Description automatique de dynamiques de groupes dans des simulations à base d'agents

Philippe Caillou, Javier Gil-Quijano

Description automatique de dynamiques de groupes dans des simulations à base d'agents

SimAnalyzer : un outil générique pour l'analyse de simulations

Philippe Caillou [1], **Javier Gil-Quijano** [2]

1. *Laboratoire de Recherche en Informatique*
   Université Paris Sud – INRIA, France
   philippe.caillou@lri.fr

2. *CEA, LIST, Laboratoire Information Modèles et Apprentissage*
   F-91191 Gif-sur-Yvette, France
   javier.gil-quijano@cea.fr

RÉSUMÉ. *Les simulations à base d'agents (MABS) ont été utilisées avec succès pour modéliser des systèmes complexes dans de nombreux domaines. Néanmoins, un problème des MABS est que leur complexité augmente avec le nombre d'agents et de types de comportements différents considérés dans les modèles. Pour des systèmes de taille moyenne à grande, il est impossible de valider, voire même d'observer simplement les trajectoires des agents individuels lors d'une simulation. Les approches de validation classiques, où seuls des indicateurs globaux sont calculés, sont trop simplistes pour permettre d'évaluer le modèle de simulation avec un degré de confiance suffisant. Il est alors nécessaire d'introduire des niveaux intermédiaires de validation et d'observation. Dans cet article, nous proposons l'utilisation de la classification automatique de données (clustering) combinée à la caractérisation automatisée de clusters pour construire, décrire et suivre l'évolution de groupes d'agents en simulation. La description de clusters est utilisée pour générer des profils d'agents qui sont réintroduits dans les simulations avec l'objectif d'étudier la stabilité des descriptions et des structures des clusters sur plusieurs simulations et de décider de leur capacité à décrire les phénomènes modélisés. Ces outils permettent au modélisateur d'avoir un point de vue intermédiaire sur l'évolution du modèle. Ils sont suffisamment flexibles pour être appliqués à la fois hors ligne et en ligne comme le montrent les analyses réalisées à la fois sur des simulations Netlogo et sur des logs de simulations.*

Revue d'intelligence artificielle – n° 6/2013, 739-764

ABSTRACT. *Multi agent based simulations (MABS) have been successfully exploited to model complex systems in different areas. Nevertheless a pitfall of MABS is that their complexity increases with the number of agents and the number of different types of behaviours considered in the model. For average and large systems it is impossible to validate the trajectories of single agents in a simulation. The classical validation approaches, where only global indicators are evaluated, are too simplistic to give enough confidence on the simulation's model. It is then necessary to introduce intermediate levels of validation. In this paper we propose the use of data clustering and automated characterization of clusters in order to build, describe and follow the evolution of groups of agents in simulations. The description of clusters is used to generate profiles of agents that are reintroduced in simulations in order to study the stability of the descriptions and structures of clusters over several simulations and decide their capability to describe the modelled phenomena. These tools provides the modeller with an intermediate point of view on the evolution of the model. They are flexible enough to be applied both offline and online, and we illustrate it with both a NetLogo and a CSV-simulation log example.*

MOTS-CLÉS : *simulation multi-agents, classification, description automatique, validation.*

KEYWORDS: *multi-agent based simulation, clustering, automated description, validation.*

DOI:10.3166/RIA.27.739-764

1. Introduction

La modélisation par simulation de systèmes complexes est un processus cyclique, le modèle etant progressivement amélioré en introduisant des nouvelles connaissances et en corrigeant des *bugs* et/ou des effets indésirables. Les systèmes multi-agents (SMA) sont particulièrement bien adaptés pour représenter les phénomènes complexes à partir de la description des comportements locaux d'agents. Une fois que les comportements des agents sont définis, le processus de modélisation cyclique lors de l'utilisation du SMA se concentre généralement sur la détermination de paramètres globaux de la simulation et/ou des états initiaux des agents qui permettent de reproduire les comportements globaux observés dans les phénomènes modélisés. Calibrer les simulations signifie trouver le bon ensemble de valeurs (ou intervalles de valeurs) pour les paramètres globaux et individuels qui permettent de minimiser la différence entre comportements aggrégés simulés et observés. Cette démarche d'optimisation est mise en oeuvre dans plusieurs outils existants comme LEIA (Gaillard *et al.*, 2008) et GAMA (Taillandier *et al.*, 2010).

Néanmoins, cette approche traditionnelle présente des limites lorsqu'il s'agit de modéliser des systèmes complexes. En effet, dans de tels systèmes, des phénomènes différents peuvent se produire simultanément à différents niveaux et s'influencer mutuellement (Gil-Quijano *et al.*, 2010). Par exemple, des groupes d'agents (vols d'oiseaux, groupes sociaux, etc) suivant des trajectoires similaires peuvent apparaître, évoluer et disparaître. Pour décrire et évaluer l'évolution de ces groupes, l'observation de variables globales n'est pas suffisante. En outre, en raison des propriétés émergentes des systèmes complexes, ces groupes peuvent être inattendus, et, pire encore, leur

présence peut passer inaperçue, car aucune variable ou tout autre mécanisme d'observation adapté n'est fourni dans le simulateur. L'importance et l'existence même des groupes peuvent alors être cachées en raison du nombre très important d'informations qui peuvent être observées dans les simulations SMA.

Dans cet article, nous introduisons l'utilisation d'outils statistiques pour aider le modélisateur dans la découverte, la description et le suivi de l'évolution des groupes d'agents. Nous proposons deux outils complémentaires, le clustering de données et le calcul de valeurs-test, utilisés pour détecter automatiquement et décrire des groupes d'agents. Après un bref état de l'art (section 2), ces outils sont présentés en section 3. Dans la section 4, nous présentons le modèle d'observation qui les utilise pour produire une analyse automatisée de l'évolution des groupes dans des simulations SMA, puis nous l'appliquons en section 5 à deux exemples, le premier en ligne (sur un modèle NetLogo) et le second sur des logs CSV.

2. Etat de l'art

Les simulations à base d'agents ont été largement utilisées pour modéliser des phénomènes économiques, géographiques ou sociaux. Il existe plusieurs plateformes de simulation, certaines plus accessibles, comme NetLogo (Wilensky, 1999), et d'autres plus ouvertes, comme par exemple Moduleco (Phan, 2004), Gama (Taillandier *et al.*, 2010) ou Repast (North *et al.*, 2006).Se basant sur ces plateformes, certains outils permettent l'analyse automatique de simulations, tels que l'analyseur Leia (Gaillard *et al.*, 2008), OpenMole (Reuillon *et al.*, 2010) et (Caillou, 2010).

Leia (Gaillard *et al.*, 2008) est un navigateur d'espace de simulations, qui donne à l'utilisateur une comparaison visuelle instantanée de N simulations. En utilisant les spécificités du modèle Ioda utilisé pour les simulations (et en particulier le fait que définir le modèle revient à définir une matrice d'interactions), Leia fournit un ensemble d'outils de transformation et de génération pour la modélisation, ainsi que des outils pour le parcours de l'espace des simulations possibles. De plus, des heuristiques permettent de juger de l'intérêt d'un jeu de paramètres par rapport aux autres, et ainsi d'identifier des simulations intéressantes pour l'utilisateur. Un état de l'art très complet sur les outils de simulation multi-agent et l'exploration des paramètres dans ce type de simulation est abordé dans (Treuil *et al.*, 2008).

SimExplorer/OpenMole (Reuillon *et al.*, 2010) est une plateforme actuellement en développement, qui vise à offrir un environnement générique pour la définition de workflow et l'exécution distribuée d'expérimentations sur des modèles complexes (et en particulier de simulations). Les objectifs sont multiples : (1) fournir un outil simple et libre de définition de workflow générique pour un grand nombre d'outils de simulations et d'analyse ; (2) favoriser la réutilisation des composants disponibles, et donc de réduire l'effort de développement visant les applications de bonne qualité pour l'exploration de modèles ; (3) permettre une distribution simple et presque transparente pour l'utilisateur de la distribution de l'exécution de simulations en parallèle.

(Caillou, 2010) présente un outil qui permet de générer et d'exécuter des nouvelles simulations jusqu'à ce que les résultats obtenus soient statistiquement validés selon un test de χ^2. Il peut générer des nouvelles simulations et effectuer des tests statistiques sur les résultats, avec une précision qui augmente progressivement à mesure que les résultats sont produits. Il est utilisable sur n'importe quelle simulation basée sur RePast.

Cependant, tous ces outils ont pour objectif d'explorer l'espace des paramètres par l'exécution/l'analyse de plusieurs simulations. Notre objectif principal est d'étudier plusieurs étapes d'une seule simulation. Pour explorer une simulation, les seuls outils existants sont des outils intégrés (tels que les graphes et les logs dans NetLogo), qui se limitent à des groupes d'agents définis par l'utilisateur, et le data mining classique sur des logs de simulation. Nous souhaitons prendre en compte les avantages de l'analyse en ligne et de l'analyse orientée agent de NetLogo et les combiner avec le potentiel de flexibilité et de description des outils de dat amining.

3. Outils d'analyse

Dans cette section nous présentons brièvement deux outils fondamentaux que nous utilisons pour analyser les simulations à base d'agents. Il s'agit de la classification automatique de données (*data-clustering*) utilisée pour retrouver des groupes d'agents, et le calcul de la valeur-test utilisé pour décrire ces groupes. Une fois ces outils définis, nous nous situons par rapport aux méthodes de visual analytics actuel dans le suivi des groupes.

