Cohérence et robustesse dans un système multiagent perturbé : application à un système décentralisé de collecte d'information distribué

Quang-Anh Nguyen Vu

Lyon 1

ED InfoMaths

No Ordre : 232 - 2012

École doctorale Informatique et Mathématiques

Cohérence et robustesse dans un système multiagent perturbé : application à un système décentralisé de collecte d'information distribué

THÈSE

présentée et soutenue publiquement le 05 décembre 2012

pour l'obtention du

Doctorat de l'Université Claude Bernard Lyon I

(spécialité informatique)

par

Quang-Anh NGUYEN VU

Composition du jury

Rapporteurs :	Michel Occello, Professeur, Université Pierre-Mendes-France (Grenoble, France) Laurent Vercouter, Professeur, INSA de Rouen (Rouen, France)
Examinateurs :	Marie-Pierre Gleizes, Professeur, Université Paul Sabatier (Toulouse, France) Giovanna Di Marzo Serugendo, Professeur, Université de Genève (Genève, Suisse)
Directeur :	Salima Hassas, Professeur, Université Claude Bernard (Lyon, France)
Encadrants :	Richard Canal, l'institut de la Francophonie pour l'Informatique (Hanoi, Vietnam) Benoit Gaudou, Maître de conférences, Université Paul Sabatier (Toulouse, France) Frédéric Armetta, Maître de conférences, Université Claude Bernard (Lyon, France)

Laboratoire LIRIS

Laboratoire MSI/IFI

UMMISCO, UMI 209

Remerciements

Je tiens à exprimer mes vifs remerciements à :

– Monsieur Michel Occello, Professeur à l'Université Pierre-Mendes-France, et Monsieur Laurent Vercouter, Professeur à INSA de Rouen, pour avoir accepté d'être rapporteurs de ce travail, pour l'honneur qu'ils me font de participer à ce jury.

– Madame Salima Hassas, Professeur à l'Université Lyon 1, qui a accepté d'être ma Directrice Thèse. Le soutien, les conseils précieux et la confiance qu'elle m'a témoigné m'ont permis de mener à bien ce travail.

– Monsieur Richard Canal, directeur de l'Institut de la Francophonie pour l'Informatique (Hanoï, Viet-Nam), mon Codirecteur de thèse, qui m'a beaucoup apporté par ses précieux conseils, ses encouragements et sa gentillesse.

– Monsieur Benoit Gaudou, Maître de conférence à l'Université Toulouse 1, et Monsieur Frédéric Armetta, Maître de conférence à l'Université Lyon 1, mes encadrants de thèse. Ils se sont complétés à merveille. Cette thèse n'aurait pas été possible sans leur aide et leur patience. Je suis extrêmement reconnaissant pour leurs conseils avisés, leur accompagnement, leur soutien, ainsi que le temps qu'ils m'ont accordé durant ces quatre années. Je souhaite adresser mes remerciements sincères à Monsieur Benoit Gaudou pour le temps qu'il a consacré à lire mon manuscrit, à ses corrections, et à ses conseils d'amélioration de ce manuscrit.

Mes remerciements vont aussi à tous les membres de l'équipe GAMA du LIRIS en France et ceux du MSI de l'IFI au Vietnam pour leur aide durant mes séjours au sein de ces équipes. Merci à tous mes collègues pour la bonne ambiance qu'ils ont apporté tout au long de cette thèse.

J'adresse enfin mes profonds remerciements à ma famille, particulièrement à ma femme, ma petite fille, mes parents pour leur soutien, leur compréhension et leur encouragement tout au long de ma vie. Merci beaucoup à tous mes amis, de près ou de loin, qui m'ont aidé et encouragé aux moments opportuns et pour tout le temps précieux que nous avons passé ensemble.

Résumé

L'objectif de cette thèse est de proposer une approche générale améliorant le maintien de la cohérence et de la robustesse dans un système multi-agents (SMA) qui recueille collectivement des informations provenant des sources distribuées où certains sources sont défectueuses (volontairement ou non). Dans ce contexte, les informations recueillies collectivement par le système sont une agrégation progressive (non linéaire) des informations recueillies individuellement par chaque agent. Par conséquent, chaque agent a des informations directes (recueillies par lui-même) et des informations indirectes (obtenues grâce à la communication avec d'autres agents). La cohérence du système est définie par la compatibilité des informations recueillies collectivement sur l'environnement exploré avec l'environnement réel. La robustesse du système est définie par la capacité du système de maintenir la cohérence de ses informations, en dépit de l'existence et de l'augmentation des agents défaillants au sein du système.
Pour assurer la cohérence du système, nous proposons un modèle de confiance appelé TrustSet, permettant aux agents de raisonner eux-mêmes sur les informations collectées et en particulier de calculer la fiabilité de ces informations. Chaque agent maintient un réseau de confiance et un système d'informations locales construit à partir de la confrontation des informations directes (collectées par l'agent) et des informations indirectes (obtenues à partir des agents rencontrés) pour développer une stratégie de communication locale ou globale garantissant la robustesse du système par rapport aux effets de la dissonance des agents. Ensuite, nous construisons un système multi-agents capable de définir dynamiquement ou de faire émerger des stratégie de déplacement ou de communication adaptées à la perturbation. Une démarche d'auto-organisation se base sur une vision systémique dans laquelle nous considérons un couplage structurel entre les deux composantes du système : le sous-système de collecte d'information directe et le sous-système de communication. Ce mécanisme agit comme un guide pour la communication et pour la limitation de la diffusion d'informations dissonantes dans le système, réduisant ainsi son impact sur le processus de la collecte d'information collective.
De nombreuses simulations ont été menées dans le cadre d'une application de cartographie collaborative, afin de montrer l'intérêt de notre approche.

Mots-clés :
Système multi-agents, Auto-organisation, Confiance, Cohérence, Robustesse, TrustNet, Mapping

Abstract

This thesis addresses the issue of maintaining information coherence and its robustness in a multi-agent system, that collectively gathers information from distributed sources and where some sources may be defective (deliberately or not). In this context, the collective information gathered by the system is a progressive (possibly non-linear) aggregation of information collected individually by each agent. Therefore, each agent has direct information, collected by itself, and indirect information, obtained through communication with other agents. System coherence is defined by the compatibility of collected information about the explored environment and its actual information. System robustness is defined by the capability to maintain information coherence, despite the existence and increase of faulty agents within the system.

To ensure the system coherence, this thesis proposes a trust model named TrustSets, allowing agents themselves to reason about collected information to ensure its consistency by using the calculation of the information reliability. Each agent maintains a trust network and can recognize direct (collected from the environment) and indirect (collected by exchanging information with other agents) information, not only in its stored data, but also in the data transmitted by agents it encounters. Then, the agents develop their own local and global communication strategies to ensure the system robustness against the effects of dissonance agents.

To ensure the system robustness, we construct a multi-agent system which brings out dynamically strategies of movement and communication automatically adapted to the perturbation. For this purpose, we propose a self-organizational approach, based on a systemic view in which we consider a structural coupling between two levels: direct information gathering and communication. This mechanism acts as a guide for communicating and for limiting the propagation of dissonant information in the system. Consequently, it reduces the impact of dissonant information on the process of gathering information collectively.

Various experiments were conducted as the part of a collaborative mapping application to show the interest of our approach.

Keywords :

Multi-Agents System, Self-organizing, Trust, Coherence, Robustness, TrustNet, Mapping

Une pensée particulière pour mes parents, ma femme et ma petite fille.

Table des matières

Table des figures

Liste des tableaux

Liste des Algorithmes

Introduction

Chapitre 1

Introduction

Sommaire

Dans ce chapitre, je présente la problématique générale de la thèse. Après avoir présenté les motivations de ce travail de recherche et le contexte scientifique dans lequel il se place à savoir l'étude des systèmes de collecte d'information distribués soumis à des perturbations, j'en précise le cadre théorique et mes contributions. Enfin, je détaille l'organisation du manuscrit.

1.1 Contexte du travail de thèse

1.1.1 Projet AROUND

Cette thèse s'inscrit dans le cadre du projet AROUND (Autonomous Robots for Observation of Urban Networks after Disasters) [Boucher et al., 2009] visant à construire un système complet d'aide à la décision en temps réel pour la gestion des catastrophes naturelles dans les zones urbaines. Le VietNam est le pays cible de ce projet car il est considéré comme l'un des plus exposés aux catastrophes naturelles.

Le projet AROUND se compose de 2 niveaux présentés dans la figure 1.1 :

- Un niveau local avec le déploiement de robots autonomes mobiles pour aider et à plus long terme pour remplacer les secouristes. Ces robots sont censés être capables de s'auto-organiser pour réaliser les diverses activités de sauvetage comme le recueil d'informations sur les sites urbains sinistrés ou établir un réseau pour la communication entre les secouristes humains.
- Un niveau global qui contient un système spatialisé d'aide à la décision (SDSS – Spatial Decision Support System).

Les flux de données entre ces niveaux sont bidirectionnels et peuvent fonctionner en mode entièrement automatique ou de manière semi-dirigée. Les données locales recueillies par les robots sont communiquées, analysées et fusionnées à chaque niveau intermédiaire.

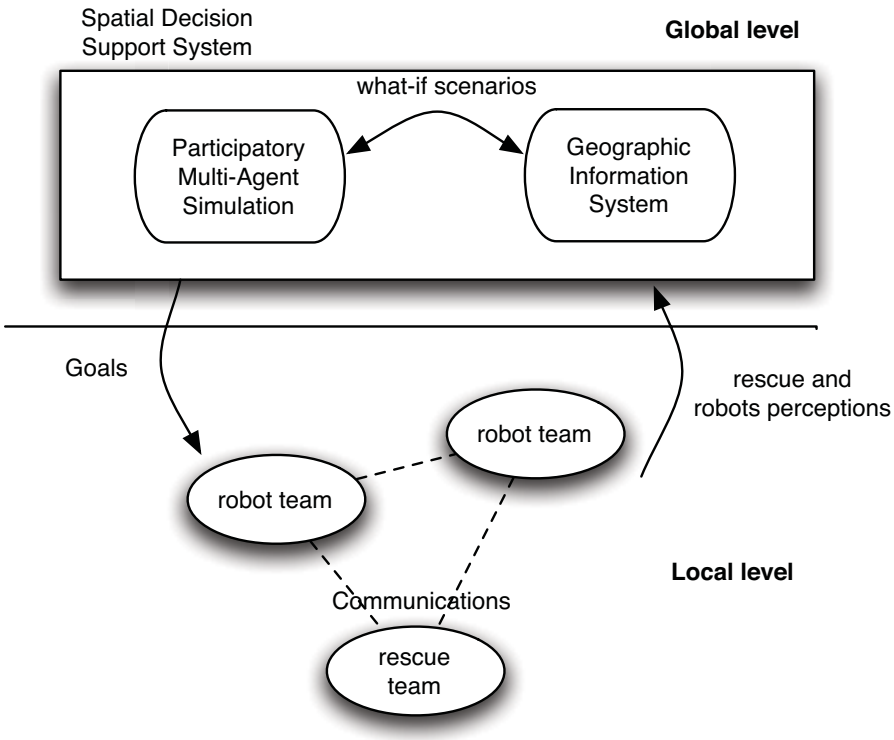


Figure 1.1: L'architecture du projet AROUND
[Boucher et al., 2009]

Le travail de cette thèse est positionné au niveau local où interviennent la communication entre robots. Le recours aux robots est justifié pour minimiser les délais de localisation des victimes, les risques pour les humains et les couts financiers. Après une catastrophe dans un site urbain, la tâche la plus importante est de localiser les victimes dans les plus courts délais possibles. Des robots s'échangent des informations concernant les positions des victimes ou les endroits dangereux. Ils cherchent à localiser les victimes et en même temps cartographient la zone.

1.1.2 Robots défectueux dans une mission de sauvetage

Dans le cadre du projet AROUND, les robots sont censés aider voire même remplacer les humains dans les opérations de sauvetage. Dès lors, les robots sont supposés être autonomes dans la réalisation des différentes tâches qui leur sont assignées. Lorsqu'une catastrophe se produit, il faut minimiser les risques pour le personnel de recherche et de sauvetage, tout en augmentant le taux de survie des victimes, en déployant des équipes de robots pouvant :

- se déplacer ;
- trouver des victimes et déterminer leur état ;
- localiser leurs emplacements ;
- communiquer pour s'échanger des informations ;
- identifier des dangers

Pour pérenniser un tel projet, les robots utilisés doivent être fabriqués localement et être bon marché. Les robots sont connectés les uns avec les autres en utilisant un réseau sans fil. En utilisant des robots bon marché, il faut être conscient que leur fiabilité n'est pas aussi élevé qu'avec plusieurs robots couteux, en particulier au niveau des capteurs. Il semble donc nécessaire de tenir compte du fait qu'un robot puisse être lui-même défectueux ou qu'il doive interagit avec des robots eux-mêmes défectueux. Ce paramètre doit donc intervenir dans la conception du comportement des robots.

1.2 Problématique de recherche

1.2.1 Problématique scientifique

La problématique scientifique s'énonce comme suit :

Comment aboutir à un système multi-agents qui soit capable dynamiquement de développer des organisations (auto-organisation) et des stratégies d'échanges d'information (communications), de déplacement et des comportements permettant de garantir le maintien de la cohérence de l'information globale recueillie par les agents ?

Cette problématique peut être détaillée par les questions sous-jacentes suivantes :

- Comment mettre en place des activités collectives des agents pour la collecte efficace de l'information ? Quels mécanismes de coordination doivent être développés ? Quelles stratégies à développer pour le déplacement ?
- Comment identifier les informations fiables et comment identifier leur source ?
- Comment mesurer le niveau de fiabilité d'une information ou d'une source d'information ?
- Quels sont les moyens de transmission de l'information dans le système et stratégies de communication qui doivent être déployés pour limiter la propagation des informations non fiables dans le système tout en fournissant un certain niveau de diversification des informations échangées ?

1.2.2 Hypothèses

Je considère un système multi-agents (SMA) **décentralisé** (il n'existe pas d'agent central, ni pour le stockage des données, ni pour le contrôle du système) et **clos** (tous les agents sont connus et définis au départ) dans lequel chaque agent vise à obtenir une représentation de son environnement la plus précise possible par la collecte d'informations, soit de manière directe (par capteurs) soit de manière indirecte (en communiquant avec d'autres agents). Je prends pour hypothèse que certains agents perturbent le système, c'est-à-dire transmettent des informations inexactes sur l'environnement, soit parce que leur perception est erronée (capteurs déficients), soit parce que leur intérêt va à l'encontre de la communauté et les pousse à déformer les informations qu'ils communiquent. Je m'intéresse à la mise en place d'un mécanisme de contrôle du comportement des agents afin d'améliorer la cohérence (adéquation entre l'image de l'environnement construite par les agents et l'environnement réel) et la robustesse (réussir à mettre en place des dispositifs permettant d'atteindre ses objectifs malgré les conditions dégradées) dans ce système d'information distribué perturbé.

1.2.3 Problématiques posées pour la modélisation multi-agents

Le système de collecte d'information et de maintien d'une information collective fiable à partir de sources d'informations distribuées dont certaines sont non fiables est représenté par un système multi-agents, dont certains agents sont défaillants. Trois problématiques liés aux SMA sont posées à partir de la problématique scientifique générale :

- Organisation de l'activité du SMA pour la collecte collective d'information en fonction de l'état de l'information collective disponible et de la proximité du but global quand celui-ci est connu. (coordination de l'activité collective).
- Organisation de la communication et des échanges entre agents en fonction de la fiabilité de l'information et de la fiabilité des sources de l'information.
- Coordination des deux organisations de collecte et de communication : la communication guide la collecte tout en étant guidée par elle.

1.3 Objectifs et contributions

L'objectif de cette thèse est de proposer une approche générale pour améliorer la résilience des systèmes complexes et en particulier des systèmes multi-agents décentralisés de collecte d'information qui sont particulièrement vulnérables à l'introduction d'agents mal conçus ou malveillants. Je cherche donc à construire un système multi-agents qui soit capable dynamiquement de développer des organisations (auto-organisation) et des comportements permettant de garantir le maintien de la cohérence de l'information globale recueillie par les agents et la robustesse qui leur permettent d'atteindre leurs objectifs malgré des conditions dégradées.

Pour cela j'ai choisi de doter les agents de capacités leur permettant d'évaluer une valeur de confiance qu'ils ont envers les autres agents. Je considère que la confiance est l'élément essentiel permettant à l'agent de décider avec qui et quoi communiquer, déterminer la fiabilité d'une information, voire même vers quelle positon l'agent se déplacera.

Mettre en place un tel SMA capable de s'auto-organiser en fonction de la confiance se décompose donc en deux sous-objectifs majeurs :

- **Construire un modèle de confiance TrustSet.** Cette phase vise à développer un modèle de confiance fiable et adapté, que j'appelle TrustSet dans la suite. Chaque agent maintient un TrustSet (et un système d'informations locales) construit à partir de la confrontation d'informations directes (collectées par l'agent) et d'informations indirectes (obtenues à partir des agents rencontrés). Il sera utilisé pour développer une stratégie de communication garantissant la robustesse du système par rapport aux effets de la dissonance des agents.
- **Caractériser une démarche d'auto-organisation à base de confiance.** Cette phase permet de définir ou de faire émerger des stratégie de déplacement ou de communication automatiquement adaptées à la perturbation. Elle se base sur une vision systémique dans laquelle je considère un couplage structurel entre les deux composantes du système : une composante de collecte d'information directe et l'autre de communication. Grâce à cette démarche, les agents peuvent optimiser l'exploration de leur environnement, mieux communiquer lors du partage des informations, mieux ajuster ses comportements en fonction de l'objectif.

1.4 Structure du manuscrit

Le manuscrit est divisé en sept chapitres. Je décris ici le contenu de chacun des chapitres.

- **Chapitre 1** : Ce chapitre présente le sujet de la thèse, en commençant par énoncer les problèmes liées au maintien de la cohérence et l'amélioration de la robustesse dans un système multi-agents perturbé. Il présente ensuite les contributions issues de cette thèse.
- **Chapitre 2** : Ce chapitre dresse un état de l'art du paradigme multi-agents en général et des SMA perturbés en particulier, ainsi que des modèles de confiance récents et des travaux sur l'auto-organisation. Cet état de l'art me permet de situer mon approche par rapport à l'ensemble des approches récemment proposées. Une comparaison des modèles de confiance est donnée.

- **Chapitre 3** : Le chapitre 3 décrit l'approche que je propose. Cette approche est composée de deux modèles qui font l'objet d'une description approfondie dans les chapitres 4 et 5 respectivement : un modèle original de confiance et son utilisation dans une approche à base d'auto-organisation.
- **Chapitre 4** : Le chapitre 4 est dédié au modèle de confiance lui-même. Il se compose de trois parties : la représentation des informations collectées, le modèle de confiance proprement dit et sa dynamique liée à l'échange d'informations entre agents et l'impact de la confiance sur la communication.
  - Le modèle TrustSets a été publié et présenté en partie dans la conférence 2010 IEEE International Conference on Computing and Communication Technologies, Research, Innovation, and Vision for the Future (**Best Student Paper Award**, RIVF 2010), ([NguyenVu et al., 2010]) et pour une version complète dans le journal Ambient Intelligence and Humanized Computing en 2012 ([Nguyen Vu et al., 2012a]).
  - La stratégie simple de communication de ce chapitre a été publié et présenté aux Journées Francophones des Systèmes Multi-Agents (JFSMA 2009) ([NguyenVu et al., 2009]).
  - L'approche de la formation de coalitions d'agents fiables de ce chapitre a été publié et présenté au workshop Principes and pratice of Multi-agent systems en 2012 ([Nguyen Vu et al., 2012b]).
- **Chapitre 5** : Ce chapitre détaille mon approche auto-organisationnelle basée sur la confiance. Je présente en détails le système de collecte directe d'informations, celui de communication, l'interaction entre la communication et le déplacement et son contrôle.
  - Ce chapitre a été publié en partie et présenté à la conférence 2011 IEEE International Conference on Self-Adaptive and Self-Organizing Systems (SASO 2011) ([NguyenVu et al., 2011]).
- **Chapitre 6** : Ce chapitre est dédié à l'évaluation de l'approche proposée. Je décris le simulateur multi-robots que j'ai développé, les paramètres utilisés, les mesures d'évaluation et les résultats obtenus.
- **Chapitre 7** : Dans ce chapitre, je donne les conclusions sur l'approche proposée. Je discute également des perspectives à court et à long-terme.

Enfin en annexe, je décris la présentation de la plate-forme de simulation GAMA et l'implémentation de l'approche que je propose sur cette plate-forme.

Première partie

État de l'art

Chapitre 2

État de l'art

Sommaire

Ce chapitre dresse un panorama des approches dédiées au maintien de la cohérence dans un système de collecte d'information perturbé et à l'amélioration de sa robustesse.

Dans la première partie, je rappelle diverses définitions à propos des systèmes et des systèmes complexes. Je présente également les thèmes de la perturbation, de la cohérence et de la robustesse dans un système complexe.

Dans la deuxième partie, je présente les systèmes d'information (et en particulier perturbés) comme des systèmes complexes. Ce type de système sera en effet le contexte de toute la thèse.

Les systèmes multi-agents (SMAs) sont introduits ensuite dans la troisième partie comme un outil de modélisation des systèmes de collecte d'informations. Pour aborder mon problème du maintien de la cohérence et de l'amélioration de la robustesse dans les systèmes de collecte d'information perturbés, je m'intéresse donc au traitement du maintien de la cohérence et de la robustesse dans les SMA.

Enfin, dans les dernières parties, je définis la notion de confiance en présentant différents modèles de confiance existants. Je considère en effet que la confiance est un outil idéal pour assurer la cohérence dans les systèmes qui m'intéressent. Je présente ensuite une analyse détaillée des approches s'intéressant à la robustesse de tels systèmes.

En conclusion, je présente les bases de l'approche que je vais présenter dans la suite, à savoir l'utilisation d'une approche auto-organisationnelle basée sur la confiance.

2.1 Introduction

Les systèmes distribués mettent en jeu de nombreuses entités (logicielles ou matérielles) qui sont souvent autonomes en terme de contrôle. Elles doivent effectuer diverses tâches complexes telles que : action, calcul, communication, etc. Chaque entité n'a qu'une vue locale, possède généralement un but individuel et est incapable de résoudre seule la tâche dévolue au système global. Dans beaucoup d'applications, il est extrêmement difficile d'avoir une vision globale du système, de vouloir prévoir toutes les situations pouvant survenir durant le fonctionnement du système. La dynamique du système provient des créations, modifications des entités toutes décidées de manière autonome, ainsi que de la pression exercée par l'environnement dans lequel le système est immergé. Parmi ces situations, certaines sont qualifiées de perturbatrices car elles dégradent l'activité collective de la société. Elles se produisent fréquemment lorsque des entités sont (ou peuvent devenir) inopérantes ou « mal opérantes », soit parce que leurs unités de perception sont déphasées, parce que leur comportement se détériore, ou encore parce que leur intérêt va à l'encontre de celui de la communauté. Il faut donc leur donner la possibilité de s'adapter aux imprévus de manière autonome.

Ma thèse s'est concentrée sur les moyens d'assurer la cohérence et d'améliorer la robustesse d'un système de collecte d'informations réparti (représenté par un système multi-agents : SMA) dont les éléments d'information sont collectés par un grand nombre d'agents dont les activités de perception peuvent être perturbées (soit volontairement soit involontairement). Un tel système sera dit cohérent lorsque la perception du monde par ses agents sera correcte et partagée par l'ensemble des agents. Il sera dit robuste s'il réussit à s'adapter et à mettre en place des dispositifs qui lui permettent d'atteindre ses objectifs malgré les conditions dégradées. Pour atteindre ces objectifs, je m'attacherai à définir et mesurer les interactions possibles entre confiance, réputation et information. L'objectif sera d'améliorer le système de communication en utilisant la confiance inter agents en vue de faire émerger l'information la plus fiable possible et la plus proche de la réalité objective. Mes travaux portent également sur les moyens que les agents doivent mettre en œuvre de manière à rendre le système robuste et à améliorer progressivement sa robustesse grâce à une démarche auto-organisationnelle.

Ce chapitre s'organise en cinq parties : après avoir présenté les systèmes complexes et la perturbation dans ces systèmes, je présenterai les systèmes de collecte d'informations perturbés qui est le contexte de mon travail de thèse. J'étudierai ensuite la modélisation d'un système complexe à l'aide du paradigme multi-agents et introduirai la perturbation dans un SMA. Je présenterai ensuite une définition de la confiance, des principaux modèles simples de confiance existants et des modèles basés sur les réseaux avant de préciser le lien entre la confiance et la cohérence. Ce chapitre se termine par la mise en évidence des liens existant entre l'auto-organisation et la robustesse du système.

2.2 Introduction aux systèmes complexes

L'objectif de cette section est d'introduire les notions de système (Section 2.2.1), de système complexe (Section 2.2.2) et de perturbation d'un système complexe (Section 2.2.3) qui me permettront de cerner plus précisément l'importance du maintien de la cohérence (Section 2.2.4) et de la robustesse (Section 2.2.5) dans un système complexe.

2.2.1 Définition d'un système

Avant de parler de **système complexe**, examinons tout d'abord les caractéristiques et les propriétés d'un **système**. Le mot **système** derive du grec « sustêma » qui signifie « *ensemble cohérent* » [Le Moigne, 1977, Weinberg, 2001]. La définition la plus courante du mot système donnée par Bertalanffy est la suivante : *« Un système est un complexe d'éléments en interaction »* [Bertalanffy and Ludwig, 1973]. Cette définition est trop générale, elle ne permet pas de fournir un cadre rigoureux pour ce concept. Une seconde définition relativement répandue est donnée par Joël de Rosnay : *« Un système est un ensemble d'éléments en interaction dynamique, organisé en fonction d'un but »* [Rosnay, 1975]. Cette définition introduit la notion de finalité que tout système poursuit. On parle d'*utilité* pour les systèmes mécaniques, de *finalité* pour les systèmes vivants et composés d'humains ou de *projet* pour les systèmes humains ou

conçus par l'homme pour désigner cette notion de but. Bien que plus complète, cette définition est encore trop générale.

Afin de décrire plus en détail ce qu'est un système, examinons les décompositions aux niveaux structurel et fonctionnel proposées par les deux systématiciens Joël de Rosnay [Rosnay, 1975] et Daniel Durand [Durand, 2002]. D'après [Rosnay, 1975, Durand, 2002] :

Au niveau structurel, un système comprend quatre composants :

> « *1.* ***les éléments constitutifs*** *: on peut en évaluer le nombre et la nature (même si ce n'est qu'approximativement). Ces éléments sont plus ou moins homogènes (ex. automobile : groupe motopropulseur, châssis, habitacle, liaison au sol, carrosserie). Dans une entreprise commerciale, les éléments sont hétérogènes (capitaux, bâtiments, personnel,...).*
>
> *2.* ***une limite (ou frontière) qui sépare la totalité des éléments de son environnement*** *: cette limite est toujours plus ou moins perméable et constitue une interface avec le milieu extérieur. C'est par exemple, la membrane d'une cellule, la peau du corps, la carrosserie d'une voiture. La limite d'un système peut être plus floue, ou particulièrement mouvante, comme dans le cas d'un groupe social.*
>
> *3.* ***des réseaux de relations*** *: les éléments sont en effet inter-reliés. Nous avons vu que, plus les interrelations sont nombreuses, plus le degré d'organisation est élevé et plus grande est la complexité. Les relations peuvent être de toutes sortes. Les deux principaux types de relations sont les transports et les communications. En fait, ces deux types peuvent se réduire à un seul, puisque communiquer c'est transporter de l'information, et transporter sert à communiquer (faire circuler) des matériaux, de l'énergie ou de l'information.*
>
> *4.* ***des stocks (ou réservoirs)*** *où sont entreposés les matériaux, l'énergie ou l'information constituant les ressources du systèmes qui doivent être transmises ou réceptionnées.* »

Sous son aspect fonctionnel, les auteurs distinguent quatre composants :

> « *1.* ***des flux*** *de matériaux, d'énergie ou d'informations, qui empruntent les réseaux de relations et transitent par les stocks. Ils fonctionnent par entrées/sorties (ou inputs/outputs) avec l'environnement ;*
>
> *2.* ***des centres de décision*** *qui organisent les réseaux de relations, c'est-à-dire coordonnent les flux et gèrent les stocks ;*
>
> *3.* ***des boucles de rétroaction*** *qui servent à informer, à l'entrée des flux, sur leur sortie, de façon à permettre aux centres de décision de connaître plus rapidement l'état général du système ;*
>
> *4.* ***des ajustements*** *réalisés par les centres de décisions en fonction des boucles de rétroaction et de délais de réponse (correspondant au temps que mettent les informations «montantes» pour être traitées et au temps supplémentaire que mettent les informations «descendantes» pour se transformer en actions).* »

Un système avion, par exemple, est considéré sous deux aspects comme dans l'exemple ci-dessous :

Exemple 1. *Un avion sous l'aspect structurel est l'ensemble des constituants en relation (moteur, aile, gouverne, etc.) et sous l'aspect fonctionnel est l'ensemble des fonctions en relation (propulser, porter, diriger, etc.).*

L'interaction (ou interrelation) est un des concepts fondamentaux d'un système. Ce concept est lié à l'échange d'informations ou d'énergie entre éléments du système. La nature du système dépend fortement de ces interactions. Les systèmes complexes présentés plus en détails dans la partie suivante se caractérisent en particulier par des interactions non-linéaires entre les éléments qui les composent.

