Une méthode pour naviguer dans l'espace des simulations

François Gaillard, Yoann Kubera, Philippe Mathieu, Sébastien Picault

Une méthode pour naviguer dans l'espace des simulations

François Gaillard[1] — Yoann Kubera[2] — Philippe Mathieu[3] — Sébastien Picault[4]

Equipe Systèmes Multi-Agents et Comportements
Laboratoire d'Informatique Fondamentale de Lille - UMR CNRS USTL 8022
Université des Sciences et Technologies de Lille - 59655 Villeneuve d'Ascq cédex
[1] francois.gaillard@isen.fr [2] yoann.kubera@lifl.fr [3] philippe.mathieu@lifl.fr
[4] sebastien.picault@lifl.fr

RÉSUMÉ. *La conception habituelle de la simulation d'un phénomène passe par sa modélisation et la réalisation d'une implémentation : son étude permet de déterminer si le modèle est plausible pour rendre compte du phénomène. Nous proposons de renverser ce processus de conception en naviguant dans l'espace des simulations possibles jusqu'à identifier des phénomènes remarquables. Cet article traite de la construction automatisée de modèles et de la manière de naviguer dans l'espace des simulations possibles à partir d'une ontologie de domaine. Afin que ceci soit possible, l'ontologie doit être constituée d'interactions génériques affectées à des familles d'agents. La méthodologie IODA et notre outil LEIA illustrent cette approche en s'appuyant sur des outils de transformation et de simplification de modèles. Les phénomènes pertinents sont identifiés grâce à une métrique spécifiée en entrée, tout en impliquant l'utilisateur dans ce processus.*

ABSTRACT. *The usual way to design a simulation of a phenomenon is to create its model and then implement it: its study allows to know if the model is adequate and can explain the phenomenon. We propose to reverse this process by exploring the simulation space in order to identify remarkable phenomena until understanding the underlying mechanisms. This paper deals with automated construction of models and the browsing of the simulation space implementations based on a domain ontology. In order to do that, we need to have generic interactions that we can assign to families of agents. The IODA methodology and our tool LEIA illustrate this approach by using tools for processing and simplifying models. The relevant phenomena are identified by using a specified input metric and by involving the user in this process.*

MOTS-CLÉS : *interaction, simulation, rétro ingénierie, observation de modèle*

KEYWORDS: *interaction, simulation, reverse engineering, model observation*

RSTI - RIA - 24/2010. Génie logiciel multi-agent , pages 575 à 600

1. Introduction

L'étude d'un phénomène au travers d'une simulation se fait en trois étapes pour le concepteur : en premier lieu, il réalise une modélisation du problème, puis en réalise une implémentation. Enfin, il peut étudier la simulation afin de déterminer si le modèle est correctement construit avec pour objectif d'expliquer le phénomène. Tout particulièrement, il doit pouvoir retrouver un certain nombre de faits stylisés liés au domaine simulé. Si le modèle ne répond pas aux attentes du concepteur, il doit alors modifier le modèle puis réaliser une nouvelle implémentation à laquelle succède une nouvelle étude. Cette construction itérative peut devenir rapidement laborieuse. En outre, elle peut être biaisée par des idées préconçues sur les changements à introduire dans le modèle (Kubera *et al.*, 2009). C'est pourquoi nous décrivons une approche nouvelle dans laquelle nous nous proposons de naviguer dans l'espace des simulations possibles.

Nous définissons un explorateur de l'espace des simulations comme l'association d'outils avec lesquels nous pouvons créer, implémenter et étudier l'ensemble des simulations possibles exprimables à partir d'une ontologie de domaine. Cette approche se différencie des méthodes classiques de conception d'une simulation car elle ne se limite pas à la variation des paramètres identifiables d'un modèle. Ici, nous proposons de faire varier les relations qui régissent le comportement des agents entre eux. L'intérêt de cette approche se place à plusieurs niveaux :

– le concepteur dispose d'un cadre efficace pour modifier un modèle et l'implémenter immédiatement ;

– le concepteur peut envisager toutes les variations de son modèle, en ayant même la possibilité d'explorer au-delà des modifications de paramètres usuelles ;

– le concepteur dispose d'outils de transformation de modèles génériques mais peut aussi créer les siens ;

– le concepteur peut quantifier l'impact des transformations de ses modèles grâce à des outils de mesure, soit génériques car portant sur la méthodologie de conception du modèle et sur le fonctionnement de la simulation, soit propres au domaine simulé car portant sur des faits spécifiques au phénomène étudié.

Le navigateur de l'espace des simulations permet donc d'effectuer un travail de *reverse engineering* désigné en tant que « design recovery » dans (Chikofsky *et al.*, 1990). L'utilisateur est capable de pleinement comprendre les simulations et les modèles sous-jacents puis d'étudier les relations entre interactions, comme nous l'illustrons dans la figure 1. De plus, par les variations des modèles, il agit comme un « chatouilleur d'idées » tel que l'envisage (Hofstadter, 1988) dans « Ma Thémagie » : « Des variations sur un thème considérées comme l'essence de la créativité ». De surcroît, un tel explorateur de l'espace des simulations doit idéalement pouvoir fonctionner selon deux modes. Le premier est un mode interactif dans lequel la découverte de modèles se fait avec la participation du concepteur qui peut évaluer les phénomènes qui apparaissent dans chaque simulation.

Le second est un mode automatique dans lequel l'exploration des modèles est régie par des outils de mesures correctement choisis.

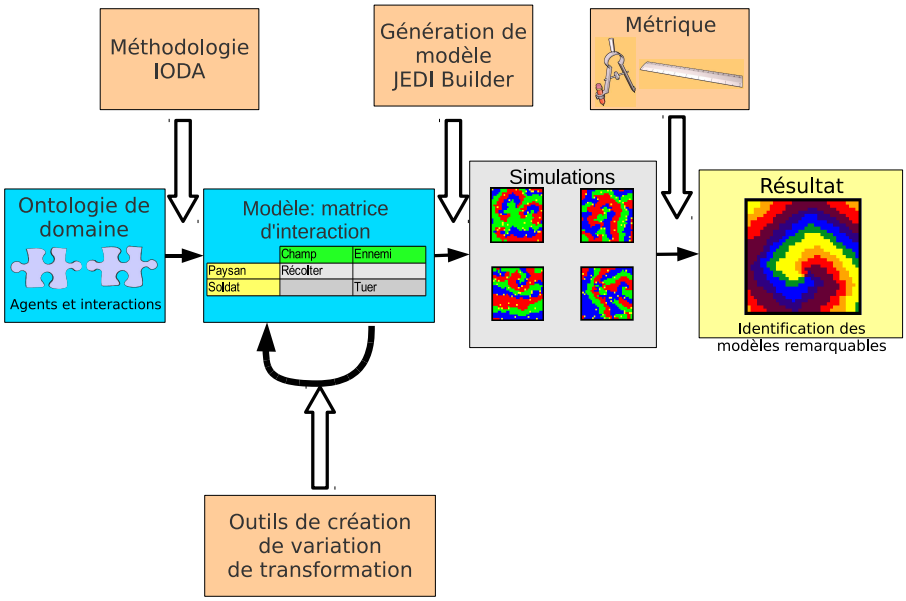

Figure 1. *Ce schéma illustre le principe de fonctionnement d'un explorateur de l'espace des simulations : nous spécifions en entrée une ontologie à base d'agents et de comportements. En se basant sur un formalisme adapté, grâce à des outils de génération de modèles et enfin des outils de transformation des modèles, on peut construire les simulations en associant les comportements aux agents sans modifier le code des agents. Nous envisageons alors d'implémenter automatiquement plusieurs simulations, les voir évoluer en parallèle et repérer des phénomènes remarquables via des outils de mesure adaptés. Nous disposons de tels outils dans l'équipe SMAC*

Une telle approche nécessite un cadre formel dans lequel la séparation des agents et de leur comportement permet de faire varier les comportements indépendamment de l'implémentation des simulations. Cette approche requiert un cadre précis que ne permet pas l'approche de conception classique des simulations. Cette dernière se retrouve dans un grand nombre de plateformes :

– orientées génie logiciel telles que Swarm (Minar *et al.*, 1996), Madkit (Gutknecht *et al.*, 2000) ou Magique (Bensaid *et al.*, 1998), qui laissent une grande liberté au concepteur pour créer ses agents, les comportements des agents et l'environnement ;

– orientées ergonomie comme Netlogo (Wilensky, 1999), conçu pour être employé par des non-informaticiens en reposant sur une forme très simple de programmation.

Cependant, l'ouverture et la généricité se font au détriment d'un cadre précis de la conception des comportements : le risque est alors de mélanger dans l'implémentation des agents le code métier et les connaissances propres au modèle. C'est pourquoi notre approche apparaît totalement novatrice.