3.1. Retrouver les groupes : la classification automatique de données

La *classification automatique* a pour objectif de « trouver la structure » d'un ensemble de données du même type. Le plus souvent les données représentent des objets ou des individus décrits par un certain nombre de variables ou de caractères (Lebart *et al.*, 2006). La structure de l'ensemble des données est présentée sous forme de partition ou de hiérarchie de partitions. Chaque donnée est assignée à un *cluster* dans la partition ou, dans le cas d'une hiérarchie de partitions, à plusieurs *clusters* (si on considère chaque branche de l'arbre comme un cluster, en plus des feuilles). On suppose que la structure existe et l'objectif est donc de la rendre évidente. Le processus de classification automatique comporte deux étapes essentielles :

– Le partitionnement

– La description des partitions

Le *partitionnement* est la phase de mise en évidence de la structure qui décrit l'ensemble des données. Il existe un très grand nombre d'algorithmes de partitionnement. Classiquement, ils se classifient en algorithmes agglomératifs et en algorithmes hiérarchiques. La phase *de description des partitions* fait l'objet de la section suivante. L'éventail d'algorithmes de partitionnement est importante, un état de l'art très complet est présenté dans différentes publications et ouvrages tels que (Nakache, Confais,

2005 ; Berkhin, 2006 ; Xu, II, 2005). Les algorithmes de partitionnement ont en commun deux objectifs : maximiser la compacité des partitions et maximiser la différenciation entre clusters. Le premier objectif consiste à trouver la partition la plus compacte possible, c'est-à-dire la partition pour laquelle la similarité entre les éléments dans un même cluster est maximale. La compacité d'un cluster est souvent mesurée à partir de sa variance interne (σw) ou variance intra-cluster. Le deuxième principe consiste à trouver la partition où les clusters sont les plus différents les uns des autres. La différenciation entre clusters est mesurée à partir de la variance inter-clusters (σb).'Ces deux principes sont illustrés dans la figure 1. Ces principes se basent sur la définition d'une mesure de distance entre éléments. Plusieurs choix sont possibles, par exemple la distance euclidienne (e.g. pour les variables continues), la distance de Hamming (e.g. pour les variables nominales), la distance de Khi2, etc. Le choix de la mesure de distance et de l'algorithme de partitionnement conditionnent la structure/partition retrouvée.

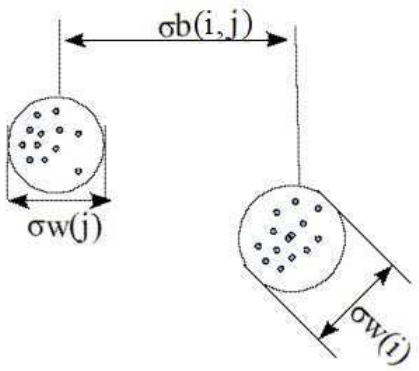


Figure 1. Illustration des principes de compacité et de différenciation des partitions dans un processus de partitionnement. $\sigma b(i,j)$ est la variance inter-clusters, $\sigma w(i)$ est la variance intra-cluster du cluster i

Étant donné que l'état d'un agent est défini par un vecteur qui inclut les valeurs instantanées sur un ensemble de variables données, il est possible de considérer l'ensemble des états pour tous les agents dans une simulation comme un ensemble de données sur lequel il est possible d'exécuter un algorithme de partitionnement. L'objectif de ce partitionnement est de déterminer des groupes d'agents dont la similarité est maximale. Pour effectuer le partitionnement, l'algorithme et la mesure de distance "adéquats" doivent être choisis. En effet, différents algorithmes de partitionnement peuvent conduire à des résultats différents en raison de leurs hypothèses sous-jacentes. L'automatisation de ce choix est un problème complexe et sort du cadre de notre travail. Nous laissons, donc la responsabilité au modélisateur de choisir l'algorithme de partitionnement ainsi que de la mesure de distance entre données. Pour permettre ce choix, nous intégrons dans SimAnalyzer la librairie d'apprentissage automatique

weka [1], qui contient une liste importante d'algorithmes de partitionnement et de fonctions de distance (notre approche et notre outil sont toutefois indépendants du choix de la librairie). Dans nos expérimentations nous avons utilisé l'algorithme X-means (Pelleg, Moore, 2000). L'algorithme *X-means* est une amélioration de l'algorithme *K-Means* (Lloyd, 1982). L'objectif de cet algorithme est de retrouver les k prototypes (vecteurs caractéristiques moyens) qui représentent au mieux les k clusters dans une partition sur un ensemble de données. Dans K-Means le nombre k de prototypes est fixe, X-Means retrouve le nombre "le plus adapté" de prototypes. Il se base sur l'exécution successive de l'algorithme K-Means pour différentes valeurs de k (entré un k_{min} et un k_{max} donnés). Alors, $k_{max} - k_{min} + 1$ partitions sont calculées. La partition choisie est celle qui maximise la compacité des clusters et maximise la distance entre prototypes.

Le choix de l'algorithme X-means est fait ici pour faciliter l'illustration. En effet, cet algorithme nécessite un paramétrage limité et présente l'avantage de calculer le nombre de clusters. Toutefois, il ne faut pas oublier que cet algorithme souffre d'un certain nombre de limitations héritées de l'algorithme k-means, notamment la dépendance de la qualité des solutions à l'initialisation des prototypes.

3.2. Description des groupes : valeur-test

La description automatique des clusters dans une partition se base généralement sur des statistiques qui comparent des moyennes ou des pourcentages à l'intérieur des clusters avec les moyennes ou les pourcentages obtenus sur l'ensemble des données à classer (Lebart *et al.*, 2006). En complément de ces statistiques, il existe également des aides graphiques pour présenter des synthèses visuelles des descriptions comme les dendogrammes, les graphiques de densité, etc (Ohsumi, 1988).

Nous nous intéressons ici à la valeur-test (VT) (Morineau, 1984), un indicateur qui permet d'opérer un tri sur les variables, et de désigner ainsi les variables les plus caractéristiques de chaque cluster dans une partition. Cette mesure peut être appliquée aussi bien aux variables continues qu'aux variables nominales. Elle se base sur le calcul de l'écart entre les valeurs relatives au cluster et les valeurs dans l'ensemble de données. Les variables/modalités représentatives d'un cluster sont celles qui donnent des valeurs fortes de la valeur absolue des VT associées.

L'hypothèse principale dans le calcul des VT est que les variables suivent une distribution normale. Sous cette hypothèse, on considère, avec un niveau de risque de 5 %, qu'une variable/modalité est significative si la valeur absolue de sa VT est supérieure à 2 (Lebart *et al.*, 2006). A partir du calcul des VT, il est possible de définir un marquage des clusters avec les variables/modalités les plus représentatives. La méthode la plus simple est de choisir les m variables/modalités avec les VT les plus significatifs.

1. http://weka.wikispaces.com/

Valeur-test pour les variables continues

Le calcul de la valeur-test pour une variable continue se base sur le centrage de la variable autour de son espérance (Morineau, 1984). Si nous considérons un ensemble de données de taille n et un cluster k de taille n_k et étant donné $E(n^j)$ et $\sigma^2(n^j)$ (respectivement l'espérance et la variance de la variable j sur l'ensemble de données) et $E(n_k^j)$ (l'espérance de j sur le cluster k), le VT de j sur k est donné par l'expression suivante :

$$VT(j,k) = \frac{(E(n_j,k) - E(n^j))}{\sqrt{(\frac{n-n_k}{n-1} \times \frac{\sigma^2(n^j)}{n_k})}} \tag{1}$$

Valeur-test pour les variables nominales

Dans le cas des valeurs-test pour les variables nominales, il s'agit de trouver les modalités dans chaque cluster dont l'abondance permet de dire qu'elles sont représentatives du cluster. L'abondance de la modalité j dans le cluster k est mesurée en comparant son pourcentage dans le cluster avec son pourcentage dans l'ensemble de données. Étant donné :

- n le nombre total de données dans l'ensemble à classer ;
- n_j le nombre de données dans l'ensemble à classer avec la modalité j ;
- n_k le nombre de données dans le cluster k,
- et n_k^j le nombre de données dans le cluster k avec la modalité j ;

la valeur-test de la modalité j dans la classe k est donnée par l'expression :

$$VT(n_k^j) = \frac{n_k^j - E(n_k^j)}{S_k(n_k^j)} \tag{2}$$

Où $E(n_k^j)$ et $S_k^2(n_k^j)$ sont respectivement l'espérance et la variance de j sur le cluster k :

$$E(n_k^j) = n_k \times \frac{n_j}{n} \tag{3}$$

$$S_k^2(n_k^j) = n_k \times \frac{n-n_k}{n-1} \times \frac{n_j}{n} \times (1 - \frac{n_j}{n}) \tag{4}$$

Justification du choix de la valeur-test

Le choix de la valeur-test comme test statistique repose sur sa simplicité d'interprétation pour un non spécialiste (différence plus ou moins forte à la moyenne) et sa capacité à ordonner les variables les unes par rapport aux autres, facilitant une description simple des groupes. D'autres alternatives auraient pu être : les analyses de

variances type ANOVA, mais qui présentent l'inconvénient de comparer les clusters entre eux, et non les clusters à l'ensemble de la population (ce qui rend leur interprétation plus complexe lorsqu'il y a un grand nombre de clusters) ; ou des indicateurs type enthropie, mais il aurait fallu les combiner à un indicateur de valeur relative pour définir si une variable était plutôt élevée ou faible (ce qui nous semble une information importante pour l'utilisateur), pour finalement redéfinir la valeur-test, qui intègre déja une grande quantité d'information sur la variance du groupe et de la population.

La valeur-test n'est toutefois pas sans défaut. En particulier, elle est sensible à la taille de l'échantillon (conduisant à une augmentation de la valeur de l'ordre de $\sqrt{n}$, ce qui peut être notable dans le cas de grande population d'agents), ce qui a conduit à une extension normalisée VT100 (Lebart *et al.*, 2006). Toutefois, cette limite ne remet pas en cause l'utilisation qui peut en être faite pour comparer les variables sur un même groupe ou sur des groupes de taille similaire dans le temps (ce que notre approche propose).