2.2.2 Les systèmes complexes

Les systèmes complexes sont des systèmes dynamiques qui exhibent des comportements chaotiques et/ou des interactions non linéaires [Beurier, 2007]. Un système dynamique linéaire est un système qui évolue au cours du temps à la fois de façon causale (c'est-à-dire que son avenir ne dépend que de phénomènes du passé ou du présent) et déterministe (c'est-à-dire qu'à une condition initiale donnée va correspondre

à chaque instant ultérieur un et un seul état possible). Le comportement d'un système non-linéaire ne peut être déduit du comportement de ses composants :

> « *la caractéristique clé des systèmes non-linéaires est que leurs comportements primaires intéressants sont des propriétés de l'interaction entre les composants, plutôt que des propriétés des composants eux-mêmes* » [Langton, 1997]

Le domaine des systèmes complexes est un domaine qui étudie comment les relations entre les parties donnent lieu aux comportements collectifs d'un système et comment le système interagit et forme des relations avec son environnement. Bien que les systèmes complexes ont été étudié depuis des milliers d'années, l'étude scientifique moderne des systèmes complexes est vraiment jeune par rapport aux sciences conventionnelles telle que la physique et la chimie (son origine est associée à la création du Santa Fe Institut aux États Unis dans les années 1980).

Un système est dit complexe,

> "*S'il est constitué d'une grande variété de composants ou d'éléments possédant des fonctions spécialisées. Les éléments sont organisés en niveaux hiérarchiques internes (par exemple, dans le corps humain : les cellules, les organes, les systèmes d'organes). Les différents niveaux et éléments individuels sont reliés par une grande variété de liaisons. Il en résulte une haute densité d'interconnexions. Les interactions entre les éléments d'un système complexe sont d'un type particulier. On dit que ces interactions sont non linéaires*"[Rosnay, 1975]

[Beurier, 2007] a proposé une définition générale (inspiré du travail de [Bar-Yam, 1997]), à défaut d'être exhaustive, des systèmes complexes. Il décompose un système complexe en plusieurs composants :

> « *1.* ***les éléments, constituants ou composants*** *(et leur nombre) : un système complexe est composé de constituants que l'on peut isoler à plusieurs niveaux d'échelle,*
>
> *2.* ***les interactions ou interrelations*** *(ainsi que leur nature et leur force) : les constituants des systèmes complexes sont en interaction (linéaire et non-linéaire),*
>
> *3.* ***l'activité*** *(et son but) : les constituants du système ont un comportement observable (qui peut être déterministe),*
>
> *4.* ***la fonction ou comportement global*** *(et son rapport au temps) : le système dans sa globalité possède un comportement observable mais généralement non déductible à cause de dynamiques non-linéaires sous-jacentes,*
>
> *5.* ***l'environnement*** *(et ses échanges avec le système) : le système est plongé dans un environnement qui peut être un support d'interaction pour ses constituants* ».

Ces caractéristiques permettant d'identifier nombre de systèmes complexes naturels ou artificiels avec des entités ou interactions très diverses. Quelques exemples de systèmes complexes sont donnés dans le tableau Figure 2.1. Cette diversité des systèmes complexes amène d'ailleurs l'apparition de travaux de recherches dans de très nombreuses disciplines.

Système	Eléments	Interactions	Comportement	Activité
Protéines	Acides aminés	Liaisons	Pliage de protéine	Activité enzymatique
Système Nerveux	Neurones	Synapses	Apprentissage	Pensée
Internet	Ordinateurs	TCP/UDP	Apprentissage ?	Client/Server
			KBase universelle	Peer2Peer
Le vivant	Organismes	Reproduction	Evolution	Survie
		Compétition		
		Prédation		
Société humaine	Etres Humains	Communication	Evolution ?	Survie ?

FIGURE 2.1 – Exemples de systèmes complexes [Beurier, 2007]

Une approche que l'on peut adopter pour l'analyse des systèmes complexes consiste à observer les relations entre le comportement global et les parties qui le composent. Dans un système complexe, on peut distinguer au moins deux niveaux :

« – un niveau micro, représentant le niveau des composants, avec des propriétés locales à chacun d'eux ;

– un niveau macro, représentant l'ensemble du système, avec des propriétés nouvelles, que l'on ne retrouve dans aucun des composants pris individuellement. On parle alors d'émergence de nouvelles propriétés » [Hassas, 2003]

On pourrait aborder cette thématique par le concept d'auto-organisation et d'émergence.

Notation d'émergence.

« Un système exhibe de l'émergence quand il y a des émergents cohérent au niveau macro qui apparaissent dynamiquement des interactions entre les parties au niveau micro. De tels émergents sont nouveaux par rapport aux parties du système prises individuellement » [De Wolf and Holvoet, 2004]

Le mot « émergence » utilisé dans cette définition désigne en fait le résultat du processus d'émergence : comportements, structures, propriétés, etc. Le niveau macro considère le système dans son intégralité, le niveau micro considère le système du point vue des différentes entités qui le composent.

Notation d'auto-organisation. Une définition intuitive et littérale de l'auto-organisation est proposée par Dempster, dans [Dempster, 1998] :

« L'auto-organisation réfère à ce qui est exactement suggéré : des systèmes qui s'organisent sans direction, manipulation ou contrôle externes »

Dans le cadre de mes travaux, je me suis intéressé à un cas particulier de système complexe, un système de collecte d'information ainsi qu'aux moyens d'assurer, au niveau microscopique, la cohérence et la robustesse du système en tenant compte du fait qu'il peut présenter un comportement imprévisible face à des perturbations inattendues de son environnement. Dans la section suivante, je m'intéresse plus particulièrement à la notion de perturbation dans les systèmes complexes.

2.2.3 Perturbation dans un système complexe

Qu'est-ce qu'une perturbation ? Le terme « perturbation » est défini de plusieurs façons dans la littérature :

- Perturbation (français) : irrégularité dans le fonctionnement d'un système (Le nouveau Petit Robert 1, édition 2008)
- Perturbation (anglais) : a small change in the quality, behaviour or movement of something (Oxford Dictionary, 2000)
- Disturbance (anglais) : actions that upset the normal state that something is in (Oxford Dictionary, 2000)

La perturbation étudiée dans ma thèse se rattache à la première définition. Elle représente les irrégularités dans le fonctionnement d'un système qui peuvent provenir d'un environnement inadapté ou partiellement défaillant qui modifie l'état interne du système.

Perturbation dans un système complexe. Pour illustrer les effets d'une perturbation dans un système complexe, je prends comme exemple de système complexe, le web (et notamment d'un point vue hardware). De nombreux chercheurs ont montré que ces systèmes peuvent présenter un comportement imprévisible face à des perturbations inattendues de leur environnement.

Une implication bien connue de couplage dans les systèmes complexes est l'effet de papillon [Lorenz, 1995] : une petite perturbation dans le système peut entrainer de grands changements globaux. Ces effet s'applique parfaitement à notre exemple. Selon Steven Gribble, dans [Gribble, 2001a], ces perturbations peuvent être petites, mais à cause de défauts dans la conception du système combiné au couplage généralisé entre ses composantes, les effets de petites perturbations peuvent être importantes et destructrices et même rendre le système inopérant. Dans la même idée, [Havlin et al., 2010] a montré qu'en raison du couplage fort entre les composants dans les systèmes complexes, un échec dans un ou plusieurs composants peut conduire à des défaillances en cascade qui peuvent avoir des conséquences catastrophiques sur

le fonctionnement du système. Selon [Geard et al., 2005], les systèmes dans le monde réel fonctionnent dans des environnements sujets à du "bruit" et sont soumis à des perturbations à partir d'une grande variété de sources internes et externes.

Par exemple, dans [Floyd and Jacobson, 1994], Floyd et Jacobson ont démontré que les signaux périodiques (tels que les diffusions d'un routeur) dans l'Internet ont tendance à devenir brusquement synchronisés, ce qui conduit à des schémas de perte et de retards. Parailleurs, dans [Druschel and Banga, 1996], Druschel et Banga ont démontré que dans les serveurs Web fonctionnant sur des systèmes d'exploitation traditionnels, une légère augmentation de la charge au-delà de la capacité du serveur peut conduire le serveur dans un état persistant de livelock, réduisant considérablement son débit efficace. Comme troisième exemple de perturbation, dans [Arpaci-dusseau et al., 1999], Arpaci-Dusseau et al. démontrent qu'avec des architectures logicielles classiques, la différence de performance résultant du déplacement des données sur les pistes intérieures par rapport aux pistes extérieures d'un seul disque peut affecter le rendement global d'un cluster de huit nœuds de workstations d'un coefficient pouvant aller jusqu'à 50%. Selon [Gribble, 2001a], les grands systèmes fonctionnent naturellement grâce aux interactions complexes entre de nombreux composants. Ces interactions conduisent à un couplage généralisé des éléments du système ; ce couplage peut être fort (par exemple, les paquets envoyés entre les routeurs adjacents dans un réseau) ou subtil (par exemple, la synchronisation de routage annonces dans un réseau étendu).

2.2.4 Cohérence d'un système complexe

Notion de cohérence. La cohérence, selon [Goldstein, 1999, Heylighen, 1999], fait référence à une corrélation logique entre les parties, impliquant une absence de contradiction entre elles. La cohérence recouvre et corrèle les composants séparés du niveau micro en une unité au niveau macro. Les corrélations entre les composants sont nécessaires pour atteindre un tout cohérent.

Cohérence dans un système. Selon [Pontisso, 2009], la cohérence des systèmes distribués est gérée d'un point de vue logique, par exemple en se concentrant sur la préservation d'un ordre causal ou total entre les opérations du système. En informatique, la cohérence est la capacité pour un système à refléter sur la copie d'une donnée les modifications intervenues sur d'autre copies de cette donnée. Cette notion est principalement utilisée dans trois domaines informatiques : les systèmes de fichiers, les base de données, et les mémoires partagées. La cohérence dans les systèmes d'information sera détaillée dans la section 2.3.3.

2.2.5 Robustesse d'un système complexe

Notion de robustesse. Les systèmes complexes peuvent fonctionner dans un environnement où certaines caractéristiques de performance du système se dégradent en raison de circonstances imprévisibles, telles que des pannes soudaines de machines, une charge supérieure à celle prévu ou des inexactitudes dans l'estimation des paramètres du système. Une question importante se pose lorsqu'on conçoit un système : dans quelle mesure, à partir de circonstances imprévues qui provoqueraient une performance dégradée du système, le système peut limiter cette dégradation ? Autrement dit, à quel point le système est-il robuste ? Avant de répondre à cette question, on doit définir clairement la robustesse.

La robustesse a été définie de différentes façons par différents chercheurs. Selon [Jensen, 2001], la robustesse est la capacité d'un système à fonctionner correctement en présence d'entrées différentes. Dans un sens plus général, [Gribble, 2001b] montre qu'un système robuste continue de fonctionner correctement sur un large éventail de conditions opérationnelles. La robustesse, selon [Castanet R., 2003] et [Saad Khorchef et al., 2006], visent à garantir le maintien de certaines caractéristiques du système souhaités en dépit des fluctuations dans le comportement de ses composants ou de son environnement. Le concept de robustesse, tel qu'il est utilisé dans ma thèse, est similaire à celui de [Castanet R., 2003] et [Saad Khorchef et al., 2006].

Système robuste vs. système résilient. Alors que les systèmes ont souvent été conçus pour être robuste (conçus pour prévenir les échecs), un système résilient accepte que l'échec est inévitable et se concentre sur la découverte précoce et la récupération rapide de l'échec. Dans mon travail, je me focalise

sur l'étude des mécanismes pour prévenir l'échec du système (système robuste) plutôt que se concentrer sur la récupération rapide de l'échec (système résilient).

La figure 2.2 montre un exemple du système robuste dans un environnement très perturbé (avec beaucoup des perturbations qui proviennent d'erreurs de conception, de défaillances des logicielles, d'attaques malveillantes, d'erreurs humaines, de transistors dégradées, etc.) qui peut produire une sortie acceptable.

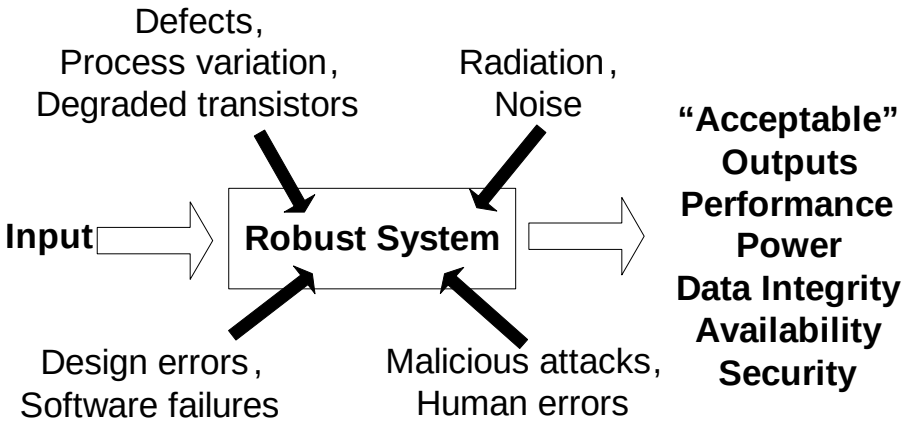


FIGURE 2.2 – Description d'un système robuste ([Mitra, 2006])

2.3 Système d'information (SI)

Dans cette partie, je présente tout d'abord la définition de système d'information (Section 2.3.1) ; la perturbation dans un système d'information est présentée ensuite en Section 2.3.2 et finalement je présente la cohérence (Section 2.3.3) et la robustesse (Section 2.3.4) dans un système d'information.

2.3.1 La notion de système d'information

L'information d'après Robert Reix [Reix, 1999] est un "*élément de connaissance susceptible d'être représenté à l'aide de conventions pour être traité, conservé, communiqué*" et un systèmes d'information (SI) désigne un "*ensemble organisé de ressources (matériel, logiciel, personnel, données, procédures...) permettant d'acquérir, de stocker, de communiquer des informations sous formes de données, textes, images, sons...dans des organisations*".

Je m'intéresse de ma thèse à des systèmes d'information ayant des sources d'informations distribuées. Les informations recueillies collectivement par le système est dans une agrégation progressive (non linéaire) des informations recueillies individuellement par chaque élément. Par conséquent, chaque élément a des informations directes (recueillies par lui-même) et des informations indirectes (obtenues grâce à la communication). Un système de collecte d'information se distingue des systèmes de recherche d'information (SRI) qui sont des systèmes permettant de retrouver dans une collection de documents ceux qui sont pertinents par rapport à une requête d'un utilisateur.

2.3.2 Perturbation dans un système d'information

Un système de collecte d'information est perturbé si des composants du système perturbent système par de fausses informations : soit la collecte des informations directes par des capteurs défectueux, soit les informations sont modifiées (volontairement ou non) dans l'échange et la propagation de ces informations dans l'environnement.

Rappelons que l'objectif de mon travail est d'étudier les moyens pour assurer la cohérence et la robustesse dans un système de collecte d'information perturbé. J'examine donc la définition de la cohérence et de la robustesse dans un système de collecte d'information dans les parties suivantes.

2.3.3 Cohérence d'un système d'information

La cohérence est une des qualités applicables à un SI car elle s'intéresse à la nature des relations entre les informations qui le composent. Il est usuel de distinguer deux aspects dans un SI : la base de données et les opérations sur les données. On ne s'intéresse ici qu'au problème de cohérence dans la base de données du système. Selon [Balter, 1985], « *Le maintien de la cohérence désigne l'ensemble des techniques mises en œuvre pour assurer la validité du système d'information par rapport au monde réel qu'il modélise* ». Balter a montré aussi que le terme générique « *maintien de la cohérence* » recouvre plusieurs aspects complémentaires.

> *« 1. La protection des informations sensibles ou confidentielles des accès non autorisés. C'est le domaine de la sécurité.*
>
> *2. Les informations enregistrées dans le système doivent être conformes à la sémantique des objets représentés. Les textes de conformité s'expriment à l'aide de conditions que les données doivent vérifier en permanence. C'est le domaine des contraintes d'intégrité.*
>
> *3. Le dernier point concerne les incohérences engendrées par des incidents non prévisibles tels que les erreurs, les défaillances ou l'accès incontrôlé à des informations partagées »* [Balter, 1985]

C'est ce dernier aspect du contrôle de la cohérence du système d'information qui a fait l'objet de mes recherches sur la cohérence et sur son maintien dans un SI (et en particulier un système de collecte d'information).

2.3.4 Robustesse d'un système d'information

La robustesse est une des qualités applicables à un SI, elle s'intéresse à la capacité du système à continuer à fonctionner malgré des conditions extrêmes ou non prévues. La robustesse donne principalement la capacité au système d'information d'atteindre ses objectifs malgré la présence de perturbations. Selon [Ali et al., 2003], la robustesse est considérée comme la capacité du système à fonctionner correctement en présence d'entrées différentes de celles prévues. Elle y garantit le maintien des caractéristiques souhaitées du système en dépit des fluctuations dans le comportement des composants du système ou de son environnement.

Une approche mathématique développée dans [Burgin, 2009], permet d'étudier la robustesse comme une propriété intrinsèque des systèmes d'information et des processus. Le terme « *robuste* » peut alors s'utiliser pour tout système logiciel ou matériel. Le système d'exploitation, par exemple, est dit robuste s'il fonctionne bien, non seulement dans des conditions ordinaires, mais aussi dans des conditions inhabituelles avec des fautes des concepteurs.

2.4 Modélisation d'un système à l'aide du paradigme multiagent

Dans les sections précédentes, j'ai présenté un système de collecte d'information perturbé (via les définitions de système, système perturbé, système de collecte d'information) et la définition de la cohérence et de robustesse dans un système. Je présente dans cette section divers outils développés pour l'étude des systèmes complexes afin de choisir la meilleure méthode pour la modélisation du système de collecte d'information. J'introduis tout d'abord différents outils d'analyse et de modélisation des systèmes complexes (statistiques, automates, ...) (Section 2.4.1) ; puis le paradigme agent que j'ai choisi pour modéliser mon système de collecte d'information (Sections 2.4.2, 2.4.3). Enfin, j'introduis la notion de perturbation dans un SMA posant ainsi le contexte de mon travail sur le maintien de la cohérence et l'amélioration de la robustesse (Sections 2.4.4, 2.4.5, 2.4.6).

2.4.1 Outils d'analyse et de modélisation des systèmes complexes

Il existe de nombreux outils pour modéliser les systèmes complexes parmi lesquels on peut citer : les fonctions itératives, la thermodynamique et les mathématiques statistiques, les automates, les graphes

hiérarchiques. Mon objectif est de chercher une méthode pour présenter un système de collecte d'information distribué (un cas particulier de système complexe). C'est pourquoi on ne présente que les outils qui s'accordent bien avec la représentation et l'échange d'information dans un système. Les autres méthodes (telles que la simulation de Monte-Carlo, etc.) ne seront pas abordées. Beurier Grégory, donne dans [Beurier, 2007], un analyse des outils d'analyse et de modélisation des systèmes complexes.

2.4.1.1 Fonctions itératives

Cette méthode consiste à modéliser les comportements et l'évolution des paramètres sous la forme d'une fonction itérative. Une fonction itérative est construite par l'application d'une fonction sur le résultat à l'instant précédent. Par exemple, les fonctions itératives linéaires sont de la forme : $s(t) = s(t-1) + v$ où $s(t)$ décrit l'état du système à l'instant t et v est une constante.

Exemple 2. *De nombreux types de fonctions itératives peuvent être utilisées* [Beurier, 2007] :

- *des fonctions itératives constantes :* $s(t) = s_0$
- *des fonctions itératives linéaires :* $s(t) = s(t-1) + v$ *où* v *est une constante.*
- *des fonctions itératives quadratiques :* $s(t) = as(t-1)(1-s(t-1))$ *où* a *est une constante. Les fonctions quadratiques produisent des comportements allant de la linéarité au chaos en passant par des phases périodiques. Elles permettent également d'observer le changement de phase du comportement simple au comportement chaotique.*
- *des fonctions itératives stochastiques :* $P_s(s'(t);t) = \sum_{s'(t)} P_s(s'(t) \mid s'(t-1)) P_s(s'(t) \mid t-1)$. *Dans cette fonction,* $P(s;t)$ *représente la probabilité de distribution (d'existence) de l'état* s *du système à un temps* t *et* $P(s \mid s')$ *représente la probabilité de transition de l'état* s' *du système vers l'état* s. *On peut donc, avec ce type de fonction, modéliser l'évolution non déterministe d'un système.*

2.4.1.2 Thermodynamique et mathématiques statistiques

La thermodynamique dans un premier temps s'intéresse à la chaleur et à la température. La thermodynamique cherche à décrire les échanges de chaleur et de travail entre systèmes. Elle se base principalement sur deux principes.

Pour les illustrer, considérons un système dans un état initial d'équilibre sous l'effet du milieu extérieur, ce système va évoluer vers un état équilibre final. Lors de la transformation, le milieu extérieur fournit un certain travail W et une certaine quantité de chaleur Q.

Le premier principe affirme que l'énergie est toujours conservée. L'énergie totale d'un système isolé reste constante, les évènements qui s'y produisent ne se traduisent que par des transformations et des échanges d'énergie entre le travail et la chaleur. Cela peut s'écrire formellement : $\triangle U = W + QoDelta_u = U_f - U_i$ avec $\triangle U$ la différence entre la quantité d'énergie U_f du système après la transformation et la quantité d'énergie U_i du système à l'état initial, W le travail fournit par le système et Q la quantité de chaleur que le système reçoit.

Le second principe introduit une dissymétrie entre ces deux formes d'échanges d'énergie en indiquant dans quel sens se développe la transformation. Claudius (cité par [Beurier, 2007]) l'a formulé de la façon suivante :

> « *il est impossible de réaliser un processus dont le seul résultat serait de transférer de la chaleur d'un corps froid à un corps chaud* »

Sadi Carnot (1824) introduit une fonction d'état du système appelée entropie, traduisant le désordre d'un système. Claudius a montré qu'un échange de chaleur Q d'un système à une température T se traduit par une variation de son entropie de Q/T. Une transformation d'un système sera dite réversible si la variation d'entropie lors de cette transformation est nulle. Si la variation est strictement positive, la transformation est dite irréversible. Cette variation ne peut pas être négative.

2.4.1.3 Les automates

La première définition des automates revient au mathématicien Ulam Stanislaw. En se basant sur la génération de motifs graphiques, il a donné le concept d'automate cellulaire suivant :

> « *Étant donné un treillis infini ou un graphe de points, chacun ayant un nombre fini de connexion avec certains de leurs voisins. Chaque point peut être dans un nombre d'états finis. Les états des voisins à l'instant t induisent, selon des règles spécifiées, l'état du point à $t+1$. L'un des principaux objectifs de la théorie est d'établir l'existence de ce système capable de se multiplier, c'est-à-dire de créer, à travers le temps, d'autres systèmes identiques congruents à eux-mêmes* ». [Ulam, 1962]

Les automates cellulaires sont utilisés dans plusieurs domaines différents : modélisation des processus physiques [Toffoli, 1984], la modélisation des dynamiques urbaines [White and Engelen, 1993] et des trafic routiers [Wagner, 1995], des outils informatique [Wolfram, 1986, Hortensius et al., 2006] et des modèles d'études pour la biologie [Barto, 1975]. Ils constituent un des modèles standards dans l'étude des systèmes complexes.

> « *Un automate cellulaire est un modèle spatial discret. Il consiste en une grille de "cellules" pouvant chacune prendre à un instant donné un nombre fini d'états. Le temps est également discret et l'état d'une cellule au temps t est fonction de l'état à $t-1$ d'un nombre fini de cellules appelé voisinage. A chaque nouvelle unité de temps, les mêmes règles sont appliquées pour toutes les cellules de la grille, produisant une nouvelle génération de cellules dépendant entièrement de la génération précédente.* » [Beurier, 2007]

Wolfram a défini, dans [Wolfram, 1986], un automate cellulaire par un quadruplet : $A = (d, G, S, f)$ avec :
- d : la dimension de l'automate
- G : le voisinage des cellules (2-connexité, 4-connexité, n-connexité, etc.)
- S : un ensemble fini d'états
- f : une fonction de transition entre les états

L'exemple de plus célébré d'automate cellulaire est certainement le jeu de la vie [Gardner, 1970].

Exemple 3. (Jeu de la Vie [Gardner, 1970]) *Le Jeu de la Vie est un automate cellulaire de dimension 2 inventé par Conway. Malgré des règles très simples, le jeu de la vie permet le développement de motifs extrêmement complexes. L'automate du Jeu de la Vie est de la forme :*

$$A = (2, Moore, \{vivant, mort\}, f)$$

avec f défini comme suit :

- *Règle 1 : une cellule morte (blanche) entourée de trois cellules vivantes (ombres) devient vivante.*

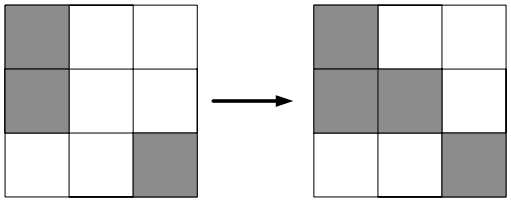


FIGURE 2.3 – Règle 1 du jeu de la vie

- *Règle 2 : une cellule vivante entourée de deux ou trois cellules vivantes, reste vivante.*

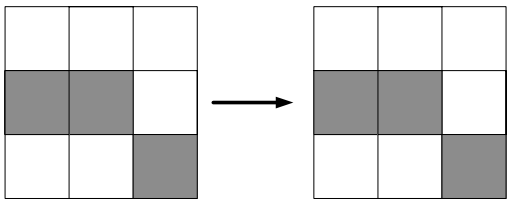


FIGURE 2.4 – Règle 2 du jeu de la vie

– *Règle 3 : dans tous les autres cas la cellule devient ou reste morte.*

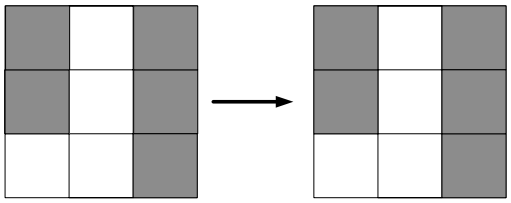
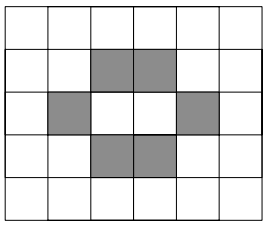
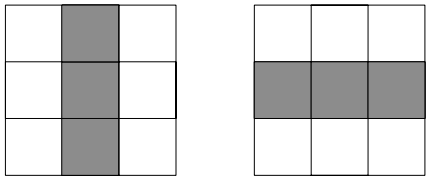


FIGURE 2.5 – Règle 3 du jeu de la vie

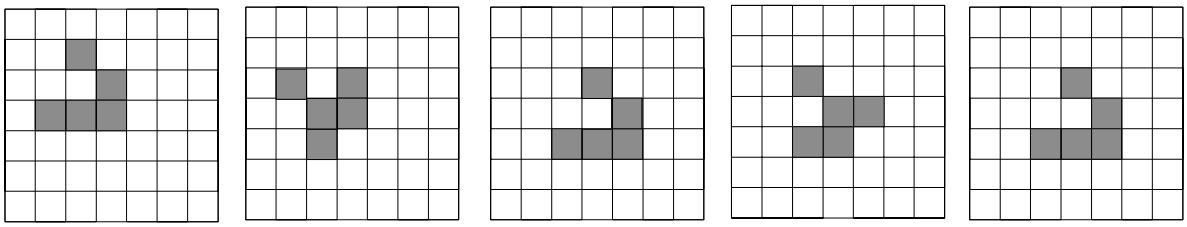
Cet automate cellulaire permettant de mettre en évidence l'apparition de phénomènes émergents avec les trois grands types de comportements ou de structures émergents dans le jeu de la vie : les blocs, les oscillateurs et les planeurs (Figure 2.6). Le jeu de la vie a été le point de départ de nombreuses recherches sur les phénomènes émergents [Beurier, 2007].



(a) Bloc du jeu de la vie



(b) Oscillateur du jeu de la vie



(c) Planeur du jeu de la vie

Figure 2.6: L'apparition de phénomènes émergents avec les trois grands types de comportements ou de structures émergents dans le « jeu de la vie ».

2.4.1.4 Inconvénients pour la modélisation de systèmes complexes

Les inconvénients des trois méthodes précédentes sont résumées dans le Tableau 2.1.

Méthodes	Inconvénients de la méthode
Fonctions itératives	– pas efficaces lorsque les systèmes mettent en œuvre de très nombreux éléments (ou paramètres) – pas adaptées aux phénomènes inscrits dans le temps ou dans la dynamique du système
Thermodynamique	– ne permet pas de décrire tous types de systèmes complexes – ne permet pas de rendre compte des dynamiques locales (à cause de l'utilisation d'un petit nombre de paramètres)
Automates	– une grande difficulté subsiste au niveau de la conception

Table 2.1: Méthodes pour la modélisation de systèmes complexes [Beurier, 2007]

C'est la raison pour la quelle, j'oriente mon travail vers un outil puissant de modélisation des systèmes complexes, qui possède un plus grand degré de liberté de modélisation : les système multi-agents (en particulier en permettant de modéliser des entités hétérogènes dans un environnement dynamique).

2.4.2 Les systèmes multi-agents (SMA)

2.4.2.1 Agent

Les systèmes multi-agents sont construit autour de la notion d'agent. Le terme « *agent* » selon Wooldridge et Jenning [Wooldridge and Jennings, 1994] (résumé dans [Beurier, 2007]) désigne une entité qui possède les propriétés suivantes :

- *« situé : un agent est situé dans un environnement. Il est capable de* percevoir *cet environnement et de* réagir *aux changements qui y interviennent par ses actions* (Figure 2.7).
- *autonome : un agent est libre d'agir à sa guise en prenant des décisions basées sur son état interne. Cet état interne est propre à chaque agent et ne peut être accédé par des agents externes.*
- *social : un agent est une entité sociale. Il est capable de communiquer ou d'interagir avec les autres agents via le médium d'interaction qu'est l'environnement.*
- *pro-actif : un agent est capable de produire des décisions liées à son état interne et ses perceptions de l'environnement ».*

Dans le domaine des systèmes multi-agents, l'architecture interne d'un agent fait référence aux mécanismes qui modélisent sa dynamique. Un agent est généralement considéré comme un processus cyclique comportant trois phases successives : perception, délibération et action (Figure 2.8).