Comme illustré par la figure 1, pour fonctionner, une telle approche nécessite :

– une méthodologie de conception de modèles définissant un cadre formel dans lequel on peut séparer agents et comportements, afin de faire varier les modèles simplement en modifiant les relations entre agents et les comportements qui leur sont affectés.

– des outils de variation et de transformation des modèles s'appuyant sur ce cadre formel, outils grâce auxquels on peut garantir l'exploration de tout l'espace des modèles ;

– un moteur générique de simulation qui puisse supporter tous les modèles envisageables ;

– des outils d'évaluations génériques ou spécifiques au domaine, afin d'identifier les phénomènes remarquables et d'en obtenir un retour quantifié.

Au sein de l'équipe SMAC du LIFL, nous avons développé de tels moyens. Nous disposons en premier lieu d'une méthodologie de conception de simulations, dénommée IODA[1] (Mathieu *et al.*, 2007), qui repose sur une séparation claire des agents, de leur comportement et du processus de sélection des actions. Dans cette méthodologie, les interactions sont réifiées indépendamment des agents qui les utilisent. De ce fait, nous pouvons établir des bibliothèques d'interactions pour un domaine particulier et adaptables à différentes familles d'agents, augmentant la généricité du travail de modélisation. Une implémentation réalisée au moyen du moteur JEDI (Kubera *et al.*, 2008a) offre un support générique à l'utilisation de cette méthodologie.

Cette généricité s'illustre parfaitement dans le cadre du générateur JEDI Builder[2] : à partir d'un modèle conçu selon IODA, nous pouvons générer le code d'une simulation exécutable avec le moteur JEDI. Grâce à la méthodologie IODA, nous avons donc un formalisme :

– qui nous permet de générer automatiquement des modèles ;

– avec lequel on peut modifier rapidement une simulation en modifiant les relations entre interactions et agents ;

– qui nous permet de spécifier des outils de mesure génériques pour caractériser les résultats en formalisant le fonctionnement du moteur de simulation.

Plus qu'un cadre formel pour l'exploration de l'espace des simulations, nous avons réalisé une implémentation de ces concepts via une application nommée LEIA, pour « LEIA lets you Explore Interactions for your Agents », qui renverse donc totalement

1. *Interaction Oriented Design of Agent simulations*
2. `www2.lifl.fr/SMAC/projects/ioda/demonstrations/`

la vue habituelle de conception de simulations multi-agents : nous générons une série de modèles aléatoires et leur implémentation à partir d'une ontologie spécifiée à l'avance, constituée d'interactions génériques que nous pouvons affecter à des familles d'agents. Ensuite, par raffinements successifs et l'utilisation des outils définis dans cet article, l'utilisateur va pouvoir créer un modèle et étudier ses caractéristiques sous-jacentes. LEIA peut évoluer tout seul en choisissant à chaque période de temps la meilleure simulation au sens de la métrique choisie. Il peut aussi laisser ce choix à l'utilisateur, ce qui semble préférable puisque la majorité des phénomènes est indécidable.

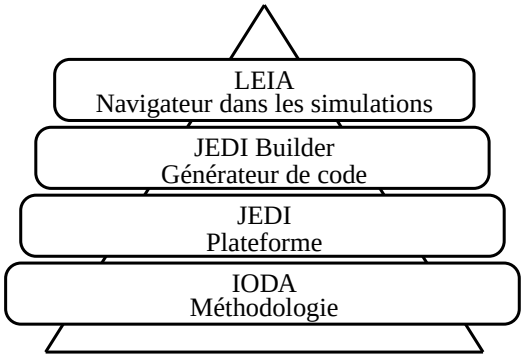

Figure 2. *Hiérarchie dans la construction du navigateur dans l'espace des simulations : IODA fournit le cadre formel dans lequel sont séparés agents et comportements ; JEDI offre un support générique pour l'exécution des simulations ; JEDI Builder permet de créer des modèles IODA exécutables dans JEDI. Enfin, LEIA permet la transformation et l'évaluation des modèles issus de l'espace des simulations*

Comme l'illustre la figure 2, nous fournissons une application, LEIA, qui s'appuie sur tous les outils de l'équipe SMAC. Ainsi, la section 2 présente la méthodologie IODA nécessaire à la conception d'un navigateur dans l'espace des navigations. Dans la section 3, nous définissons les concepts nécessaires à un explorateur de l'espace des simulations. Nous décrivons dans la section 4 des outils de mesure génériques qualifiant la complexité de systèmes conçus suivant la méthodologie IODA . Nous développons ensuite dans la section 5 la conception du navigateur LEIA. Finalement, dans la section 6, nous proposons des perspectives de travail.

2. La méthodologie IODA

La méthodologie de conception de simulation IODA (Mathieu *et al.*, 2007) est une méthodologie centrée sur les interactions : elles sont réifiées indépendamment des agents. Les agents de $\mathbb{A}$, ensemble des agents présents dans l'environnement, disposent de primitives de base :

– primitives de perception relatives à l'état global de la simulation (noté $\mathbb{E}$), à l'environnement, à son état interne et à la communication avec les autres agents ;

– primitives d'action qui permettent de modifier l'état de la simulation : son propre état, l'état de l'environnement ou l'état d'autres agents.

Ces primitives permettent de définir le rôle que peut jouer l'agent dans des interactions spécifiques.

Dans la méthodologie IODA, les agents se réduisent à des spécifications simples et sont représentés de manière homogène, ce qui permet d'intégrer n'importe quel agent au modèle de simulation centré interaction : soit $\mathbb{F}$ l'ensemble des familles d'agents d'une simulation donnée, $\forall a \in \mathcal{A}$ et $\forall \mathcal{F} \in \mathbb{F}$, un agent a est une entité autonome et instanciée à partir d'une famille d'agents $\mathcal{F}$, notée $a \prec \mathbb{F}$.

Une famille d'agents $\mathcal{F}$ parmi l'ensemble des familles $\mathbb{F}$ est l'abstraction d'un ensemble d'agents partageant tout ou partie de leurs primitives de perception ou d'action. Un agent a dispose également d'un halo de perception $\mathcal{H}_{\mathcal{F}}(a) \subseteq \mathbb{E}$: c'est-à-dire le sous-ensemble de l'état global de la simulation que l'agent a discerne au travers de ses primitives. Le voisinage $\mathcal{N}(a)$ d'un agent a est l'ensemble des agents présents dans son halo de perception : $\mathcal{N}(a) = \mathbb{A} \backslash \{a\} \bigcap \mathcal{H}_{\mathcal{F}}(a)$.

Une interaction est une séquence de primitives d'action s'appliquant à plusieurs agents, pouvant jouer le rôle de source ou de cible (respectivement notés $\mathcal{S}$ ou $\mathcal{T}$), et est soumise à des conditions d'activation. Les interactions apparaissent totalement dissociées des agents qui les utilisent, augmentant leur réutilisabilité à travers les simulations et applicables à des familles d'agents différentes.

On définit la cardinalité d'une interaction $\mathcal{I}$ comme le couple composé du nombre d'agents sources $\mathrm{card}_{\mathcal{S}}(\mathcal{I})$ et du nombre d'agents cibles $\mathrm{card}_{\mathcal{T}}(\mathcal{I})$ nécessaires à la réalisation de l'interaction.

Les assignations sont l'ensemble des interactions qu'un agent source peut faire avec des agents cibles. On appelle assignation $a_{\mathcal{S}/\mathbb{T}}$ d'un ensemble d'interactions $(\mathcal{I}_k)_{k\in[1,n]}$ entre une famille d'agents sources $\mathcal{S} \in \mathbb{F}$ et un q-uplet de familles d'agents cibles $(\mathcal{T}_j)_{j\in[1,q]} \subseteq \mathbb{F}^q$, l'ensemble des interactions appartenant aux agents $\mathcal{S}$ qu'ils peuvent effectuer en tant que sources avec les agents de $\mathbb{T}$ en tant que cibles. On définit un ensemble de tuples $(\mathcal{I}_k, p_k, d_k)$, $k \in [1, n]$, appelés éléments d'assignation, avec :

– $\mathcal{I}_k$: l'interaction pouvant être effectuée par toute source $a \prec \mathcal{S}$ et subie par les cibles $\{t_j, j \in [1, q] / t_j \prec \mathcal{T}_j \bigwedge (\forall l \in [1, q], t_j = t_l \Rightarrow j = l)$;

– $\mathrm{card}_{\mathbb{T}}(\mathcal{I}_k) = q$: le nombre de cibles dans $\mathbb{T}$;

– p_k : la priorité donnée à l'interaction $\mathcal{I}_k$;

– d_k : la garde de distance en deçà de laquelle l'interaction est réalisable.

On appelle **matrice d'interaction** la matrice $\mathcal{M} = (a_{\mathcal{S}/\mathbb{T}})_{\mathcal{S}\in\mathbb{F},\mathbb{T}\subseteq\mathbb{F}^{\mathbb{N}}}$ de toutes les assignations entre sources $\mathcal{S}$ et cibles $\mathbb{T}$ possibles (voir pour exemple le tableau 1).