3.3. Retracer la dynamique des groupes : analyse et visualisation

La classification automatique de données (voir section 3.1) permet de découvrir la structure d'un ensemble de données de manière statique (i.e. à un instant déterminé dans le temps), puis les statistiques comme la valeur-test (voir section 4.3.4) de les décrire. Pour pouvoir saisir l'évolution de la structure des données dans le temps, il existe des méthodes récentes basées sur l'analyse et sur la visualisation interactive de l'évolution des partitions. Ces travaux se basent principalement sur la détection de changements dans la structure, la taille et/ou la forme des clusters. Ainsi, par exemple, dans (Spiliopoulou *et al.*, 2006), l'algorithme MONIC permet la détection de *transitions internes* (i.e. les changements qui concernent le contenu et la forme d'un cluster) et *transitions externes* (i.e. les changements d'un cluster qui concernent ses relations avec d'autres clusters). L'algorithme MONIC permet d'identifier des transitions internes telles que la division, l'absorption et la disparition d'un cluster, et des transitions externes telles que le changement de taille, de compacité ou de déplacement du centroïde du cluster. A la suite de ces travaux, dans (Held, Kruse, 2013), les auteurs proposent une visualisation sous forme de graphe de l'évolution des clusters, qui se base sur des représentations graphiques spécifiques à chacune des formes de transitions détectables par l'algorithme MONIC.

D'autres travaux se basent sur le calcul de partitions qui inclut la dimension temporelle dans le processus de clustering. L'objectif est de déterminer des partitions cohérentes, à la fois sur l'espace des variables et sur l'espace temporel. C'est le cas de l'algorithme *TDCK-Means* (Rizoiu *et al.*, 2012) incluant une mesure de dissemblance temporelle qui, tout en introduisant le temps dans le clustering, ne réduit pas la pertinence des partitions sur l'espace des variables.

L'intérêt croissant pour l'étude des réseaux sociaux a donné également lieu à un nombre important d'algorithmes d'analyse de la dynamique de clusters. Ces travaux portent sur l'analyse et la visualisation de l'évolution des graphes d'interactions entre

individus, et sont souvent contraints par la taille importante des données à traiter. Parmi le nombre important de ces travaux, nous pouvons citer *ContextTour* (Lin *et al.*, 2010), qui comporte un algorithme de clustering du graphe de relations (Dynamic Relation Clustering) et un algorithme de visualisation des résultats du clustering (Dynamic Network Contour-Map). ContexTour permet de prendre en compte de manière simultanée plusieurs relations (e.g. des relations entre individus, entre objets et individus, etc.) dans le suivi de l'évolution d'une communauté. Pour traiter le problème de la taille des données, la visualisation de la "carte du réseau" se concentre sur l'affichage d'entités représentatives des différents clusters.

Dans le cadre de simulations multi-agents, l'utilisation d'outils offrant de suivre l'évolution de clusters permettrait à l'expérimentateur d'avoir une vision synthétique de l'évolution du système, et pourrait l'aider dans la détermination et la caractérisation des transitions du système. Nous pensons ici à la formation, à l'évolution et à la disparition de structures ordonnées. Nous faisons l'hypothèse que ces transitions s'accompagneront de changements importants dans la structure interne des clusters. Dans notre travail, nous proposons des outils d'analyse et de visualisation interactive qui permettent le suivi de l'évolution des clusters, en se focalisant, soit sur des agents précis (suivi par extension), soit sur une méthode d'identification fixe (suivi par intension). Ces outils sont présentés dans la section 4.3.5.

Cette distinction nous permet d'éviter le problème de la mise en relation de clusters sur deux periodes, car nous ne faisons pas (et n'avons pas besoin de faire) de lien entre les k clusters de t et les k' de t+1. En effet, à partir des k clusters de t, nous conservons soit l'intention soit l'extension, mais dans les deux cas on se retrouve avec k nouveaux clusters parfaitement identifiés (donc 2k). Nous calculons également k' nouveaux clusters, qui peuvent avoir dans ce cas une intension et extension tres différentes des 2k précédents, mais ils ne sont pas mis en relation avec les 2k précédents. Ce lien entre les deux, ne serait-ce que pour réduire le nombre de clusters (nous en sommes a 2k+k' en t=2), sort du cadre d'analyse du modèle et est présenté en perspective.

4. Modèle d'observation

4.1. Présentation

Notre objectif est de décrire, en ligne ou hors ligne, ce qui se passe dans une simulation au niveau des clusters d'agents. Notre modèle d'observation peut être décrit à travers plusieurs étapes présentées figure 2 :

1. Model selection : que veut-on analyser ?
2. Data processing : quelles sont les données et comment sont-elles retraitées ?
3. Clustering : comment identifier des groupes homogènes ?
4. Cluster description : comment décrire ces groupes ?
5. Cluster evolution : comment ces groupes évoluent-ils au cours de la simulation ?

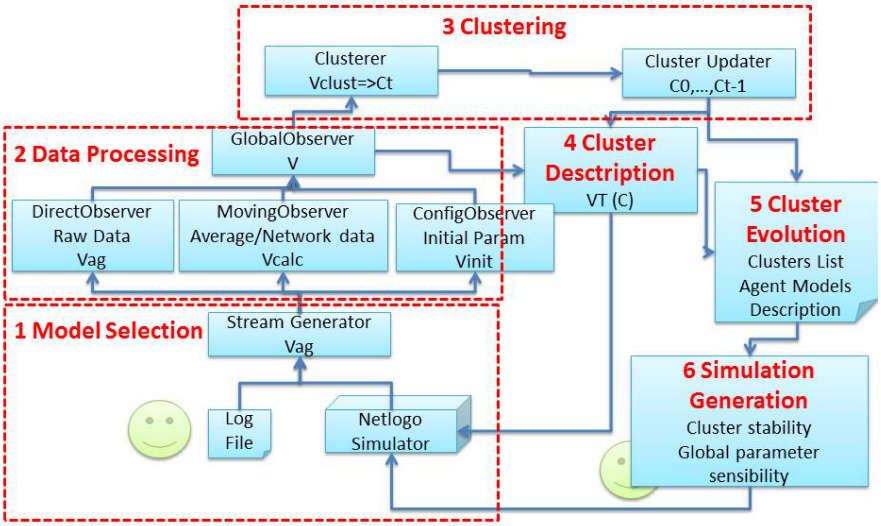


Figure 2. Présentation globale du modèle d'analyse

6. Simulations generation : les groupes identifiés sont-ils stables sur plusieurs simulations ?

4.2. Notations

Etant donné l'ensemble de simulations $\mathbb{S} = \{s_1, ..., s_s\}$, l'ensemble d'agents $\mathbb{A}_t^s = \{a_{t,1}^s, ..., a_{t,n}^s\}$ au pas de temps t dans une simulation donnée $s \in \mathbb{S}$ et l'ensemble $\mathbb{V} = \{v_1, .., v_m\}$ des variables utilisées pour analyser les simulations dans $\mathbb{S}$.

L'ensemble $\mathbb{V}$ est donnée par : $\mathbb{V} = \mathbb{V}ag \cup \mathbb{V}init \cup \mathbb{V}calc$ où :

– $\mathbb{V}ag = \{v_{ag,1}, ..., v_{ag,d}\}$ est l'ensemble des variables qui décrivent les états des agents. Pour l'instant, nous ne considérons qu'un seul type d'agent, donc $\mathbb{V}ag$ est identique pour chaque agent.

– $\mathbb{V}init = \{v_{init,1}, ..., v_{init,d}\}$ est l'ensemble des variables virtuelles qui contient les valeurs initiales de $\mathbb{V}ag$. $\mathbb{V}init$ peut être considéré comme l'ensemble des paramètres des agents.

– $\mathbb{V}calc = \{v_{calc,1}, ..., v_{calc,q}\}$ est l'ensemble des variables statistiques calculées sur chaque agent. Par exemple la moyenne mobile de chaque variable dans $\mathbb{V}ag$.

Nous définissons l'ensemble $\mathbb{V}clust \subset \mathbb{V}$ qui contient les variables utilisées dans le clustering.

Le sous-ensemble $\mathbb{V}mod^C \subseteq \mathbb{V}init$ contient l'ensemble de variables initiales utilisées pour définir un *modèle d'agent* associé au cluster C.

Nous notons $x_j^{a,t}$ la valeur de la variable v_j pour l'agent a au pas de temps t, et $X_{\mathbb{V}}^{a,t} = \{x_j^{a,t}/v_j \in \mathbb{V}\}$ l'ensemble de ces valeurs pour toutes les variables de $\mathbb{V}$.

Nous définissons l'ensemble de clusters d'agents au pas de temps t dans la simulation s comme $\mathbb{C}_t^s = \{C_{t,1}^s, ..., C_{t,c}^s\}$.
$\mathbb{C}_t^s$ a une fonction d'intension associée :
$intension_{\mathbb{C}_t^s}(a \in \mathbb{A}_t^s, X_{\mathbb{V}_{clust}}^{a,t}) \rightarrow C \in \mathbb{C}_t^s$. Cette fonction permet d'affecter un agent a au cluster C le plus indiqué en tenant compte de ses valeurs pour les variables de clustering $\mathbb{V}clust$. Par exemple dans le cas de l'algorithme classique de clustering k-means
$intension_{\mathbb{C}_t^s}(a, X_{\mathbb{V}_{clust}}^{a,t}) = \underset{C}{\operatorname{argmin}} \sum_{C \in \mathbb{C}_t^s} \sum_{j=1}^{c} |x_{clust,j}^a - \mu_j^C|^2$, où μ_j^C est la valeur moyenne de la variable j dans le cluster C. Pour l'algorithme k-means, la fonction de définition affecte l'agent a à son cluster le plus proche.
Nous appelons $extension(C_t^s)$ l'ensemble d'agents contenus dans le cluster C_t^s. $\forall C_t^s \in \mathbb{C}_t^s, extension(C_t^s) \subset \mathbb{A}_t^s$.

Etant donné un cluster C_t^s, nous pouvons définir l'ensemble $\mathbb{V}sig^{C_t^s} \subset \mathbb{V}$ de variables significatives, comme l'ensemble de variables dont la valeur-test (VT) est significative (voir section 4.3.4) : $\forall v \in \mathbb{V}sig^{C_t^s}, |VT(v, C_t^s)| \geq 2$.

En fin, nous définissons l'ensemble $\mathbb{GP}_s = \{gp_{1,s}, ..., gp_{p,s}\}, s \in \mathbb{S}$ de paramètres globaux pour chaque simulation $s \in \mathbb{S}$.