Ainsi, la nature de l'architecture interne d'un agent est souvent caractérisée par celle de son processus décisionnel, bien plus que par ses mécanismes de perceptions et d'actions. Suite à cette définition, je présente un système multi-agent dans la section suivante.

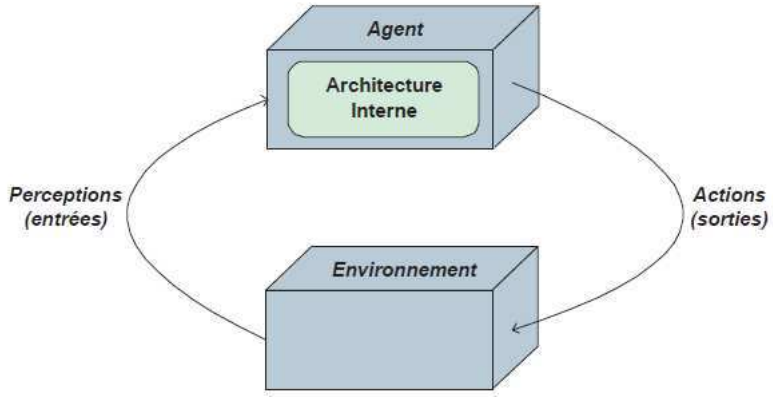


FIGURE 2.7 – Représentation classique d'un agent et de son environnement [Wooldridge and Jennings, 1994]

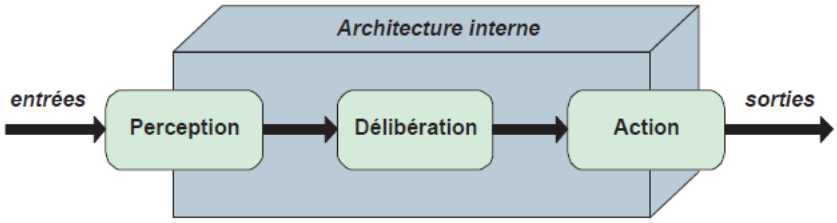


FIGURE 2.8 – L'agent, un processus cyclique à trois phases : perception, délibération puis action [Wooldridge and Jennings, 1994]

2.4.2.2 Le système multi-agents

Le domaine des systèmes multi-agents (SMA) est actuellement un domaine de recherche qui suscite beaucoup d'intérêt. Ce domaine est né à la fin des années 70 et le début des années 80, de l'idée de distribuer les connaissances et le contrôle dans les systèmes d'Intelligence Artificielle. Cette idée a émergé d'une part du besoin de faire face à la complexité croissante de ces systèmes et a été favorisée d'autre part par l'émergence des modèles et machines parallèles, rendant possible la mise en œuvre opérationnelle de la distribution. Un SMA est composé par plusieurs entités appelées agents qui interagissent dans un environnement partagé visant à atteindre un objectif individuel ou collectif.

On peut considérer qu'un système multi-agents est tout simplement un ensemble d'agents partageant un environnement commun. La figure 2.9 représente schématiquement un système multi-agents.

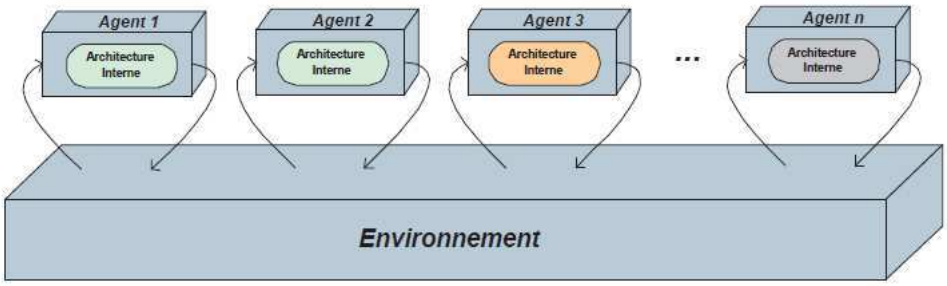


FIGURE 2.9 – Représentation schématique d'un système multi-agents [Wooldridge and Jennings, 1994]

Définition d'un système multi-agents par [Bonabeau et al., 1999] : *Un système multi-agents est un ensemble d'entités (physiques ou virtuelles) appelées agents, partageant un environnement commun (physique ou virtuel), qu'elles sont capables de percevoir et sur lequel elles peuvent agir. Les perceptions permettent aux agents d'acquérir des informations sur l'évolution de leur environnement, et leurs actions leur permettent entre autres de modifier l'environnement. Les agents interagissent entre eux directement ou indirectement, et exhibent des comportements corrélés créant ainsi une synergie permettant à l'ensemble des agents de former un collectif organisé.*

2.4.2.3 A + E + I + O : L'approche VOYELLES

L'analyse ou le développement d'un système multi-agents, nécessite donc la prise en compte d'au moins quatre dimensions, comme identifiées dans l'approche voyelle : AEIO définie par Y. Demazeau [Demazeau, 1995] : l'**A**gent, l'**E**nvironnement, les **I**nteractions, et l'**O**rganisation (Figure 2.10). Une description simplifiée est comme suit :

- **A**gents : les modèles (l'architectures internes) des agents
- **E**nvironnement : le milieu dans lequel évoluent les agents
- **I**nteractions : les moyens par lesquels les agents interagissent
- **O**rganisation : les moyens qui structurent les agents en groupes, hiérarchies, relations, etc.

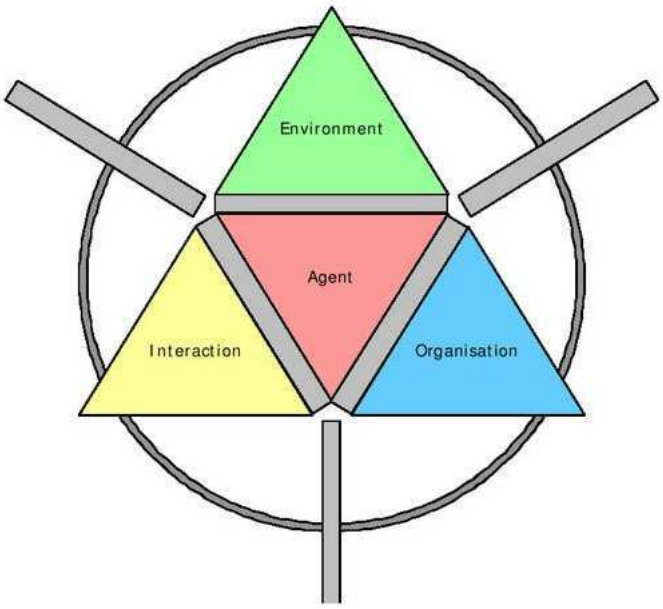


FIGURE 2.10 – Représentation symbolique d'un système multi-agents [Ricordel, 2001].

Selon [Ricordel, 2001], l'approche **Voyelles** est guidée par trois principes :

- Le principe déclaratif reflet de la décomposition précédente (**SMA = Agents + Environnement + Interactions + Organisations**).
- Le principe fonctionnel : les fonctionnalités du système entier incluent les fonctionnalités individuelles des agents auxquelles on ajoute les fonctionnalités résultant de la valeur ajoutée par la cohésion du système (**Fonction(SMA) = S Fonction(Agents) + Fonction collective**).
- Le principe de récursion : les systèmes multi-agents, dans un niveau d'abstraction supérieur, sont considérés à leur tour comme des entités d'un système multi-agents (**SMA* = SMA**).

Le SMA que j'ai développé dans cette thèse est basé sur cette approche.

2.4.3 Les SMA pour représenter un système de collecte d'information

Comme nous avons pu le voir dans ce chapitre, la dynamique et des interactions non-linéaires entre les constituants d'un système confèrent à ce dernier un caractère complexe, rendant son étude difficile et l'usage des approches purement analytiques et formelles insuffisante pour rendre compte de l'intégralité des propriétés et de la dynamique du système. Une approche par simulation permet de mieux appréhender cette complexité en offrant la possibilité d'expérimenter directement ces interactions non-linéaires. Le paradigme agent est parfaitement adapté à la modélisation de systèmes complexes car il permet de décrire les dynamiques des système complexes en s'intéressant aux comportements des agents au niveau micro. Ils permettent aussi d'aborder des phénomènes tels que l'émergence et l'auto-organisation dans la dynamique du système.

Afin de modéliser un système de collecte d'information, je considère un système multi-agents (SMA) dans lequel chaque agent vise à obtenir une représentation de son environnement la plus précise possible par la collecte d'informations de manière soit directe (par capteurs) soit indirecte (en communiquant avec d'autres agents). Je prends pour hypothèse que certains agents perturbent la communication du système, c'est-à-dire transmettent des informations fausses ou inexactes sur l'environnement, soit parce que leur perception est erronée (capteurs déficients), soit parce que leur intérêt va à l'encontre de la communauté et les pousse à déformer les informations qu'ils communiquent.

Mes travaux s'inscrivent dans une vision systémique, enactive et auto-organisationnelle. La vision systémique consiste à considérer le système dans son intégralité en considérant les interdépendances entres ses différents composants. Ainsi le système global à concevoir, doit considérer les influences mutuelles entre les activités de collecte et de communication et doit s'intéresser à leurs co-évolutions et leurs influences mutuelles et rétro-actives. La vision enactive consiste à considérer le couplage structurel existant entre le système et son environnement (le système structure l'environnement, tout en étant structuré par lui). Par exemple, les effets de la collecte d'information influencent les comportements de collecte du système et vice versa. La vision auto-organisationnelle consiste à considérer le système à travers son organisation et s'intéressent aux mécanismes ou dynamiques internes qui mènent le système vers une organisation, sans que cette organisation ne soit dictée par une entité externe au système.

J'introduis dans la section suivante la notion de perturbation dans un SMA.

2.4.4 Perturbation dans un SMA

Mes travaux portent sur l'étude d'un système de collecte d'information perturbé modélisé en utilisant un SMA. Pour cette raison, je vais tout d'abord introduire la notion de perturbation dans un SMA.

Pour ce faire, je me baserai sur la théorie de la dissonance cognitive présentée en 1957 par Festinger [Festinger, 1957] qui est l'une des théories les plus importantes de la psychologie sociale. Dans sa version originale, cette théorie considère que deux éléments de cognition (perceptions, attitudes propositionnelles ou comportements) sont en rapport selon qu'un lien pertinent les relie ou non. Deux cognitions liées sont soit consonantes, soit dissonantes. Elles sont dites consonantes si l'une entraîne ou corrobore l'autre. A l'inverse, deux cognitions sont dites dissonantes si l'une entraîne ou soutient le contraire de l'autre [Pasquier and Chaib-draa, 2002]. La théorie suppose que les agents dissonants vont essayer de réduire cette dissonance, par exemple en changeant ou en oubliant certaines attitudes mentales.

Je définis les termes suivants relatif à la perturbation dans un SMA :

Définition 1. (Acte dissonant)
En appliquant la théorie de Festinger [Festinger, 1957] *au contexte des SMA, j'appelle* ***'acte dissonant'*** *tout acte allant à l'encontre de la réalisation de l'objectif d'une communauté.*

Définition 2. (Agent dissonant)
Tout agent qui accomplit un acte dissonant est appelé ***'agent dissonant'****, autrement dit tout agent fournissant des informations qui présentent un écart par rapport aux croyances de l'agent récepteur dans le contexte à base de communication que nous avons choisi, sera considéré comme dissonant par le récepteur.*

Définition 3. (Information dissonante)

Une information sera alors qualifiée de **'dissonante'** *par rapport à un ensemble d'informations si elle diverge un tant soit peu d'une des informations correspondantes de l'ensemble ; on dit que ces 2 informations sont dissonantes.*

Définition 4. (Système multi-agents perturbé)
Nous appelons **'système multi-agents perturbé'**, *tout SMA contenant un ou plusieurs agents dissonants.*

On peut se demander dans quelles circonstances la **dissonance** survient. [Pasquier and Chaib-draa, 2002] a donné quatre situations dans lesquelles la dissonance est presque inévitable :

> *« **1. contact direct initial avec une situation** : une situation entièrement nouvelle est susceptible d'introduire un certain nombre de nouveaux éléments de cognition dissonants avec ceux qui pré-existent ;*
> *__2. un changement dans la situation__ : de la même façon, un changement dans la situation peut amener des items de cognition jusqu'alors consonants à devenir dissonants ;*
> *__3. communication__ : la communication avec les autres est susceptible d'introduire de nouveau items qui sont dissonants avec ceux de l'agent ;*
> *__4. existence simultanée de différentes cognitions dont certaines sont consonantes et d'autres dissonantes__ : dans le cas général, une cognition est liée à plusieurs autres dont certaines sont consonantes et d'autres dissonantes ».*

C'est l'aspect **communication** (parmi les quatre situations ci-dessus données par Pasquier) qui sera la situation introduisant des perturbations dans le système étudié. Je me concentre sur la communication, je considère que l'état de l'environnement est inchangé (je ne traite donc pas les situations *« **un changement dans la situation** »* et *« **contact direct initial avec une situation** »*).

Après avoir défini la notion de perturbation dans un SMA, j'examine maintenant la cohérence, la robustesse et les moyens existants pour assurer la cohérence d'un SMA et renforcer sa robustesse.

2.4.5 Cohérence dans un SMA

2.4.5.1 Notion de cohérence

Dans le cadre des systèmes multi-agents, les mesures de cohérence définissent une métrique qui peut être appliquée à l'agent comme aux groupes d'agents. Ainsi, la cohérence est considérée autour de deux axes :

> *« 1. la cohérence comme moteur cognitif et comportemental : un agent travaille à maintenir une cohérence interne la plus élevée possible.*
> *2. la cohérence comme moteur social : dans un cadre coopératif, les agents travaillent à maintenir la cohérence du groupe la plus élevée possible »*[Pasquier and Chaib-draa, 2002].

La cohérence d'un SMA dans mon contexte est considérée comme **le moteur cognitif et comportemental** des agents et est lié à la validité des informations collectées par les agents par rapport aux données réelles de l'environnement.

Définition 5. (Système multi-agents cohérent)
Un SMA sera dit **cohérent** *lorsque la perception du monde par ses agents sera correcte et partagée par l'ensemble des agents.*

2.4.5.2 Travaux existants pour assurer la cohérence dans un SMA

En posant l'hypothèse selon laquelle les agents tentent de réduire leur dissonance cognitive (section 2.4.4), nous supposons que l'objectif de la communauté est de réduire la dissonance, en écartant ou minimisant l'apport d'informations fausses ou incohérentes et par conséquent en évitant ou sériant les interactions avec les agents dissonants, ce qui se traduit par la mise en place d'une stratégie de communication adaptée à la dissonance du système.

Pour assurer la cohérence dans un SMA perturbé, l'agent peut agir de différentes façons pour contrer les situations provoquant la dissonance [Pasquier and Chaib-draa, 2002] :

– changer la situation afin qu'elle soit consonante avec ses cognitions,
– changer ses cognitions afin qu'elles soient consonantes avec la situation.

Pour ces 2 objectifs, les chercheurs ont développé diverses stratégies de résolution des conflits pour assurer la cohérence du système multi-agents [Barber et al., 1999] :

– **Arbitrage** : solution de l'agent omniscient [Steeb et al., 1988] ;
– **Convention de priorité** : inégalité de poids des agents en cas de conflit [Ioannidis and Sellis, 1989] ;
– **Procédure de vote et élection** : élection de la résolvante du conflit [Ephrati and Rosenschein, 1991] ;
– **Négociation** : communication et échanges d'intérêts [Sycara, 1988] ;
– **Hypothèses de surface** : retour aux hypothèses qui ont menées au conflit [Mason and Johnson, 1990] ;
– **Standardisation et normes sociales** : permettent d'éviter les conflits, ou de les résoudre de manière protocolaire [Castelfranchi, 1992] ;
– **Relaxation de contraintes** : élimination ou reformulation des contraintes problématiques [Sycara et al., 1991] ;
– **Argumentation** : construction d'arguments pour défendre un point de vue particulier [Loui, 1987, Hewitt, 1986] ;
– **Persuasion** : tentative pour faire changer les croyances, les actions des autres [Sycara, 1990] ;
– **Élaboration de conjecture commune** [Fiorino, 1998].

Ces stratégies diverses de résolution des conflits ont chacune leurs forces et faiblesses en fonction du contexte. Contrairement aux méthodes énumérées ci-dessus, afin d'assurer la cohérence du SMA, je propose de m'appuyer sur un modèle de confiance, permettant aux agents de raisonner eux-mêmes sur l'information collectée pour assurer sa cohérence.

2.4.6 Robustesse dans un SMA

2.4.6.1 Notion de robustesse

La robustesse d'un système est présentée dans la section précédente. Dans cette section, j'examine plus en détails les définitions de la robustesse dans un SMA existant dans la littérature.

Construire un SMA robuste, hautement évolutif et performant est l'un des objectifs majeurs pour les chercheurs en (D)AI (Intelligence Artificielle Distribuée) [Bond and Gasser, 1988, Weiss, 1999]. Bien que la littérature traite la robustesse comme une caractéristique inhérente, il n'y a pas beaucoup de discussion sur la notion de robustesse. Comme n'importe quel système complexe artificiel, un SMA a besoin d'être spécifiquement conçu pour être robuste. La robustesse des SMA est différente par rapport aux systèmes de l'informatique classique. La plupart des systèmes sont des système de transformation, ce qui signifie qu'ils calculent une fonction sur certaines entrées. Toutefois, les SMA sont des systèmes ouverts, qui n'ont pas pour but de calculer explicitement une fonction (Internet en est bon exemple).

Quelques définitions de la robustesse dans un SMA sont données comme suit :

> *La robustesse d'un SMA est la capacité à montrer un comportement prédéfini qualitatif en présence d'évènements non-pris en compte et de perturbations techniques.*
> *(Traduction personnelle à partir de* [Nimis and Lockemann, 2004]*).*

Du point de vue de [Schillo et al., 2001], la robustesse d'un SMA peut seulement être définie par rapport aux définitions de mesure de performance.

> *La robustesse d'un SMA est définie quantitativement par la baisse attendue de la mesure de performance dans les quatre scénarios de perturbation (i) augmentation de la taille de la population, (ii) le changement de profil de tâche au fil du temps, (iii) d'intrusion agent malveillant, et (iv) les abandons d'agent.*
> *(Traduction personnelle à partir de* [Schillo et al., 2001]*).*

Selon Schillo, la robustesse peut alors être définie et mesurée par la diminution de la mesure de la performance dans différents cas : doubler la taille de la population, ajouter 5% abandons d'agents, etc.

En général, je définis un système multi-agents robuste comme suit :

Définition 6. (Système multi-agents robustesse)
Un SMA sera dit ***robuste*** *s'il réussit à mettre en place des dispositifs qui lui permettent d'atteindre ses objectifs malgré les conditions dégradées.*

2.4.6.2 Travaux existants pour assurer la robustesse dans un SMA

Jens Nimis, donne dans [Nimis and Lockemann, 2004], un aperçu des différentes approches visant à accroitre la robustesse dans un SMA. Les approches existants sont présentées du point vue des développeurs sur des problèmes différents apparaissent lors du processus de développement. Les tâches de développement peuvent être classées par l'ordre croissant d'abstraction représentant par une architecture des couches qui est montré dans la Table 2.2.

Table 2.2: Architecture d'implémentation d'agents en différentes couches et approches de la robustesse correspondants [Nimis and Lockemann, 2004]

Couche	**Références**
7. Couche de l'utilisateur	
6. Couche mécanisme	[Lesser and Corkill, 1981, Klein and Dellarocas, 1999, Kaminka et al., 2001, Kumar and Cohen, 2000, Sandholm and Lesser, 2002, Browning et al., 2002, Parsons and Klein, 2004, Amza, 2011]
5. Architecture d'agent	[Nagi, 2001a, Nagi, 2001b, y López et al., 2002]
4. Couche de l'ontologie	[Mena et al., 2000]
3. Couche de conversation	[Smith, 1980, Alan Galan, 1999, Nodine and Unruh, 2000, Hannebauer, 2002, Paurobally et al., 2003, Nimis and Lockemann, 2004]
2. Couche d'infrastructure SMA	[Poslad et al., 2000, Bellifemine et al., 2001, Campo, 2002]
1. Couche de communication	[Halsall, 1992]

Je résume dans la suite la fonction des différentes couches avec les problèmes possibles et certaines des solutions qui tentent de faire face à ces problèmes :

- Dans la couche de communication, un SMA est considéré comme un type particulier de système logiciel distribué. Les agents sont affectés par tous les problèmes de communication de données classiques (par exemple perte de l'information, scission du réseau, etc.). Il existe de nombreuses approches pour améliorer de la robustesse d'un réseau ([Halsall, 1992]). Mais elles ne sont pas spécifiques aux SMA.
- Dans la couche d'infrastructure d'un SMA, il ne me semble pas nécessaire d'examiner plus en détails la robustesse dans cette couche car elle n'a pas d'influence directe sur la robustesse de la coopération entre agents lors de l'exécution d'une tâche commune.
- Dans la couche de conversation, la communication entre agents dans un SMA est structurée dans une conversation qui suit un protocole de communication (par exemple le protocole Contract-Net [Smith, 1980]). Un certain nombre d'approches traitent ce problème s'intéressant au processus de conception du protocole tels que [Alan Galan, 1999, Nodine and Unruh, 2000, Hannebauer, 2002, Paurobally et al., 2003, Nimis and Lockemann, 2004].
- Dans la couche de l'ontologie, le problème d'incompatibilité de l'ontologie est abordé par diverses approches, comme Mena et al. [Mena et al., 2000] qui proposent un mécanisme de négociation des termes de l'ontologie en cours d'interaction les agents.

- La couche de l'architecture de l'agent traite de la façon dont les connaissances et le comportement des agents sont ne présentées et traitées. Pour donner un exemple [Nagi, 2001a, Nagi, 2001b] présentent un plan des étapes à suivre par un agent sous la forme d'un arbre des transactions ouverts.
- Dans la couche de mécanisme, un SMA est considéré comme une société d'agents. Il existe plusieurs approches comportementales et organisationnelles afin de réduire l'incertitude dans les sociétés d'agents comme par exemple : « *redundant broker teams* » par [Kumar and Cohen, 2000], « *leveled-commitment contracting* » par [Sandholm and Lesser, 2002] ou encore « *knowledge-based exception handling* » par [Klein and Dellarocas, 1999]. [Parsons and Klein, 2004] propose également l'utilisation des services de gestion des exceptions indépendants du domaine pour développer un SMA robuste à partir de composants non fiables. Cette approche est très proche des travaux de [Kaminka et al., 2001, Browning et al., 2002] sur le suivi de l'exécution.
- La couche de l'utilisateur introduit la possibilité pour les agents d'indiquer des problèmes insolubles aux utilisateurs et de demander leur intervention du point de vue de la robustesse.

Mon travail sur la robustesse se situe sur les deux couches (5) et (6). J'introduis la confiance des agents sur le couche (5) et l'approche de l'auto-organisation basée sur la confiance dans le couche (6).

2.5 Confiance et cohérence d'un système d'information

Afin d'assurer la cohérence d'un système de collecte d'information perturbé, je donne aux agents la capacité de raisonner sur les informations collectées directement ou indirectement. Pour cela, l'un des outils les plus efficaces consiste à introduire le concept de confiance. Les agents utiliseront la confiance qu'ils ont envers les autres agents pour :

- calculer la fiabilité de l'information reçue.
- identifier et isoler les agents dissonants, évaluer l'utilité et décider de l'opportunité d'interagir avec un agent.

Dans cette partie, je présente tout d'abord la définition de la confiance dans les SMA (Section 2.5.1) ; le lien entre la confiance et la cohérence dans un SMA est présentée ensuite en Section 2.5.2 et finalement je présente un aperçu des modèles de confiance existants (Section 2.5.3) et expliquerai pourquoi j'ai choisi d'utiliser le modèle TrustNet de Schillo et al. ([Schillo et al., 1999, Schillo et al., 2000]) comme base de mon travail (Section 2.5.4).

2.5.1 Confiance : définition

La confiance est au cœur de plusieurs disciplines. Il n'est donc pas surprenant qu'elle ait suscité beaucoup d'attention de la part de nombreux chercheurs. Une grande quantité de travaux a été menée sur la confiance et de nombreuses définitions et caractérisations ont été proposées en sciences sociales ([Castelfranchi and Falcone, 1998]), en e-commerce ([Guibert, 1999]) et en SMA ([Huynh, 2006]). Tous les auteurs essaient de construire une définition générale de la confiance et de distinguer ce concept d'autres comme la réputation, la sécurité, le risque, la sincérité, l'honnêteté, etc. Des recherches ont été effectuées pour comparer les différents points de vue sur la confiance dans les diverses disciplines ([Mui et al., 2002b, Castelfranchi and Falcone, 1998]). Certaines définitions sont propres au point de vue du domaine des auteurs ou au point vue de l'auteur lui même, par l'exemple la confiance est définie comme des croyances sur divers attributs d'une personne [Castelfranchi and Falcone, 1998], comme une attente sur le comportement de l'autre [Dasgupta, 2000], comme des processus de construction dans le domaine du management, par exemple fondés sur la dissuasion, sur la connaissance ou sur l'identification [Lewicki and Bunker, 1995]. Traditionnellement, il y a deux visions principales de la confiance. D'abord, dans la vision cognitive [Castelfranchi and Falcone, 1998], la confiance est une fonction des croyances et autres attitudes mentales de l'agent. Le second point de vue est la vue probabiliste qui adopte une métrique pour modéliser une probabilité subjective avec laquelle un agent effectuera une action particulière [Yu and Singh, 2000]. Par la suite, j'utiliserai la définition de [Huynh et al., 2006] car elle est très générale et assez complète pour prendre en compte la plupart des modèles de confiance et de réputation existants (par exemple [Abdul-Rahman and Hailes, 2000], [Dimitrakos and Bicarregui, 2001], [Mui et al., 2002b],

[Teacy et al., 2006], et [Yu and Singh, 2000]) et elle peut être simplement définie comme la probabilité selon laquelle un agent pense que l'interaction avec un autre agent sera bénéfique pour lui :

> *La confiance est définie comme un niveau mesurable de la probabilité subjective avec laquelle un agent A estime que l'agent B effectuera une action particulière d'une façon favorable à A avant que A puisse vérifier une telle action dans un contexte où il affecte sa propre action. (Traduction personnelle à partir de* [Huynh et al., 2006]*).*

Cette définition ne fait aucune hypothèse sur l'architecture des agents contrairement à la définition de Castelfranchi [Castelfranchi and Falcone, 1998] qui fait appel à des croyances (et autres attitudes mentales). Suivant cette définition, dans la suite, les valeurs de confiance seront des probabilités et donc dans l'intervalle $[0, 1]$.

2.5.2 Lien confiance - information fiable

Les informations extraites de systèmes ouverts répartis peuvent être d'une fiabilité incertaine et doivent être vérifiée par comparaison avec une grande quantité de données. La fiabilité de l'information peut-être assurée en utilisant les mécanismes de gestion de conflits présentés précédemment (section 2.4.5.2). Un des outils les plus efficaces est l'utilisation des mécanisme sociaux de confiance ou de réputation. L'utilisation de la confiance se fait généralement dans 2 buts différents :

- **calculer la fiabilité de l'information reçue** : le travail de [Abdul-Rahman and Hailes, 1999] suggère que l'utilisation de mécanismes de confiance peuvent aider à maintenir la cohérence de l'information dans Internet en attachant plus de poids aux opinions des sources qu'on connait et en qui on a confiance. En tenant compte de ces sources, il est possible d'obtenir des informations plus fiables que par des sources aléatoires. Le problème de la surcharge d'information est réduit automatiquement car on ne s'intéresse qu'à des sources d'informations en qui on a confiance. C'est une forme de contrôle social qui se met en place en regroupant les résultats de sources ayant une bonne réputation.
- **identifier et isoler les sources dissonantes** : des travaux récents [Conte and Paolucci, 2002, Sabater and Sierra, 2002, Vercouter and Muller, 2010] suggèrent d'introduire la notion de confiance pour limiter l'influence des agents malveillants dans un système multi-agents ouvert et décentralisé pour contrer la propagation via la communication des informations fausses. Par exemple, le modèle L.I.A.R [Vercouter and Muller, 2010]se déroule selon les étapes suivantes : tout d'abord, l'auteur définit précisément les cas possibles de mensonge et la manière dont les communications entre agents sont représentées. Cette représentation est ensuite utilisée dans la deuxième phase pour formaliser la notion de mensonge, puis pour détecter des agents menteurs. Enfin, le résultat de la détection est intégré dans un mécanisme plus général de gestion des réputations d'agents.

Dans la suite, je présente un bref résumé de l'évolution des modèles de confiance. Je distingue les modèles simples de réputation et de confiance et les modèles plus complexe basés sur des réseaux.

2.5.3 Modèles de confiances existants

2.5.3.1 Principaux modèles simples de confiance existants

Modèle de Marsh. Marsh [Marsh, 1994] a proposé un des premiers modèles de confiance. Son modèle ne tient compte que des informations directes et des expériences (tout ce qui provient des interactions passées entre agents) pour calculer la confiance. Dans ce modèle, un agent ne tient pas compte des interactions entre les autres agents (et donc des réputations) quand il calcule ses confiances. En outre, Il n'y a pas de propagation et aucun processus d'évaluation n'est explicité.

Modèle de Abdul-Rahman et Hailes. Dans le modèle proposé par [Abdul-Rahman and Hailes, 2000] (figure 2.11), la confiance est divisée en confiance directe et en recommandation. Quatre degrés de croyance pour présenter la confiance de l'agent sont donnés : très fiable, fiable, peu fiable et très peu fiable. Pour chaque partenaire et dans un contexte donné, l'agent conserve un tuple avec les évaluations passées. Ensuite, du point vue de l'interaction directe, la confiance d'un partenaire dans un contexte donné est calculée par la valeur maximale stockée dans ce tuple. Contrairement à d'autres modèles de confiance où

l'information des témoins est fusionnée avec l'information directe pour obtenir la confiance dans un sujet particulier, ce modèle n'évalue que la fiabilité de l'information donnée par les témoins. Des expériences directes sont utilisées pour comparer le point de vue de ces témoins avec la perception directe de l'agent et être capable d'ajuster ensuite les informations venant d'eux en conséquence.