Source \ Cible	Environnement	Loup	Mouton	Herbe
Loup	Mourir 2 / Chasser 0	Reproduire 1	Dévorer 1	-
Mouton	Errer 0	-	Reproduire 1	Brouter 1
Herbe	Pousser 0	-	-	-

Tableau 1. *Matrice d'interaction d'un modèle de type proie/prédateur. Dans cette matrice d'interaction, l'ensemble des assignations* $a_{\mathrm{Loup/Mouton}}$ *comporte l'interaction Dévorer. Une instance de Loup peut donc effectuer l'interaction Dévorer sur une instance de Mouton si les conditions d'activation de Dévorer sont vérifiées. La priorité* $p = 1$ *attribuée à Dévorer signifie qu'elle sera effectuée de préférence à Chasser par exemple (avec l'Environnement) dont la priorité vaut* 0*, si les deux sont réalisables*

La construction d'un modèle dans la méthodologie IODA se fait en 6 étapes :

1) identification des familles d'agents et des interactions, comme constituants d'une matrice d'interaction ;

2) écriture des conditions d'activation et séquences d'actions qui sont primitives de chaque interaction ;

3) identification des primitives d'action et de perception des agents ;

4) spécification de la priorité et de la garde de distance de chaque interaction ;

5) détermination de la dynamique, c'est-à-dire l'évolution de la matrice d'interaction construite aux étapes précédentes ;

6) détermination des spécificités requises par le modèle.

Dans le cadre de l'élaboration de modèles dans le navigateur LEIA, les familles d'agents et d'interactions sont spécifiées à l'avance. Les conditions d'actions et de perception des interactions sont également fixées. La conception du modèle peut alors se limiter à choisir les interactions et les familles d'agents sans avoir besoin de générer du code pour les employer. Nous pouvons ainsi modifier à la volée le modèle d'une simulation sans avoir à arrêter l'application.

3. Vers un explorateur de l'espace des simulations

Depuis des années, le concepteur amené à travailler dans le domaine de la simulation est confronté à un problème récurrent : la conception habituelle d'une simulation d'un phénomène passe par sa modélisation et la réalisation d'une implémentation. Son étude permet de déterminer si le modèle est correctement

construit et peut expliquer le phénomène. Si ce n'est pas le cas, il doit modifier son modèle, le plus souvent par modifications progressives de certaines propriétés puis il doit évaluer à nouveau son implémentation. Ce processus apparaît fastidieux : Comment guider le concepteur afin qu'il puisse étudier efficacement toutes les simulations possibles ? Comment aider le concepteur à trouver les simulations qu'il jugera intéressantes vis-à-vis du domaine simulé ? Notre objectif dans cet article est de décrire pour la première fois une méthodologie générique permettant au concepteur de faire évoluer un modèle jusqu'à obtenir une simulation affichant des phénomènes remarquables en s'appuyant sur des outils de mesure adaptés au domaine simulé. Pour cela, nous sommes confrontés aux deux problèmes suivants :

– quels outils pour transformer un modèle ?

– quels outils pour évaluer une simulation ?

3.1. *Les outils de transformation de modèles*

De tels outils de transformation de modèles nécessitent en premier lieu une méthodologie permettant de modifier le comportement des agents indépendamment de l'implémentation de la simulation. En effet, il ne s'agit pas ici de générer à la volée le code de chaque simulation que l'on souhaite tester. Une telle séparation apparaît possible grâce à la méthodologie IODA qui repose sur une séparation claire des agents, de leur comportement et du processus de sélection des actions. Ainsi, nous pouvons définir une ontologie de domaine à base d'agents et d'interactions qui vont définir l'espace des modèles possibles. L'assignation paramétrée des interactions aux agents va définir leur comportement et de manière plus générale l'évolution de la simulation. Précisément, définir un modèle décrit selon la méthodologie IODA revient en particulier à faire varier la **matrice d'interaction** $\mathcal{M}$. Ainsi, à partir d'une ontologie, nous pouvons agir sur :

– les interactions assignées $\mathcal{I}_k$;

– les sources $\mathcal{S}$ et cibles $\mathbb{T}$ de chaque interaction $\mathcal{I}_k$;

– les priorités p_k des interactions $\mathcal{I}_k$;

– les gardes de distance d_k des interactions $\mathcal{I}_k$.

Nous pouvons également agir sur la situation initiale de la simulation :

– le nombre d'agents Pop dans la simulation ;

– les paramètres d'initialisation des agents ;

– la répartition initiale des agents ;

– la forme de l'environnement.

Nos outils de transformation doivent couvrir tout l'espace des modèles possibles à partir d'une ontologie de domaine spécifiée. Le nombre de modèles que l'on peut créer est parfaitement quantifiable :

– Soit $n_I = \mathrm{card}(\mathbb{I}_{res})$ le nombre d'interactions dans l'ontologie ;

– Soit $n_F = \mathcal{F}$ le nombre de famille d'agents dans l'ontologie ;

– Soit la priorité $p \in [0, p_{\max}] \subset \mathbb{N}$;

– Soit la garde de distance $q \in [0, q_{\max}]$ et $q \in \mathbb{N}$.

Placer un élément d'assignation dans la matrice d'interaction $\mathcal{M}$ revient à former le tuple $\mathrm{Assi} = (\mathrm{Interaction}, \mathrm{Source}, \mathrm{Cible}, \mathrm{Priorité}, \mathrm{Garde\ de\ distance})$. Rappelons que si le nombre de sources correspond au nombre d'agents n_F, le nombre de cibles est le nombre d'agents auquel nous ajoutons l'environnement, soit $n_F + 1$. Nous pouvons donc générer $n_{\mathrm{Assi}} = n_I \times n_F \times (n_F + 1) \times (p_{\max} + 1) \times (q_{\max})$ assignations. Remplir la matrice $\mathcal{M}$ consiste donc à choisir n_{assi} parmi les n_{Assi}, sans répétition dans le choix des assignations et sans limites sur le nombre d'apparitions d'une même interaction. Au final, nous pouvons donc créer $C_{n_{\mathrm{Assi}}}^{n_{\mathrm{assi}}}$ modèles et cela sans compter toutes les variations à l'initialisation.

L'espace des simulations possibles est inclus voire égal au mieux à l'espace des modèles : en effet, nous considérons que, parmi tous les modèles ainsi obtenus, certains modèles donnent des simulations équivalentes par simplification des gardes de distance (voir la section 5 consacrée à LEIA) ou par le comportement résultant du choix des interactions et du jeu des priorités.

À titre d'exemple, nous détaillons le calcul de la taille de l'espace des modèles en nous basant sur l'ontologie ayant servi aux premiers tests de LEIA. Nous y avons fixé les paramètres suivants :

– $n_I = 10$ interactions ;

– $n_F = 4$ familles d'agents ;

– $p_{\max} = 10$ et $q_{\max} = 4$ (rayon de perception des agents) ;

– $n_{\mathrm{assi}} = 6$ assignations générées au hasard.

Nous pouvons donc obtenir dans ce contexte environ $2,4 \times 10^{21}$ modèles différents, toujours sans compter toutes les initialisations possibles. Pour parcourir correctement un espace de cette taille, nous devons définir des outils de transformations qui peuvent modifier les parties variables des modèles :

– modifier l'interaction d'une assignation ;

– modifier les sources et cibles d'une assignation ;

– augmenter ou diminuer la priorité d'une assignation ;

– augmenter ou diminuer la garde de distance d'une assignation ;

– ajouter ou retirer des assignations ;

– modifier indépendamment les paramètres d'initialisation des agents et d'une simulation.

Les outils de transformation ou leur composition doivent garantir ces propriétés afin de parcourir l'intégralité de l'espace des modèles. Pour cela, on peut toujours

envisager de démarrer l'exploration à partir de modèles existants, qui se trouvent dans des parties pertinentes de l'espace des simulations. Le choix des outils de transformation spécifiques aux domaines revient, dans le cas général, à choisir les bons outils de transformation définis précédemment et d'en limiter les variations pour rester dans un espace de transformation crédible par rapport au domaine simulé. Par exemple, il n'est peut-être pas pertinent de faire intervenir des gardes de distance excédant le millimètre (voire beaucoup moins) quand on simule le fonctionnement d'une cellule.

Le second point à développer est l'efficacité de la recherche de simulations dans un espace qui, nous le constatons d'après l'exemple précédent, peut rapidement devenir très vaste. Pour cela, il faut développer des outils efficaces d'évaluation d'une simulation afin de guider le concepteur dans sa recherche, voire de parcourir automatiquement cet espace.