4.3. Description du modèle

Nous présentons ici une description détaillée de chaque étape du processus d'observation.

4.3.1. Sélection du modèle

Ce modèle est générique pour l'analyse de flux de données d'une simulation. Il peut être appliqué online directement avec des données en provenance d'une plateforme de simulation (par exemple NetLogo) ou offline avec des données extraites d'un fichier de logs (en simulant un flux de données). Cette généricité permet une adaptation aisée à de nouvelles plateformes sans changer le reste de l'analyse.

4.3.2. Traitement des données

Une matrice de données est générée tous les st pas de simulation par le modèle choisi précédemment. Une ligne dans cette matrice représente l'état d'un agent. Ces données brutes ne sont pas les seules variables intéressantes pour la génération et l'analyse de clusters. A partir de l'ensemble $\mathbb{V}_{ag}$ de variables qui décrivent les agents, nous générons les ensembles $\mathbb{V}calc$ et $\mathbb{V}init$. Dans $\mathbb{V}calc$, plusieurs filtres ou agrégateurs peuvent être considérés pour enrichir le flux de données, tels que des filtres de génération et d'analyse de graphe ou des filtres statistiques. Dans notre exemple, nous

utilisons en particulier un filtre Moyenne Mobile : pour chaque variable de $\mathbb{V}_{ag}$ nous définissons une nouvelle variable qui correspond à sa moyenne mobile.

$\mathbb{V}init$ contient les valeurs initiales de $\mathbb{V}_{ag}$ de chaque agent dans la simulation. Par défaut, ces variables *initiales* ne sont pas utilisées dans le calcul de clusters, mais dans la description a posteriori des clusters. Le sous-ensemble des variables utilisées pour la classification $\mathbb{V}clust$ doit être sélectionné au début de la simulation (il est maintenu constant pour le reste de l'analyse). Par défaut, $\mathbb{V}clust = \mathbb{V}ag \cup \mathbb{V}calc$. Il peut toutefois être intéressant de sélectionner un sous-ensemble plus restreint de variables afin de cibler l'analyse, par exemple analyser ce qui est lié aux variables de performances ou afin de faire un clustering géographique.

4.3.3. Clustering : est-il possible de retrouver des groupes homogènes ?

Le clustering est effectué sur l'ensemble de données $\mathbb{V}clust$ afin de générer des groupes d'agents homogènes. Notre objectif n'est pas de proposer un nouvel algorithme de clustering, n'importe quel algorithme peut être utilisé (dans notre application, n'importe quel algorithme de la librairie Weka [2] et ses paramètres associés). Par défaut, l'algorithme de clustering XMeans (Pelleg, Moore, 2000) est utilisé avec la fonction de similarité classique basée sur la distance euclidienne (voir section 3.1 pour plus de détails sur le clustering et les algorithmes associés).

4.3.4. Description : comment décrire les clusters ?

Une fois que les clusters sont identifiés, et si nous considérons $\mathbb{A}_t^s$ l'ensemble d'agents et $\mathbb{A}_{C_t^s} = extension(C_t^s)$ l'ensemble d'agents dans le cluster C_t^s, il est possible d'obtenir des descriptions claires et précises de chaque cluster en se basant sur le calcul des valeurs-test (VT, voire section 3). La VT représente alors l'importance de la valeur moyenne $E(\mathbb{A}_{C_t^s}, v)$ d'une variable v sur l'ensemble $\mathbb{A}_{C_t^s}$ par rapport à la distribution de v sur l'ensemble $\mathbb{A}_t^s$.

Adapté à nos notation, le calcul des valeurs-test pour une variable quantitative devient :

$$VT(v, C_t^s) = \frac{(E(\mathbb{A}_{C_t^s}, v) - E(\mathbb{A}_t^s, v))}{\sqrt{(\frac{card(\mathbb{A}_t^s) - card(\mathbb{A}_{C_t^s})}{card(\mathbb{A}_t^s) - 1} \times \frac{\sigma^2(\mathbb{A}_t^s, v)}{card(\mathbb{A}_{C_t^s})})}} \quad (5)$$

où $card$ est la fonction de cardinalité qui renvoie la taille d'un ensemble.

v est significative pour l'ensemble $\mathbb{A}_{C_t^s}$ si $|VT(v, C_t^s)| \geq 2$ (Lebart *et al.*, 2006). Si $VT(v, C_t^s) \geq 2$, la valeur de v dans $A_{C_t^s}$ est en moyenne plus grande que sa valeur dans $\mathbb{A}_t^s$ (et plus petite si $VT(v, C_t^s) \leq -2$).

Pour faciliter la comparaison des clusters découverts lors d'une simulation, nous les présentons sous une vue globale qui permet de visualiser les variables les plus

2. http://weka.wikispaces.com/

significatives soit ordonnées pour chaque cluster (voir par exemple la figure 8), soit par ordre alphabetique pour comparer facilement les clusters sur une variable donnée (voir par exemple la figure 3).

4.3.5. Evolution des clusters

Afin de décrire l'évolution des clusters, nous considérons deux hypothèses alternatives : soit l'extension, soit l'intension est fixée pour l'analyse.

Evolution des clusters en fixant leur extension

La première possibilité est de fixer l'extension (la population) d'un cluster, à un pas de simulation donné, $extension(C_t^s) = extension(C_{t'}^s), \forall t' > t$. La description du cluster $C_{t'}^s$ est mise à jour à chaque pas de simulation. Etant donné que les agents du cluster restent les mêmes, $VT(v_{init}, C_{t'}^s) = VT(v_{init}, C_t^s)$ (à moins que certains agents sortent, ou que de nouveaux agents entrent dans la simulation). Pour les autres variables (en particulier les variables dans $\mathbb{V}ag$ et $\mathbb{V}calc$), VT peut évoluer car l'état des agents peut changer.

Description de la stabilité. Pour évaluer la stabilité d'un cluster avec une extension fixe, nous définissons la mesure *Desc-stability* (eq. 7). Elle se base sur le calcul de la *stabilité* du cluster C sur une variable v par rapport à un cluster C', comme suit :

$$\underset{v}{stability}(C, C') = 1 - \frac{|VT(v, C) - VT(v, C')|}{\max(|VT(v, C)|, |VT(v, C')|)} \quad (6)$$

La valeur maximale de stabilité est de 1, ce qui signifie que la valeur de VT de v est la même pour les deux clusters. L'équation 6 retourne des valeurs négatives lorsque les deux valeurs de VT ont des signes différents. *Desc-stability* mesure la stabilité moyenne du cluster initial par rapport à toutes les descriptions avec extension fixe :

$$Desc - stability(C^s) = \frac{\sum_{t' \neq t} \sum_{v \in \mathbb{V}} \underset{v}{stability}(C_t^s, C_{t'}^s)}{card(\mathbb{V}) \times card(\{t'/t' \neq t\})} \quad (7)$$

L'interprétation de cette mesure est assez intuitive : plus la valeur est proche de 1, plus les valeurs de VT sont proche de la valeur d'origine sur toutes les périodes considérées (la description du groupe - les variables qui le caractérisent - est stable). Une valeur de 0 signifie que les variables ne sont plus significatives, alors qu'une valeur de -1 sigifierait que les VT ont changé de signe (les variables significativement supérieures à la moyenne sont passées inférieures et inversement).

Evolution des clusters en fixant leur intension

L'alternative est de fixer l'intension des clusters. A chaque nouvelle étape, la fonction $intension$ est utilisée pour déterminer quels agents affecter à chaque cluster (certains clusters peuvent être vides) et les nouvelles descriptions sont calculées. Pour un cluster C_t^s, pour chaque pas de temps $t' > t$ un nouveau cluster $C_{t'}^s$ est construit où $\forall a \in extension(C_{t'}^s), intension_{\mathbb{C}_t^s}(a \in \mathbb{A}_{t'}^s, X_{\mathbb{V}_{clust}}^{a,t'}) \rightarrow C_{t'}^s$. La description de chaque cluster $C_{t'}^s$ peut être facilement comparée à celle de C_t^s. Etant donné que les intentions des clusters sont les mêmes, la VT des variables utilisées dans le clustering

($\mathbb{V}clust$) sont très enclins à rester les mêmes. Mais les VT des autres variables, en particulier ceux des variables initiales ($\mathbb{V}init$) des agents peuvent évoluer (puisque la population du cluster change). De même, la population du cluster peut évoluer.

Stabilité de la population La mesure *Pop-stability* permet de calculer la proportion de la population qui reste dans l'extension d'un cluster alors que l'intension est fixe :

$$Pop - stability(C^s) = \frac{\sum_{t' \neq t} \frac{card(C_t^s \cap C_{t'}^s)}{max(card(C_t^s), card(C_{t'}^s))}}{card(\{t'/t' \neq t\})} \quad (8)$$

Une valeur de 1 indique que la population du groupe reste la même (la description du groupe en termes de population est donc stable), alors qu'une valeur de 0 indique qu'aucun des agents présents dans le groupe en t n'est plus présent lors des autres périodes considérées.

4.3.6. Génération de simulations

Afin de valider les descriptions des clusters, nous étudions d'une part, leur stabilité à travers différentes simulations, et d'autre part, leur sensibilité à la variation de paramètres globaux. Pour cela, nous définissons deux méthodes de génération et d'analyse de simulations. La première méthode conserve l'intension du cluster et applique simplement la même intension sur les nouvelles simulations. La seconde méthode suppose que les valeurs significatives initiales identifiées pour un cluster définissent un "*modèle d'agent*". Ce modèle est utilisé dans de nouvelles simulations et l'analyse vérifie si les autres variables des agents générés ont un comportement similaire à celui du cluster initial.