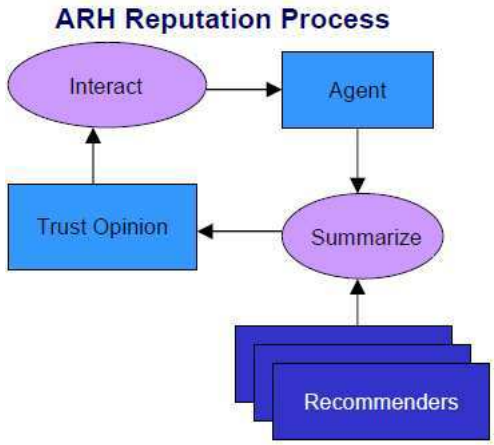


FIGURE 2.11 – Une vue synthétique du modèle ARH [Abdul-Rahman and Hailes, 2000]

Modèle Sporas et Histos. Les sites tels que eBay ou Amazon sont de bons exemples de marchés en ligne qui utilisent des modèles de confiance avec punition. eBay est l'un des plus grand marché en ligne avec une communauté de plus de 50 millions d'utilisateurs enregistrés. La plupart des produits sur eBay sont vendus aux enchères. Chaque utilisateur possède une évaluation basée sur les évaluations données par les utilisateurs après une transaction. La valeur de réputation est calculée en ajoutant les évaluations des six derniers mois. SPORAS introduit par [Zacharia and Maes, 2000] est une adaptation du modèle de réputation en ligne : seule l'évaluation la plus récente entre deux utilisateurs est utilisée. Cependant, elle a été construite sans considérer le problème des rapports non fiables, elle est donc influencée lors de la réception de fausses informations.

Modèle de Sen et Sajja. Dans le modèle de réputation Sen et Sajjas [Sen and Sajja, 2002], les interactions et les observations des expériences directes sont considérés. Ce travail se focalise sur la façon dont les agents utilisent les informations transmises de bouche à oreille (word-of-mouth) pour calculer des valeurs de confiance et de réputation. L'apprentissage par renforcement est utilisé pour mettre à jour la valeur de la réputation. Ce mécanisme est adapté pour détecter les menteurs si on suppose que les menteurs donnent toujours des informations fausses.

La confiance est utilisée non seulement dans des applications axées sur les affaires et le commerce mais aussi dans d'autres applications. Une des applications les plus récentes est les réseaux de capteurs (Wireless SenSor Networks - WSNs). Jusqu'à présent, de nombreux modèles de confiance et de réputation ont été publiés dans ce domaine tels que le modèle de confiance utilisant la logique floue introduit par [Kim and Seo, 2008]. Il prend en compte à la fois la fiabilité et la non fiabilité des capteurs pour distinguer les capteurs normaux et les capteurs anormaux. Un autre modèle introduit par [Zia, 2008], utilise des votes pour établir la confiance entre les capteurs. Plus de détails sur ces deux modèles seront fournis dans le chapitre 6 où ils sont comparés à notre modèle.

Comme on peut le constater à partir de ce bref aperçu, les modèles de confiance ont tendance à de plus en plus s'appuyer sur les expériences antérieures des agents et des témoignages dans le calcul de valeur de la confiance. Alors que les modèles présentées ci-dessus ne prennent pas pleinement en compte l'environnement social, la deuxième catégorie de modèles, qui est présenté dans la section suivante, fait une présentation de l'usage d'une représentation plus riche de l'environnement social en utilisant des réseaux.

Modèle ReGreT. Un modèle plus complet, ReGreT (Reputation in gregarious societies) a été introduit par Sabater [Sabater and Sierra, 2002]. Dans ce modèle (Figure 2.12), l'agent calcule sa confiance en un autre agent à partir de ses expériences directes et de la réputation de l'agent afin de tenir compte de ses interactions avec les autres agents. Il dispose d'un module de crédibilité pour évaluer la fiabilité des informations reçues des agents témoins. Il utilise également l'analyse de réseau social pour améliorer les connaissances sur la société (spécialement quand aucune expérience directe n'est disponible). Ce modèle fournit un degré de fiabilité pour les valeurs de confiance, de réputation et de crédibilité qui aident l'agent dans le processus de décision. Cependant, cette approche ne peut pas faire face aux situations où les comportements des agents cibles peuvent changer, donc il peut donner des évaluations fausses sur les agents honnêtes.

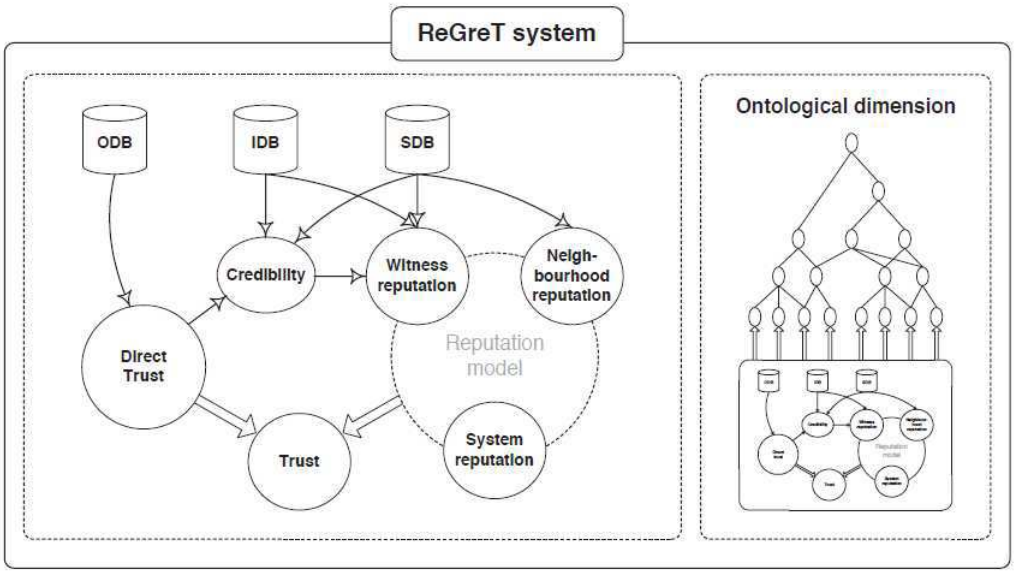


FIGURE 2.12 – Une vue synthétique du système de ReGreT [Sabater-Mir, 2002]

Modèle RepAge. [Sabater-Mir et al., 2006] a introduit le modèle RepAge (REPutation and ImAGE Among Limited Autonomous Partners) fondé sur ReGreT ([Sabater and Sierra, 2002]). C'est un modèle informatique qui apporte un soutien aux architectures d'agents qui veulent faire la distinction entre l'image et la réputation. Même si ces deux concepts sont des évaluations sociales, l'image est une simple croyance évaluant si un agent donné est bon ou mauvais par rapport à un certain contexte (rôles). La réputation est un méta-croyance qui dit ce que les autres pensent d'un agent donné dans un contexte donné. Le modèle calcule l'image et la réputation d'un agent grâce à l'évaluation des contrats passés par l'agent considéré et de ce qu'il a accompli dans ces contrats (qui mettent en œuvre des interactions directes) et des communications tiers. Les auteurs affirment que l'agrégation des évaluations ne doit pas dépendre de l'ordre dans lequel ces évaluations sont agrégées. Cependant les auteurs ne définissent pas comment ces évaluations sont utilisées dans la prise de décision.

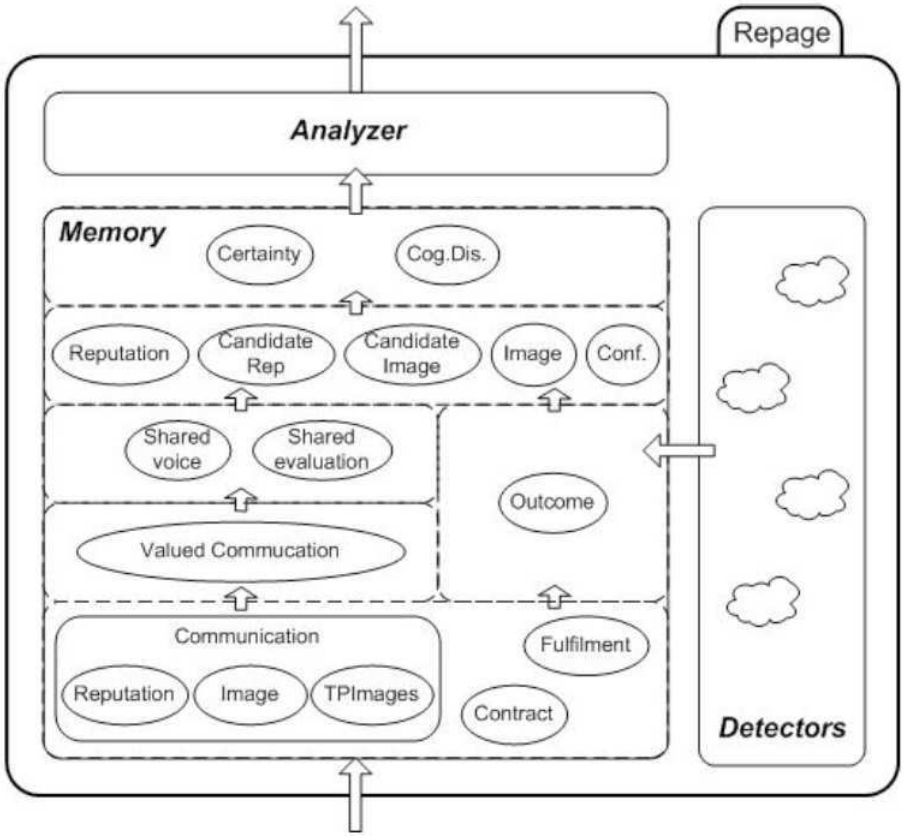


FIGURE 2.13 – L'architecture de Repage [Sabater-Mir et al., 2006]

Modèle de Torres da Silva et al. [Silva et al., 2009] présente un modèle hybride de réputation dans les structures organisationnelles basé sur un modèle de réputation à la fois centralisé et décentralisé (Figure 2.14). Dans ce modèle, les agents évaluent non seulement les comportements des autres agents et stockent les valeurs de réputation mais aussi envoient ces informations à un mécanisme centralisé. Ils peuvent également lui demander la réputation d'eux-même et celle des autres agents. L'objectif principal de cet approche est de permettre aux agents de raisonner sur les valeurs de réputation qu'ils reçoivent. En utilisant les valeurs de réputation, les agents stockent et transmettent des informations au sujet des violations des normes, des accomplissements et des faits liés aux violations ou aux accomplissements. Ces évaluations et les raisons de ces évaluations sont ensuite envoyées à l'organisation centralisé.

Un avantage de leur proposition est que les évaluations des agents sont distribuées car elles sont effectuées par les agents. Comme dans notre approche, le modèle de confiance est fondé sur les informations acquises directement et échangées. Mon modèle est totalement décentralisé et donne la capacité à chaque agent d'auto-évaluer sa fiabilité et de se construire un réseau d'agents fiables.

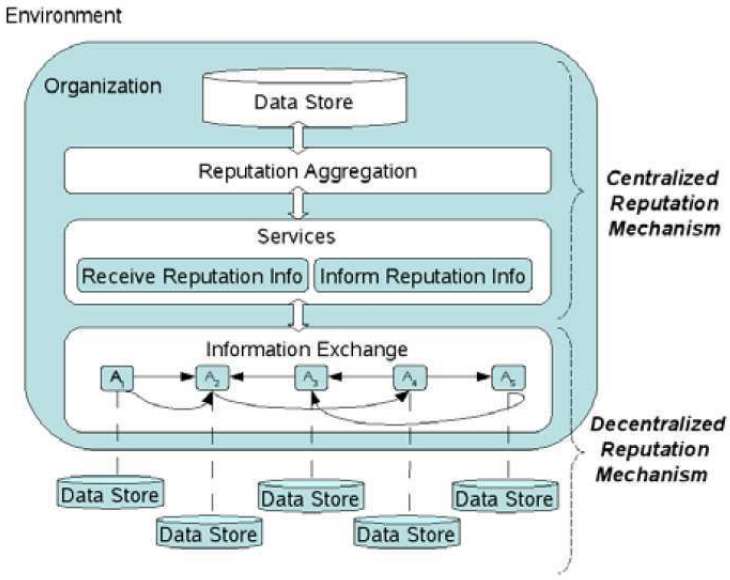


FIGURE 2.14 – L'architecture du modèle de Torres da Silva et al. [Silva et al., 2009]

Modèle TRSIM [Caballero et al., 2009] présente un modèle de réputation basé sur des critères de similarité entre les tâches (Figure 2.15) : le modèle TRSIM est composé d'un ensemble d'expériences sur le comportement des autres agents stockées par chaque agent et un ensemble de fonctions pour traiter ces expériences. Les fonctions apparaissant dans la figure sont représentées par des boîtes arrondies. Elles permettent de produire des valeurs basées sur des concepts de confiance et de réputation pour guider les interactions entre les agents. L'idée de base du modèle est que les agents peuvent modifier leur comportement en fonction de changements de l'environnement. Si les agents dans le système maintiennent un niveau adéquat d'interactions entre eux, TRSIM est capable de détecter les changements de comportement dans les individus de la société et de mettre à jour les confiances et les réputations en conséquence. Cependant l'auteur ne discute pas du nombre d'interactions nécessaires au modèle pour s'adapter aux changements et ni de la manière de s'adapter.

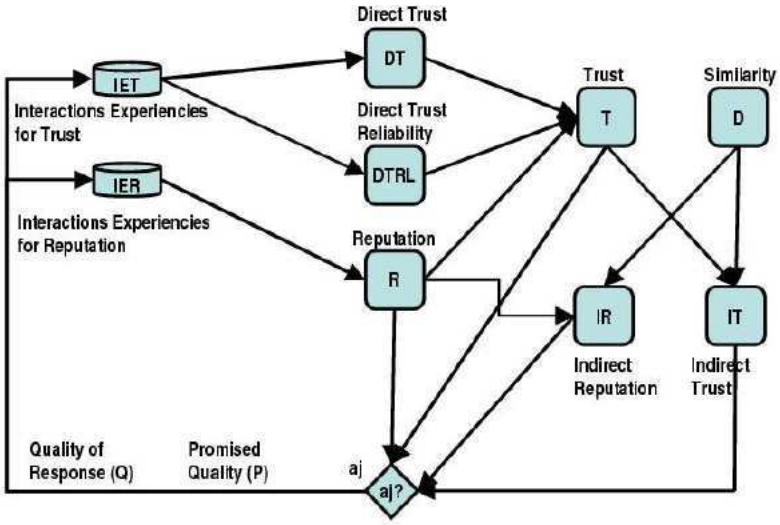


FIGURE 2.15 – Relation entre les différentes parties du modèle TRSIM [Caballero et al., 2009]

Modèle RepAge + BDI Le modèle RepAge + BDI ([Pinyol et al., 2012]) intègre le modèle Repage présenté ci-dessus ([Sabater-Mir et al., 2006]) dans un architecture BDI multi-valuée, fournissant un cadre basé sur la logique pour effectuer du raisonnement pratique en utilisant des informations d'image et de réputation. Ce modèle est présenté sous la forme d'un système multi-contexte, permettant à plusieurs composants distingués d'être connectés par un ensemble de règles transitives. Ces règles transitives sont des mécanismes utilisés pour transmettre des informations d'un contexte à un autre. Ce modèle intègre un contexte pour chaque attitude (la croyance, le désir et l'intention), un pour le modèle RepAge et deux contextes fonctionnels (communication et planificateur). Les auteurs ne parlent pas de la confiance dans leur approche. Cependant, leur architecture effectue un raisonnement pratique à déterminer à quel agent pour interagir.

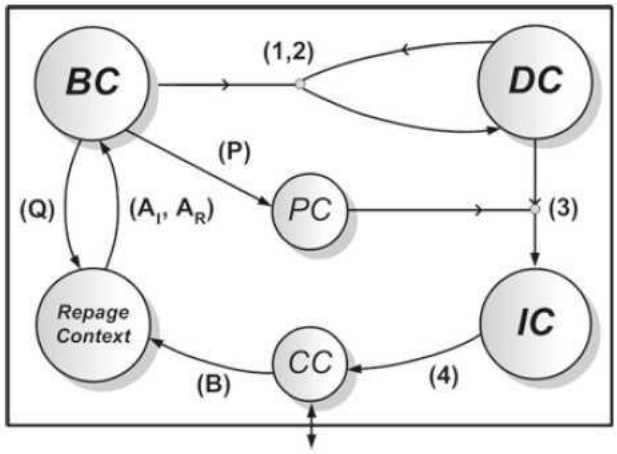


FIGURE 2.16 – Le contexte RepAge incorporé dans un contexte multi-agents BDI. Les cercles représentent les contextes et les flèches représentent les règles transitives [Pinyol et al., 2012]

2.5.3.2 Principaux modèles de confiance basées sur les réseaux

De manière générale, dans le domaine des SMA, les réseaux de confiance sont composés de relations de confiance entre agents connectés. L'hypothèse est qu'un agent peut calculer une valeur de confiance en un autre agent qu'il n'a jamais rencontré en analysant les chemins de confiance le liant à cet autre agent par une forme de transitivité de la confiance. Ainsi, si A fait confiance à B et B fait confiance à l'agent C, nous pouvons en déduire que A a un certain degré de confiance en C. Il s'agit d'un modèle inductif de confiance. Le calcul de la valeur de confiance en un agent cible exige une compréhension de la façon dont la confiance se propage le long d'un chemin de confiance. Deux opérateurs sont essentiels pour construire de tels réseaux de confiance : *l'agrégation* pour gérer les confiances sur le même objet mais provenant de sources différentes et *la propagation* pour calculer la confiance le long d'un chemin. Plusieurs modèles de réseaux de confiance, utilisant des implémentations différentes de ces opérateurs, ont été développés. Dans [Buchegger and Le Boudec, 2004], des observations directes sont échangées entre les nœuds voisins et un agent ne va fusionner à ses propres observations les observations reçues que dans le cas où les opinions de ses voisins sont proches des siennes. Dans [Kamvar et al., 2003], afin d'agréger les confiances locales, un agent demande à ses voisins leur avis sur les autres agents (dans le cas de EigenTrust, des pairs). Les opinions des voisins sont ensuite pondérées par la confiance que l'agent à en eux. [Marti and Garcia-Molina, 2004, Xiong and Liu, 2004] propose des algorithmes similaires permettant d'évaluer la confiance en combinant les opinions d'un groupe d'agents sélectionné. D'autres approches, tels que [Jiang et al., 2005, Jiang and Baras, 2006], sont basées sur des règles d'interaction locale en utilisant la théorie des graphes.

Dans mon modèle, les valeurs de confiance intrinsèques peuvent être agrégées en utilisant à la fois les confiances directes et indirectes stockées dans un réseau de confiance (appelé TrustGraph) ce qui combine la propagation de la confiance le long d'un chemin et

la combinaison des confiances sur différents chemins.

Modèle de Schillo et al. Un des premiers modèles de confiance basé sur les réseaux a été proposé par [Schillo et al., 1999]. Dans ce modèle, le résultat d'une interaction entre deux agents (du point de vue de la confiance) est une expression booléenne : bon ou mauvais, il n'y a pas de degré de satisfaction. Le scénario que l'auteur utilise est un dilemme du prisonnier modifié, se déroulant comme une enchère. Un agent évalue l'honnêteté de ses voisins en observant leur comportement qui est stocké dans une structure de données appelée TrustNet (Figure 2.17). TrustNet est un graphe orienté où les nœuds représentent les agents (témoins) et les arrêtes portent les informations sur les observations que l'agent dit au propriétaire du graphe. Les agents communiquent à la fois des informations factuelles et les confiances qu'ils ont sur les autres agents. La valeur de confiance finale d'un agent envers un autre agent est alors une combinaison des expériences directes et des témoignages des autres agents.

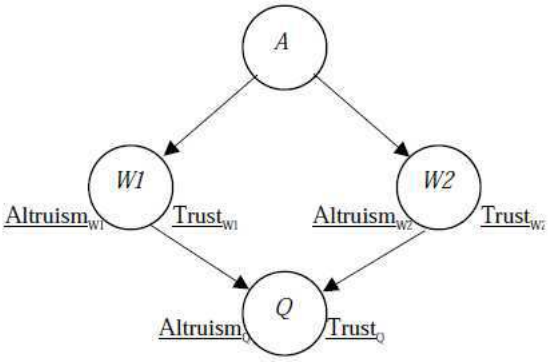


FIGURE 2.17 – Exemple de TrustNet [Schillo et al., 2000]

Modèle de Yu et Singh. [Yu and Singh, 2000] a étudié la gestion de la réputation distribuée dans un réseau social d'agents où les agents calculent la fiabilité des autres agents pour coopérer avec eux. Ils utilisent la théorie Dempster-Shafer pour représenter la confiance de l'agent en des agents fournisseur de services. Quand un agent veut trouver la fiabilité d'un fournisseurs de services, il utilise le réseau de référence pour trouver des témoins, et combine les confiances de ces témoins dans les fournisseurs de services. La figure 2.18 montre comment les témoignages se propagent à travers le réseau de confiance. Afin de garder le graphe limitée (plus la chaine de référence est longue, moins l'information obtenue est fiable), ils ont introduit une limite de profondeur du graphe.

Il y a certaines limitations à cette approche. Tout d'abord, un service de qualité moyenne est considéré comme une qualité inconnue au lieu d'une qualité moyenne connue. Deuxièmement, tous les témoins sont considérés comme des agents de même type alors qu'ils sont des agents différents. Troisièmement, ce modèle ne combine pas les informations directes et les informations indirectes en même temps (si l'information directe est disponible, seule cette source est utilisée pour déterminer la confiance de l'agent cible).

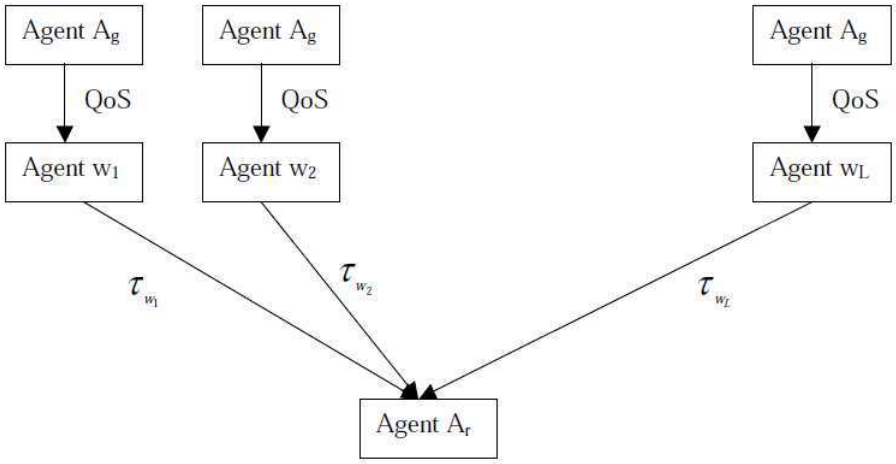


FIGURE 2.18 – Propagation de témoignages par des témoins [Yu and Singh, 2002]

Modèle de Mui et al. [Mui et al., 2002a] propose un modèle de confiance basé sur des réseaux Bayésiens. La réputation est considérée comme la probabilité de succès variant de 0 à 1. La confiance est définit comme une quantité dyadique entre le constituant et le fiduciaire, qui peut être déduite à partir des données de réputation sur le fiduciaire. La réputation est propagée via le réseau de communication de l'agent (Figure 2.19). Afin d'éliminer les difficultés avec des références circulaires, ce réseau bayésien a été limité à la profondeur 1. Une fonction de sommation pondérée est utilisée pour agréger les différents valeurs de réputations. Cependant Mui et al. n'examinent pas les effets des agents non fiables dans ce modèle.

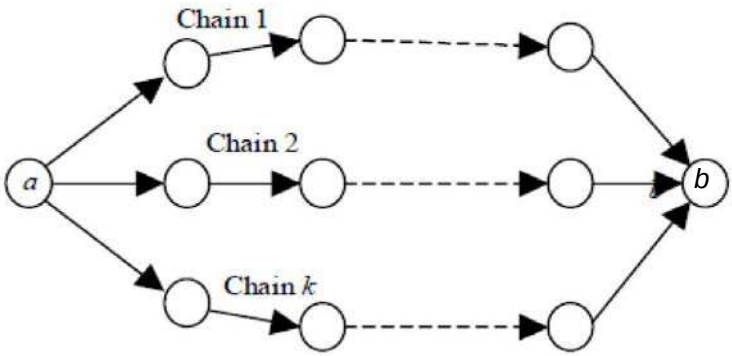


FIGURE 2.19 – Illustration d'un réseau parallèle entre deux agents a et b [Mui et al., 2002a]

Modèle de Richardson. [Richardson et al., 2003] a proposé une gestion de la confiance pour le web sémantique. Chaque utilisateur maintient une valeur de confiance dans un petit nombre d'utilisateurs. La confiance d'un utilisateur envers les autres peut être calculée via les informations de webs de manière récursive comme suit : (i) énumérer tous les chemins entre l'utilisateur et tous les autres utilisateurs qui ont une confiance locale dans un énoncé donné, (ii) calculer la confiance associée à chaque chemin d'accès en utilisant une fonction de concaténation prédéfinie le long de chaque chemin et la croyance de l'utilisateur final, (iii) les confiances associées à tous les chemins peuvent être combinées par une fonction d'agrégation prédéfinie. La Figure 2.20 illustre certaines fonctions de concaténation possibles comme la multiplication et la valeur minimale. Toutefois, ce modèle pourrait bien demander des coûts élevés de temps et de ressources pour localiser les témoins.

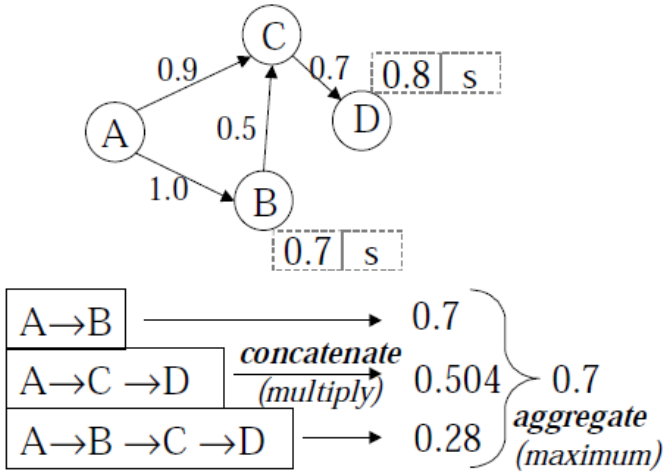


FIGURE 2.20 – La fusion de la confiance sur le Web utilisant l'algèbre chemin [Richardson et al., 2003]

Modèle F.I.R.E. [Huynh et al., 2006] a introduit un modèle de confiance, FIRE, qui considère les confiances directes et indirectes pour évaluer la fiabilité des agents et permet de choisir de bons partenaires pour les interactions. FIRE utilise les résultats de l'expérience directe, des relations basées sur les rôles, des rapports de témoins et de tiers de référence pour fournir des mesures de confiance. La réputation des témoins est obtenue à partir du réseau social de l'agent en suivant la méthode proposée par [Yu and Singh, 2000] (Figure 2.21) et à chaque témoin est assigné un poids. Par conséquent, les requêtes doivent se propager à travers le réseau afin de calculer la réputation des témoins, ce qui implique un coût plus élevé en terme de quantité d'informations transmises.

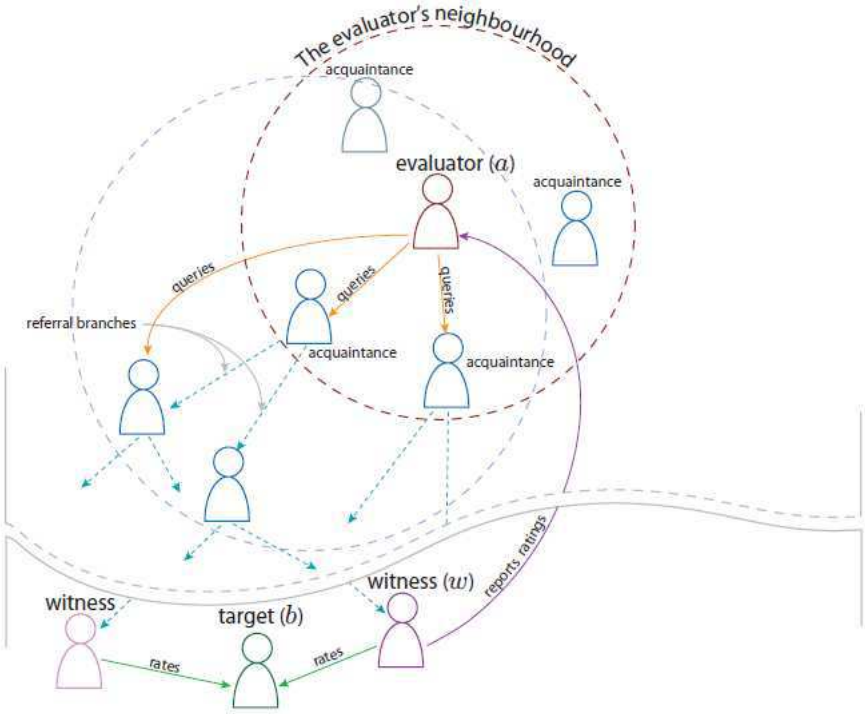


FIGURE 2.21 – Processus de références pour l'évaluation la réputation des témoins [Huynh, 2006]

Je pense que ce modèle est le plus complet, combinant les avantages des modèles de Sabater et Sierra [Sabater and Sierra, 2002] et de l'approche décentralisée par réseau de référence de [Yu and Singh, 2000]. Il a plusieurs points communs avec notre modèle. Le premier est que les expériences directes peuvent être à notre confiance directe puisque les agents utilisent leurs propres expériences pour évaluer la cible eux-mêmes. Le second est les rapports de témoins et les tiers références qui sont semblables à notre confiance indirecte. Confiance basée sur des relations fondées le rôle ne seront pas utilisées dans notre modèle puisque les rôles de nos agents sont les mêmes.

