendrait davantage sur les mécanismes mis en œuvre par les organismes multicellulaires pour survivre.

4.2 Cas d'utilisation de l'exploration

Nous envisageons améliorer l'outil et augmenter ses possibilités d'utilisation telles que :

- ajouter des génomes à la base déjà constituée au cours de la simulation
- mettre au début de chaque génome de la base des gènes identiques pour contraindre les génomes à avoir tous la même base génétique
- quand il y a correspondance du nombre de cellules au choix de paramètres, pour le même génome, passer au choix de paramètre suivant pour vérifier la correspondance demeure. Cela teste la robustesse de la correspondance.
- tester l'algorithme avec comme cible le lignage cellulaire (voir figure 22) du Caenorhabditis elegans (C. elegans) pour vérifier si on peut trouver un génome avec le même nombre de gènes activés que de types cellulaires du C. elegans et donnant un nombre identique de cellules que lui.

Le lignage cellulaire du C. elegans est choisi car c'est un eucaryote, ce qui signifie qu'il partage les structures cellulaires, moléculaires et des voies de

contrôle avec des organismes supérieurs. Ainsi, l'information biologique du C. elegans(l'embryogenèse, la morphogénèse, la croissance ,etc.) pourrait être directement applicable à des organismes plus complexes, comme celui de l'homme. En plus, il possède un nombre fixe de cellules. L'adulte hermaphrodite est composé de 959 noyaux somatiques et l'adulte mâle de 1031 tandis que le jeune est constitué de 1090 noyaux somatiques. Chaque cellule a sa propre fonction et position qui déterminées par les communications cellulaires. Une mutation ou un défaut se produisant chez une cellule peut avoir une grande incidence sur tout l'organisme.

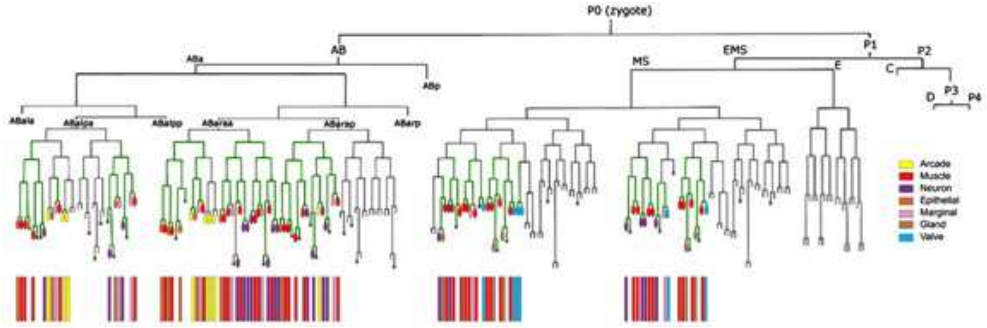

FIGURE 22 – Reconstruction du lignage cellulaire du C-elegans | Sources :

Conclusion

Les données issues de la biologie deviennent de plus en plus disponibles et ouvrent de larges champs d'études des systèmes complexes que constituent les cellules. Nous avons vu que les travaux dans le domaine de la modélisation mathématique et de la simulation informatique sont nombreux et ne manquent pas de susciter quelques réflexions de la part des biologistes. En effet, ils interpellent sur l'appropriation de la réalité si complexes de la biologie cellulaire en biologie virtuelle. En effet, parmi quelques modèles que nous avons vu, nous avons soulevé que les processus d'expression génétique jouaient un rôle primordial. Alors que d'autres facteurs comme l'acquisition de propriétés différentielles d'adhésion, les mécanismes cellulaires, l'influence de l'environnement jouent un rôle tout aussi important dans l'apparition de formes. En ce sens, comprendre les paramètres épigénétiques qui mènent à l'émergence de certaines formes dans la nature est primordial. Cela permettrait par exemple d'ouvrir des perspectives dans l'identification des perturbations qui mènent au cancer et d'agir sur elles en vue de les modifier. C'est ainsi que durant ce stage, nous nous sommes intéressés à la modélisation mathématique des mécanismes cellulaires avec une approche permettant de mieux prendre en compte la nature des formes du point de vue mathématique. Et à développer des algorithmes permettant de mesurer l'impact de l'environnement sur l'activation de gènes en simulant une base de génomes créée à partir d'un ensemble de gènes et vérifier sous quelles conditions des certains gènes particuliers pouvaient être activées.
Leur utilisation peut bien être étendue et leur développement amélioré. Nous avons particulièrement constaté une montée en charge lors des tests réalisés lorsqu'on disposait d'une base de génomes assez grande. En plus, avec un nouveau modèle mathématique généralisé qui permettra d'avoir plus de mécanismes cellulaires, il faudra envisager une parallélisation des algorithmes. En ce sens, un contrat doctoral a été établi en vue de poursuivre ces mêmes questions de recherche.
Ce stage a m'a permis de me confronter aux métiers de chercheur, de l'étude bibliographique à la rédaction d'articles scientifiques, en passant par la recherche et l'implémentation de résultats scientifiques.