4.3.6.1. Génération en fixant la définition du groupe.

La définition d'un groupe d'agents (l'intension du cluster) est maintenue constante. Pour un groupe cible C_t^s, ns nouvelles simulations sont générées. Au pas de temps t de chaque nouvelle simulation s', un nouveau cluster $C_t^{s'}$ est calculé à partir de l'intension cible :
$\forall a \in extension(C_t^{s'}), intension_{\mathbb{C}_t^s}(a \in \mathbb{A}_t^{s'}, X^{a,t}_{\mathbb{V}_{clust}}) \rightarrow C_t^{s'}$. Par exemple, pour un cluster identifié autour d'un prototype P à la période t, ce même prototype est réappliqué lors d'une nouvelle simulation à la même période t (de la nouvelle simulation) afin de comparer les populations appartenant au cluster. La population de ces nouveaux groupes, ainsi que la description des variables non utilisées dans le clustering (en particulier $\mathbb{V}_{init}$) peuvent être comparées au cluster d'origine afin d'évaluer la stabilité des résultats.

4.3.6.2. Génération en fixant le modèle d'agent.

A partir de l'extension d'un cluster cible C_t^s, il est possible de construire une distribution des variables significatives initiales $\mathbb{V}mod^{C_t^s} = \mathbb{V}init \cap \mathbb{V}sig^{C_t^s}$, définissant ainsi un *modèle d'agent* pour C_t^s. Nous émettons l'hypothèse que ces variables suivent des distributions gaussiennes.

Etant donné un cluster C_t^s, son modèle d'agent est défini par la distribution de ses variables significatives initiales comme suit :

$$\{v_{mod} \sim \mathcal{N}(\mu(v_{mod}, C_t^s), \sigma(v_{mod}, C_t^s)), \forall v_{mod} \in \mathbb{V}mod^{C_t^s}\}.$$

où $\mathcal{N}$ est la distribution gaussienne, $\mu(v_{mod}, C_t^s)$ et $\sigma(v_{mod}, C_t^s)$ sont respectivement la moyenne et l'écart type de la variable v_{mod} au pas t de la simulation s pour les agents dans $extension(C_t^s)$ (étant donné que les variables $\mathbb{V}_{init}$ sont fixées, le temps t n'est pas important).

Taille des clusters : nous définissons la *proportion* de la population dans C_t^s par rapport à la taille de l'ensemble de la population en s au pas de temps t comme il suit : $proportion(C_t^s) = \frac{card(C_t^s)}{card(\mathbb{A}_t^s)}$

Génération d'agents : nous générons ns nouvelles simulations. Pour chaque simulation s' un ratio fixe ($proportion(C_t^s)$) d'agents est choisi au hasard et le modèle d'agent est appliqué pour biaiser les valeurs initiales de ces agents. Le biais est appliqué en changeant les valeurs initiales de toutes les variables $v_{mod} \in \mathbb{V}mod^{C_t^s}$ en suivant le modèle de distribution.

Nous appelons $\mathbb{C}_{gen}^{s',C_t^s}$ le cluster défini par l'ensemble des agents biaisées dans la simulation s' au pas de simulation initial suivant le modèle de C_t^s. Une fois ce groupe défini, il est possible de le décrire dans la nouvelle simulation à l'instant t, afin de le comparer avec la description du cluster original C_t^s. Si la description est stable, cela signifie que le modèle d'agent est suffisant pour expliquer la description du cluster.

4.3.7. Analyses de stabilité et de sensibilité aux paramètres globaux

4.3.7.1. Analyse de stabilité

Pour évaluer la stabilité d'un cluster cible $C_t^{s_0}$ par rapport à un cluster équivalent $C_t^{s'}$ dans une nouvelle simulation, nous utilisons la stabilité définie dans l'équation 6. La stabilité d'un cluster $C_t^{s_0}$ sur une variable v sur un ensemble de simulations $\mathbb{S}'$ est donnée par :

$$\underset{v}{stability}(C_t^{s_0}, \mathbb{S}') = \frac{\sum_{s' \in \mathbb{S}', s' \neq s_0} \underset{v}{stability}(C_t^{s_0}, s')}{card(\mathbb{S}') - 1} \quad (9)$$

4.3.7.2. Sensibilité aux paramètres globaux

Dans le but d'étudier la sensibilité d'un cluster donné $C_t^{s_0}$ par rapport à un paramètre global $gp \in \mathbb{GP}_{\mathbb{S}}$, nous régénérons des nouvelles simulations en considérant différentes valeurs de gp et en calculant la stabilité de $C_t^{s_0}$ dans ces nouvelles simulations.

5. Résultats expérimentaux

Pour illustrer notre modèle d'observation, nous présentons quelques expériences effectuées avec notre outil d'analyse. L'outil a été implémenté en Java, utilisant à la fois Weka et l'API de NetLogo. Par souci de généricité, nous avons implémenté des générateurs de flux de données pour NetLogo et pour des fichiers de log au format CSV. Nous présentons les résultats obtenus sur un modèle simple NetLogo et sur l'analyse de fichiers de logs d'une simulation plus complexe.

5.1. Expérimentation sur NetLogo : le modèle Bank Reserves

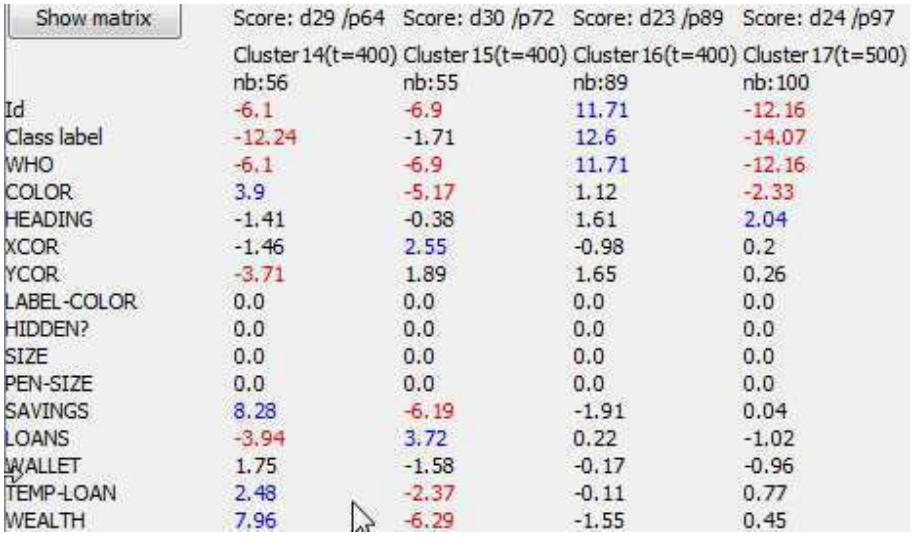
5.1.1. Présentation du modèle

Le modèle *Bank Reserves*, fourni avec Netlogo, est un modèle très simple où lorsque deux agents se rencontrent, un agent tente d'acheter (donner de l'argent) à l'autre. S'il n'a pas d'argent, il tente d'emprunter à la banque. La banque prête de l'argent (et crée de l'argent) tant que le montant total de prêts n'atteint pas le montant total des dépôts ($savings$) multiplié par un paramètre ($1 - Reserves$). Dans ce modèle, des variables globales donnent un aperçu d'une expérience : on peut ainsi observer différents phénomènes, comme par exemple la stabilisation du total d'argent ($money$) lorsque le montant maximal de prêts ($loans$) est atteint, ou encore suivre des groupes fixes d'agents qui dépendent de paramètres globaux (par exemple, 3 groupes : un groupe avec richesse ($wealth$) négative, un avec $wealth > richThreshold$ et le reste). Même si ces informations sont intéressantes, une compréhension plus précise du comportement du modèle ne peut pas être obtenue avec une telle observation globale/locale. Par exemple, il est difficile de savoir *qui sont les agents riches ?* ou encore si *les riches restent riches ?* Ce serait encore plus vrai pour des modèles plus complexes, pour lesquels les variables d'interaction sont beaucoup plus difficiles à déduire que pour une simulation jouet très simple. Pour notre expérience illustrative, on utilise $Reserves = 70$, $People = 200$ et $richThreshold = 20$.

5.1.2. Description des clusters

"Qui sont les riches ?" Nous avons utilisé notre outil d'analyse avec la configuration par défaut : $\mathbb{V}clust = \mathbb{V}ag \cup \mathbb{V}calc$. Le clustering est réalisé tous les $st = 100$ pas de simulation. A chaque étape de clustering, les clusters sont identifiés, visualisés sur NetLogo (avec des couleurs), et leur extension et description sont présentées. Par exemple, dans la figure 3 à $t = 400$, trois groupes sont identifiés : un cluster *pauvre* ($cluster15$, 55 agents, avec une faible richesse $VT(wealth, pauvre) = -6,29$), un cluster *moyen* ($cluster16$, 89 agents) et un cluster *riche* : $cluster14$.

"Quelles sont les caractéristiques des riches à chaque étape ?" A la fin de la simulation, une description de tous les clusters obtenus à chaque étape donne un aperçu global de la simulation (la figure 3 est en fait un extrait de cette fenêtre). Dans notre expérience, il est toujours possible d'identifier un cluster *riche* et un cluster *pauvre*, et parfois (comme dans $t = 400$) un cluster *moyen*. A partir de leur description, il



Show matrix	Score: d29 /p64	Score: d30 /p72	Score: d23 /p89	Score: d24 /p97
	Cluster 14(t=400)	Cluster 15(t=400)	Cluster 16(t=400)	Cluster 17(t=500)
	nb:56	nb:55	nb:89	nb:100
Id	-6.1	-6.9	11.71	-12.16
Class label	-12.24	-1.71	12.6	-14.07
WHO	-6.1	-6.9	11.71	-12.16
COLOR	3.9	-5.17	1.12	-2.33
HEADING	-1.41	-0.38	1.61	2.04
XCOR	-1.46	2.55	-0.98	0.2
YCOR	-3.71	1.89	1.65	0.26
LABEL-COLOR	0.0	0.0	0.0	0.0
HIDDEN?	0.0	0.0	0.0	0.0
SIZE	0.0	0.0	0.0	0.0
PEN-SIZE	0.0	0.0	0.0	0.0
SAVINGS	8.28	-6.19	-1.91	0.04
LOANS	-3.94	3.72	0.22	-1.02
WALLET	1.75	-1.58	-0.17	-0.96
TEMP-LOAN	2.48	-2.37	-0.11	0.77
WEALTH	7.96	-6.29	-1.55	0.45

Figure 3. Vue d'ensemble des clusters. Les trois premiers groupes correspondent à des clusters d'agents identifiés à l'instant $t = 400$. Les valeurs supérieures à 2 sont celles significativement plus élevées pour ce cluster ($VT > 2$) et les valeurs inférieures à -2 celles significativement plus basses ($VT < -2$)

est possible de constater que le lien entre la richesse ($wealth$) et la richesse initiale ($T0Wallet$) n'est plus significatif après $t = 800$ ($|VT(T0Wallet, riche)| < 2$ pour $t \geq 800$). Cela peut être lié au fait (observé avec l'observation globale sur NetLogo) que la banque a atteint sa limite max de prêts (la somme totale s'arrête d'augmenter à environ $t = 400$).