Modèle SinAlpha [Urbano et al., 2009] présente le modèle SinAlpha, un modèle de confiance se basant sur trois propriétés importantes de la dynamique de la confiance : l'asymétrie (elle stipule que la confiance est difficile à gagner et facile à perdre), la distinction des preuves du passé et la considération des phases de maturité distincts sur le comportement des agents cibles. L'auteur propose d'un moteur d'agrégation non statistique qui utilise une courbe en forme de S pour calculer les scores de confiance (Figure 2.22), en tenant compte de ces trois propriétés. L'utilisation d'un moteur d'agrégation qui englobe les expériences passées de l'agent fiable en tenant compte de la dynamique fondamentale de la confiance permettant une meilleure estimation de la fiabilité de l'agent que les approches probabilistes et statistiques qui existent dans la littérature. Cependant, l'auteur ne montre pas comment calculer la confiance le long d'un chemin.

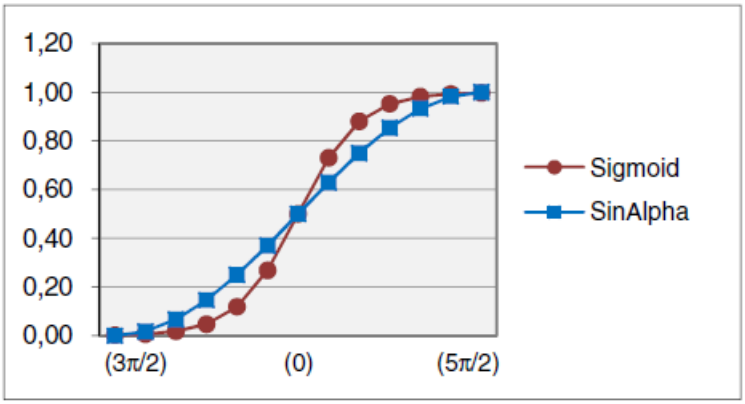


FIGURE 2.22 – Deux courbes en forme de S, une exponentielle (Sigmoïde) et une trigonométriques (SinAlpha) [Urbano et al., 2009]

Modèle L.I.A.R. L.I.A.R. ("Liar Identification for Agent Reputation") [Vercouter and Muller, 2010] est un modèle de confiance pour la mise en œuvre d'un contrôle social des interactions entre agents appliqué à un réseau pair-à-pair. Un contrôle social est établi par un mécanisme de gestion de la confiance. Il fournit un formalisme pour représenter les règles de communication qui devraient être respectées dans le système. Il donne également des outils et des modèles pour les agents afin de raisonner sur les interactions avec les autre agents, de détecter s'ils violent les règles. Cinq types de réputation sont proposés : la réputation basée sur des interactions Directes (DIbRp), la réputation basée sur des Interactions Indirectes (IIbRp), la réputation basée sur des Recommandation d'Observations (ObsRcbRp), la réputation basée sur des Recommandation d'Évaluations (EvRcbRp), la Réputation basée sur des Recommandations de Réputation (RpRcvRp). Tout cela est utilisé par l'agent pour décider s'il fait confiance ou non à un autre agent. Selon les auteurs une agrégation en une valeur unique n'est pas souhaitable. Ils proposent une approche de conservation des différentes évaluations selon la provenance des informations. La figure 2.23 représente la manière dont un agent utilise les réputations pour créer une intention de faire ou non confiance.

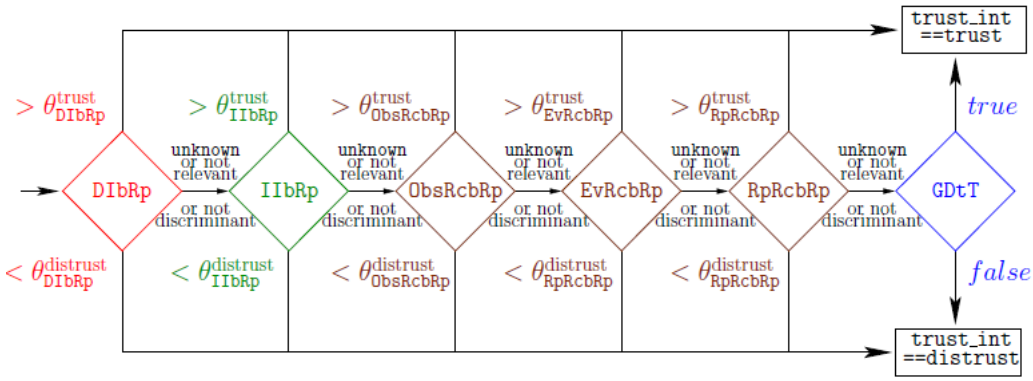


FIGURE 2.23 – Processus de raisonnement utilisant le modèle L.I.A.R [Vercouter and Muller, 2010]

2.5.4 Discussion

La confiance a été étudié depuis une longue période par des chercheurs de divers domaines scientifiques. Chaque domaine s'est intéressé à la modélisation, et au calcul de la confiance en utilisant des techniques différentes telles que le vote, les probabilités, la logique floue, la pondération, l'intelligence en essaim, etc. De mon point de vue, une des techniques les plus prometteuses est le modèle proposé par [Schillo et al., 1999, Schillo et al., 2000].

Dans ce modèle, les agents communiquent à la fois des informations factuelles et les confiances qu'ils ont sur les autres agents. Le modèle est basé sur la théorie des probabilités en proposant le jeu du dilemme du prisonnier pour calculer la valeur de confiance d'un partenaire. Mon travail est basé sur ce modèle. Dans la suite, je distingue deux types d'informations : les **données** (informations recueillies par les agents dans l'environnement) et les **métadonnées** (informations sur les autres agents, c'est-à-dire les confiances). Mes agents, comme ceux de Schillo, sont capables d'échanger à la fois des données et des métadonnées. Les agents peuvent ainsi construire un réseau de valeurs de confiance, appelé TrustNet chez Schillo *et al.* La valeur de confiance finale d'un agent envers un autre est alors une combinaison des expériences directes et des témoignages des autres agents.

J'ai choisi d'utiliser le TrustNet comme base de travail pour deux raisons.

- Premièrement, dans des systèmes de collecte d'informations distribués, les agents doivent partager toutes leurs données, c'est-à-dire à la fois celles qu'ils collectent eux-mêmes (directes) et celles qu'ils reçoivent des autres agents (indirectes). Dans ce contexte, nous soutenons que la pénalité (en termes de diminution de confiance) ne doit pas être attribuée à l'agent qui a transmis de fausses informations mais à l'agent défectueux qui a lui-même collecté ces informations. C'est pourquoi le chemin parcouru par les données nous parait important.
- Deuxièmement, comme nous considérons un système distribué sans contrôle centralisé ou base de données commune des informations, la communication s'effectuera essentiellement de manière directe entre deux agents. Ainsi, un système de réputation centralisé classique ne peut pas être utilisé dans ce cas. Donc l'utilisation du TrustNet à la place d'une simple valeur de confiance nous permet de travailler sur des confiances qu'on pourrait qualifier à la fois de locales et distribuées. Les agents peuvent calculer les confiances en se basant sur l'expérience directe (ce qui leur fournit une confiance directe sur les autres agents) et sur les expériences des autres agents (confiances indirectes), qui ne sont pas représentées ici sous la forme de réputation, mais par des confiances portées par le TrustNet.

Pour améliorer la robustesse du système je vais utiliser une approche auto-organisationnelle basée sur la confiance. Dans ce sens, je vais commencer par présenter l'auto-organisation et la robustesse dans les SMAs dans la partie suivante.

2.6 Auto-organisation et robustesse d'un système d'information

J'introduis dans cette section la notion d'auto-organisation pour les systèmes complexes immergés dans un environnement dynamique. L'auto-organisation des parties d'un système peut être vue comme un moyen d'adaptation du système de manière autonome, pour faire face à des perturbations de l'environnement. Je présente tout d'abord une définition de l'auto-organisation issue de la littérature (Section 2.6.1), puis le lien entre l'auto-organisation et la robustesse du système (Section 2.6.2) et enfin les travaux sur l'auto-organisation dans les SMA (Section 2.6.3).

2.6.1 Auto-organisation : définitions

Dans une vision systémique qui est la mienne tout au long de cette thèse, un système présente une forme d'**organisation** si les composants sont disposés de manière non-aléatoire, cette disposition se faisant au moyen de processus internes ou externes. **L'auto-organisation** pourraient ainsi être définie comme la capacité d'un système à atteindre une organisation, par le biais d'un processus interne, sans aucune forme (implicite ou explicite) de contrôle ou d'intervention externe.

Le concept d'un système d'auto-organisation a été introduit en 1947 par le psychiatre et ingénieur [Ashby, 1947]. Dans les années 1950, un système auto-organisé était considéré comme un système modifiant sa structure en fonction de son expérience et de son environnement [Banzhaf, 2009]. Ce terme a été utilisé par Farley et Clark en 1954 pour décrire l'apprentissage et les mécanismes d'adaptation [Farley and Clark, 1954]. Ashby, en 1965, a redéfini un système auto-organisé afin d'inclure l'environnement dans le système approprié [Ashby, 1965]. Ce terme ne devient commun dans la littérature scientifique que lors de son adoption par les physiciens et autres chercheurs du domaine des systèmes complexes dans les années 1970 et 1980. Cette notion a été développée par Haken [Haken, 1977] en 1977 qui appelle la coopération globale des éléments d'un système dynamique - résultant en l'assumant un état d'attracteur (auto-organisation). Edgar Morin, dans les années 1980, voit l'auto-organisation comme un « *désordre organisateur* » (cf. Figure 2.24) et il la définit par :

> *« Une propriété d'un système rendant compte de sa capacité à transformer et se transformer, et produire et se produire, et relier et se relier, et maintenir et se maintenir »* [Edgar, 1981]

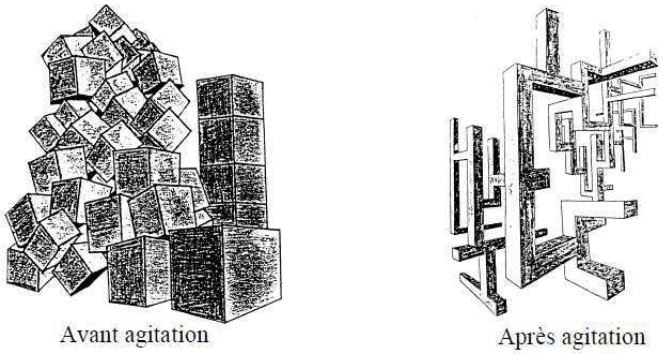


FIGURE 2.24 – L'auto-organisation (figure reprise de l'article de H. Von Foerster [von Foerster, 1960])

Ces dernières années, l'auto-organisation des systèmes ont pris une place centrale dans la sciences naturelles ([Camazine et al., 2003, Sole and Bascompte, 2011]), les sciences sociales ([Luhmann et al., 1996, Focardi et al., 2002]) et les systèmes multi-agents (SMAs) ([Holland and Melhuish, 1999, Anderson, 2002, Marzo Serugendo et al., 2005, Picard et al., 2009]).

Pour les SMAs, l'auto-organisation définie par [Marzo Serugendo et al., 2005] est proche de celle de mes travaux. Les auteurs donnent la définition suivante :

> *« L'auto-organisation est définie comme le mécanisme ou le processus permettant à un système de modifier son organisation sans commande externe explicite lors de son temps d'exécution ».*

En se basant sur cette définition, on peut observer plusieurs niveaux d'auto-organisation, en fonction du degré d'explicitation du processus interne. Le niveau le plus faible de l'auto-organisation est celui où le processus interne est entièrement explicite, alors que le niveau le plus fort est celui où ce processus interne est implicite et produit par la dynamique interne du système.

2.6.2 Lien entre auto-organisation et robustesse

Systèmes auto-organisés. Selon [Banzhaf, 2009], le terme « les systèmes auto-organisés » se réfère à une classe de systèmes capables de modifier leur structure interne et leur fonctionnement en réponse aux circonstances extérieures. Par l'auto-organisation, les éléments d'un système sont capables de manipuler ou d'organiser d'autres éléments du même système de manière à stabiliser la structure ou la fonction du système pour contrer les fluctuations externes. Afin de décrire plus en détail ce qu'est un système auto-organisé, examinons les éléments d'un système auto-organisé proposés par [Marzo Serugendo, 2009] comme les suivants :

- **l'environnement** dans lequel le système évolue (système d'exploitation, le réseau, etc.)
- les autonomes individuels des entités actives **- les agents -** qui constituent le système lui-même
- **le mécanisme d'auto-organisation** définissant des règles que tous les agents peuvent appliquer dans leur environnement et leur permettant de ré-organiser en cas de changements ou des défauts
- **les objets façonné**s, qui sont les entités passives tenues par l'environnement, sont créés, modifiés ou détectés par les agents (par exemple phéromones propagés dans l'environnement ou les informations échangés entre les agents)

En se basant sur les éléments d'un système auto-organisé ci-dessus, mon système auto-organisé construit s'appuie sur les éléments suivants :

- les agents déploient sur l'environnement physique qui représente le **territoire** (espace physique dans lequel évoluent les agents)
- **les agents** peuvent se déplacer et communiquer avec d'autres agents pour collecter plus les informations fiables possibles
- **le mécanisme d'auto-organisation** basé sur le couplage de l'organisation spatiale et l'organisation sociale des agents. Ce sont les règles prédéfinies que les agents peuvent appliquer lorsqu'ils évoluent dans l'environnement
- les agents échangent les informations (**les objets façonnés**) avec d'autre d'agents via la communication

Je présente dans la suite la perturbation et les types des fautes possibles dans les systèmes auto-organisés.

Perturbations et types des défauts dans les systèmes auto-organisés. [Marzo Serugendo, 2009] donne une liste de fautes qui peuvent avoir lieu dans un système auto-organisé (Figure 2.25). Ce sont les défauts de l'environnement, de l'agent, du mécanisme de l'auto-organisation et des objets façonnés. Je résume par la suite les caractéristiques de chaque type de fautes.

- Défaut de l'environnement : il inclut tous les défauts liés au réseau et les défauts de communication entre les agents, les défauts de fonctionnement des entités de calcul présentes dans l'environnement et tout les défauts de stockage (base de données ou de la mémoire)
- Défaut de l'agent : les agents défauts peuvent être des défauts physiques (matériel ou défaut logiciel) ou des anomalies de développement du système
- Défaut du mécanisme de l'auto-organisation : ce sont les défauts de développement, les erreurs dans la conception et dans la mise en œuvre les règles du mécanisme de l'auto-organisation.
- Défaut des objets façonnés : ce sont les défauts causés par l'environnement, les agents ou le mécanisme de l'auto-organisation, puisque les objets façonnés eux-mêmes sont essentiellement passifs.

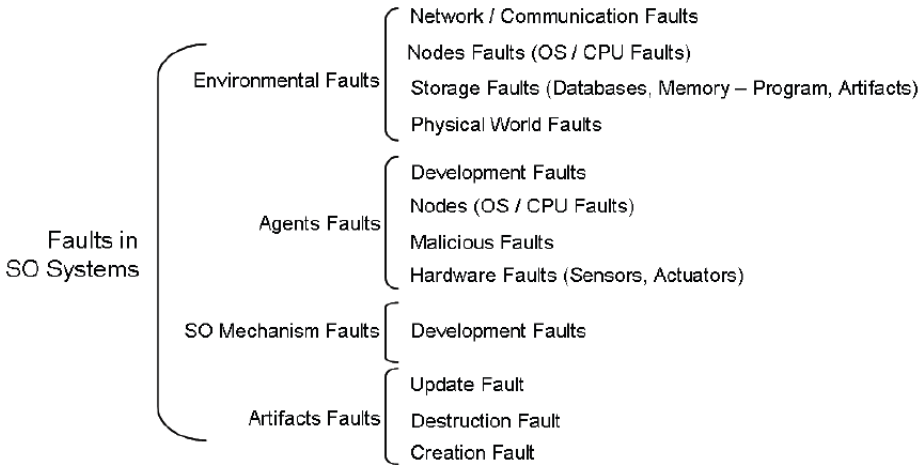


FIGURE 2.25 – Caractérisation des types de défauts dans les systèmes auto-organisés ([Marzo Serugendo, 2009])

On a définit les éléments de conception constituant d'un système auto-organisé (l'environnement, les agents, le mécanisme d'auto-organisation et les objets façonnés) dans la partie précédente. La perturbation sera définie comme tous les défauts ou les modifications du système provenant de ces éléments.

Dans mon cas, la perturbation provient des défauts de capteur de l'agent (défaut de l'agent). Les agents sont incapables de collecter correctement des informations sur l'environnement. Par conséquence, ils ne peuvent pas détecter correctement le niveau de danger d'une zone, comme si le spectre de leurs capteurs avait été décalé.

Lien entre auto-organisation et robustesse. La robustesse est une propriété importante pour un grand nombre de systèmes organisés. La plupart de ces systèmes sont complexes et utilisent la redondance pour atteindre la robustesse malgré les perturbation externes [Banzhaf, 2009]. Giovanna Marzo Serugendo, donne dans [Marzo Serugendo, 2009], une analyse de la robustesse des systèmes organisés aux perturbations (des pannes ou des changements de l'environnement). Selon elle, lorsque nous concevons un système artificiel, nous souhaitons qu'il soit auto-organisationnel afin de bénéficier des propriétés de l'auto-organisation telles que la robustesse, l'adaptabilité, etc. Les propriétés de robustesse sont détaillés par la convergence (une propriété importante des systèmes auto-organisés pour savoir si le système converge effectivement vers le but souhaité), la vitesse de convergence (à quelle vitesse le système atteint son objectif ?), la stabilité (une fois que l'objectif est atteint, est-ce que le système peut le maintenir ?) et l'évolutivité (comment le système est affecté par le nombre d'agents ?).

C'est aussi mon objectif d'utiliser l'auto-organisation dans un système pour améliorer sa robustesse. Dans mon système, les propriétés de robustesse sont les problèmes suivants et sont examinés dans le chapitre 6 :

- La convergence : les agents du système peuvent-ils obtenir une carte exacte ?
- La vitesse de convergence : le temps maximal d'exploration nécessaire pour que tous les agents obtiennent une carte exacte
- La stabilité : une fois que tous les agents obtiennent une carte exacte, peuvent-ils la maintenir ?
- L'évolutivité : comment les différents paramètres de la simulation influencent la robustesse du système ? (par exemple le taux de robot défectueux, etc.)

Mécanismes de l'auto-organisation. Le mécanisme d'auto-organisation est une composante importante dans les systèmes organisés. Les chercheurs ont développé diverses approches pour ces mécanismes de l'auto-organisation [Marzo Serugendo et al., 2006] :

- **Mécanismes basés sur des interactions directes** : un algorithme simple d'élection [Mamei et al., 2005], le système de modélisation de la dynamique des fluides

- **Mécanismes basés sur la stigmergie** : contrôle de fabrication [Karuna et al., 2005], gestion du réseau [Reitbauer et al., 2005], gestion de la sécurité des réseaux informatiques [Foukia, 2005], coordination des véhicules sans contrôle [Van Dyke ParunaK et al., 2002], araignées sociales pour détecter des régions d'une scène [Bourjot et al., 2003]
- **Mécanismes basés sur le renforcement** : le modèle d'agents adaptatifs [Weyns et al., 2004], système social [Sansores and Pavón, 2008]
- **Mécanismes basés sur la coopération** : la théorie des AMAS [Gleizes and Camps, 1999, Capera et al., 2003], framework OSD [Ishida et al., 1992]
- **Mécanismes basés sur une architecture générique** : les architectures de référence génériques [Maturana and Norrie, 1996, Bongaerts, 1998], méta-modèles [Dowling and Cahill, 2001]

Dans mon travail, le mécanisme de l'auto-organisation s'inscrit dans une méthodologie de développement de systèmes informatiques auto-organisants développé par [Hassas, 2003] : des agents réactifs utilisent l'environnement comme moyen d'interaction et aussi comme support d'inscription des effets de leurs actions. Le mécanisme de l'auto-organisation est basé sur le couplage de l'organisation spatiale et l'organisation sociale des agents. Plus précisément, je propose une démarche d'auto-organisation basée sur la confiance, puis établie un système multi-agents qui est capable dynamiquement de définir ou de faire émerger des stratégie de déplacement ou de communication automatiquement adaptée à la perturbation. Dans la section suivante, je dresse un état de l'art sur les différentes approches récentes relatives à l'auto-organisation dans les SMA.

2.6.3 Auto-organisation dans un SMA

L'activité de recherche sur l'auto-organisation, en particulier dans les SMA, est un champ fertile, où beaucoup de travaux existent [Marzo Serugendo et al., 2005]. Les chercheurs ont expérimenté de nombreux mécanismes conduisant à auto-organisation sur différents types d'applications pour résoudre des problèmes complexes [Bernon et al., 2006].

Gleizes et ses collègues [Gleizes and Camps, 1999, Capera et al., 2003, Marzo Serugendo et al., 2005] ont proposé une approche théorique et pratique pour la conception de systèmes auto-adaptatifs artificiels plongés dans un environnement dynamique. Elle est basée sur la théorie des AMAS et sur la méthode de conception ADELFE. L'idée principale est que le comportement des agents est défini sans aucune connaissance de la fonction globale réalisée par le système. Cette approche est soutenue par le théorème des AMAS [Gleizes, 2004] : *"Pour tout système fonctionnellement adéquat, il existe un système à milieu intérieur coopératif qui réalise une fonction équivalente dans le même environnement"*. Cette méthode est appropriée lorsque des évènements imprévisibles se produisent réellement dans l'environnement.

Tom de Wolf et Tom Holvoet, proposent dans [De Wolf and Holvoet, 2004, De Wolf and Holvoet, 2005], une méthodologie qui construit un cycle de vie complet basé sur les processus unifiés. Cette méthodologie est personnalisée pour se concentrer explicitement sur le comportement macroscopique des SMAs auto-organisés. À la phase d'analyse, l'identification des propriétés macroscopiques doit être indiquée par le système d'exécution. Puis, la phase de conception contient deux étapes particulières : une pour décider d'utiliser ou non un système auto-organisé et l'autre pour l'exploitation des pratiques et des expériences existantes. À la phase de vérification et de test, une approche empirique basée sur une rétroaction itérative de développement est proposée. Cette technique donne des résultats plus fiables sur le comportement du système que la simple observation des résultats de simulation, à un coût abordable de calcul. Les algorithmes agissent au niveau du système et orientent les simulations. Cependant, cette méthode ne fournit pas d'outils afin de choisir l'approche existante pour coder l'auto-organisation.

[Gershenson, 2007] a proposé une méthodologie indépendante du domaine pour la conception et le contrôle des systèmes auto-organisés. La méthodologie proposée reçoit les demandes d'un système (ce que le système devrait faire) et permet au concepteur de produire un système qui satisfait tous les demandes. Cette méthodologie itérative et incrémentale comprends une série d'étapes à suivre : la représentation, la modélisation, la simulation, l'application et l'évaluation qui sont des étapes interdépendantes (Figure 2.26). L'objectif principal de cette méthode est de distribuer le contrôle dans le but d'influencer le système (en réduisant la friction et la promotion des synergies) et d'assurer la production des comportements souhaités. Même si ce travail est principalement destiné aux ingénieurs, il est plutôt philosophique et vise à comprendre et concevoir les systèmes complexes. Ce travail ne doit pas être considérée comme une

recettes qui fournit des solutions concrète mais plutôt comme un guide pour rechercher des solutions.

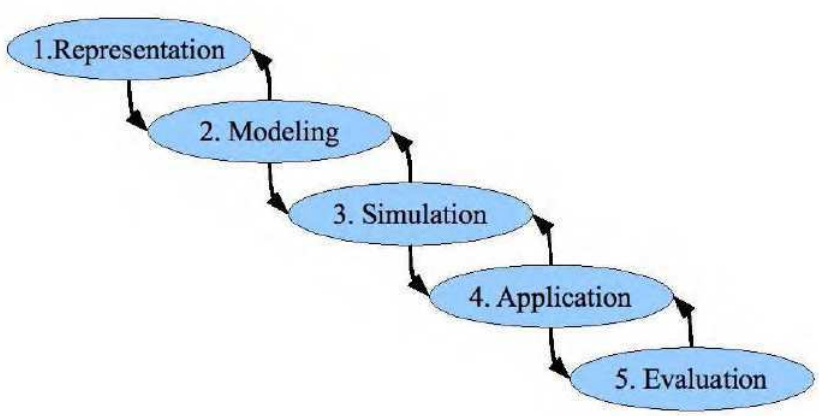


FIGURE 2.26 – Différentes étapes de la méthodologie [Gershenson, 2007].

[Abdallah and Lesser, 2007] propose une nouvelle méthode d'auto-organisation qui non seulement permet aux agents de s'auto-organiser en un réseau situé au cours du processus d'apprentissage mais aussi d'utiliser les informations apprises pour guider le processus d'auto-organisation. En particulier, en utilisant ce mécanisme, un agent peut ajouter et supprimer des agents voisins tout en continuant à apprendre. Le mécanisme utilise des heuristiques pour transférer les connaissances apprises à travers les différentes étapes de l'auto-organisation. Ce travail a été appliqué au problème de la répartition des tâches distribuées (Figure 2.27). Des résultats expérimentaux montrent une amélioration significative de la performance du système grâce à l'auto-organisation.

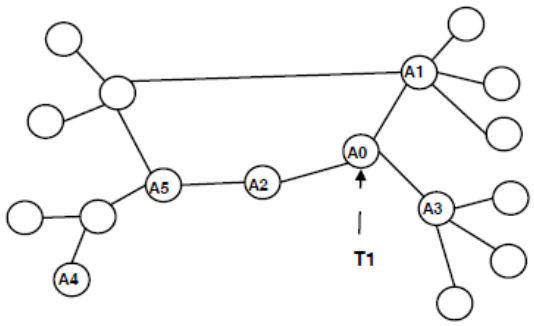


FIGURE 2.27 – Répartition des tâches en utilisant un réseau d'agents [Abdallah and Lesser, 2007].

[Zhang et al., 2009] a proposé un cadre de supervision utilisant le contrôle périodique de l'organisation pour coordonner et orienter l'apprentissage des agents. Ce travail définit une structure multi-niveau d'organisation et un protocole de communication pour échanger des informations entre les agents au niveau inférieur (ou subordonnés) et des agents de supervision au niveau supérieur (ou superviseurs) au sein d'une organisation. Comme l'illustre de la figure 2.28, les subordonnés rapportent leur état abstrait et leurs récompenses à leurs superviseurs, qui à leur tour génèrent et transmettent l'information de surveillance. Le cadre de supervision a également précisé une politique d'adaptation de surveillance et a intégré les informations de surveillance dans le processus d'apprentissage pour guider l'exploration des subordonnées dans leur espace d'état. Cependant, la limite du cadre de la surveillance est que l'organisation hiérarchique donnée est fixe, les auteurs ne précisent pas comment les organisations de supervision sont formées.

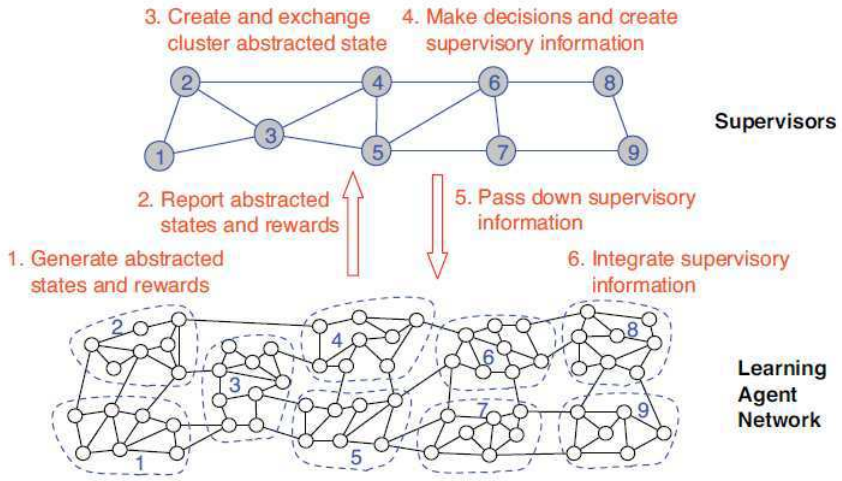


FIGURE 2.28 – Processus de supervision dans un cadre du contrôle[Zhang et al., 2009].

[Sansores and Pavón, 2008] présente une approche de l'auto-organisation pour modéliser des systèmes sociaux auto-organisés dont les individus présentent une adaptation aux circonstances changeantes dans l'environnement et sont dotés de mécanismes d'évaluation des options possibles et décident des actions à faire. Afin de modéliser ces caractéristiques, ils proposent une architecture d'agent basé sur le but dynamique selon la théorie de l'action sociale [Castelfranchi and Paglieri, 2007]. Basés sur cette théorie, les buts disposent d'un un cycle de vie. L'auteur utilise cette perspective pour inclure un état de motivation à ces buts pour augmenter ou diminuer les motivations des agents pour la poursuite de leurs buts. L'approche est complétée par un mécanisme permettant à un agent d'adapter son comportement dynamiquement (en jouant des rôles différents) en réponse à une rétroaction de sa propre expérience. Cette rétroaction est fournie par les interactions précédente avec d'autres agents. L'auto-organisation proposée par ce modèle est fondée sur la capacité des agents à changer leur comportement dynamiquement par apprentissage par renforcement. La figure 2.29 décrit tous les éléments de ce modèle d'auto-organisation. Cependant, la plupart des éléments de cette approche conduisent à des ambigüités de conception. Je pense que cette méthode est mieux adaptée aux petites sociétés d'agents.