3.2. *Les outils d'évaluation d'une simulation*

Lorsqu'on étudie une simulation, les mesures apparaissent identiques suivant le sens qu'en donne le concepteur. Par exemple en biologie, compter le nombre d'enzymes n'est pas particulièrement différent de la mesure de l'activité d'un gène. Pourtant, le nombre d'enzymes est un simple décompte d'agents, c'est-à-dire que cette mesure est réalisable grâce au moteur de simulation, alors que la mesure de l'activité d'un gêne est une mesure spécifique au domaine simulé. Ainsi, les outils de mesure sur les simulations peuvent être envisagés de deux manières différentes :

– des outils génériques qui comprennent :

- les outils liés à la dynamique des simulations multi-agents, c'est-à-dire le nombre d'agents et de familles d'agents, leur répartition dans l'environnement ;

- les outils liés à IODA et JEDI, tels que les interactions résolues, présence de cycles dans les modèles... Ces éléments sont étudiés dans la section suivante.

– des outils spécifiques au domaine simulé, développés par le concepteur pour repérer certains faits spécifiques aux phénomènes étudiés. Par exemple, des mesures de fécondité des proies, un taux de satisfaction des clients... Ces outils spécifiques sont à envisager comme des modules complémentaires pour l'explorateur de l'espace des simulations.

Dans les deux cas, ces outils de transformations doivent être définis en connaissant leur impact sur les mesures effectuées par nos outils. En effet, afin de garantir une recherche efficace, il faut vérifier la convergence des mesures vers un modèle jugé bon, voire optimal. Cette étude est un problème difficile.

Dans tous les cas, ces outils de mesure constituent une aide au concepteur qui peut rester maître de son évaluation, bien que nous puissions envisager une recherche automatique telle que celle proposée dans LEIA.

3.3. *Un navigateur dans l'espace des simulations*

Comme représenté dans la figure 1, pour définir un navigateur dans l'espace des simulations, nous avons besoin :

– d'une ontologie de domaine composée d'agents et d'interactions, rédigée selon la méthodologie IODA ;

– d'outils de mesure pour évaluer les simulations ;

– d'outils de transformation, capable de couvrir tout l'espace des simulations, et dont l'impact sur les mesures garantit une convergence vers un modèle bon, voire optimal.

Impliqué dans le processus d'évaluation des simulations et aidé par les outils d'évaluation, le concepteur peut alors découvrir des modèles aux caractéristiques remarquables par utilisation des outils de transformation.

4. Quelques outils de mesure

Nous spécifions dans cette section des mesures heuristiques qui ont pour objectif de caractériser la complexité d'un système conçu suivant la méthodologie IODA et simulé grâce au moteur JEDI.

Les outils présentés sont implémentés dans le moteur JEDI (Kubera *et al.*, 2008a) qui est un moteur séquentiel et dont la représentation du temps est discrète. Une interaction $\mathcal{I}$ n'a qu'une seule source $\mathcal{S}$ ($\mathrm{card}_S(\mathcal{I}) = 1$). À chaque pas de temps, pour chaque source $\mathcal{S}$ potentielle, on sélectionne un couple (interaction $\mathcal{I}$, cible $\mathcal{T}$). L'interaction est choisie suivant l'assignation de priorité la plus élevée parmi toutes les assignations réalisables pour cette source $\mathcal{S}$. La cible $\mathcal{T}$ de cette interaction $\mathcal{I}$ doit également se trouver dans le voisinage de la source $\mathcal{N}(\mathcal{S})$. Pour le couple ainsi sélectionné, nous résolvons alors l'action de l'interaction $\mathcal{I}$ entre la source $\mathcal{S}$ et la cible $\mathcal{T}$. Cette interaction $\mathcal{I}$ est enregistrée en tant qu'« interaction résolue », c'est-à-dire que son action a été appliquée une fois entre la source $\mathcal{S}$ et la cible $\mathcal{T}$. À chaque pas de temps, nous pouvons donc avoir un retour quantifié sur l'ensemble des événements de la simulation dont découlent nos outils de mesure.

Ces outils ont pour objectif de révéler les qualités et défauts des modèles simulés, relatifs au domaine simulé. Dans ce qui suit, nous nous plaçons, dans le cadre de LEIA, dans une situation simplifiée où chaque case de l'environnement ne peut être occupée que par un seul agent. La distribution initiale des agents et leurs primitives sont donc implémentées en conséquence.

4.1. *Activité dans la simulation*

Grâce au moteur JEDI nous pouvons suivre l'**activité des agents**, en particulier si un agent est en mesure d'effectuer des interactions. Cette mesure appelée « Activité

des agents » caractérise l'interactivité de la simulation, c'est-à-dire la capacité de la simulation à fonctionner et donc évoluer.

Définition 4.1 ***Activité***
Avec $\mathbb{I}_{\text{res}}(t)$ *l'ensemble des « interactions résolues » au pas de temps t, soient* $\mathbb{A}(t)$ *l'ensemble des agents dans une simulation donnée, l'****activité*** *des agents est donnée par :*
$Activite(t) = \text{card}(\mathbb{I}_{\text{res}}(t)) / \text{card}(\mathbb{A}(t))$

Le score fourni est donc le ratio entre le nombre d'agents qui sont sources d'une interaction et la totalité des agents de la simulation. Il faut noter que, dans le paramétrage par défaut du moteur JEDI, la source $\mathcal{S}$ et la cible $\mathcal{T}$ d'une « interaction résolue » sont désactivés (Kubera *et al.*, 2008a) : la cible $\mathcal{T}$ ne peut alors plus participer à une autre interaction au cours du même pas de temps, que ce soit en tant que cible ou source. Plus cette mesure sera faible, plus les agents seront passifs, leur état n'évoluant qu'au gré des quelques interactions qui ont été résolues (en considérant que leur état interne n'évolue pas de lui-même). D'un point de vue interaction, cette mesure permet de révéler des modèles très complexes, comme la vente ou l'achat d'objets, mais qui ne génèrent pas de modifications de l'environnement et de sa représentation.

4.2. *Environnement*

4.2.1. *Modifications de l'environnement*

Le moteur JEDI fournit un environnement assimilable à un ensemble de cellules que peuvent occuper les agents (les agents disposent de coordonnées réelles (Kubera *et al.*, 2008a)). L'environnement est représenté graphiquement dans la simulation comme une grille en deux dimensions, fabriquée à partir des cellules dans lesquelles sont représentés les agents. Il dispose d'un ensemble de primitives, telles que *Placer un agent* ou *Retirer un agent* de l'environnement, qui nécessitent un renouvellement de sa représentation lors de leur utilisation. Nous proposons de mesurer, au pas de temps t, le nombre d'appels à ces primitives de l'environnement, noté $\mathcal{E}(t)$, par rapport au nombre d'agents dans la simulation.

Définition 4.2 ***Modifications***
Au pas de temps t*, le nombre de* ***modifications*** *des agents est donné par :*
$Modifications(t) = \mathcal{E}(t) / \text{card}(\mathbb{A}(t))$

Nous obtenons un indicateur de l'évolution de la représentation de l'environnement entre deux pas de temps. Cette mesure n'est pas toujours bornée : elle n'est comprise entre 0 et 1 que si les interactions résolues ne font appel qu'à une seule primitive de l'environnement nécessitant son rafraîchissement.

4.2.2. *Stabilité de l'environnement*

Cet indicateur est une mesure des points stables de la simulation, effectuée grâce à l'évolution de l'occupation des cases de l'environnement par rapport à chaque famille d'agents. Un grand écart type pour certaines cases montre que celles-ci sont occupées de manière récurrente par la famille d'agents considérée : on peut alors en conclure qu'il s'agit d'un point stable dans la simulation.

Définition 4.3 ***Stabilité***
Au pas de temps t, soient :

– $N_{\mathcal{F}}(t) = card(\mathbb{A}_{\mathcal{F}}(t))$ le nombre d'agents de la famille $\mathcal{F}$;

– p le nombre de cases de l'environnement ;

– $\mathcal{O}_{c,\mathcal{F}}(t)$ la présence cumulée d'agents de la famille $\mathcal{F}$ depuis le début de la simulation dans la case c ;

– $\mathcal{M}_{\mathcal{F}}(t) = \frac{N_{\mathcal{F}}(t) \times t}{p}$ la moyenne d'occupation des cases de l'environnement.

L'écart type d'occupation des cases est : $\sigma_{\mathcal{F}}(t) = \sqrt{\frac{1}{p} \times \sum_{c=1}^{p} (\mathcal{O}_{c,\mathcal{F}}(t) - \mathcal{M}_{\mathcal{F}}(t))^2}$

Imaginons un modèle dans lequel il n'y a absolument aucun mouvement : depuis le départ, chaque agent occupe une case différente, ne peut pas se déplacer dans une autre case et la population Nb est absolument la même depuis le début de la simulation. Au pas de temps t, l'écart type le plus défavorable est :
$\mathrm{def}_{\mathcal{F}}(t) = \sqrt{\frac{1}{p} \times (\sum_{c=0}^{p-Nb} (0 - \mathcal{M}_{\mathcal{F}}(t))^2 + \sum_{c=p-Nb}^{p} (t - \mathcal{M}_{\mathcal{F}}(t))^2)}$
*La **stabilité** est alors définie comme : $Stabilite_{\mathcal{F}}(t) = \sigma_{\mathcal{F}}(t) / \sigma\mathrm{def}_{\mathcal{F}}(t)$*

Si à l'instant t, la population Nb est à son maximum et, que de surcroît, l'occupation des cellules stagne, l'écart type va converger vers l'écart type défavorable. Une simple mesure de la stabilité des agents aurait consisté à déterminer la proportion d'agents restés au même endroit entre deux pas de temps. Notre mesure tient compte ici de la présence cumulée d'agents depuis le début de la simulation : elle nous permet de révéler des zones de convergence des agents dans l'environnement.