Cependant, comparer des clusters de différentes étapes dans cette vue d'ensemble s'avère difficile car ils sont différents à la fois en intension et en extension (voir la section 4.3.5). Dans un modèle plus complexe, un cluster peut avoir une signification complètement différente à chacune des étapes. Afin de décrire des évolutions, il est donc nécessaire de suivre un cluster, défini en extension ou en intension.

5.1.3. Evolution des clusters

5.1.3.1. Extension fixe : *les riches restent-ils riches ?*

La première possibilité est de fixer l'extension (population) d'un cluster, à un pas t de simulation donné. La description du cluster C est mise à jour à chaque étape. Etant donné que les agents du cluster restent les mêmes, $VT(v_{init}, C_{t'}) = VT(v_{init}, C_t) \forall t' > t$ (à moins que certains agents sortent ou des nouveaux agents rentrent dans la simulation).

La figure 4 décrit le $cluster14$ et son évolution, à la fois en extension et en intension. Sur la partie droite de la fenêtre, pour chaque variable, deux barres représentent

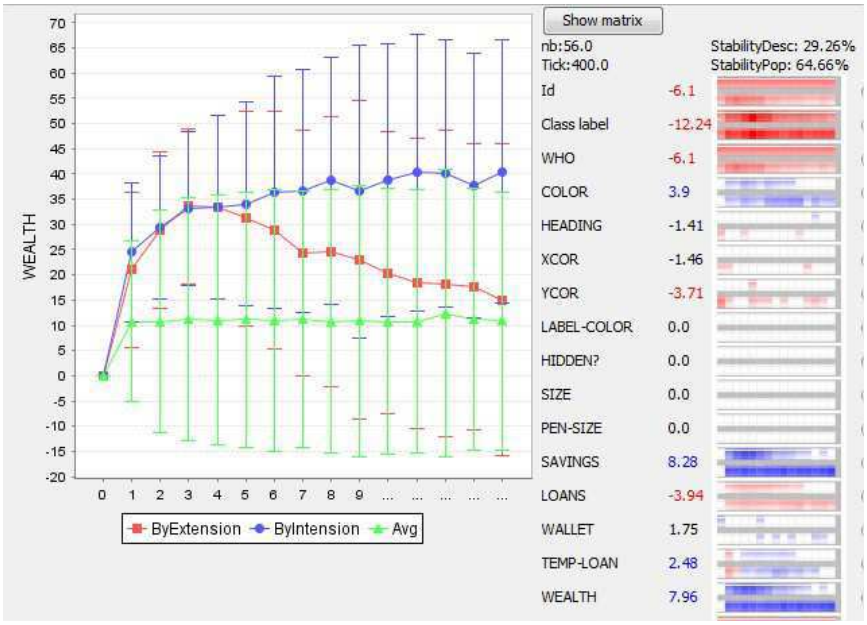


Figure 4. Analyse du $cluster14$: sur la droite, pour chaque variable, son VT à l'instant t = 400, et deux barres qui correspondent à l'évolution de la valeur de VT pour chaque étape avec extension fixe (barre du haut, la population du cluster reste la même) ou intension fixe (barre inférieure, la définition du cluster reste la même). Sur la gauche, on observe l'évolution des valeurs d'une variable sélectionnée ($wealth$) pour toute la population (triangles), pour le cluster fixé en extension (carrés) et en intension (ronds). $t = 1$ dans le graphique correspond à $t = 100$ pas de simulation

la VT de la variable avec extension fixe (barre du haut) ou intension fixe (barre inférieure). La couleur est bleue quand $VT > 2$ et rouge quand $VT < -2$. Par exemple, en extension, nous pouvons observer que pour toutes les variables associées à la richesse ($wealth$, $savings$, $loans$), la différence avec les autres agents diminuent : les valeurs absolues des VT des variables $wealth$, $saving$ et $loans$ diminuent (couleur de plus en plus claire et blanche). Cela signifie que, en moyenne, les gens riches de $t = 400$ sont de moins en moins riches. Aucune de ces variables n'est significativement différente de la moyenne après $t = 1200$. Ceci est représenté à gauche de l'image, où l'on visualise l'évolution de la variable $wealth$: la richesse moyenne de la population est de 10. La richesse moyenne du $cluster14$ en extension est en nette diminution à partir de $t = 400$ ($t = 4$ dans la figure 4 puisque $st = 100$) tout en convergeant vers la moyenne. Cette convergence est également confirmée par le faible score de $Desc - stability$ du cluster (29 %). Même si elles n'apparaissent pas sur l'image, les valeurs initiales des paramètres ($\forall init$) sont par définition stables pour cette évolution.

Tableau 1. Génération de simulations : analyse de la stabilité avec définition de cluster similaire (type : Def) et configuration initiale des agents biaisée (type : Modèle)

Type	Var.	Init	Sim1/2/3	Sim4/5/6	Sim7/8/9
Def	Reserves	70	45	70	95
	Size	56	57/46/55	61/51/42	43/41/46
	TOWallet	3,4	3,0/3,3/2,8	3,1/2,7/1,5	-0,7/2,0/1,9
	Stability		0,87	0,69	0,32
Model	Wealth	7,96	0,1/0,2/-1,6	0,7/0,1/-0,6	0,5/-0,2/1,6
	Stability		-0,05	0,01	0,08

5.1.3.2. Intension fixe : *les riches restent-ils les mêmes ?*

L'alternative est de fixer l'intension (définition) des clusters. La ligne intermédiaire (ronds, graphique de gauche) et les barres inférieures [3] (partie droite) de la figure 4 représentent la description des clusters identifiés à chaque étape avec la fonction d'intension de $t = 400$. Toutes les variables prises en compte dans le clustering ($\mathbb{V}clust$) sont, par définition, à peu près semblables (comme par exemple $wealth$, $savings$, ...), puisque l'intension des clusters est la même. Toutefois, les autres variables peuvent évoluer (dans notre exemple, les paramètres initiaux des agents $\mathbb{V}init$). Par exemple, le $cluster14$ regroupe à chaque étape les agents riches, mais le nombre et les propriétés initiales de ses agents peuvent évoluer. Il est intéressant de voir que le nombre d'agents riches ($card(cluster14, t)$) reste approximativement constant (56, 53, 52, 48, 50, 53, ...). Mais l'évolution des paramètres initiaux confirme l'observation faite avec la vue d'ensemble : la richesse initiale des agents ($TOWallet$) n'est plus significative après $t = 800$.

5.1.4. Stabilité d'un cluster avec définition de groupe fixe

"était-ce un effet du hasard et comment réagit le cluster au paramètre $Reserve$?" Une fois que nous avons identifié un groupe intéressant et sa description (les agents riches du $cluster14$ qui sont initialement riches), il est important de savoir si la description du groupe est stable lorsque la simulation est reproduite avec des paramètres similaires, puis d'étudier l'impact de certains paramètres globaux. Ici, nous pouvons faire l'hypothèse que les gens riches ne peuvent s'enrichir que lorsque la banque est encore capable de faire des prêts. Ainsi, en présence d'une valeur importante du paramètre $Reserves$ le lien entre la richesse initiale et la richesse devrait s'affaiblir. Afin d'étudier à la fois la stabilité et l'effet de paramètres globaux, des nouvelles simulations sont générées automatiquement avec les mêmes valeurs de paramètres globaux,

3. Sur la partie droite du graphique, chaque variable est décrite par sa VT, puis par deux barres : la barre supérieure décrit l'évolution de la VT avec extension fixe, alors que la barre inférieure décrit l'évolution de la VT avec intension fixe

sauf pour le paramètre $Reserves$, qui change de valeur (45, 70 et 95). L'intension du $cluster9$ est fixée et son extension et sa description sont calculées à $t = 400$ pour chaque simulation. Les résultats de l'évaluation de la stabilité sur 9 simulations différentes sont présentés dans le tableau 1 (type Def). Puisque nous avons utilisé toutes les variables, sauf les paramètres initiaux pour le clustering, seulement $\mathbb{V}init$ et la taille du cluster sont intéressants.

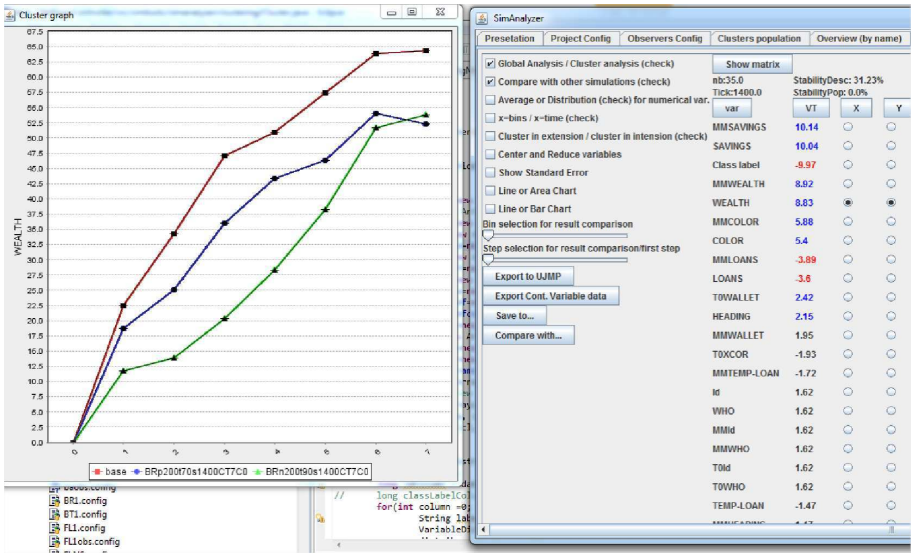


Figure 5. Comparaison de la richesse ($wealth$) du cluster des riches identifié en t=1400 sur 3 simulations avec des valeurs de réserves de la banque de 90 (triangles), 70 (ronds) et 50 (carrés)

Les résultats confirment que les gens riches sont initialement plus riches que la moyenne : $VT(T0Wallet)$ est presque toujours significativement positif ($VT > 2$) pour $Reserve = 45$ et 70. Il n'est cependant pas significatif pour deux des trois simulations avec $Reserve = 95$. Les valeurs de stabilité montrent que notre première simulation avait une valeur de $T0Wallet$ exceptionnelement élevée (stabilité de $TOWallet$ plus importante pour $Reserve = 45$ que pour $Reserve = 70$).