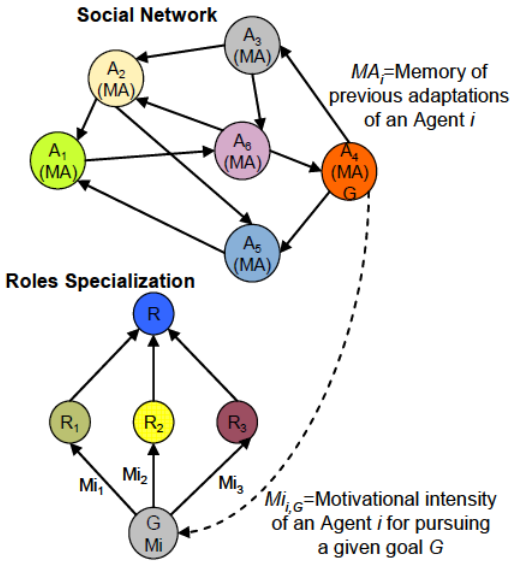


FIGURE 2.29 – Éléments du modèle d'auto-organisation [Sansores and Pavón, 2008].

[Lefevre et al., 2009] décrit un framework où l'ensemble du système et ses composants sont définis par leur structure d'accouplement avec leurs environnements respectifs. Cette approche suit l'approche décrite dans [Hassas, 2003]. Elle est basée sur la représentation d'un problème comme étant un ensemble d'agents qui sont situés dans un environnement dans lequel ils ont des perceptions locales et sur lequel ils agissent à travers des actions locales. La résolution de problèmes est ainsi effectué par un comportement global qui se dégage des multiples interactions, qui se déroulent entre les agents et par l'environnement partagé. Ces interactions induisent des échanges complexes entre les agents. Un mécanisme d'auto-organisation est proposé en se basant sur les effets macroscopiques de l'activité microscopique (l'activité des agents). Ce mécanisme permet aux agents de percevoir les effets macroscopique et donc de guider leurs choix vers une solution globale satisfaisante.

Dans le travail de Ye et al. [Ye et al., 2010], un mécanisme d'auto-organisation dans un réseau d'agents est proposé. Le but de ce mécanisme est de parvenir à une répartition des tâches efficace dans le réseau d'agents à travers la modification dynamique des relations structurelles entre les agents (i.e. modifier la structure de réseau). Ce mécanisme permet aux agents d'apprendre quelles relations modifier et comment faire en utilisant uniquement des informations locales. Ces informations locales sont accumulées à partir des interactions passées entre agents.

2.6.4 Discussion

On peut remarquer que la plupart des approches actuelles de l'auto-organisation sont orientées vers les mécanismes à mettre en place pour changer dynamiquement l'organisation du système (dynamique de réorganisation au moment de l'exécution), soit par le biais de mécanismes bio-inspirés ou par l'utilisation de règles de transformation spécifiées. Ces règles sont généralement définies au moment de la conception du système [Gleizes and Camps, 1999, Capera et al., 2003, De Wolf and Holvoet, 2004, Gershenson, 2007]. Récemment, une plus grande attention a été accordée à l'influence mutuelle entre la structure du système qui gère son niveau d'information et la dynamique du système qui gère son niveau comportemental [Abdallah and Lesser, 2007, Sansores and Pavón, 2008, Lefevre et al., 2009, Ye et al., 2010]. L'approche que je développe dans ma thèse est basée sur cette idée, tout en le poussant vers l'avant, en envisageant un couplage à 3 niveaux entre le système gérant l'information, celui gérant le comportement et un troisième contrôlant les deux précédente. Cette approche s'inscrit dans une méthodologie de développement de systèmes informatiques auto-organisants développé par [Hassas, 2003] . Ce cadre stipule que l'auto-organisation qui nait et se manifeste par le couplage de l'organisation spatiale et l'organisation sociale des agents. Ce couplage s'appuie sur les éléments suivants :

- Une représentation de l'environnement : les agents déploient sur l'environnement physique qui représente le territoire (espace physique dans lequel évoluent les agents).
- L'organisation sociale : l'ensemble des rôles sociaux (comportements de communication des agents) qui permettent à réaliser des tâches du système ; et des relations entre ces rôles.
- L'organisation spatiale : l'ensemble des rôles spatiaux (comportements de déplacement des agents) et relations entre ces rôles. Elle permet d'exprimer la réalisation de l'organisation sociale (des tâches à réaliser) dans l'environnement spatial pour tenir compte par exemple des contraintes de l'environnement physique.

Ces deux organisations émergent de la dynamique du système, tout en agissant sur cette dynamique qui découle elle-même de l'influence des règles locales et des contraintes de l'environnement. Grâce à cette démarche, malgré la perturbation, les agents peuvent optimiser l'exploration de leur environnement, mieux communiquer lors de partage des informations et mieux ajuster leurs comportements en fonction de l'objectif.

2.7 Conclusion

J'ai proposé dans cette partie une description générale des systèmes. Ces système sont généralement divisés en plusieurs typologies classes (systèmes ouverts, systèmes fermés sur leur environnement, systèmes naturels, systèmes artificiels, systèmes sociaux, systèmes organisés hiérarchiquement, systèmes en réseau, etc.) sur lesquels le contexte de mon recherche est présenté via la présentation des système de collecte

d'information perturbés. Un système sera dit perturbé si des éléments du système perturbent le système par la caption de fausses informations dans l'environnement, et leur propagation par la communication.

Les SMAs sont présentés ensuite comme un outil de modélisation, de mise en œuvre des systèmes complexes et des systèmes de collecte d'information en particulier de systèmes de collectes d'information.

Dans le cadre de ce travail de thèse, qui porte sur le maintien de la cohérence et la robustesse dans un SMA perturbé, je m'intéresse plus particulièrement aux moyens permettant s'adapter aux perturbations. J'ai choisi d'utiliser pour cela les notions de confiance et réputation ; un panorama de tell modèles dans le cadre des SMA est donc présenté. L'étude des diverses méthodes fait apparaitre deux types de modèles de confiance : des modèle simples de confiance (sans représentation explicite de l'environnement social) et d'autres basés sur les réseaux. J'ai proposé donc dans la chapitre 4 un modèle de confiance TrustSets basé sur les réseaux.

Les définitions de l'auto-organisation et des différents types de l'auto-organisation sont présentées ensuite, car il vont constitues dans ma thèse un puissant moyen pour renforcer la robustesse du système, adaptatif à la perturbation.

Deuxième partie

Contributions

Chapitre 3

Aperçu de l'approche proposée

Sommaire

Je présente brièvement dans ce chapitre l'approche que je propose pour assurer la cohérence et améliorer la robustesse d'un système de collecte d'information perturbé. L'approche proposée est composée de deux phases : dans un premier temps j'introduis un modèle original de confiance pour améliorer la cohérence de l'information collectée et dans un second temps, je propose un mécanisme d'auto-organisation basée sur la confiance pour renforcer la robustesse du système. Ces deux étapes sont détaillées respectivement dans les chapitres 4 et 5.

3.1 Description générale du modèle

3.1.1 Cadre théorique de l'approche développée

L'approche que je développe s'inscrit dans une vision systémique, enactive et auto-organisationnelle.

1. La vision systémique consiste à considérer le système dans son intégralité (selon une vision holiste) en considérant les interdépendances entre ses différents composants. Ainsi le système global à concevoir doit considérer les influences mutuelles entre les activités de collecte et de communication et doit s'intéresser à leurs co-évolutions et leurs influences mutuelles et rétro-actives.
2. La vision enactive consiste à considérer le couplage structurel existant entre le système et son environnement (le système structure l'environnement, tout en étant structuré par lui). Par exemple, les effets de la collecte d'information doivent influencer les comportements de collecte du système.
3. La vision auto-organisationnelle consiste à considérer le système à travers son organisation et s'intéresser aux mécanismes internes qui mènent le système vers une organisation, sans que cette organisation ne soit dictée par une entité externe au système.

3.1.2 Hypothèses

Mon approche se base sur les hypothèses suivantes :

- Hypothèse 1 : bien que mon approche puisse être étendue pour travailler dans un environnement ouvert, je me limite ici à un environnement fermé (en particulier, tous les agents définis au départ. Néanmoins, les agents ne connaissent pas initialement les agents crées d'où la possibilité d'étendre l'approche à un système ouvert).
- Hypothèse 2 : les informations dans l'environnement restent inchangées, l'environnement est donc considéré statique ;
- Hypothèse 3 : je ne travaille que sur des agents défectueux (et non sur des agents menteurs) définis comme des agents incapables de collecter correctement des informations sur l'environnement. Par exemple, ils ne peuvent pas détecter correctement le niveau de danger d'une zone, comme si le spectre de leurs capteurs avait été décalé.
- Hypothèse 4 : les agents peuvent stocker des informations. Ils peuvent "*se déplacer*" dans l'environnement (changer de position dans un environnement réel, aller sur des sites internet ou aller là où il y a l'information, etc.). On suppose que chaque agent a une perception locale (avec un rayon de perception) et que la communication dans le système est également locale, chaque agent a donc un rayon de communication donné. L'énergie des agents est illimitée.

Il est important de noter que, bien que l'environnement soit considéré statique, le comportement des agents peut évoluer au cours de la simulation. Un agent correct peut par exemple être victime d'une panne et devenir défectueux.

3.1.3 Résumé de mon approche

Le premier aspect de mon travail est la construction d'un modèle de confiance appelé TrustSet. Dans le TrustSet, je considère que les agents conservent en mémoire les données sur l'environnement qu'ils ont eux-mêmes recueillies (appelées « informations directes » dans la suite) et les données sur l'environnement qu'ils ont reçues d'autres agents (appelées « informations indirectes »). A chaque étape, les agents se déplacent, rencontrent d'autres agents et échangent des informations sur l'environnement et sur la confiance qu'ils ont en les autres agents. En particulier, afin de calculer la fiabilité des données sur l'environnement, l'agent doit garder dans sa mémoire les agents qui ont collectés ces informations. La fiabilité d'un élément d'information est calculée à partir du nombre d'agents qui ont fournis la même information sur une position en fonction du nombre total d'agents ayant fournis une information sur cette position et la confiance dans chaque agent.

Le second aspect de mon travail est l'intégration d'un mécanisme d'auto-organisation basée sur le modèle de confiance TrustSet. Cette auto-organisation se base sur une vision systémique dans laquelle on considère un couplage structurel entre les deux niveaux : la collecte d'information directe et la communication. Plus précisément, un ensemble de règles locales de comportements est défini pour chaque

agent. La notion de confiance est intégrée dans ces règles. On peut distinguer trois catégories de règles : les règles de déplacement, les règles de communication et les règles de rétro-action entre la communication et le déplacement. La confiance aura un rôle de pression sélective conduisant à l'émergence de deux organisations qui co-évoluent de façons rétroactive : une au niveau social et l'autre au niveau spatial. L'organisation sociale reflète les relations de confiance établies entre les différents agents. L'organisation spatiale reflète le déploiement des agents dans l'environnement pour favoriser son exploration.

3.2 Introduction d'un modèle de confiance pour améliorer la cohérence de l'information

Le modèle de confiance TrustSet utilise le réseau des interactions entre agents pour calculer la confiance ainsi que la distance entre les informations collectées par l'agent et par les autres agents. C'est la raison pour laquelle je commence par présenter la représentation de l'information sur l'environnement (et les agents) avant de décrire comment les agents calculent les valeurs de confiance. Enfin, je discute l'impact de la confiance sur le système de communication.

3.2.1 Représentation des informations sur l'environnement et sur les agents

Dans un système de collecte d'information contenant des agents perturbés, les agents ont besoin de représenter et stocker deux types d'informations : les informations sur l'environnement et les informations sur les autres agents. Ces dernières informations sont nécessaires à l'agent pour qu'il raisonne sur les autres agents et détecte ceux qui sont défectueux. Pour les informations sur l'environnement, les agents doivent stocker l'objet sur lequel porte l'information (une localisation géographique par exemple) et l'information elle-même. De plus les informations sur l'environnement sont stockées dans deux bases différentes en fonction de la provenance de l'information : les informations directes (collectées par l'agent) et les informations indirectes (reçues d'autre agents). Cette dissymétrie tient au fait que l'agent va considérer les informations qu'il a collecté dans l'environnement comme la référence à partir de laquelle il va évaluer les autres agents. Il part du principe qu'il est un agent fiable, jusqu'à ce que les autres agents tendent à lui monter le contraire.

Les informations sur les agents sont stockées sous forme d'un graphe, pour les confiances communiquées par d'autres agents (TrustGraph) et dans un tableau (TrustTable) pour les confiances que l'agent attribue aux autres. Ces concepts seront décrits dans le chapitre 4.

3.2.2 Modèle de confiance : initiation et évolution

Dans le modèle de confiance TrustSet, les valeurs de confiance des agents sont mises à jour sous l'effet des interactions entre les agents. Lorsqu'un agent reçoit des informations d'un autre agent, il compare chacune des informations reçues avec ses propres informations pour ajuster la confiance qu'il a en cet agent. En se basant sur le résultat de cette comparaison, chaque agent va mettre à jour ses informations sur l'environnement et ses informations sur les agents du système.

Dans la mise à jour des informations sur l'environnement, il n'y a que les informations indirectes de l'agent qui vont être modifiées : les informations directes ne sont pas modifiées car elles contiennent seulement les informations recueillies par l'agent directement sur le terrain.

La mise à jour des informations sur les agents (c.à.d des confiances) sera également effectuée après chaque rencontre. L'agent remet à jour son réseau de confiance à partir des informations reçues, en intégrant de nouveaux agents dans le réseau si nécessaire. Puis il remet à jour son tableau de confiance.

3.2.3 Utilisation de la confiance

Le modèle de confiance est utilisé pour définir plusieurs mécanismes pour limiter l'impact des agents défectueux : un mécanisme d'auto-évaluation, une stratégie simple de communication, une méthode de calcul de la fiabilité de l'information et un algorithme pour la formation de coalitions d'agents fiables.

Auto-évaluation. L'objectif de l'auto-évaluation est de donner à chaque agent la capacité de procéder à un examen critique de sa situation actuelle (par exemple pour savoir si ses capteurs sont en panne ou non) basé sur les confiances des autres agents du système en lui-même. L'idée principale est de donner une méthode du calcul de la confiance de l'agent en lui-même.

Stratégie de communication. L'objectif d'une stratégie de communication consiste à séparer les agents peu fiables (avec lequel il est inutile ou nuisible de communiquer) des agents fiables (qui envoient des informations utiles à la communauté). Ce processus permet à un agent de renforcer la communication avec les agents fiables et de refuser les communications provenant d'agents jugés non fiables, afin de réduire la dissonance.

Calcul de la fiabilité de l'information. Comme les agents reçoivent potentiellement beaucoup d'informations contradictoires, il est important de les doter d'une méthode de calcul de la fiabilité des informations afin de pouvoir déterminer quelles informations sont correctes.

Formation de coalitions d'agents fiables. Pour améliorer la coopération entre les agents, minimiser les redondances de données et améliorer la qualité des informations échangées, on a proposé un mécanisme de formation de coalitions d'agents fiables. L'idée principale de cette approche est que l'organisation sociale du système (construite à partir des relations de confiance) se traduit dans une organisation spatiale par la formation de clusters spatiaux.

3.3 Utilisation d'un mécanisme d'auto-organisation basée sur la confiance pour améliorer la robustesse du système

Pour renforcer la robustesse d'un système de collecte d'information distribué je propose de développer un mécanisme d'auto-organisation basée sur la confiance. La confiance est un moyen qui permet de structurer les échanges au niveau social (interactions et échanges entre agents). Cette structuration peut avoir des incidences sur la structuration spatiale (ou plus généralement physique) des agents. Je vais présenter la mise en œuvre d'une démarche d'auto-organisation qui se base sur une vision systémique dans laquelle je considère un couplage structurel entre les deux sous systèmes : le sous-système de collecte d'information et le sous-système de communication.

3.3.1 Sous-système de collecte d'informations

On a caractérisé le sous-système de collecte d'information par un ensemble des règles de déplacement dans lequel la confiance est utilisée comme outil pour guider l'agent. Plus précisément, chaque agent va choisir un certain nombre de points de la frontière de sa zone connue et chercher le meilleur, compte tenu de ses objectifs, qu'il ciblera dans son futur déplacement. Ces règles de déplacement sont de trois types : les règles génériques, les règles associées au terrain et les règles associées aux agents.

Règles génériques. Ces règles génériques concernent le choix d'une cible quand l'agent n'a pas de cible ou quand plus de deux cibles ont la même attractivité (que l'on représente par un coefficient). Dans ces deux cas, le choix est aléatoire.

Règles associées au terrain. Les règles associées au terrain déterminent l'attractivité des cibles possibles en fonction de critères spatiaux (distance de la position actuelle de l'agent, par exemple) ou liés au renouvèlement des informations (nombre d'informations nouvelles ou fiables apportées par une cible). La distance spatiale, la fiabilité de l'information des candidats et le nombre de nouvelles informations possibles sont les trois éléments principaux qui vont influencer le coefficient d'attractivité lié à ces règles.

Règles associées aux agents. Ces règles permettent de calculer l'influence des autres agents sur le déplacement d'un agent. Elles sont sous la forme de forces d'attraction / répulsion. La confiance est l'élément principal pour déterminer ces forces. Plus l'agent a confiance en un autre agent, plus il va s'en rapprocher.

3.3.2 Sous-système de communication

La décision de communiquer pour un agent est régie par un ensemble de règles. On peut distinguer 3 niveaux différent de règles. Ces trois niveaux sont hiérarchiques dans le sens où les règles d'un niveau supérieur sont utilisées avant celles des niveaux inférieurs. Ces trois niveaux permettent donc de séparer les différentes règles en trois ensembles, chaque ensemble décrivant une certaine tendance à communiquer.

Tendance générale à communiquer. Chaque agent a une tendance générale à communiquer en fonction de sa situation par rapport à la cible vers laquelle il a choisi de se déplacer et plus généralement de la distance (au sens large) qui le sépare de sa cible. Cela permet à un agent de se déplacer sans trop communiquer à partir du moment où il a choisi une cible. Cela lui évite de changer de cible trop souvent et économise le temps de calcul.

Tendance à communiquer avec un agent. Si l'agent décide de communiquer, c'est à dire si la communication globale se déclenche (règle précédente), il va calculer une tendance à communiquer avec chacun des agents qu'il perçoit dans son environnement en fonction de la distance à ces agents, du temps de déconnexion avec chaque agent et de la nouveauté des l'information reçues. Cela permet à un agent de renforcer la communication avec des agents fiables. Cela lui évite de recevoir trop d'informations redondantes qu'il pourrait recevoir en communiquant plusieurs fois de suite avec les mêmes agents. Cela le fait également progresser plus vite vers son objectif.

Tendance à communiquer avec les informations. Un agent ne communique pas toujours la totalité de ses informations. On définit plusieurs règles destinées à sérier les informations (soit informations directes, soit informations indirectes) qu'un agent transmet à un autre. Cela permet à un agent de choisir un comportement robuste en fonction de la perturbation de l'environnement. Si l'agent détecte que son capteur est en panne, alors il ne transmet pas ses informations directes. Par contre, s'il y a beaucoup d'agents non fiables dans l'environnement, il n'envoie que ses informations directes, ses informations indirectes ayant une forte probabilité à être fausses.

3.3.3 Interaction entre communication et déplacement

Les deux sous-systèmes de collecte et de communication interagissent l'un avec l'autre de façon à ce que les comportements de collecte soient influencés par les comportements de communication et réciproquement. Cela se traduit par exemple par la définition de règles locales permettant de spécialiser les comportements en fonction des informations collectées.

3.3.4 Contrôle de l'auto-organisation

Toutes ces règles peuvent donner des résultats contradictoires. Des poids sont associés à chacune d'elles pour leur donner plus ou moins d'importance.

Les poids que j'ai affectés aux différentes règles sont donc autant de réglages sur lesquels je vais, en tant qu'utilisateur, pouvoir jouer de manière à «diriger» l'auto-organisation du système. Les poids associés aux règles peuvent être décomposés en trois ensembles : déplacement, communication et interdépendance entre la communication et le déplacement. Le nombre de combinaisons possibles est très important, donc la richesse du système de guidage de l'auto-organisation est très importante. Chaque combinaison représente un comportement possible pour un agent. Je définis par la suite un sous-ensemble de comportements archétype que j'essaierai ensuite de retrouver sous la forme de rôles dans l'auto-organisation qui se dessinera au fil des simulations.

3.4 Conclusion

Mes contributions sont donc de deux types.

1. Le modèle de confiance basé sur le réseau des interactions entre agents : TrustSet. Ce modèle inclut :
   - **une représentation des informations** sur l'environnement et sur les agents indépendant du problème ;
   - **un modèle de confiance**, nommé TrustSet, basé sur le réseau des interactions entre les agents et sur les informations échangées entre les agents ;
   - une méthode donnant aux agents la possibilité de calculer une confiance en eux-mêmes, leur permettraient de faire **un auto-diagnostic** sur leurs propres capacités et de détecter leur rôle en tant qu'élément perturbateur du système ;
   - **une stratégie de communication** consistant à séparer les agents peu fiables (avec lequel il est inutile ou nuisible de communiquer) des agents fiables (qui envoient des informations utiles à la communauté) ;
   - **une méthode du calcul de la fiabilité des informations** nécessaire car les agents reçoivent beaucoup d'informations contradictoires ;
   - **un mécanisme de formation de coalitions des agents fiables** permettant une meilleure coopération entre agents, une minimisation de la redondance des données et une amélioration de la qualité des informations échangées.
2. L'approche auto-organisationnelle basée sur la confiance. Cette auto-organisation est basée sur le découpage du système en deux sous-systèmes couplés par un système de contrôle. Je définis donc les trois systèmes suivants :
   - **un système de déplacement** représenté par un ensemble des règles de déplacement ;
   - **un système de communication** représenté par un ensemble des règles de communication ;
   - **un système de contrôle** qui permet d'agir sur la coévolution du système de communication et du système de déplacement. Il est représenté par un ensemble de règles contrôlant la rétroaction entre les 2 systèmes.

L'approche proposée se veut générique, dans le sens où elle peut s'appliquer à différents cas de systèmes de collecte d'information. Dans notre approche le concept de confiance est central : il est utilisé à la fois pour maintenir la cohérence des informations (en raisonnant sur les informations collectées et reçues) et pour améliorer la robustesse du système (en raisonnant sur les agents). La confiance est également centrale dans les mécanismes d'auto-organisation mis en place. Cette auto-organisation est basée sur un couplage structurel entre les systèmes de communication et de déplacement.

Chapitre 4

Vers un modèle de confiance basé sur le réseau des interactions entre agents : TrustSet

Sommaire

Dans ce chapitre, je vais présenter un modèle de confiance permettant d'assurer la cohérence et la robustesse dans un système de collecte d'information réparti dont les éléments sont collectés par un grand nombre d'agents dont les activités de perception peuvent être perturbées (soit volontairement soit involontairement). Pour ce faire, chaque agent maintient un réseau de confiance de type graphe et un système d'informations locales, construits à partir de la confrontation d'informations directes (collectées par l'agent) et d'informations indirectes (obtenues à partir des agents rencontrés). L'objectif sera d'améliorer le système de communication entre agents en vue de faire émerger l'information la plus fiable possible et la plus proche de la réalité objective. Les agents débutent sans connaissance sur le comportement (et en

particulier sur le fait qu'ils sont défectueux ou non) des autres agents. Ils améliorent cette connaissance en utilisant les interactions directes et indirectes et le chemin suivi par l'information depuis l'agent source qui l'a collecté directement. Grâce aux interactions, le réseau de confiances est progressivement amélioré et utilisé pour juger de la fiabilité des informations ou pour rejeter les communications indésirables.

4.1 Introduction

Dans un système de collecte d'information distribué, les agents ont pour but de collecter le plus de données possible en utilisant le moins de ressources possible. Collecter toutes les informations par lui-même n'est pas, a priori, la meilleure stratégie pour un agent. Une solution consiste à utiliser des informations provenant d'autres agents. Mais ce n'est pas facile de savoir quel agent est honnête et quel agent ne l'est pas. La communication est une source d'enrichissement et de perturbation dans un système de collecte d'information quand l'information peut être altérée par des agents non fiables. Pour limiter l'influence de ces agents, je propose de travailler sur les stratégies de communication dans une tentative d'éliminer les fausses informations et les agents qui sont responsables de la production de ces fausses informations dans le système. Je donne aux agents la capacité de raisonner sur les autres agents et de choisir avec lesquels interagir. Pour cela, l'un des outils les plus efficaces est le concept de confiance. J'introduis une structure de données étendant le TrustNet de [Schillo et al., 2000] que j'appelle 'TrustSet'. Cette structure permet aux agents de combiner les observations faites par tous les agents (lui-même et les autres). Des algorithmes originaux sont ensuite décrits pour construire cette structure et la faire évoluer. Dans cette approche, la confiance est utilisée pour identifier et isoler les agents non fiables : elle est utilisée pour limiter les communications avec eux et pour éliminer les informations qu'ils ont transmis. La confiance dans notre contexte peut être simplement définie comme la probabilité selon laquelle un agent pense que l'interaction avec un autre agent sera bénéfique pour lui. Dans la suite, les valeurs de confiance seront donc dans l'intervalle $[0, 1]$.

Avant de décrire le modèle de confiance proposé, il est important de noter que l'agent associé à une information indirecte est l'agent qui a collecté l'information, ce n'est pas forcément l'agent qui a transmis à l'agent l'information.

4.2 Représentation de l'information

Notre objectif est de développer un modèle de confiance basé sur les informations échangées et sur le réseau des interactions entre agents. Parmi les informations stockées dans la mémoire de l'agent et échangées lors de la communication avec un autre agent, on distingue deux types d'information : les informations sur l'environnement (nommées *données*) qui dépendent du problème et les informations sur les agents (appelées *méta-données*), qui sont indépendantes du problème.

J'introduis tout de suite un certains nombre de notations qui seront utilisées tout au long du mémoire. L'ensemble des agents du système est notée $V = \{i, j, k, m...\}$. De plus, COULEUR représentera l'ensemble des couleurs représentant les différents niveaux de danger dans l'exemple d'application Danger Mapping.

4.2.1 Représentation de l'information sur l'environnement

4.2.1.1 Représentation de l'environnement

Une information sur l'environnement est représentée généralement par une paire $info = \langle clé, valeur \rangle$. Un ensemble de données peut donc être vu comme un dictionnaire, chaque clé étant en effet unique car elle caractérise un objet (position ou entité) de l'environnement.

Dans l'exemple du « Danger Mapping », on note une information recueillie comme suit :

- $\langle (x, y), couleur \rangle$ pour une zone explorée $(x, y) \in \mathbb{Z}^2$, avec un niveau de danger codé par $couleur \in COULEUR$ (un ensemble des couleurs prédéfinis) ;

Les agents du système de collecte d'information vont explorer l'environnement pour collecter des informations et vont également en recevoir d'autres agents. On sépare en interne les informations en deux ensembles en fonction de l'origine de l'information.

4.2.1.2 Représentation des informations collectées : les informations directes

L'ensemble des informations directes, noté $D_i = \{info\}$, représente les informations collectées directement par l'agent i. Il a simplement la forme d'un dictionnaire de toutes les informations.

Dans l'exemple du « Danger Mapping », un exemple des informations directes collectées par l'agent i par l'intermédiaire de ses capteurs est $D_i = \{\langle (x_1, y_1), bleu\rangle, \langle (x_2, y_2), rouge\rangle\}$. Dans cet exemple, l'agent i a détecté une zone dangereuse de niveau bleu en (x_1, y_1) et une zone dangereuse de niveau rouge en (x_2, y_2).

4.2.1.3 Représentation des informations reçues : les informations indirectes

L'ensemble des informations indirectes, noté $IND_i = \{\langle clé, \{\langle valeur, \{a_x\}\rangle\}\rangle\}$, représente les informations que l'agent i reçoit d'autres agents. D'une manière générale, une information indirecte sera composée de 2 parties :

1. une clé identifiant l'information recueillie, notée *clé*, dépendant du problème (coordonnées dans l'exemple du Danger Mapping) ;
2. un ensemble de paires $\langle valeur; \{a_x\}\rangle$ avec $a_x \in V$, l'ensemble des agents du système. Les agents de $\{a_x\}$ sont les agents qui ont recueillis directement l'information *valeur* pour la *clé* considérée.

Dans l'exemple Danger Mapping, on note une information indirecte :
– $\langle (x, y), \{\langle couleur, \{a_x\}\rangle\}\rangle$ avec $(x, y) \in \mathbb{Z}^2$, $couleur \in COULEUR$, $a_x \in V$.
Par exemple, l'ensemble $IND_i = \{\langle (x_1, y_1), \{\langle bleu, \{k, l\}\rangle, \langle rouge, \{m, n\}\rangle\}\rangle, \langle (x_2, y_2), \{\langle rouge, \{j\}\rangle\}\rangle\}$ signifie que l'agent i a reçu une information indirecte sur une zone dangereuse de niveau bleu en (x_1, y_1) que deux agents k, l ont précédemment recueillies directement et une autre information indiquant que la zone (x_1, y_1) est dangereuse avec un niveau $rouge$ depuis deux agents m et n qui l'ont collecté directement. De plus, il a reçu une information sur une zone dangereuse de niveau rouge en (x_2, y_2) que l'agent j a recueillie directement.

4.2.2 Représentation de l'information sur les agents

Les informations sur les agents (*méta-données*) sont des confiances calculées à partir des informations sur l'environnement (*données*) collectées et/ou communiquées par les agents. Nos agents, comme ceux de Schillo [Schillo et al., 2000], sont capables d'échanger à la fois des données et des métadonnées. Les agents peuvent ainsi construire un réseau social pondéré par les valeurs de confiance, appelé 'TrustNet' chez Schillo *et al.* La valeur de confiance d'un agent envers un autre est alors une combinaison des expériences directes et des témoignages des autres agents.

J'introduis ci-dessous plusieurs définitions qui seront utilisées tout au long du mémoire. J'ai délibérément omis les notations de temps pour simplifier les expressions. Chaque agent met son TrustSet à jour à chaque pas de la simulation. Ainsi, les valeurs du TrustSet sont calculées à chaque instant t à partir des valeurs de confiance disponibles à l'instant considéré.