Références

[1] J.-P. Aubin. *Viability theory*. Birkhauser, 1991.

[2] J.-P. Aubin and A. Lesne. *Analyse morphologique et mutationnelle : des outils pour la morphogenèse*, chapter 17, pages 309–325. 2006.

[3] W. S. Bainbridge and M. C. Roco. Managing nano-bio-info-cogno innovations : Converging technologies in society. *Springer Science and Business Media*, 2006.

[4] P. Ballet. *Intérêt mutuel des systèmes multi-agents et de l'immunologie*. PhD thesis, Université de Bretagne Occidentale, Janvier 2000.

[5] S. Bonneaud, P. Redou, G. Desmeulles, and P. Chevaillier. Biais computationnels dans les modèles de peuplement d'agents. In *JFSMA 09*, 2009.

[6] M. Campana, B. Rizzi, C. Melani, P. Bourgine, N. Peyriéras, and A. Sarti. A framework for 4-d biomedical image processing, visualization and analysis. In *GRAPP'08*, pages 403–408, 2008.

[7] Gireg Desmeulles. *Réification des interactions pour l'expérience in virtuo de systèmes biologiques multi-modèles*. PhD thesis, Université de Bretagne Occidentale, décembre 2006.

[8] P. Dittrich, J. Ziegler, and W. Banzhaf. Artificial chemistries-a review. *Artificial Life*, 7(3), 2001.

[9] René Doursat. *Organically Grown Architectures : Creating Decentralized, Autonomous Systems by Embryomorphic Engineering*, chapter 8, pages 167–200. Springer-Verlag, 2008. Organic computing.

[10] René Doursat. The self-made puzzle. *Inter Journal : Complex Systems*, 2292, 2008.

[11] J. Echasserieau. Outils de modèlisation pour la morphogénèse, 2009.

[12] H. De Garis. Artificial embryology and cellular differentiation. *Evolutionary Design by Computers*, 1999.

[13] I. Harvey. The microbial genetic algorithm. *Advances in artificial life*, pages 126–133, 2009.

[14] P. Hogeweg and S. Marée. How amoeboids self-organize into a fruiting body : multicellular coordination in dictyostelium discoideumm. *Proc. Natl. Acad. Sci. U.S.A.*, 98(7) :3879–3883, 2001.

[15] B.G. Lawson and S.Park. Asynchronous time evolution in an artificial society mode. *Journal of Artificial Society and Social Simulation*, 3(1), 2000.

[16] T. Lorenz. *Mutational Analysis A Joint Framework for Cauchy Problems In and Beyond Vector Spaces*. Springer, 2010.

[17] N. Marion, C. Septseault, A. Boudinot, and R. Querrec. Gaspar : Aviation management on an aircraft carrier using virtual reality. volume 1434, pages 15–22, 2007.

[18] C. Melani, N. Peyriéras, K. Mikula, C. Zanella, M. Campana, B. Rizzi, F. Veronesi, A. Sarti, B. Lombardot, and P. Bourgine. Cells tracking in the live zebrafish embryo. In *Conf. Proc. IEEE Eng. Med. Biol. Soc.*, volume 1, pages 1631–1634, 2007.

[19] J.D. Murray. *Mathematical Biology.* 2002.

[20] Nicolas Olivier, Miguel A. Luengo-Oroz, Louise Duloquin, Emmanuel Faure, Thierry Savy, Israël Veilleux, Xavier Solinas, Delphine Débarre, Paul Bourgine, Andrés Santos, Nadine Peyriéras, and Emmanuel Beaurepaire. Cell lineage reconstruction of early zebrafish embryos using label-free nonlinear microscopy. *Science*, 329(5994) :967–971, 2010.

[21] A. Paldi. *Génétiquement indéterminé, le vivant auto-organisé.*, chapter 2. Expression aléatoire des gènes au cours de la différenciation cellulaire., pages 59–76. 2007.

[22] Arturo Chavoya Pena. *Un modèle de développement artificiel pour la génération de structures cellulaires.* PhD thesis, Université de Toulouse, décembre 2007.

[23] N. Peyriéras. *Morphogenèse animale*, pages 179–201. 2006.

[24] S. Stoma, J. Chopard, C. Godin, and J. Traas. Using mechanics in the modelling of meristem morphogenesis. *5th International Workshop on functional-structural plant models, Napier, New-Zeland*, 52 :1–4, 2007.

[25] C.H. Waddington. *Organisers and Genes.* 1940.