Ce résultat est confirmé si on compare l'évolution de la richesse pour un même cluster de riches sur 3 simulations avec des taux de réserves différents (voir figure 5). Lorsque le taux de réserve est plus élevé (courbe verte), l'accumulation de richesse est limitée par l'interdiction rapide d'emprunter faite par la banque du fait de la limite assez faible de prets total dans l'économie.

5.1.5. Stabilité d'un cluster avec modèle d'agent fixe

"est-ce que la richesse peut-être expliquée uniquement à partir de sa valeur initiale ?" Une autre possibilité pour valider nos résultats est d'essayer de créer des agents à partir des variables significatives initiales et de voir s'ils se comportent comme nous

pensons qu'ils devraient le faire. Par exemple, avec $cluster14$ nous avons identifié un *profil d'agent riches* caractérisé par sa richesse initiale élevée et sa richesse très élevée à $t = 400$. Pour tester la stabilité de ce cluster, nous pouvons changer les paramètres initiaux de certains agents dans une nouvelle population pour les faire correspondre avec les caractéristiques initiales significatives du $cluster14$ et évaluer en $t = 400$ si leur description est semblable à celle du $cluster14$. Un seul paramètre initial est significatif ($T0Wallet$ qui, nous l'espérons, devrait conduire à une richesse élevée en $t = 400$). Une proportion similaire de la population dans chaque nouvelle simulation (56/200 agents) est changée de manière à obtenir une distribution de $T0Wallet$ similaire à celle du $cluster14$. Pour l'analyse de la stabilité, nous avons également modifié le paramètre $Reserve$ afin d'étudier son impact sur le comportement des agents. Contrairement à nos attentes, les résultats (tableau 1(type Model)) ne montrent pas de lien significatif avec les variables de richesse ($wealth$, $savings$, $loans$). Notre hypothèse n'est pas validée : définir un agent avec une richesse initiale légèrement plus élevée ne suffit pas pour obtenir une richesse significativement plus grande à $t = 400$. Des expériences complémentaires montrent que considérer des valeurs élevées du seuil de richesse ($rich-threshold$) (qui impliquent une plus grande diversité des richesses) ou effectuer une observation plus rapide (en $t = 200$ par exemple) sont suffisants pour obtenir un effet de causalité significatif.

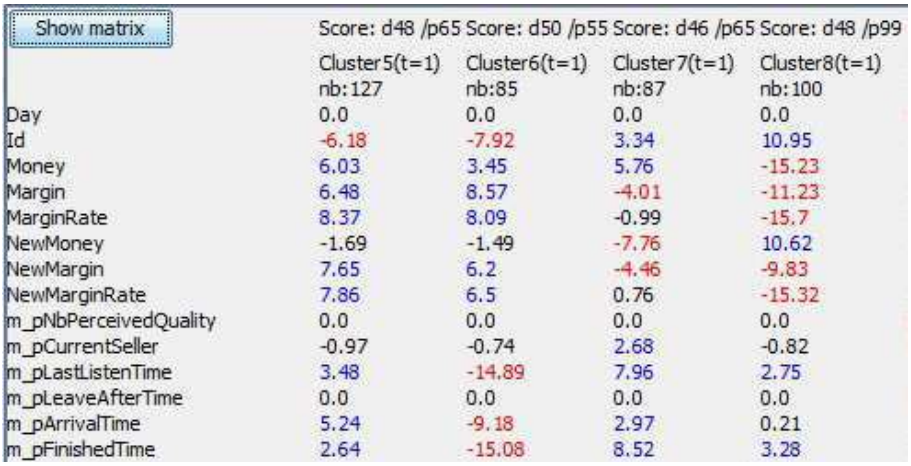
5.2. Exemple d'analyse sur des logs de simulation

5.2.1. Description du modèle

Pour illustrer la gestion de simulations plus complexes par notre modèle d'observation, nous l'avons également appliqué à un modèle qui suit l'approche de modélisation KIDS (Edmonds, Moss, 2004), ce qui signifie que le nombre de paramètres et des variables observées est maintenu élevé pour être plus descriptif et réaliste plutôt que synthétique. La simulation du marché de Rungis (Caillou *et al.*, 2009) a été développée avec le framework BitBang (Baptista *et al.*, 2006). Elle reproduit un marché de gros de fruits et de légumes. Un type d'agent vendeur et 4 types d'agents acheteurs sont considérés dans la simulation (avec des nombreux paramètres, 20 paramètres en moyenne par type d'agent) : les *restaurateurs* qui cherchent à acheter rapidement tout ce dont ils ont besoin, les *TimeFree* qui sont à la recherche de bonnes occasions, les *Barbes* qui cherchent des produits de basse qualité peu chers pour revendre sur leur marché, et les *Neuilly* qui cherchent de la haute qualité. Le fichier csv contient une ligne par couple jour/agent, avec 49 variables de sortie. Ces variables comprennent les marges globale et quotidienne, le nombre de vendeurs visités, les vendeurs habituels, la qualité et la quantité des produits ainsi que les prix. Nous n'analysons ici que les 400 agents acheteurs au cours des 10 premiers jours de la simulation.

Description des clusters

Quatre clusters sont généralement identifiés à chaque étape. Par exemple, au jour 1 de l'analyse, les $cluster5$ à $cluster8$ avec une taille assez homogène (figure 6). Un de ces clusters a la particularité d'être parfaitement stable par rapport à sa popula-



Show matrix	Score: d48 /p65	Score: d50 /p55	Score: d46 /p65	Score: d48 /p99
	Cluster5(t=1) nb:127	Cluster6(t=1) nb:85	Cluster7(t=1) nb:87	Cluster8(t=1) nb:100
Day	0.0	0.0	0.0	0.0
Id	-6.18	-7.92	3.34	10.95
Money	6.03	3.45	5.76	-15.23
Margin	6.48	8.57	-4.01	-11.23
MarginRate	8.37	8.09	-0.99	-15.7
NewMoney	-1.69	-1.49	-7.76	10.62
NewMargin	7.65	6.2	-4.46	-9.83
NewMarginRate	7.86	6.5	0.76	-15.32
m_pNbPerceivedQuality	0.0	0.0	0.0	0.0
m_pCurrentSeller	-0.97	-0.74	2.68	-0.82
m_pLastListenTime	3.48	-14.89	7.96	2.75
m_pLeaveAfterTime	0.0	0.0	0.0	0.0
m_pArrivalTime	5.24	-9.18	2.97	0.21
m_pFinishedTime	2.64	-15.08	8.52	3.28

Figure 6. Vue d'ensemble des clusters après 10 jours de simulation du "Marché de Rungis". dxx et pxx représentent respectivement la description et la stabilité de la population

tion ($Cluster8$, 100 agents, stabilité de la population : 99 %). Lors de l'examen des identifiants des agents, il apparaît que ce groupe contient tous les agents restaurateurs, qui semblent avoir un comportement nettement différent du reste (les 3 autres clusters ont une population mixte). Une de leurs caractéristiques est (par construction) de passer des accords ($TrustAgreements$) avec un unique vendeur pour chaque produit dont ils ont besoin, tandis que les autres agents négocient en général avec plusieurs vendeurs réguliers. La description de ce groupe et son évolution (figure 7) donne un aperçu très instructif sur leur comportement. Le VT et les graphiques d'évolution sont identiques pour l'analyse en intension et en extension, car la population reste la même. La première propriété très visible est la variable $NewMargin$, qui mesure la marge quotidienne. Cette variable a une valeur très basse au début de la simulation (jour 1, $VT = -9,83$), et devient progressivement non significative, puis fortement positive après le jour 6 ($VT > 10$). Ceci est représenté sur l'image par l'évolution rouge-blanc-bleu. Cette évolution de la marge (d'abord inférieure à la moyenne, puis supérieure), peut être expliquée en analysant les autres variables significatives représentées sur le graphique figure 7 : $m_pMarketToursToday$ qui mesure le nombre de fois qu'un agent fait un "tour du marché", ce qui signifie qu'il se balade à la recherche d'un produit qu'il n'a pas trouvé chez ses vendeurs habituels, et $m_pIsInPrefferedList$, correspondant au nombre de vendeurs habituels chez qui l'acheteur se rendra en priorité. Les agents $Restaurateur$ ont un nombre sensiblement inférieur de vendeurs habituels (la probabilité reste stable autour de 0,1 au bout de 2 jours, alors qu'elle est en constante augmentation pour l'ensemble de la population, la VT passe de -6,49 à -17,1), et une probabilité plus élevée de faire un tour du marché (la probabilité reste plus élevée que 0,25, alors qu'elle décroît pour l'ensemble de la population, la VT passe de -0,82 à 14,2). L'interprétation est que la stratégie du restaurateur (quelques

vendeurs habituels avec des prix fixes), est moins efficace au début parce que les premiers prix négociés sont élevés. Mais étant donné qu'il ne visite pas beaucoup de vendeurs (peu de vendeurs habituels), il fera souvent des tours du marché, et trouvera alors des nouvelles opportunités (et des prix mieux négociés), alors que d'autres acheteurs resteront dans leur réseau fermé de vendeurs habituels.