Définition 7. (Liens entre agents)
Un lien entre deux agents i et j est noté $\overline{ij}$.
L'ensemble des liens possibles entre les agents de V *est noté* $E(V) = \left\{\overline{ij} \mid i, j \in V, i \neq j\right\}$.

Définition 8. (Liens orientés entre agents)
Un lien orienté de l'agent i vers l'agent j est noté $\overrightarrow{ij}$.
L'ensemble des liens orientés possibles entre les agents de V *est noté* $E'(V) = \left\{\overrightarrow{ij} \mid i, j \in V, i \neq j\right\}$.

Comme précisé au chapitre précédent, je considère que le monde est clos et donc que l'ensemble V est fixé. Les ensembles $E(V)$ et $E'(V)$ ne sont que les ensembles des liens possibles qui peuvent se créer entre les agents. Les liens existants sont stockés dans le TrustGraph présenté ci-dessous.

Définition 9. (TrustGraph)
Un TrustGraph est un digraphe (graphe orienté) pondéré avec une origine. Si $i \in V$ *est l'origine du digraphe,* $TG_i = (V_i, E'_i, w_i)$ *est le TrustGraph de l'agent i, avec* $V_i \subseteq V$ *l'ensemble des nœuds,* $E'_i \subseteq E'(V_i)$ *l'ensemble des arcs orientés sur les agents* V_i *du graphe et* $w_i : E'_i \to [0, 1]$ *une fonction de poids sur les arcs représentant les confiances.*

J'impose qu'un **TrustGraph** soit un graphe orienté (*i.e.* chemins joignant un nœud à lui-même). Il associe un nœud racine qui est le propriétaire du TrustGraph (nœud i dans la figure 4.1), un ensemble d'agents qui est relié à i (soit des agents que i a précédemment rencontré, ou dont il a entendu parler au cours de ses communications avec d'autres agent). Deux nœuds reliés par un arc représentent deux agents qui se sont déjà rencontrés. Par exemple, si j a rencontré k d'abord et rencontré i ensuite, deux arcs orientés sont ajoutés au TrustGraph de l'agent i : $\overrightarrow{ij}$ et $\overrightarrow{jk}$. Par ailleurs les arêtes portent des informations sur l'estimation de la confiance des agents.

Définition 10. (Confiance directe)
La valeur de confiance $w_i\left(\overrightarrow{ix}\right)$ attribuée à un arc qui relie l'origine i de TG_i à un nœud x est appelée confiance directe de i en l'agent x et est notée DT_{ix}.

Chaque valeur $w_i\left(\overrightarrow{ix}\right)$ représente la *confiance directe* du propriétaire de TG_i envers un autre agent (par exemple j dans le Figure 4.1). Elle est calculée à partir de la comparaison des informations recueillies par l'agent lui-même (D_i) avec les informations collectées par l'agent qu'il a rencontré (D_j).

Définition 11. (Confiance indirecte)
La valeur de confiance $w_i(\overrightarrow{xy})$ attribuée à un arc reliant un nœud x différent de l'origine à un nœud y ($y \neq x$) dans TG_i est appelée la confiance indirecte de l'agent x en l'agent y et est notée IT_{xy}.

Tout arc, n'ayant pas pour origine le propriétaire du graphe, par exemple l'arc entre le nœud j et le nœud k dans la figure Figure 4.1 porte une *confiance indirecte*, notée IT_{jk} et obtenue par le biais de la communication. IT_{jk} représente la confiance de l'agent j envers l'agent k communiquée par j au propriétaire i du TrustGraph.

Définition 12. (Confiance intrinsèque)
La confiance intrinsèque T_{ix} représente la confiance du propriétaire i envers un autre agent x en tenant compte de sa propre confiance directe DT_{ix} et de toutes les confiances sur des chemins différents reliant i à x dans TG_i.

Du point de vue du propriétaire du graphe, tous les nœuds accessibles seulement par un chemin de longueur strictement supérieure à 1 sont des agents qu'il n'a jamais rencontrés. A chaque nœud du TrustGraph TG_i, l'agent i associe une valeur, la confiance intrinsèque, notée T_{ix}, qui représente la confiance du propriétaire i envers un autre agent x en tenant compte de toutes les confiances directes et indirectes. On peut remarquer que $T_{ix} = DT_{ix}$ quand un seul arc relie i à x et T_{ii} vaut 1 (la confiance de l'agent d'origine en lui-même est initialisée à 1 car on considère qu'il n'a aucune raison d'avoir des doutes sur sa propre fiabilité). Les confiances intrinsèques seront mémorisées dans un tableau appelée TrustTable.

Définition 13. (TrustTable)
L'ensemble des confiances intrinsèques de l'agent i noté $\{T_{ix} \mid x \in V_i\}$ est stocké dans une table appelée TrustTable et notée TT_i.

Les confiances intrinsèques ne seront pas transmises aux autres agents car elles découlent d'un calcul personnel de l'agent origine du graphe. Les confiances intrinsèques ne seront recalculées que si l'un des éléments de base du calcul a été modifié, ou si un nouvel élément devant entrer dans le calcul a été introduit dans le TrustGraph. Ce seront les confiances intrinsèques qui décideront de la stratégie de communication de l'agent origine avec tel ou tel agent.

Définition 14. (TrustSet)
On appelle TrustSet TS_i de l'agent i la paire formée par son TrustGraph TG_i et sa TrustTable TT_i : $TS_i = (TG_i, TT_i)$.

Chaque agent stocke son propre TrustSet. Quand un agent i veut coopérer avec un autre agent j, i peut estimer sa confiance en j grâce à son TrustSet. Les détails de ces calculs sont présentés en section 4.3.2.

Le TrustGraph est construit grâce aux informations collectées (par le propriétaire lui-même) et transmises (par d'autres agents) et contient des informations qui ne sont pas personnelles à un agent, il représente donc *la partie publique* du TrustSet et sera communiqué aux autres agents. Les arcs reliant le propriétaire à d'autres nœuds sont affectés par la valeur de confiance du propriétaire envers les agents qu'il a déjà rencontré. Elle sera communiquée à d'autres agents à chaque rencontre. A l'inverse, la TrustTable est calculée grâce à des algorithmes pouvant être spécifiques à un agent particulier, elle représente donc *la partie privée* du TrustSet et ne sera pas communiquée aux autres agents.

Exemple 4. Un exemple de TrustSet construit par l'agent i est proposé dans la Figure 4.1 : il comprend un TrustGraph représenté de manière formelle par $TG_i = \left(\{i,j,k\}, \left\{\overrightarrow{ij}, \overrightarrow{jk}\right\}, w_i\right)$ avec $w_i\left(\overrightarrow{ij}\right) = DT_{ij}$ et $w_i\left(\overrightarrow{jk}\right) = IT_{jk}$ et une TrustTable représentée par $TT_i = \{T_{ii}, T_{ij}, T_{ik}\}$. Les nœuds représentent les trois agents i, j, k. DT_{ij} est la confiance directe de i en j, IT_{jk} la confiance de j en k communiquée à i par j. T_{ii} est la confiance intrinsèque de i en lui-même, T_{ij} représente la confiance intrinsèque de i en j (calculée grâce à la confiance directe DT_{ij} et à toutes les confiances indirectes IT_{xj} associées aux arcs menant à j).

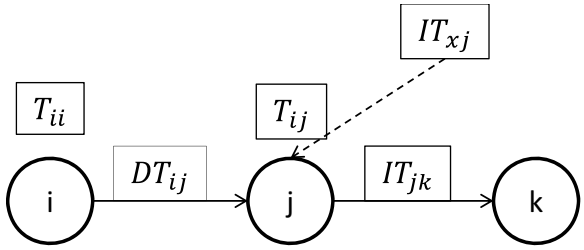


Figure 4.1: Exemple de TrustSet de l'agent i

4.3 Évolution de la confiance via les échanges d'information entre agents

Après avoir présenté la partie statique du modèle de confiance dans la section précédente, c'est-à-dire comment l'agent représente ses données sur les autres agents (ses *méta-données*), je détaille ici comment cette confiance et tout le TrustSet sont calculés et mis à jour en fonction des rencontres et des échanges.

Lors d'une rencontre, les agents échangent un certain nombre d'informations (des informations sur l'environnement et des informations sur les agents). Dans un SMA perturbé, les agents peuvent modifier les informations qu'ils ont recueillies eux-mêmes sur l'environnement et les informations reçues d'autres agents. Ils peuvent aussi déformer les confiances qu'ils ont envers les autres agents. Dans ce travail, je me limite à des agents défectueux, donc les TrustGraph transmis ne contiennent pas d'erreur.

Lorsqu'un agent reçoit des informations d'un autre agent, il compare chacune des informations reçues avec ses propres informations pour ajuster la confiance qu'il a en cet agent. J'étends ici la distinction de Festinger [Festinger, 1957] sur les cognitions au cas des informations à comparer :

- **Informations incomparables** : ce sont des informations qui ne peuvent être comparées. Ce sont notamment des informations qui portent sur des objets différents. Dans l'exemple du Danger Mapping, ce sont les informations qui portent sur des coordonnées différentes.
- **Informations consonantes** : ce sont des informations comparables (c.à.d sur un même objet) et qui donnent la même information (c.à.d la même valeur). Dans l'exemple du Danger Mapping, ce sont des informations qui portent sur les mêmes coordonnées avec les mêmes couleurs.
- **Informations dissonantes** : ce sont des informations comparables et qui donnent des informations différentes, (c.à.d des valeurs différentes). Dans l'exemple du Danger Mapping, ce sont des informations qui portent sur les mêmes coordonnées avec des couleurs différentes.

Par la suite, je considère la rencontre et l'échange d'informations entre deux agents i et j. Je ne présente le processus que du point de vue de l'agent i, il est identique pour j. Je considère aussi qu'il y a simultanéité des transferts d'information entre les agents, les calculs internes s'effectuant seulement ensuite. Je détaille tout d'abord l'évolution des données directes et indirectes de l'agent suite à cette rencontre avant de m'intéresser à celle du TrustSet.

4.3.1 Mise à jour des informations sur l'environnement

Initialement (au début de la collecte d'information), les base $(D_i)_0$ et $(IND_i)_0$ sont vides. Supposons que i rencontre j à l'instant t. $(D_i)_t$ et $(IND_i)_t$ représentent les ensembles d'informations directes et indirectes de l'agent i à ce moment-là. L'agent i fait la fusion des informations transmises par j avec ses propres informations et produit les ensembles mis à jour $(D_i)_{t+1}$ et $(IND_i)_{t+1}$.

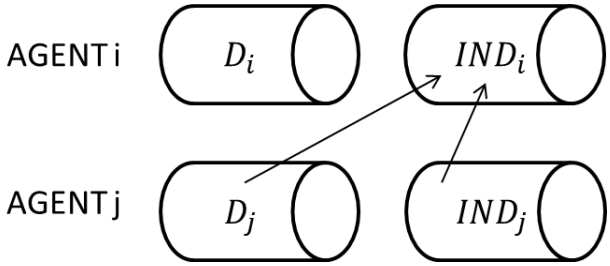


FIGURE 4.2 – Influence des informations de j sur les informations de i

Je considère que les informations directes et indirectes de j ne modifient pas D_i car D_i représente les informations recueillies par i directement sur le terrain. Pour calculer $(IND_i)_{t+1}$, l'agent i fusionne les informations transmises par j avec ses propres informations indirectes par une fonction de fusion Ψ^{Data}. La Figure 4.2 illustre les influences des informations de j sur celles de i.

On obtient donc :

- $(D_i)_{t+1} = (D_i)_t$
- $(IND_i)_{t+1} = \Psi^{Data}\left((IND_i)_t, (D_j)^*_t, (IND_j)_t\right)$

Je considère que i fusionne les informations reçues de j avec ses informations indirectes existantes. Il doit indiquer que les informations directes de j ont été collectés par j. Il ajoute ainsi aux informations directes de j un ensemble d'agents qui ont recueilli directement ces informations. Ce groupe est initialisé avec l'agent j. La fonction de fusion peut donc être réduite à l'union de l'ensemble des informations indirectes de i avec les deux ensembles d'informations de j mis à jour :

$$\Psi^{Data}\left((IND_i)_t, (D_j)^*_t, (IND_j)_t\right) = (IND_i)_t \cup (D_j)^*_t \cup (IND_j)_t$$

avec :

- $\left(D^*_j\right)_t = \{\langle clé, \langle valeur, \{j\}\rangle\rangle\}$ pour tout $\langle clé, valeur\rangle \in D_j$

La mise à jour des informations sur l'environnement est résumée dans l'Algorithme 1.

Exemple 5. Pour illustrer le processus de fusion, considérons l'exemple suivant :

- $(D_i)_t = \{\langle (x,y), blue\rangle\}$;
- $(IND_i)_t = \{\langle (x,y), red, \{k,l\}\rangle\}$;
- $(D_j)_t = \{\langle (x,y), yellow\rangle\}$;
- $(IND_j)_t = \{\langle (x,y), \{\langle blue, \{m\}\rangle\}\rangle\}$.

1. D_i n'est pas modifié : $(D_i)_{t+1} = \{\langle (x,y), blue\rangle\}$.
2. L'ensemble $(D_j)_t$ reçu est mis à jour par adjonction de l'agent j dans l'ensemble les agents qui ont recueillis directement l'information pour permettre la fusion de cet ensemble avec $(IND_i)_t$. La fusion de $(D_j)_t$ et $(IND_j)_t$ avec $(IND_i)_t$ a pour résultat :
   $(IND_i)_{t+1} = \{\langle (x,y), \{\langle red, \{k,l\}\rangle, \langle yellow, \{j\}\rangle, \langle blue, \{m\}\rangle\}\rangle\}$.

Après avoir mis à jour ses informations sur l'environnement, l'agent i doit réévaluer sa confiance dans les agents, c.à.d. mettre à jour ses informations sur les agents.

Algorithme 1: Algorithme pour la mise à jour des informations sur l'environnement

Entrées : $(D_i)_t, (IND_i)_t$: ensemble des informations **directes** et **indirectes** de l'agent i à l'instant t
$(D_j)_t, (IND_j)_t$: ensemble des informations **directes** et **indirectes** de l'agent j à l'instant t
Sorties : $(D_i)_{t+1} = (D_i)_t$: ensemble des informations **directes** de l'agent i à l'instant $t+1$
$(IND_i)_{t+1} = \Psi^{Data}\left((IND_i)_t, (D_j)^*_t, (IND_j)_t\right)$: les ensembles d'informations **indirectes** de l'agent i à l'instant $t+1$

$(clésIND_i)_t \leftarrow$ ensemble les clés différentes de $(IND_i)_t$
pour chaque *information directe* $d_j = \langle clé_j, valeur_j\rangle$ *telles que* $d_j \in (D_j)_t$ **faire**
  si $clé_j \in (clésIND_i)_t$ **alors**
    $ind_i = \langle clé_i, \{\langle valeur_i, \{a_x\}\rangle\}\rangle$ de IND_i tel que $clé_i = clé_j$
    $ATTs \leftarrow \{valeur \mid \exists valeur \in \{\langle valeur_i, \{a_x\}\rangle\}\}$
    si $valeur_j \notin ATTs$ **alors**
      $ind_i = \langle clé_i, \{\langle valeur_i, \{a_x\}\rangle \cup \langle valeur_j, \{j\}\rangle\}\rangle$
    sinon
      $ind_i = \langle clé, \{\langle valeur_i, \{a_x\} \cup \{j\}\rangle\}\rangle$
    fin
  sinon
    $d^*_j = \langle clé_j, \langle valeur_j, \{j\}\rangle\rangle$
    $(IND_i)_{t+1} = (IND_i)_t \cup d^*_j$
  fin
fin
pour chaque *information indirecte* $ind_j = \langle clé_j, \{\langle valeur_j, \{b_x\}\rangle\}\rangle$ *de* $(IND_j)_t$ **faire**
  si $clé_j \in (clésIND_i)_t$ **alors**
    $ind_i = \langle clé_i, \{\langle valeur_i, \{a_x\}\rangle\}\rangle$ de IND_i tel que $clé_i = clé_j$
    $ATTs \leftarrow \{valeur \mid \exists valeur \in \{\langle valeur_i, \{a_x\}\rangle\}\}$
    si $valeur_j \notin ATTs$ **alors**
      $ind_i = \langle clé_i, \{\langle valeur_i, \{a_x\}\rangle \cup \langle valeur_j, \{j\}\rangle\}\rangle$
    sinon
      $ind_i = \langle clé_i, \{\langle valeur_i, \{a_x\}\rangle \cup \langle valeur_j, \{b_x\}\rangle\}\rangle$
    fin
  sinon
    $(IND_i)_{t+1} = (IND_i)_t \cup ind_j$
  fin
fin

4.3.2 Mise à jour des informations sur les agents

4.3.2.1 Initialisation du TrustSet

Initialement, chaque agent construit son propre TrustSet vide. Le TrustGraph est initialisé seulement avec le nœud racine (c.à.d l'agent lui-même) et ne contient aucun arc. La TrustTable contient une seule valeur, la confiance de l'agent racine en lui-même, initialisée à 1, car il n'a aucune raison de douter de lui-même.

- $TS_i = (TG_i, TT_i)$:
  - $TG_i = (\{i\}, \{\emptyset\}, w_i)$
  - $TT_i = \{T_{ii}\}$

Les TrustGraph et TrustTable sont ensuite mis à jour grâce aux informations échangées avec d'autres agents. Je suppose aussi que chaque agent a une valeur de confiance *a priori* envers les agents qu'il n'a pas encore rencontré. Cette valeur est utilisée comme valeur initiale la première fois qu'il communique avec un autre agent et est notée dans la suite T_{init}.

4.3.2.2 Communication du TrustSet

A chaque interaction avec un agent j, l'agent i communiquera éventuellement ses données à j, mais aussi certaines de ses métadonnées. En particulier, il partagera son TrustGraph qui contient toutes les métadonnées publiques, mais pas sa TrustTable car elle est construite par des algorithmes de calcul personnels à l'agent et contient donc des informations privées.

Lorsqu'un agent reçoit un TrustGraph, il l'intègre à son propre TrustGraph. Il utilise ensuite le TrustGraph obtenu pour mettre à jour sa TrustTable.

4.3.2.3 Fusion des TrustGraphs

On suppose un échange d'informations entre les agents i et j. Je prends le point de vue de i, mais le processus est le même pour j. La mise à jour du TrustGraph de l'agent i s'effectue en 3 étapes et est résumée dans Algorithme 2 :

- i calcule d'abord sa confiance DT_{ij} en j ou met à jour la valeur existante en comparant ses propres informations avec celles reçues ;
- i connecte le TrustGraph de l'agent j à son propre TrustGraph ;
- i corrige toutes les incohérences sur les chemins partagés.

Étape 1 - Calcul de la confiance directe en comparant les informations. Pour calculer la confiance directe de i en j, l'agent i compare ses propres données directes et celles transmises par j. La comparaison se fait en trois étapes :

- i sélectionne uniquement les données comparables, c'est-à-dire les données portant sur les mêmes objets. Leur nombre est noté β.
- i calcule le niveau d'incohérence entre ses propres données directes et celles reçues de j. Pour ce faire, il utilise une distance notée δ, où $\delta(x, y)$ représente la distance entre les données x de l'ensemble des informations directes de l'agent i D_i et y de D_j. Intuitivement, $\delta(x, y) = 0$ signifie que les informations sont cohérentes (*c.à.d* portent les mêmes informations sur le même objet), alors que $\delta(x, y) > 0$ représente le degré d'incompatibilité entre les deux informations. Le niveau d'incohérence total entre les informations de i et de j (noté Inc_level) est calculé par la formule suivante :

$$Inc\_level = \frac{\sum_{k \in \{1,\beta\}} \delta(x_k, y_k)}{\beta * \delta_{Max} info} \tag{4.1}$$

  avec (x_k, y_k) une paire de données dépendantes et $\delta_{Max} info$ la plus grande distance possible entre informations incohérentes (cette valeur va dépendre du problème modélisé et en particulier des valeurs possibles pour les informations).
- i compare cette valeur à un seuil μ qui représente le niveau d'incohérence maximum acceptable par un agent avant de diminuer la valeur de sa confiance en un autre agent.

- Si $Inc\_level < \mu$: i augmente sa confiance en j par un facteur τ^+(défini ci-dessous) ;
- Si $Inc\_level > \mu$: i diminue sa confiance en j par un facteur τ^- ;
- Si $Inc\_level = \mu$: i laisse sa confiance en j inchangée.

La valeur des facteurs τ^+et τ^- est calculée en fonction des propriétés que le concepteur veut donner au système. J'introduis deux seuils de confiance : un seuil haut (Upp) et un seuil bas (Low). Entre Upp et 1, les agents sont considérés comme "*fiables*". Entre Upp et Low, les agents sont en instance d'évaluation ("*indéterminés*"). Entre 0 et Low, les agents sont considérés comme "*non fiables*". Je note NI une estimation du nombre d'interactions nécessaires pour avoir une image complète de l'environnement et ρ_{stab} ($\rho_{stab} \in [0,1]$) le taux de stabilisation représentant le pourcentage de la simulation après lequel le concepteur veut qu'une classification entre agents fiables et non fiables se dégage. $\rho_{stab} * NI$ représente donc le nombre d'interactions nécessaires pour passer de T_{init} à Upp ou Low (ce qui signifie que $T_{init} + NI * \rho_{stab} * \tau^+ = Upp$). Je peux donc exprimer les deux facteurs τ^+ et τ^- sous la forme :

$$\tau^+ = \frac{(Upp - T_{init})}{\rho_{stab} * NI} \text{ et } \tau^- = \frac{(T_{init} - Low)}{\rho_{stab} * NI} \tag{4.2}$$

Lors de la première rencontre entre deux agents i et j, i choisi comme confiance directe DT_{ij} la valeur T_{init}, qui sera augmentée ou diminuée en fonction des informations transmises.

Étape 2 - Fusion des TrustGraphs. Dans une deuxième étape, i construit un nouveau TrustGraph intermédiaire $(TG_i)_{t+1} = (V_i^*, E_i'^*, w_i^*)$ à partir de $(TG_i)_t = (V_i, E_i', w_i)$ et $(TG_j)_t = \left(V_j, E_j', w_j\right)$.

Le nouvel ensemble de nœuds est formé de l'ensemble des nœuds des deux TrustGraphs :

$$V_i^* = V_i \cup V_j \tag{4.3}$$

On procède de même pour l'ensemble des arcs : le nouvel ensemble d'arcs est formé de tous les arcs de E_i' et de E_j', auquel il faut rajouter un arc $\overrightarrow{ij}$ entre les agents i et j (pour "relier" les deux TrustGraphs) et enlever tout arc revenant vers i :

$$E_i'^* = \left\{\overrightarrow{ij}\right\} \cup E_i' \cup E_j' \setminus \left\{\overrightarrow{yi}, y \in V_i \cup V_j\right\} \tag{4.4}$$

Bien que ces arcs soient supprimés, la valeur de confiance qui leur est associée n'est pas perdue : elle est prise en compte dans la mise à jour de la TrustTable et influence la confiance que l'agent a en lui-même.

On associe ensuite la valeur DT_{ij} à l'arc entre i et j, $w_i^*\left(\overrightarrow{ij}\right) = DT_{ij}$ et pour chaque autre arc la valeur qu'il avait dans son TrustGraph d'origine :

$$w_i^*\left(\overrightarrow{xy}\right) = \begin{cases} w_i\left(\overrightarrow{xy}\right) & if\ \overrightarrow{xy} \in E_i', \overrightarrow{xy} \notin E_j' \\ w_j\left(\overrightarrow{xy}\right) & if\ \overrightarrow{xy} \in E_j', \overrightarrow{xy} \notin E_i' \end{cases} \tag{4.5}$$

L'étape suivante explicite la façon dont je gère l'incohérence lorsqu'un arc appartient aux deux TrustGraphs et que la valeur de confiance et différente dans les deux ($\overrightarrow{xy} \in E_i' \cap E_j'$ avec $w_i\left(\overrightarrow{xy}\right) \neq w_j\left(\overrightarrow{xy}\right)$).

Étape 3 - Gestion des incohérences sur les chemins partagés. On dit qu'il y a incohérence sur un arc partagé lorsque cet arc apparait dans les 2 TrustGraphs à fusionner avec des valeurs de confiance différentes. Ce cas apparait typiquement lorsque i et j rencontrent k à deux instants différents. Pour éviter cette incohérence, on calcule une nouvelle valeur à partir des deux valeurs de confiance incohérentes en utilisant l'algorithme suivant. Considérons $\overrightarrow{xy}$ un arc commun à TG_i et TG_j. $\overrightarrow{xy}$ porte la valeur de confiance IT_{xy_i} dans TG_i et IT_{xy_j} dans TG_j.

- Je construis les deux ensembles $Path_i$ et $Path_j$ de tous les chemins de TG_i et TG_j contenant l'arc $\overrightarrow{xy}$. Pour éviter les problèmes de cycles, je ne choisis que les plus courts chemins. Tous le chemins de $Path_i$ et $Path_j$ sont représentés par la liste des agents composant ce chemin, l'origine i exclus.
- On calcule ensuite la valeur de confiance $IT_{xy_{ij}}$ de l'arc xy dans le TrustGraph résultant de la fusion des TrustGraphs de i et de j avec la formule suivante :

$$IT_{xy_{ij}} = \frac{\sum_{u \in Path_i} T_{iu} * IT_{xy_i} + \sum_{v \in Path_j} T_{iv} * IT_{xy_j}}{\sum_{u \in Path_i} T_{iu} + \sum_{v \in Path_j} T_{iv}} \tag{4.6}$$

Cette nouvelle valeur de confiance est sur la moyenne de $Path_i$ et $Path_j$ des confiances IT_{xy} dans chacun des chemins au long du chemin menant à l'arc commun $\overrightarrow{xy_{ij}}$ avec la valeur de confiance sur $\overrightarrow{xy}$.

4.3.2.4 Mise à jour de la TrustTable

Les confiances intrinsèques doivent être recalculées après la fusion des TrustGraphs si l'un des éléments de base du calcul a été modifié, ou si un nouvel élément devant entrer dans le calcul a été introduit dans le TrustGraph. Ainsi, après la mise à jour des confiances directes et indirectes dans le TrustGraph, les agents mettent à jour leurs propres confiances intrinsèques dans leur TrustTable privé. Les deux étapes suivantes sont nécessaires pour calculer les confiances intrinsèques dans la TrustTable de l'agent i :

- Étape 1 : Calculer les confiances intrinsèques sur tous les nouveaux nœuds transmis par j, *c.à.d.* les nœuds qui n'existaient pas dans sa TrustTable avant la rencontre avec j.
- Étape 2 : Recalculer toutes les confiances intrinsèques de la TrustTable ayant été influencées par la fusion des TrustGraphs, sauf pour les nœuds dont la confiance a été calculée à l'étape 1.

La valeur de confiance intrinsèque de l'agent i envers un agent x est calculée par la formule suivante :

$$T_{ix} = \frac{T_{ii} * DT_{ix} + \sum_{y \in \left\{z \in V_i^* | \overrightarrow{zx} \in E_i'^*\right\}} (T_{iy} * IT_{yx})}{T_{ii} + \sum_{y \in \left\{z \in V_i^* | \overrightarrow{zx} \in E_i'^*\right\}} T_{iy}} \tag{4.7}$$

La confiance intrinsèque de i en x est calculée comme la moyenne de la confiance directe DT_{ix} et des confiances indirectes IT_{yx} que tous les agents de TG_i ont en x, pondérée par les confiances intrinsèques de i en les agents sources de ces DT_{ix} et IT_{yx} (c.à.d confiance en lui-même et en tous les agents qui ont rencontré x). Notons que $T_{ix} = DT_{ix}$ quand il n'y a pas d'autre chemin de i à x qu'un arc direct $\overrightarrow{ix}$ et que T_{ii} est initialisée à 1. La mise à jour de la confiance intrinsèque de chaque agent a deux objectifs. Premièrement, si un agent est compromis (il devient un agent défectueux) et envoie fréquemment des informations fausses ou inexactes sur l'environnement, sa valeur de confiance intrinsèque va être diminuée. Une fois que la valeur de la confiance intrinsèque d'un agent est inférieure à un seuil, l'agent peut être considéré comme agent défectueux. Deuxièmement, la valeur de la confiance intrinsèque est utilisée pour décider de la stratégie de communication de l'agent d'origine avec un agent particulier.