Nous fournissons également deux indicateurs sur l'environnement : le mélange et la cohésion. Le mélange correspond à la moyenne du pourcentage d'agents d'autres familles dans le voisinage de chaque agent. De même, la cohésion correspond à la moyenne du pourcentage d'agents de la même famille dans le voisinage de chaque agent. L'idée générale est l'idée communément admise du pourcentage de similarité dans le voisinage de Moore, défini comme les huit cellules entourant une cellule centrale centrée sur l'agent considéré.

Définition 4.4 ***Mélange et Cohésion***
Au pas de temps t, soient :

– $\mathcal{N}(x)$ le voisinage d'un agent x ;

– $\mathcal{F}(x)$ la famille dont l'agent est une instance ;

– $Diff(x) = card(\{a \in \mathcal{N}(x) | \mathcal{F}(a) \neq \mathcal{F}(x)\})$ *le nombre d'agents dans le voisinage d'un agent* x *dont la famille est différente de la sienne ;s*

– $Pareil(x) = card(\{a \in \mathcal{N}(x) | \mathcal{F}(a) = \mathcal{F}(x)\})$ *le nombre d'agents dans le voisinage d'un agent* x *dont la famille est la même que la sienne ;*

– $CasesP(x)$ *le nombre de cases possibles dans le voisinage d'un agent* x *;*

– $Nb = card(\mathbb{A})$ *le nombre d'agents ;*

– p *le nombre de cases de l'environnement.*

Le ***mélange*** *est défini comme :* $Melange = (1/p) \times \sum_{x=1}^{Nb}(Diff(x)/CasesP(x))$ *;*
la ***cohésion*** *est définie comme :* $Cohesion = (1/p) \times \sum_{x=1}^{Nb}(Pareil(x)/CasesP(x))$.

Nous utilisons également une définition classique de la densité.

Définition 4.5 ***Densité***
Au pas de temps t*, soient* $Nb(t) = \mathrm{card}(\mathbb{A}(t))$*, le nombre d'agents et* p*, le nombre de cases de l'environnement, la* ***densité*** *de population est définie comme :* $Densite = \frac{Nb(t)}{p}$.

4.3. *Etude de l'évolution des populations et de la résolution des interactions*

4.3.1. *Usage des interactions*

Grâce à la méthodologie IODA, la séparation des interactions et des agents nous permet d'enregistrer facilement les « interactions résolues » de chaque entité. L'analyse de ces enregistrements nous révèle le comportement des agents et l'usage global des interactions, tout particulièrement l'apparition de cycles dans leur usage voire un ordre entre elles, comme les exemples suivants le montrent.

Source \ Cible	Envir.	Végétal	Sauterelle
Végétal	-	-	-
Sauterelle	-	Brouter 1	Dévorer 0

Tableau 2. *Matrice d'interaction d'un modèle simple de recherche de nourriture : Brouter a une priorité supérieure à Dévorer*

Prenons l'exemple de deux familles d'agents *Végétal* et *Sauterelle* et deux interactions *Brouter* et *Dévorer*. Dans le tableau 2, les *Sauterelles* vont *Brouter* en priorité les végétaux puis, lorsque les *Végétaux* auront disparu, les *Sauterelles* vont se *Dévorer* entre elles. L'interaction *Dévorer* ne sera effectuée que lorsqu'il n'y aura plus de végétaux. Imaginons que dans cet exemple, de nouveaux *Végétaux* puissent pousser en cours de simulation en nombre suffisant pour nourrir les sauterelles présentes : l'interaction *Dévorer* ne pourra alors jamais s'opérer, d'où une étape de

simplification envisageable dans ce modèle. Bien que nous n'ayons pas connaissance a priori de l'évolution de la simulation, nous pouvons détecter de manière heuristique les interactions qui n'apportent rien au modèle.

4.3.2. *Dynamique des interactions*

Dans un phénomène de rétroaction, le résultat du phénomène considéré agit en retour sur lui-même. De telles rétroactions peuvent apparaître dans la résolution des interactions dans nos simulations.

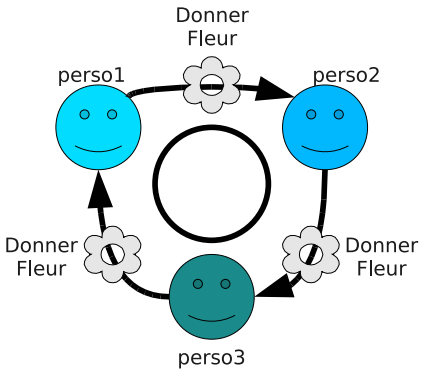

Figure 3. *Expérience simple de transmission d'objet entre trois individus sans remerciement*

Prenons un exemple simple (figure 3) : 3 agents sont placés autour d'une table. Au début de la simulation, un objet *Fleur* est donné à l'un des *Personnages*. En cours de simulation, si un agent a une *Fleur* en main, il effectue l'interaction *Donner*, en l'occurrence la *Fleur*, à l'un de ses voisins. En considérant un des agents, l'interaction *Donner* l'objet *Fleur* ne sera effectuée qu'une fois tous les trois pas de temps, le temps que l'objet *Fleur* fasse le tour de la table : le phénomène est périodique.

La mise en place d'un enregistrement des « interactions résolues » pour chaque agent dans le moteur JEDI permet de détecter l'usage cyclique d'interactions qui font partie d'un éventuel phénomène de rétroaction.

Reprenons l'exemple précédent que nous modifions (figure 4) : après avoir reçu l'objet *Fleur*, l'agent effectue d'abord l'interaction *Remercier* dont la cible est son voisin donateur, puis effectue au pas de temps suivant l'interaction *Donner*, en l'occurrence la fleur, à un autre voisin. Si on étudie les interactions d'un agent, on constate que les interactions *Remercier* et *Donner Objet* sont effectuées tous les 6 pas de temps (2 pas de temps pour chaque personne). *Donner Objet* ne peut être fait qu'après *Remercier* : il y aura donc une phase de $\frac{2\pi}{6}$ entre les deux interactions. Si l'on suit les interactions de manière générale, *Remercier* et *Donner Objet* sont effectuées alternativement une fois à chaque pas de temps.

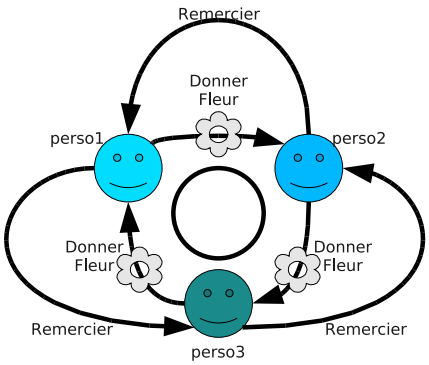

Figure 4. *Expérience simple de transmission d'objet entre trois individus avec remerciement*

L'ordre entre *Remercier* et *Donner Objet* est visible, avec un phénomène périodique de période deux pas de temps. Dans ce cas, nous avons une connaissance de bas niveau sur le scénario modélisé : nous proposons donc l'utilisation d'outils d'analyse fréquentielle afin de déterminer les phénomènes périodiques dans l'usage (fréquence) et l'ordre (phase) des interactions.

4.3.3. *Transformation de Fourier Discrète*

En voyant l'usage des interactions comme un signal discret, nous pouvons utiliser la définition classique de la Transformation de Fourier Discrète (**TFD**) qui nous permet d'étudier les interactions dans l'espace des fréquences sans contrainte dans le choix des fréquences recherchées.

Définition 4.6 ***TFD***
Soient $\mathrm{s}_{\mathcal{I}}(n)$ *l'évolution de l'utilisation d'une interaction* $\mathcal{I}$ *et* Npt *le nombre de pas de temps utilisés par la TFD, la* ***TFD*** *est définie comme :*
$\mathcal{S}_{\mathcal{I}}(k) = \sum_{0}^{\mathrm{Npt}-1} \mathrm{s}_{\mathcal{I}}(n) \times e^{-2ikn/\mathrm{Npt}}$

4.3.4. *Fréquences remarquables*

Nous proposons la recherche de maxima locaux dans le spectre de fréquences obtenu par transformée de Fourier de l'évolution des interactions.