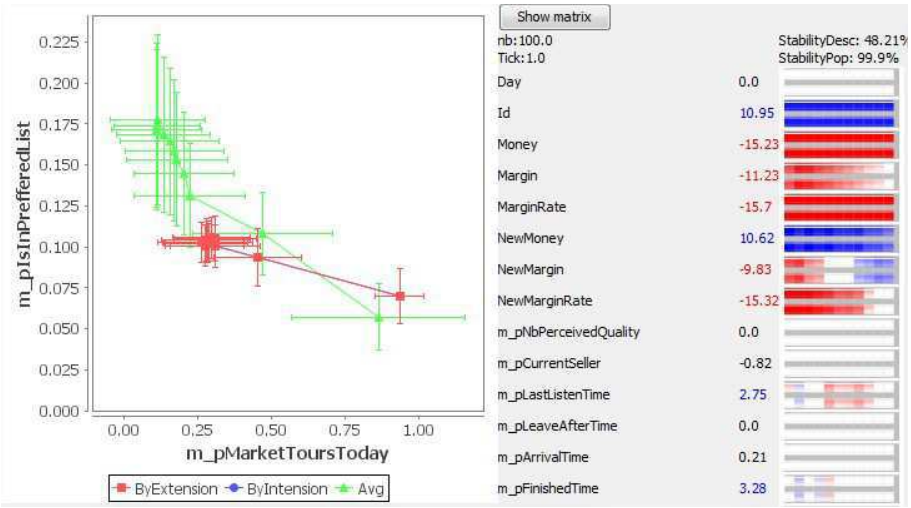


Figure 7. Analyse détaillée du Cluster8. Modèle du marché de Rungis

Cette analyse est validée par une autre expérimentation, réalisée avec 100 agents, dont les résultats au jour 20 (t = 20) sont présentés figure 8. Cette figure permet de plus d'illustrer la description directe des clusters par variable significative ainsi que l'utilisation des VT sur variables qualitatives. La variable $m_pTypeAgent$ est une variable textuelle contenant le type de l'agent. Les agents de type restaurateurs ($BuyerResto$) sont significativement surreprésentés dans le $C4$ qui représente le cluster des restaurateurs décrit précédemment. On retrouve bien les caractéristiques de la simulation précédente : les restaurateurs passent des accords ($TrustAgreements$ est la variable la plus discriminante pour le groupe), et leur stratégie est payante sur le long terme (au jour 20, la variable $NewMargin$ reste significativement supérieure à la moyenne : VT=3,73). Cette analyse a de plus été effectuée en utilisant un filtre de traitement de variables de graphe afin d'analyser les graphes de transaction et de négociation. Ces variables confirment en particulier que les restaurateurs vont négocier avec significativement moins de vendeurs que les autres ($grOutDegreeNbVisitToday$ à -2,43). Par contre, ils n'apparaissent pas comme achetant chez significativement moins de vendeurs. Les autres agents négocient donc avec plus de vendeurs, mais finalement ont autant de fournisseurs que les restaurateurs.

Ce type d'interprétation peut également être faite pour les clusters qui ne correspondent pas à un type d'agent spécifique. Par exemple, le $cluster6$ de la figure 6 qui regroupe 85 agents dont la caractéristique principale est d'arriver tôt sur le marché ($m_pArrivalTime$) et de dégager des marges élevées ($NewMargin$). Une analyse

Figure 8. Vue d'ensemble des clusters après 20 jours de simulation du "Marché de Rungis", par ordre décroissant de variables significatives

plus complète de leur profil (que l'on retrouve dans la nouvelle expérimentation en $Cluster3$ dans la figure 8) et de leur évolution montre que c'est effectivement une bonne stratégie. Ceux qui arrivent tôt et qui possèdent un vaste réseau de vendeurs habituels (valeur élevée de $IsInPrefferedList$) vont négocier beaucoup (nombre élevé de visites par vendeur), mais vont voir moins de vendeurs que la moyenne. Et ils vont maintenir une marge plus élevée que la moyenne. Tout ceci peut être interprété à partir de la représentation de l'évolution du cluster, alors que cela aurait été très difficile à identifier à partir d'une analyse classique de logs.

6. Conclusions

Le modèle d'observation des simulations que nous présentons ici fournit au modélisateur des outils génériques qui lui permettent d'avoir une vision synthétique et descriptive des simulations. Il peut être utilisé pour comprendre la dynamique des simulations et pour faciliter leur validation. Grâce au mécanisme de génération des simulations, des agents créés en tenant compte des caractéristiques d'un cluster cible sont réintroduits dans les simulations, les clusters peuvent être reconstruits et les variables globales de simulation peuvent être comparées à leurs valeurs précédentes. De

cette façon, la stabilité des clusters et leur "expressivité" peuvent être mesurées sur différentes simulations. Pour diminuer la quantité d'information présentée à l'utilisateur et les données traitées, une évolution possible de ce travail consiste à relier les clusters identifiés entre deux périodes afin de ne conserver que les nouveaux clusters apportant une information/caractéristique non présente dans les clusters existants. Cela permettra de réduire grandement le nombre de clusters à suivre pour le programme, mais aussi de suivre les fusions et éclatements de clusters entre les périodes. Enfin, pour permettre l'analyse d'un nombre important de types de simulations à grand nombre d'agents (plusieurs centaines de milliers d'agents, par exemple pour les simulations urbaines), le modèle d'observation est en train d'être adapté. Cette dernière amélioration se fera par l'intégration de notre modèle au moteur OpenMole et/ou à la plateforme GAMA, qui prévoient tous les deux, entre autres facilités, une utilisation quasi transparente de grilles de calcul pour gérer des simulations en parallèle.

Bibliographie

Baptista T., Menezes T., Costa E. (2006). Bitbang: A model and framework for complexity research. In *Eccs 2006*.

Berkhin P. (2006). *A survey of clustering data mining techniques*. Rapport technique. Consulté sur http://dx.doi.org/10.1007/3-540-28349-8_2

Caillou P. (2010). Automated multi-agent simulation generation and validation. In *Prima 2010*, p. 16p. LNCS.

Caillou P., Curchod C., Baptista T. (2009). Simulation of the rungis wholesale market: Lessons on the calibration, validation and usage of a cognitive agent-based simulation. In *Iat*, p. 70-73.

Edmonds B., Moss S. (2004). From kiss to kids - an 'anti-simplistic' modelling approach. In *Mabs*, p. 130-144.

Gaillard F., Kubera Y., Mathieu P., Picault S. (2008). A reverse engineering form for multi agent systems. In *Esaw 2008*, p. 137–153.

Gil-Quijano J., Louail T., Hutzler G. (2010). From biological to urban cells : lessons from three multilevel agent-based models. In *Pracsys 2010*. Kolkota, India, LNCS.

Held P., Kruse R. (2013). Analysis and visualization of dynamic clusterings. In *System sciences (hicss), 2013 46th hawaii international conference on*, p. 1385-1393.

Lebart L., Piron M., Morineau A. (2006). *Statistique exploratoire multidimeensionnelle : visualisation et inférence en fouilles de données*. Fourth edition, Dunod, Paris.

Lin Y.-R., Sun J., Cao N., Liu S. (2010). Contextour: Contextual contour analysis on dynamic multi-relational clustering. In *Proceedings of the siam international conference on data mining, sdm 2010, april 29 - may 1, 2010, columbus, ohio, usa*, p. 418-429.

Lloyd S. P. (1982). Least squares quantization in pcm. In *Ieee transactions on information theory*, vol. 28, p. 129–137.

Morineau A. (1984). Note sur la caractérisation statistique d'une classe et les valeurs-tests. In *Bull. techn. du centre de statistique et d'informatique appliquées*, vol. 2, p. 20–27.

Nakache J., Confais J. (2005). *Approche pragmatique de la classification*. Editions Technip, Paris.

North M. J., Collier N. T., Vos J. R. (2006). Experiences creating three implementations of the repast agent modeling toolkit. In *Acm transactions on modeling and computer simulation*, vol. 16, p. 1–5.

Ohsumi N. (1988). Role of computer graphics in interpretation of clustering results. In D. E. (Ed.), *Recent developments in clustering and data analysis.* Academic Press, Boston.

Pelleg D., Moore A. (2000). Xmeans : Extending k-means with efficient estimation of the number of clusters. In *17th international conference on machine learning*, p. 727–734.

Phan D. (2004). From agent-based computational economics towards cognitive economics. In *Cognitive economics, handbook of computational economics*, p. 371–398. Springer Verlag.

Reuillon R., Chuffart F., Leclaire M., Faure T., Dumoulin N., Hill D. (2010). Declarative task delegation in openmole. In *High performance computing and simulation (hpcs), 2010 international conference on*, p. 55-62.

Rizoiu M., Velcin J., Lallich S. (2012). Structuring typical evolutions using temporal-driven constrained clustering. In *Tools with artificial intelligence (ictai), 2012 ieee 24th international conference on*, vol. 1, p. 610-617.

Spiliopoulou M., Ntoutsi I., Theodoridis Y., Schult R. (2006). Monic: modeling and monitoring cluster transitions. In *Proceedings of the 12th acm sigkdd international conference on knowledge discovery and data mining*, p. 706–711. New York, NY, USA, ACM.

Taillandier P., Drogoul A., Vo D.-A. (2010). Gama: a simulation platform that integrates geographical information data, agent-based modeling and multi-scale control. In *Prima 2010.*

Treuil J.-P., Drogoul A., Zucker J.-D., Bourgine P., Perrier d. (2008). *Modélisation et simulation à base d'agents : exemples commentés, outils informatiques et questions théoriques.* Paris, Dunod.

Wilensky U. (1999).
In *Netlogo. http://ccl.northwestern.edu/netlogo/. center for connected learning and computer-based modeling, northwestern university. evanston, il.*

Xu R., II D. C. W. (2005). Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, vol. 16, n° 3, p. 645-678.