Définition 4.7 ***Fréquences remarquables***
Soit $\mathcal{S}_{\mathcal{I}}(k)$ *la TFD de* $\mathrm{s}_{\mathcal{I}}(n)$*, l'ensemble* Freq *des* ***fréquences remarquables*** *est :*
$\mathrm{Freq} = \{k / \mathcal{S}_{\mathcal{I}}(k-1) < \mathcal{S}_{\mathcal{I}}(k)$ *et* $\mathcal{S}_{\mathcal{I}}(k) > \mathcal{S}_{\mathcal{I}}(k+1)\}$

Si l'échantillonnage est correctement choisi, nous pouvons ainsi détecter des utilisations périodiques d'une interaction. L'échantillon est limité dans le temps : nous

ne pouvons donc trouver que des usages périodiques d'une interaction qui se répète dans cet échantillon. Comme nous ne pouvons faire une simulation durant un temps infini, nous approximons l'usage répété d'une interaction à l'existence d'un cycle.

Nous pouvons dégager deux niveaux d'analyse dans le suivi des interactions :

– Un suivi macroscopique, c'est-à-dire tenant compte de l'ensemble des interactions qui révèlent la dynamique globale du système. La découverte de fréquences remarquables peut mettre en évidence le couplage de certaines interactions.

– Un suivi microscopique, centré sur un agent, permet de suivre la résolution des interactions. Cette analyse permet de révéler la dépendance entre le comportement de l'agent et ses assignations (Kubera *et al.*, 2007). Elle se heurte toutefois à l'espérance de vie des agents dans certains modèles (par exemple, le modèle proie/prédateur). De plus, certaines interactions peuvent désactiver l'agent (au choix du développeur) : cet agent, après avoir effectué une interaction (en tant que source), se voit alors interdire de participer à d'autres interactions en tant que source pour le pas de temps en cours ; il n'a plus que la possibilité d'être cible. Le passage de la simulation au pas de temps suivant réactive ensuite tous les agents desactivés.

L'analyse des fréquences reste à pondérer car certaines interactions peuvent suivre un cadençage, une périodicité qui leur est intrinsèque (et qui est donc indépendante du déroulement de la simulation elle-même).

4.3.5. *Analyse de l'évolution de la population*

Certains modèles, comme le modèle proie/prédateur, vont conduire à la variation périodique des populations, révélateur de phénomènes de rétroaction. La détection de ces phénomènes périodiques est effectuée par analyse de Fourier, comme décrit pour l'analyse des interactions.

5. LEIA : le navigateur dans l'espace des simulations

Le navigateur LEIA[3] est une application embarquant n instances du moteur JEDI. Il permet à l'utilisateur de faire instantanément une comparaison visuelle de n simulations en les voyant toutes évoluer en parallèle (figure 5). Ces n instances sont créées à partir d'un modèle de référence comme dans JEDI Builder. Ce modèle des références est basé sur une bibliothèque (Gruber, 1993), constituée de familles d'agents et d'interactions préchargées. Nous pouvons de surcroît modifier le nombre initial d'agents et leur distribution initiale. LEIA est capable de construire automatiquement plusieurs simulations issues de l'espace des simulations associé à l'ontologie de domaine donnée en entrée.

3. Pour *LEIA lets you Explore Interactions for your Agents* `www2.lifl.fr/SMAC/LEIA/applet.html`

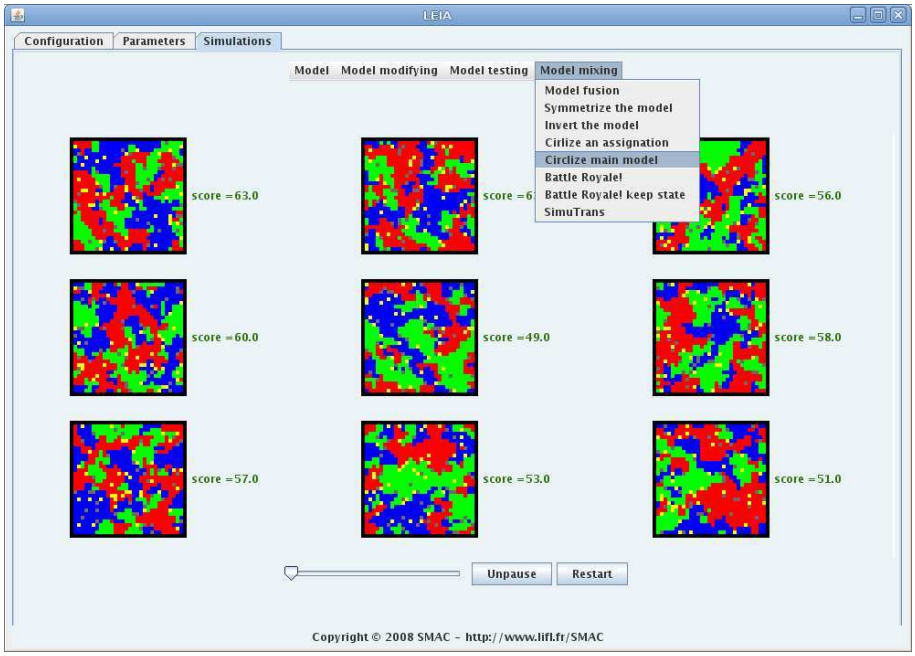

Figure 5. *LEIA, le navigateur dans l'espace des simulations*

5.1. *Manipulation du modèle*

LEIA fournit à l'utilisateur un ensemble d'outils de transformations et de génération de modèles, ainsi qu'un ensemble de jeux de tests afin de parcourir l'espace des simulations. Ces outils permettent de faire varier les paramètres du modèle, que ce soit par l'ajout ou la suppression d'assignations, la modification des priorités ou des gardes de distance, la variation des populations initiales ou des opérations sur la matrice d'interaction toute entière telle que sa symétrisation, voire la fusion des matrices d'interactions de deux modèles. L'utilisateur peut alors, par ces outils, modifier automatiquement le modèle de référence en générant n sous-modèles. Ces modèles sont alors chargés dans les n instances du moteur JEDI.

LEIA peut être utilisé avec autant d'instances que l'on souhaite. Bien entendu, LEIA sera d'autant plus lent qu'un grand nombre d'instances seront chargées. Néanmoins, chaque instance possède son propre fil léger afin de profiter des architectures multicœurs. Sur une architecture à quatre cœurs, nous faisons fonctionner 4 instances avec les mêmes performances qu'une seule.

5.2. *Analyse du modèle*

Le navigateur est secondé par un moteur de statistiques, basé sur les outils de mesures présentés dans la section précédente, afin d'aider l'utilisateur à apprécier les qualités et différences des modèles visualisés. Nous obtenons un retour quantifié de chaque simulation dans lequel nous considérons :

– l'ensemble des interactions résolues par pas de temps par rapport aux interactions envisagées ;

– le nombre de modifications de l'environnement ;

– la répartition des familles d'agents : cohésion et mélange des familles d'agents ;

– l'évolution de l'occupation de l'environnement ;

– évolution remarquable des individus ;

– évolution remarquable des interactions.

Ces mesures heuristiques permettent à l'utilisateur d'accéder à des informations sur chaque modèle par un détail des scores et l'affichage de graphiques mis à jour au fur et à mesure de la simulation.

En plus des outils de mesure déjà présentés, nous pouvons étudier dans le cadre précis de LEIA la construction de la matrice d'interaction du modèle lui-même. Sont remarquables les assignations dites circulaires : c'est-à-dire que pour la même interaction, priorité et garde de distance, les sources et cibles varient de manière cyclique.

Définition 5.1 ***Assignations Cycliques***
Soit $e = (I, p, d)$ *un élément d'assignation.*
On définit : $Assi(e) = \{(\mathcal{S}, \mathcal{T}) \in \mathbb{F}^2 | e \in \text{assi}_{\mathcal{S}/\mathcal{T}}\}$ *l'ensemble des assignations issues d'une matrice d'assignation ayant la même interaction* I*, priorité* p *et garde de distance* d*.*
Si $\text{assi} \in \text{assi}_{\mathcal{S}/\mathcal{T}}$*, alors :*

– $\text{Src}(a) = \mathcal{S}$ *est la famille d'agents étant source de* assi *;*

– $\text{Tgt}(a) = \mathcal{T}$ *est la famille d'agents étant cible de* assi*.*

Les ***assignations cycliques*** *sont l'ensemble des couples* $(\mathcal{S}, \mathcal{T})$ *participant à un cycle* e *:* $\text{Assicyliques}(e) = \{(\mathcal{S}, \mathcal{T}) \in Assi(e) / \exists(\text{assi}_i)_{i\in[1,n]} \subseteq Assi(e) / \text{Src}(\text{assi}_1) = \mathcal{S} \wedge \text{Tgt}(\text{assi}_1) = \mathcal{T} \wedge (\forall i \in [1, n-1], \text{Src}(\text{assi}_{i+1}) = \text{Tgt}(\text{assi}_i)) \wedge \text{Src}(\text{assi}_1) = \text{Tgt}(\text{assi}_n)\}$.
*L'****aspect Cyclique*** *est défini comme :* $\text{Cyclique} = \text{card}(\text{Assicyliques})/\text{card}(\text{Assi})$

L'aspect cyclique d'un modèle est donc la proportion des assignations cycliques parmi l'ensemble des assignations. Son étude permet de mettre en valeur des cycles dans la construction du modèle qui peuvent éventuellement se traduire par des boucles de rétroaction. Plus généralement, l'étude du modèle ouvre aussi les perspectives de

la simplification automatisée du modèle, dont nous posons les bases dans LEIA en éliminant les interactions inatteignables de par leurs priorités et gardes de distance.

5.3. *Score d'une simulation*

Nous fournissons à l'utilisateur un score global afin de réunir les résultats de tous les outils de mesure.

Définition 5.2 ***Score global***
Soit $\mathbb{S} = \{S_1, ..., S_N\}$ *l'ensemble des scores fournis par les outils de mesure (scores compris entre* 0 *et* 100*), le* ***score global*** *est défini comme :*
$ScoreGlobal = 1/N \times \sum_{i=1}^{N}(S_i - 50)$

Nous avons fait le choix de ramener le score global à une note typiquement entre -50 et 50 afin de ne privilégier aucun score par rapport à un autre. La note est centrée sur 0 afin de donner un repère simple à l'utilisateur. Nous ne faisons pas le produit au cas où un score particulier serait nul. Le modèle d'une simulation jugée intéressante parmi les simulations en parallèle peut être défini comme nouveau modèle de référence par l'utilisateur. Il peut alors réitérer le processus de modification du modèle de référence, que ce soit manuellement ou en utilisant nos outils de transformation.

LEIA permet donc d'explorer l'espace des simulations générées à partir de l'ontologie de domaine embarquée. Il faut bien noter que le navigateur LEIA est ouvert à n'importe quelle ontologie de domaine, tant que les interactions et agents sont conçus selon IODA.

5.4. *Un exemple de découverte automatique de phénomènes*

Les outils que nous présentons permettent à l'utilisateur d'effectuer un travail de rétro-ingénierie sur les simulations en explorant l'espace des simulations. À l'instar du « Continuator » (Pachet, 2007) qui exacerbe la créativité musicale, LEIA se révèle être un « chatouilleur d'idées » pour la découverte de nouveaux modèles. Même avec une ontologie simple, avec quelques interactions, les bénéfices tirés du navigateur LEIA sont étonnants, comme nous pouvons le constater avec le « modèle d'infection » présenté ensuite.

L'observation d'une expérience montrant un phénomène de synchronisation entre plusieurs familles d'agents a révélé une organisation intéressante des interactions à l'origine de ce phénomène : la première clone l'agent source dans une case voisine libre ; l'autre tue un agent ciblé. Cet ensemble a été simplifié en une seule interaction. Brièvement, le but de cette nouvelle interaction est de détruire l'agent ciblé trouvé dans le voisinage de la source puis de le remplacer par une copie de la source. Ce

modèle, obtenu avec LEIA, affecte de manière cyclique plusieurs familles d'agents avec cette interaction appelée « Infecter » (tableau 3).

Source \ Cible	Envir.	Rouge	Bleu	Vert
Rouge	-	-	Infecter 0 (1,0)	-
Bleu	-	-	-	Infecter 0 (1,0)
Vert	-	Infecter 0 (1,0)	-	-

Tableau 3. *La matrice d'interaction du modèle d'infection, extensible à n'importe quel nombre de familles d'agents supérieur à deux. « Infecter 0 (1.0) » signifie que l'interaction Infecter ne peut se produire qu'à une distance maximale de 1.0*

Au moins trois familles d'agents sont nécessaires pour éviter le blocage de cette simulation. À partir d'une distribution initiale aléatoire des agents, ce modèle conduit les agents à former des spirales d'infection (voir la figure 6).

LEIA révèle ainsi que le « modèle d'infection » ne peut fonctionner sans remplir l'environnement avec un nombre important et égal d'agents de chaque famille. Plus le nombre d'agents est grand, plus grande est la probabilité qu'un agent source trouve un agent cible. De plus, la distance de garde a un réel impact : une plus grande distance pour une interaction implique pour un agent source d'avoir une plus grande probabilité de trouver une cible pour cette interaction. Donc, plus cette limite est grande, meilleures sont les chances de réaliser « Infecter ». Bien entendu, augmenter cette distance permet d'augmenter le nombre de familles d'agents dans ce modèle.

Nous avons également noté la robustesse de ce modèle à l'ajout d'obstacles dans notre expérience. Ces obstacles sont des agents vides qui ne peuvent interagir avec d'autres agents de la simulation : ils occupent juste une cellule de l'environnement. Des spirales apparaissent malgré la présence de ces obstacles et peuvent même faciliter l'apparition de spirales suivant leurs positions, telle que le montre la figure 7. Cette dynamique[4] est bien connue en chimie dans les réactions cycliques telle que la réaction de Belousov-Zhabotinsky (Zhabotinsky, 1964; Belousov, 1959). De tels modèles sont également étudiés dans le domaine des automates cellulaires via le modèle de Greenberg-Hastings (Fisch *et al.*, 1993).

6. Vers une évolution génétique

Dans le navigateur LEIA, nous pouvons tester plusieurs simulations en même temps, dont le modèle implémenté dynamiquement peut être conçu automatiquement par tirage aléatoire des assignations. L'utilisateur peut alors juger de sa qualité grâce aux outils de mesure. En effet, ceux-ci facilitent la recherche de phénomènes particuliers, par exemple pour repérer :

4. `www2.lifl.fr/SMAC/LEIA/spirale.html`

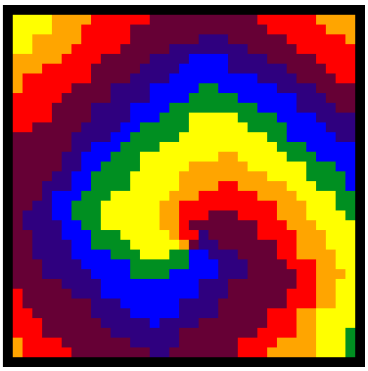

Figure 6. *Capture d'écran de l'environnement du moteur JEDI utilisant le « modèle d'infection » illustrant la formation de spirale entre 7 couleurs*

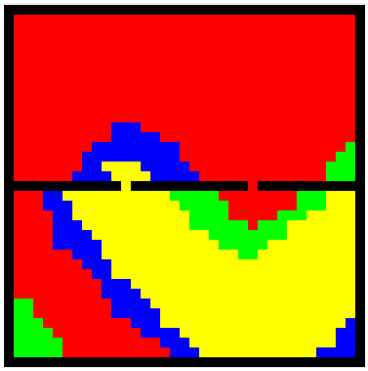

Figure 7. *Capture d'écran de l'environnement du moteur JEDI utilisant le « modèle d'infection » illustrant l'ajout d'un obstacle sauf en 2 points*

– des phénomènes de ségrégation en utilisant les outils donnant la cohésion, le mélange et la stabilité ;

– des évolutions cycliques de la population dont on peut détecter les fréquences remarquables ;

– des modèles convergeant vers des positions stables en observant les variations de l'occupation des cases par les familles d'agents ;

– des modèles provoquant un grand renouvellement de l'environnement par l'étude des modifications.

L'utilisateur peut donc, par itérations successives, raffiner un modèle jusqu'à satisfaire certains critères. Tout comme l'utilisateur peut repérer des biomorphes (Dawkins, 1986) dont l'évolution répond à ses envies, il peut générer visuellement des modèles correspondant à ses besoins. On peut lier la manière de créer de nouveaux modèles aux travaux sur Imagine (N.Monmarché *et al.*, 1999) : ses concepteurs proposent grâce à l'utilisation d'un algorithme génétique une technique originale pour construire des feuilles de styles CSS par évaluations successives de l'utilisateur. Ici, chaque paramètre de la feuille de style est un gène pouvant être croisé ou muté. L'algorithme génère aléatoirement des feuilles de styles, appliquées à un même texte. Un utilisateur peut ensuite choisir une ou plusieurs feuilles de styles aux caractéristiques plaisantes. L'algorithme génère alors des nouvelles feuilles en tenant compte des choix précédents jusqu'à converger, par itérations successives, vers une feuille de style que l'utilisateur jugera à son goût.

Un tel fonctionnement s'assimile à un processus classique de variation/sélection. En effet, les outils de transformation, liés au cadre formel de IODA, ont pour

objectif de créer de la diversité dans les modèles. Sans préjuger des conséquences « phénotypiques » (c'est-à-dire les résultats de la simulation), les outils de mesure et l'utilisateur, guidés par un but qui est ici l'identification de phénomènes remarquables liés au domaine simulé, participent à un processus d'évaluation et de sélection des simulations ainsi créées.

Ainsi, le navigateur dans l'espace des simulations ouvre la perspective de génération de modèles rédigés selon IODA grâce à un algorithme génétique. Lorsqu'un problème admet un ensemble de solutions, un algorithme génétique le résout en évaluant une famille de solutions paramétrées par des gènes en nombre fixe. Ces gènes peuvent évoluer par croisements et mutations des solutions jusqu'à maximiser une fonction d'évaluation (Holland, 1975). L'algorithme converge ainsi vers une solution jugée bonne. Les variations des gènes permettent donc de parcourir l'espace des solutions. Le concepteur définit également une fonction de fitness permettant de cadrer la solution recherchée. Les choix par l'utilisateur d'outils de mesure permettent la création de fonctions de fitness. Tous les scores sont des fonctions d'évaluation jugeant de la qualité des modèles. Nous pouvons voir les assignations comme des gènes auxquels sont associés un facteur de mutation. La mutation peut alors concerner tous les éléments variables d'une assignation : priorité, garde de distance, source, cible, interaction avec des probabilités différentes suivant l'impact supposé du paramètre, ainsi modifier la garde de distance changera moins radicalement le comportement d'un modèle que de modifier l'interaction.

7. Conclusions et perspectives

Dans l'approche classique de la conception de simulations multi-agents, lorsqu'un modèle ne répond pas aux attentes, le concepteur ne dispose d'aucun cadre formel pour le réviser, et se trouve donc réduit à redévelopper certaines parties de sa simulation selon des critères arbitraires, en suivant un processus lent et non guidé. De plus, il lui incombe de prévoir seul l'intégralité des moyens d'évaluation de ses simulations, moyens qui précisément vont lui permettre d'établir plus ou moins finement l'écart entre les attentes et les résultats de simulation.

Dans cet article, nous avons proposé une approche novatrice de la conception de la simulation. Nous avons d'abord montré quelles caractéristiques théoriques doit posséder la représentation des connaissances relatives à la simulation (structure des agents, définition des comportements, affectation des comportements aux agents) afin d'automatiser le passage du modèle à son implémentation. Ces pré-requis sont réalisés en l'occurrence dans la méthodologie IODA et le moteur de simulation JEDI, qui fournissent les moyens de créer et de transformer des modèles extrêmement diversifiés et d'exécuter la simulation correspondante. La méthodologie IODA permet également de concevoir (et d'implémenter) des indicateurs génériques quant à la dynamique d'une simulation, ainsi que des mesures dépendantes du domaine concerné. Ainsi des modèles aux caractéristiques particulièrement remarquables peuvent être repérés et sélectionnés en vue d'itérer le processus de création de variations.

Cette approche tripartite – création automatique de modèles diversifiés, moteur générique de simulation, batterie d'indicateurs pour mesurer la « qualité » des résultats – permet d'explorer l'espace des simulations associé à une bibliothèque d'agents et de comportements, soit sur un mode interactif à la façon d'un navigateur, soit de façon automatique dans une approche sélectionniste.

La méthodologie IODA permet la construction d'un tel explorateur, notamment en établissant une séparation logicielle entre les agents et interactions qui définissent leurs comportements, de sorte que les modèles sont essentiellement décrits par la façon dont les interactions sont affectées aux agents (la matrice d'interaction). Dès lors, la création de variantes à partir d'un modèle consiste principalement à transformer le contenu de ces affectations. Il n'en reste pas moins que toute autre méthodologie susceptible d'engendrer automatiquement des variations sur un modèle de simulation donné permettrait également la réalisation d'un explorateur du même genre.

Afin de valider concrètement les trois prérequis théoriques que nous avons défendus ci-dessus, nous avons implémenté, sur la base de notre moteur de simulation JEDI, le navigateur dans l'espace des simulations LEIA qui permet effectivement la construction itérative de modèles, et ce dans n'importe quel domaine d'application. LEIA fournit un ensemble d'outils génériques pour la transformation et la simplification de modèles. Les simulations qui en sont issues peuvent être visualisées en parallèle et sont soumises à des outils de mesure génériques. Celles-ci mettent en relief certaines caractéristiques remarquables de ces modèles : activité du système, répartition spatiale, stabilité au cours du temps, bouclages, etc. Ce ne sont que des propositions de mesure et le concepteur peut greffer ses propres outils, liés au domaine simulé. Elles facilitent néanmoins la compréhension des modèles construits. Ainsi, dans l'un des domaines où LEIA a été expérimenté, le navigateur a exhibé un modèle à la dynamique similaire au modèle de Greenberg-Hastings.

Cette approche consiste en fait à renverser la vue habituelle de la conception de modèles de simulation, en commençant par l'observation et la sélection des résultats pour, dans un second temps, étudier le comportement sous-jacent. Un tel explorateur dans l'espace des possibles d'un domaine de simulation agit comme un « chatouilleur d'idées » qui suggère des pistes au chercheur et stimule sa curiosité en l'immergeant dans un processus de variation/sélection.

8. Bibliographie

Belousov B. P., « A periodic reaction and its mechanism », *Compilation of Abstracts on Radiation Medicine*, 1959.

Bensaid N., Mathieu P., « A Framework for Cooperation in Hierarchical Multi-Agent Systems », *Johofstadterurnal of Mathematical Modelling and Scientific Computing*, September, 1998.

Chikofsky E. J., II J. H. C., « Reverse Engineering and Design Recovery : A Taxonomy », *IEEE Software*, vol. 07, n° 1, p. 13-17, 1990.

Dawkins R., *The Blind Watchmaker*, W.W. Norton & Company, Inc., New York, 1986.

Fisch R., Gravner J., Griffeath D., « Metastability in the Greenberg-Hastings Model », *eprint arXiv :patt-sol/9303005*, p. 3005-+, March, 1993.

Gruber T. R., *Towards Principles for the Design of Ontologies Used for Knowledge Sharing in Formal Ontology in Conceptual Analysis and Knowledge Representation*, Kluwer Academic Publishers, London, 1993.

Gutknecht O., Ferber J., « The MADKIT Agent Platform Architecture », *Agents Workshop on Infrastructure for Multi-Agent Systems*, p. 48-55, 2000.

Hofstadter D., *Ma thémagie : En quête de l'essence de l'esprit et du sens*, Intereditions, London, 1988.

Holland J., *Adaptation in natural and artificial systems*, University of Michigan Press, 1975.

Kubera Y., Mathieu P., Picault S., « La complexité dans les Simulations Multi-Agents », *Actes des 15e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2007)*, Cépaduès, p. 139-148, 2007.

Kubera Y., Mathieu P., Picault S., « Formalisation et Implémentation des Interactions pour la Simulation Centrée Individu », *Revue d'Ingéniérie des Systèmes d'Information*, vol. 14, n° 1-2, p. 9-33, Janvier-Juin, 2008a. Numéro spécial Architectures Logicielles.

Kubera Y., Mathieu P., Picault S., « Interaction-Oriented Agent Simulations : From Theory to Implementation », ECAI 08 July 21-25, 2008b.

Kubera Y., Mathieu P., Picault S., « How To Avoid Biases In Reactive Simulations », *in* Y. Demazeau, J. Pavòn, J. Corchado, J. Bajo (eds), *Proceedings of the 7th International conference on Practical Applications of Agents and Multi-Agents Systems (PAAMS'2009)*, vol. 55 of *Practical Advances in Intelligent and soft computing*, Springer, p. 100-109, 2009.

Mathieu P., Picault S., Routier J.-C., « Donner corps aux Interactions », *Actes des 4e Journées Francophones sur les Modèles Formels de l'Interaction (MFI'07)*, p. 333-340, 2007.

Minar N., Burkhart R., Langton C., Askenazi M., The Swarm Simulation System, A Toolkit for Building Multi-Agent Simulations, Working paper n° 96-06-042, Santa Fe Institute, 1996.

N.Monmarché, G.Nocent, M.Slimane, G.Venturini, « Imagine : a tool for generating HTML style sheets with an interactive genetic algorithm based on genes frequencies », *IEEE International Conference on Systems, Man, and Cybernetics (SMC'99)*, vol. 3, p. 640-645, 1999.

Pachet F., « De la co-construction d'un langage homme-machine : quelques expériences en musique », *Actes des 15e Journées Francophones sur les Systèmes Multi-Agents (JFSMA'2007)*, p. 9, 2007.

Ricordel P.-M., Demazeau Y., « La plate-forme Volcano - Modularité et réutilisation pour les systèmes multi-agents », *Technique et Science Informatiques*, vol. 21, n° 4, p. 447-471, 2002.

Wilensky U., « NetLogo (and NetLogo User Manual) », 1999.

Zhabotinsky A. M., « Periodic processes of malonic acid oxidation in a liquid phase », *Biofizika*, 1964.