4.3.2.5 Exemple

Pour illustrer le processus de fusion, considérons l'exemple de la figure 4.3. On considère les 2 TrustSets de l'agent i et de l'agent j à l'instant t :

- $(TS_i)_t = ((TG_i)_t, (TT_i)_t)$:
  - $(TG_i)_t = (V_i, E_i', w_i) = \left(\{i, o, n, m\}, \left\{\overrightarrow{io}, \overrightarrow{on}, \overrightarrow{nm}\right\}, w_i\right)$
    avec $w_i\left(\overrightarrow{io}\right) = 0.8, w_i\left(\overrightarrow{on}\right) = 0.9$ et $w_i\left(\overrightarrow{nm}\right) = 0.5$
  - $TT_i = \{T_{ii}, T_{io}, T_{in}, T_{im}\} = \{1, 0.8, 0.6, 0.7\}$
- $(TS_j)_t = \left((TG_j)_t, (TT_j)_t\right)$:
  - $(TG_j)_t = \left(V_j, E_j', w_j\right) = \left(\{j, u, o, n, m\}, \left\{\overrightarrow{ju}, \overrightarrow{un}, \overrightarrow{uo}, \overrightarrow{nm}\right\}, w_j\right)$
    avec $w_j\left(\overrightarrow{ju}\right) = 0.6, w_j\left(\overrightarrow{un}\right) = 0.7, w_j\left(\overrightarrow{uo}\right) = 0.3$ et $w_j\left(\overrightarrow{nm}\right) = 0.2$
  - $(TT_j)_t = \{T_{jj}, T_{ju}, T_{jo}, T_{jn}, T_{jm}\} = \{1, 0.6, 0.4, 0.8, 0.3\}$

La mise à jour du TrustSet de l'agent i sera calculée en 4 étapes :

1. **Étape 1 :** i calcule la confiance qu'il a en j, DT_{ij}, en comparant ses propres informations directes avec celles que j lui envoie.
   - Par exemple, on suppose que la confiance intrinsèque de i en j est $DT_{ij}^* = 0.5$. Considérons que l'agent i a reçu 50 informations directes de l'agent j sur lesquelles il n'y a que 20 données comparables (donc $\beta = 20$). Parmi ces 20 données directes transmises par j, il y a 15 informations

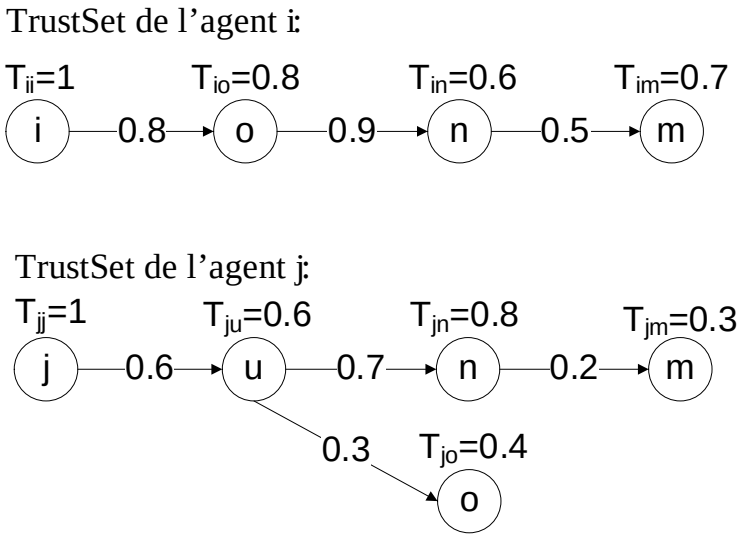


FIGURE 4.3 – TrustSet de l'agent i et celui de l'agent j

incohérentes, c.à.d qui ont $\delta(x,y) > 0$ avec 10 informations qui ont $\delta(x,y) = 2$, 5 informations qui ont $\delta(x,y) = 3$ et 5 informations cohérentes qui ont $\delta(x,y) = 0$.
- Si on prend $\delta_{Max}info = 4$, le niveau d'incohérence total entre les informations de i et de j est calculé par $Inc\_level = \frac{10*2+5*3+5*0}{20*4} = 0.4375$;
- on prend un seuil $\mu = 0.1$ pour le niveau d'incohérence maximum acceptable par un agent. $Inc\_level > \mu$ donc i diminue sa confiance en j par un facteur τ^- : $DT_{ij} = DT^*_{ij} - \tau^- = 0.5 - 0.02 = 0.48$ ($\tau^- = \frac{(T_{init}-Low)}{\rho_{stab}*NI} = \frac{0.5-0.4}{0.5*10} = 0.02$ en prenant par exemple $T_{init} = 0.5$, $Low = 0.4$ et $\rho_{stab} * NI = 10$).

2. **Étape 2 :** i connecte le TrustGraph envoyé par j à son propre TrustGraph pour créer un nouveau TrustGraph intermédiaire $(TG_i)_{t+1} = (V_i^*, E_i'^*, w_i^*)$ à partir de $(TG_i)_t = (V_i, E_i', w_i)$ et $(TG_j)_t = (V_j, E_j', w_j)$ avec :
   (a) $V_i^* = V_i \cup V_j = \{i, o, j, u, n, m\}$
   (b) $E_i'^* = \left\{\overrightarrow{ij}\right\} \cup E_i' \cup E_j' \setminus \left\{\overrightarrow{yi}, y \in V_i \cup V_j\right\} = \left\{\overrightarrow{ij}, \overrightarrow{ju}, \overrightarrow{un}, \overrightarrow{ub}, \overrightarrow{io}, \overrightarrow{on}, \overrightarrow{nm}\right\}$
   (c) w_i^* pour tous les arcs $E_i'^*$
   - $w_i^*\left(\overrightarrow{ij}\right) = DT_{ij} = 0.48$
   - $w_i^*\left(\overrightarrow{ju}\right) = 0.6$
   - $w_i^*\left(\overrightarrow{un}\right) = 0.7$
   - $w_i^*\left(\overrightarrow{ub}\right) = 0.3$
   - $w_i^*\left(\overrightarrow{io}\right) = 0.8$
   - $w_i^*\left(\overrightarrow{on}\right) = 0.9$
   - $w_i^*\left(\overrightarrow{nm}\right) = IT_{nm}$

3. **Étape 3 :** i résout les problèmes de chemins partagés sur $\overrightarrow{nm} \in E_i' \cap E_j'$ avec $w_i\left(\overrightarrow{nm}\right) \neq w_j\left(\overrightarrow{nm}\right)$
   - $Path_i = \{o, n\}$
   - $Path_j = \{j, u, n\}$
   - $w_i\left(\overrightarrow{nm}\right) = IT_{nm} = \frac{\sum_{u\in Path_i} T_{iu}*IT_{nm_i} + \sum_{v\in Path_j} T_{iv}*IT_{nm_j}}{\sum_{u\in Path_i} T_{iu} + \sum_{v\in Path_j} T_{iv}} = \frac{(T_{io}+T_{in})*IT_{nm_i} + (T_{ij}+T_{iu}+T_{in})*IT_{nm_j}}{(T_{io}+T_{on})+(T_{ij}+T_{iu}+T_{in})}$
     $= \frac{(0.8+0.6)*0.5+(0.48+0.5+0.6)*0.2}{(0.8+0.6)+(0.48+0.5+0.6)} = 0.34$

4. **Étape 4 :** i met à jour sa TrustTable : $TT_i = \{T_{ii}, T_{iu}, T_{io}, T_{in}, T_{im}, T_{ij}\}$

(a) La confiance de i en lui-même T_{ii} sera mentionnée dans la partie 4.4.1.

(b) i calcule les confiances intrinsèques sur tous les nouveaux nœuds transmis par j :
$T_{iu} = \frac{T_{ii}*DT_{iu}+\sum_{y\in\{j,o,n,m\}}(T_{iy}*IT_{yu})}{T_{ii}+\sum_{y\in\{j,o,n,m\}}T_{iy}} = \frac{1*0.5+(0.48*0.6)}{1+0.48} = 0.53$

(c) Enfin, recalculer toutes les confiances intrinsèques de la TrustTable ayant été influencées par la fusion des TrustGraphs, sauf des nœuds dont la confiance a été calculée à l'étape ci-dessus. On recalcule donc $\{T_{io}, T_{in}, T_{im}, T_{ij}\}$:
- $T_{io} = \frac{T_{ii}*DT_{io}+\sum_{y\in\{j,u,n,m\}}(T_{iy}*IT_{yo})}{T_{ii}+\sum_{y\in\{j,u,n,m\}}T_{iy}} = \frac{1*0.8+(0.5*0.3)}{1+0.5} = 0.63$
- $T_{in} = \frac{T_{ii}*DT_{in}+\sum_{y\in\{j,o,u,m\}}(T_{iy}*IT_{yn})}{T_{ii}+\sum_{y\in\{j,o,u,m\}}T_{iy}} = \frac{1*0.6+(0.63*0.9+0.53*0.7)}{1+0.63+0.53} = 0.71$
- $T_{im} = \frac{T_{ii}*DT_{im}+\sum_{y\in\{j,o,n,u\}}(T_{iy}*IT_{ym})}{T_{ii}+\sum_{y\in\{j,o,n,u\}}T_{iy}} = \frac{1*0.7+(0.63*0.34)}{1+0.6} = 0.57$
- $T_{ij} = \frac{1*0.48}{1} = 0.48$

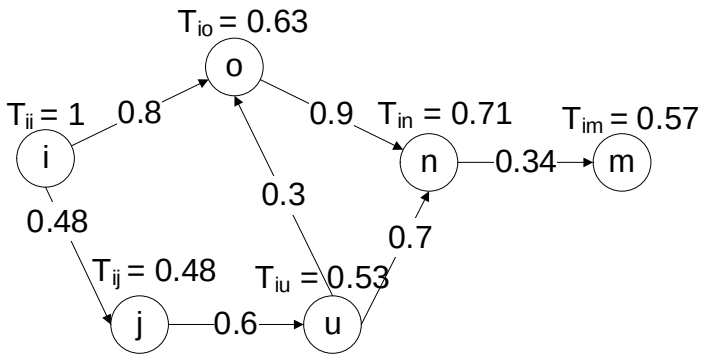


FIGURE 4.4 – Le TrustSet de l'agent i après avoir fusionné avec celui de j

En sortie, Le TrustSet de l'agent i mis à jour est :
- $TS_i = (TG_i, TT_i)$:
  - $TG_i = \left(\{i,o,j,u,n,m\}, \left\{\overrightarrow{ij}, \overrightarrow{ju}, \overrightarrow{un}, \overrightarrow{uo}, \overrightarrow{io}, \overrightarrow{on}, \overrightarrow{nm}\right\}, w_i\right)$
    avec $w_i\left(\overrightarrow{ij}\right) = DT_{ij} = 0.48, w_i\left(\overrightarrow{ju}\right) = 0.6, w_i\left(\overrightarrow{un}\right) = 0.7, w_i\left(\overrightarrow{uo}\right) = 0.3, w_i\left(\overrightarrow{io}\right) = 0.8, w_i\left(\overrightarrow{on}\right) = 0.9, w_i\left(\overrightarrow{nm}\right) = 0.34$
  - $TT_i = \{T_{ii}, T_{io}, T_{ij}, T_{iu}, T_{in}, T_{im}\} = \{1, 0.63, 0.48, 0.53, 0.71, 0.57\}$

4.4 L'impact de la confiance sur la communication

J'ai construit un modèle de confiance à partir d'un réseau de confiance basé sur les informations échangées entre les agents. L'étape suivante est de comprendre comment utiliser cette confiance pour limiter l'impact de la dissonance. Ainsi, je propose d'utiliser la confiance pour :

1. Construire une méthode permettant à chaque agent de procéder à un examen critique de sa situation actuelle, de s'auto-diagnostiquer pour savoir s'il est lui-même fiable ou non. L'agent va changer ainsi son comportement en fonction de sa situation.
2. Mettre en place une stratégie consistant à séparer les agents peu fiables (avec lequel il est inutile ou nuisible de communiquer) des agents fiables (qui envoient des informations utiles à la communauté). Ce processus permet à un agent de renforcer la communication avec les agents fiables et de refuser les communications provenant d'agents jugés non fiables, afin de réduire la dissonance dans le SMA perturbé. La robustesse du système aux agents non fiables est donc améliorée.
3. Calculer la fiabilité des informations qu'un agent reçoit, et donc de choisir parmi les informations contradictoires qu'il reçoit.

Algorithme 2: Algorithme pour la mise à jour des informations sur les agents

Entrées : $(TT_i)_t = \{T_{ix} \mid x \in V_i\}$, $(TG_i)_t = (V_i, E'_i, w_i)$, $(TG_j)_t = \left(V_j, E'_j, w_j\right)$
Sorties : $(TG_i)_{t+1} = (V_i^*, E_i'^*, w_i^*)$ provenant de la fusion des $(TG_i)_t, (TG_j)_t$,
$(TT_i)_{t+1} = \{T_{ix} \mid x \in V_i^*\}$

$\tau^+ \leftarrow \frac{(Upp - T_{init})}{\rho_{stab} * NI}$
$\tau^- \leftarrow \frac{(T_{init} - Low)}{\rho_{stab} * NI}$
$Inc\_level \leftarrow \frac{\sum_{k \in \{1,\beta\}} \delta(x_k, y_k)}{\beta * \delta_{Max} info}$
$V_i^* = V_i \cup V_j$
si $\overrightarrow{ij} \notin E'_i$ **alors**
  $DT_{ij} = T_{init}$
  $E_i'^* = \left\{\overrightarrow{ij}\right\} \cup E'_i \cup E'_j \setminus \{\overrightarrow{yi}, y \in V_i \cup V_j\}$
fin
si $Inc\_level < \mu$ **alors**
  $DT_{ij} = DT_{ij} + \tau^+$
sinon
  $DT_{ij} = DT_{ij} - \tau^-$
fin
pour tous les *arc* $\overrightarrow{xy}$ *de* $E_i'^*$ **faire**
  si $\overrightarrow{xy} = \overrightarrow{ij}$ **alors**
    $w_i^*(\overrightarrow{ij}) = DT_{ij}$
  sinon si $\overrightarrow{xy} \in E'_i$ && $\overrightarrow{xy} \notin E'_j$ **alors**
    $w_i^*(\overrightarrow{xy}) = w_i(\overrightarrow{xy})$
  sinon si $\overrightarrow{xy} \notin E'_i$ && $\overrightarrow{xy} \in E'_j$ **alors**
    $w_i^*(\overrightarrow{xy}) = w_j(\overrightarrow{xy})$
  sinon /* $\overrightarrow{xy} \in E'_i$ && $\overrightarrow{xy} \in E'_j$ */
    $Path_i \leftarrow$ tous les chemins de TG_i contenant l'arc xy
    $Path_j \leftarrow$ tous les chemins de TG_j contenant l'arc xy
    $w_i^*(\overrightarrow{xy}) = \frac{\sum_{u \in Path_i} T_{iu} * IT_{xy_i} + \sum_{v \in Path_j} T_{iv} * IT_{xy_j}}{\sum_{u \in Path_i} T_{iu} + \sum_{v \in Path_j} T_{iv}}$
  fin
fin
pour tous les *nœud* x *de* V_i^* **faire** /* Mise à jour de la TrustTable */
  $T_{ix} = \frac{T_{ii} * DT_{ix} + \sum_{y \in \{z \in V_i^* \mid \overrightarrow{zx} \in E_i'^*\}} (T_{iy} * IT_{yx})}{T_{ii} + \sum_{y \in \{z \in V_i^* \mid \overrightarrow{zx} \in E_i'^*\}} T_{iy}}$
fin

4. Viser à la formation de coalitions d'agents fiables. Ce processus permet d'améliorer la coopération entre les agents, de minimiser les redondances de données et d'améliorer la qualité des informations échangées.

4.4.1 Auto-évaluation

$(T_{ii})_t$ représente la confiance intrinsèque de i en lui-même à l'instant t. Je propose ici une manière de calculer l'évolution de cette valeur. Lorsqu'un agent i reçoit un TrustGraph d'un agent j, il est possible que dans ce TrustGraph il y ait des arcs qui pointent sur i. La fusion des TrustGraph brise les cycles en particulier en supprimant ces arcs. Or ces arcs portent des informations sur la confiance qu'ont les autres agents sur i. A chaque fois qu'un arc pointe sur i lors de la fusion des TrustGraph, on stocke la valeur de la confiance attachée et l'agent origine dans une base de données. On obtient le tableau $(TE_i)_t$ donnant à i la confiance des autres en lui-même à l'instant t. Il est représenté dans le Tableau 4.1. L'objectif de l'auto-évaluation est de donner à chaque agent la capacité de procéder à un examen critique de sa situation actuelle (par exemple, déterminer si un capteur est en panne ou non) en se basant sur les confiances des autres agents du système en lui-même.

	j	k	l	m	...	i
i	IT_{ji}	IT_{ki}	IT_{li}	IT_{mi}	...	T_{ii}

TABLE 4.1 – Tableau des évaluations $(TE_i)_t$ de i à l'instant t

La question est de déterminer la valeur de confiance T_{ii} à l'instant $t+1$. Pour cela, je propose un calcul de confiance de l'agent i en lui-même à l'instant suivant $t+1$ basé sur les trois type d'informations suivantes :

- $(T_{ii})_t$ la confiance de i en lui-même à l'instant t
- $(TE_i)_t$ le tableau des évaluations de i à l'instant t
- $(IT_{kj})_t$ la confiance d'un agent k en j calculée par i à l'instant t

Définition 15. (RÉPUTATION OBJECTIVE)
La réputation objective d'un agent j calculée par l'agent i, notée $(R_j)_t$, représente la confiance en j de tous les agents ayant communiqué avec i avant l'instant t.

$(R_j)_t$ est calculée par la moyenne des confiances de tout agent k en j $(IT_{kj})_t$ ayant communiqué avec l'agent i au cours de la simulation.

$$(R_j)_t = \frac{\sum_{k \in A_{com}(j)} (IT_{kj})_t}{|A_{com}(j)|} \tag{4.8}$$

où :

- $(IT_{kj})_t$ la confiance indirecte de k en j à l'instant t
- $A_{com}(j) = \{j, k, m, ...\} \subset V_i$ l'ensemble des agents du TrustGraph de i, ayant communiqué avec l'agent j
- $|A_{com}(j)|$ le nombre des agents ayant communiqué avec l'agent j

On remarque que $(R_i)_t$ est la réputation locale objective de l'agent i, calculée par lui-même, représentant la confiance de tous les agents ayant communiqué avec lui avant l'instant t.

Définition 16. (RÉPUTATION SUBJECTIVE)
La réputation subjective de l'agent i, notée $(R_{*i})_t$, représente la confiance en i de tous les agents du TrustGraph de i ayant une meilleure réputation que l'agent i à l'instant t.

$(R_{*i})_t$ est calculée par la moyenne des confiances indirectes en i de tous les agents j $(IT_{ji})_t$ ayant communiqué avec l'agent i, pondérée par la réputation des agents j $(R_j)_t$.

$$(R_{*i})_t = \frac{\sum_{j \in A_{reli}(i)} (R_j)_t * (IT_{ji})_t}{\sum_{j \in A_{reli}(i)} (R_j)_t} \tag{4.9}$$

avec :
- $(R_j)_t$ est la réputation objective de l'agent j à l'instant t
- $(IT_{ji})_t$ la confiance indirecte de j en i à l'instant t
- $A_{reli}(i)$ est l'ensemble des agents j ayant communiqué avec l'agent i qui satisfont la condition $(R_j)_t \geq (R_i)_t$. On a donc en particulier $A_{reli}(i) \subset A_{com}(i)$.

Définition 17. (AUTO-ÉVALUATION)
La confiance intrinsèque $(T_{ii})_{t+1}$ représente la confiance de i en lui-même en tenant compte d'une variation entre sa réputation subjective à l'instant t, noté $(R_{*i})_t$, et celle à l'instant précédent $(R_{*i})_{t-1}$.

On a donc :

$$(T_{ii})_{t+1} = (T_{ii})_t + \Delta T_{ii} \tag{4.10}$$

$$\Delta T_{ii} = (R_{*i})_t - (R_{*i})_{t-1} \tag{4.11}$$

l'Algorithme 3 résume le calcul de l'auto-évaluation effectué par chaque agent.

Algorithme 3: L'algorithme de l'auto-évaluation à exécuter sur chaque agent.

Entrées : $(R_{*i})_{t-1}$: la réputation subjective de i à l'instant $t-1$, $(T_{ii})_t$: la confiance de i en lui-même à l'instant t , $(TE_i)_t$: tableau des évaluations de i
Sorties : $(T_{ii})_{t+1}$: la confiance de i en lui-même à l'instant $t+1$

$A_{com}(i) \leftarrow$ l'ensemble des agents ayant communiqué avec l'agent i
$(T_{ii})_{t+1} = (T_{ii})_t$
$x \leftarrow 0$
$y \leftarrow 0$
$(R_i)_t \leftarrow \frac{\sum_{k \in A_{com}(i)} (IT_{ki})_t}{|A_{com}(i)|}$
pour tous les *agent* $j \in A_{com}(i)$ **faire**
    $(R_j)_t = \frac{\sum_{k \in A_{com}(j)} (IT_{kj})_t}{|A_{com}(j)|}$
    si $(R_j)_t > (R_i)_t$ **alors**
        $x = x + (R_j)_t * (IT_{ji})_t$
        $y = y + (R_j)_t$
    fin
fin
si $y \; != 0$ **alors**
    $(R_{*i})_t = \frac{x}{y}$
    $(T_{ii})_{t+1} = (T_{ii})_t + \frac{(R_{*i})_t - (R_{*i})_{t-1}}{T^t_{*i}}$
fin

Exemple 6. Supposons que l'agent i veuille réévaluer sa confiance en lui-même. On considère que par exemple $R^t_{ii} = 1$, $R^{t-1}_{*i} = 0.6$ et qu'il a rencontré les agents j, k et m. On a donc $A_{com}(i) = \{j, k, m\}$. De plus, d'après les différents échanges, i sait que j a communiqué avec m et i, que m a communiqué avec j, k, i et k avec m et i. On a donc $A_{com}(j) = \{m, i\}$, $A_{com}(m) = \{j, k, i\}$ et $A_{com}(k) = \{m, i\}$. Le TrustGraph de l'agent i, $TG(i)$, est représenté dans la figure 4.5.

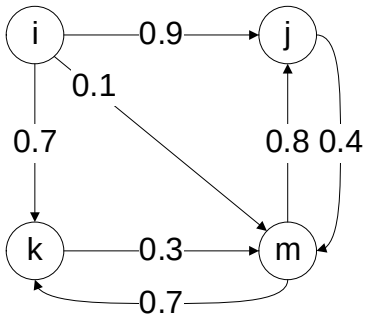


FIGURE 4.5 – Un exemple du TrustGraph de i

et son tableau des évaluations $(TE_i)_t$ est donné dans le tableau 4.2 :

	j	m	k
i	0.5	0.4	0.1

TABLE 4.2 – Un exemple du tableau des évaluations $(TE_i)_t$ de i à l'instant t : on veut calculer $(T_{ii})_{t+1}$

Les réputations objectives des agents i, j ,k et m notée $(R_i)_t, (R_j)_t, (R_k)_t, (R_m)_t$ sont calculées par l'agent i comme suit :

- $(R_i)_t = \frac{IT_{ji}+IT_{ki}+IT_{mi}}{|A_{com}(i)|} = \frac{0.5+0.1+0.4}{3} = 0.33$
- $(R_j)_t = \frac{IT_{mj}+IT_{ij}}{|A_{com}(j)|} = \frac{0.8+0.9}{2} = 0.85$
- $(R_k)_t = \frac{IT_{mk}+IT_{ik}}{|A_{com}(k)|} = \frac{0.7+0.7}{2} = 0.7$
- $(R_m)_t = \frac{IT_{jm}+IT_{km}+IT_{im}}{|A_{com}(m)|} = \frac{0.4+0.3+0.1}{3} = 0.26$

On en déduit que $A_{reli}(i) = \{j, k\}$. Donc i ne prend pas en compte m dans son auto-évaluation car $(R_m)_t < (R_i)_t$. La réputation subjective de l'agent i, noté $(R_{*i})_t$ est calculée comme suit :

$$(R_{*i})_t = \frac{0.85 * 0.5 + 0.7 * 0.1}{0.85 + 0.7} = \frac{0.495}{1.55} = 0.32$$

$$(T_{ii})_{t+1} = (T_{ii})_t + \Delta T_{ii} = (T_{ii})_t + \left((R_{*i})_t - (R_{*i})_{t-1}\right) = 1 + (0.32 - 0.6) = 0.72$$

4.4.2 Une stratégie simple de communication

A chaque instant un agent peut avoir plusieurs agents dans son voisinage de communication. Pour choisir ceux avec lesquels il va communiquer, on a décidé de réutiliser les seuils fixés précédemment. Un agent décide d'interrompre la communication avec un autre agent dès que la valeur de confiance calculée pour cet agent devient inférieure au seuil Low. De cette façon, les mauvaises informations cessent de se rependre dans le réseau de communication et vont même tendre à disparaitre progressivement, en raison du mode de calcul de la fiabilité des informations décrites dans la section suivante. Selon les valeurs de confiances intrinsèques calculées dans leur TrustTable, les agents peuvent modifier leur stratégie de communication. L'idée principale est de séparer l'ensemble des agents en distinguant les agents en qui on ne peut pas avoir confiance (avec lesquels il a déjà été prouvé qu'il est inutile voire nuisible de communiquer) des autres agents (ceux qui pourraient encore fournir des informations fiables et donc utiles). Cette séparation sera de plus en plus précise à chaque pas de l'expérimentation grâce aux apports de nouvelles informations sur l'environnement et sur les agents.

L'Algorithme 4 résume cette stratégie de communication simple effectuée par chaque agent. Cette stratégie reste très simple, le chapitre 5 proposera des règles plus évoluées pour régir la communication.

Algorithme 4: Algorithme pour une stratégie simple de communication de l'agent i

Entrées : $TS_i = (TG_i, TT_i)$: TrustSet de l'agent i
Sorties : i communique ou non avec agent $j \in A^t_{com}(i)$ se trouvant dans le zone de communication de l'agent i

$A^t_{com}(i) \leftarrow$ l'ensemble des agents se trouvant dans le zone de communication de l'agent i à l'instant t
pour tous les *agent* $j \in A^t_{com}(i)$ **faire**
  si $T^t_{ij} > Low$ **alors**
    j reçoit les information directes et indirectes et le TrustGraph transmis par j (c.à.d i communique avec j)
    j exécute l'algorithme 1 de mise à jour des informations de l'environnement sur l'agent i
    j exécute l'algorithme 2 de mise à jour des informations sur les agents
    j exécute l'algorithme 5 d'évaluation des informations contradictoires
    j exécute l'algorithme 3 d'auto-évaluation
  fin
fin

4.4.3 Le calcul de la fiabilité de l'information

En communiquant, les agents reçoivent beaucoup d'informations contradictoires. Parmi ces informations, un certain nombre sont fausses ou inexactes. Les agents doivent donc pouvoir opérer un tri entre les informations fiables et celles qui ne le sont pas. Pour calculer la fiabilité d'une information, chaque agent utilise un arbre de probabilité pour la représenter en mémoire. Chaque valeur ou chaque intervalle de valeurs pour un élément d'information est associé à un nœud de l'arbre. J'associe à chaque arc de l'arbre le couple $(\theta, \{i...n\})$, où θ est la probabilité que l'information soit vraie (sa fiabilité) et $\{i...n\}$ les sources de l'information. Chaque fois qu'une nouvelle information arrive, l'agent met à jour ces valeurs dans l'arbre.

Un exemple d'arbre de probabilité d'information est proposé dans la Figure 4.6 : il représente une information complexe représentée de manière formelle par $info = [u_i, v_i]$ avec $u_i \in Info_1, Card(Info_1) = n$ et $v_i \in Info_2, Card(Info_2) = m$ qui doivent être stockés dans la mémoire de l'agent. On note $\sum_{i=1}^{n} p_i = 1, \sum_{j=1}^{n} p_{ij} = 1, A_{i(i\in[1,R])} \subset V$ et $A^*_{i(i\in[1,S])} \subset V$. La probabilité que l'information u_3, v_2 soit vraie est $p3 * p32$.

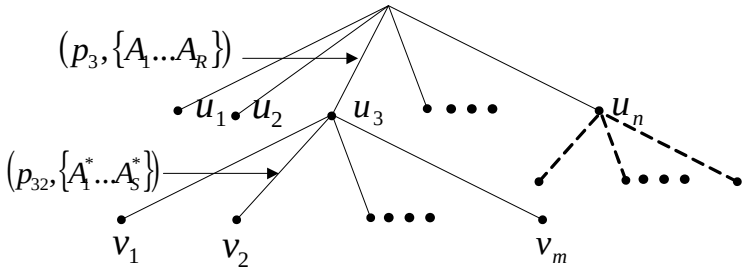


Figure 4.6: Exemple de l'arbre de probabilité d'information

Un des problèmes majeurs est de déterminer la fiabilité d'une information lorsque plusieurs groupes d'agents apportent des données différentes sur un même objet, l'agent doit comparer l'impact de ces groupes. Pour évaluer l'impact (qu'on appelle TrustWeight ou TW dans la suite) d'un groupe d'agents, l'agent divise d'abord chaque groupe en trois ensembles (fiables, non fiables et indéterminés) en réutilisant les deux seuils Upp et Low introduits précédemment. Un poids est attribué à chaque membre d'un ensemble (α pour un agent fiable, γ pour un agent non fiable, β pour un agent indéterminé). Puis, compte tenu du cardinal de ces ensembles, l'agent calcule le TrustWeight de chaque groupe par la somme

pondérée suivante :

$$\begin{array}{c} TW\left(\{a_1,...,a_n\}\right) = \\ \alpha * Card\left(\{j \mid j \in \{a_1,...,a_n\}, Upp \leq T_{ij}\}\right) \\ +\beta * Card\left(\{j \mid j \in \{a_1,...,a_n\}, Low < T_{ij} < Upp\}\right) \\ +\gamma * Card\left(\{j \mid j \in \{a_1,...,a_n\}, T_{ij} \leq Low\}\right) \end{array} \tag{4.12}$$

où T_{ij} est la confiance intrinsèque dans l'agent j stockée dans la TrustTable de i. En fixant des valeurs appropriées pour α, β et γ, cette méthode de calcul vise à créer un équilibre entre la qualité des agents dans un groupe et leur quantité. Le TrustWeight (TW) exprime l'impact d'un groupe sur la fiabilité des informations transmises par ce groupe en relation avec le TrustWeight d'autres groupes. Pour chaque objet de l'environnement p, on note $\{G_1(p), ..., G_n(p)\}$ l'ensemble des groupes qui ont transmis des données différentes sur p et p_x la valeur associée à p et fournie par les agents du groupe $G_x(p) \in \{a_{1x}, ..., a_{nx}\}$. L'agent calcule la fiabilité de chaque donnée p_x par la formule suivante :

$$f(p_x) = \frac{TW\left(G_x(p)\right)}{\sum_{k\in[1,n]} TW\left(G_k(p)\right)} \tag{4.13}$$

Illustrons ce processus avec l'exemple suivant. Supposons que l'agent i ait reçu $\{\langle(x,y), red\rangle\}$ de j, $\{\langle(x,y), blue\rangle\}$ de k, $\{\langle(x,y), blue\rangle\}$ de l et $\{\langle\neg(x,y)\rangle\}$ de m. i peut stocker ces informations dans un arbre de probabilité présenté dans la Figure 4.8. En outre, nous supposons que i dispose du TrustGraph de la Figure 4.7 : $T_{ij} = 0.8$, $T_{ik} = 0.4$, $T_{il} = 0.5$, $T_{im} = 0.3$.

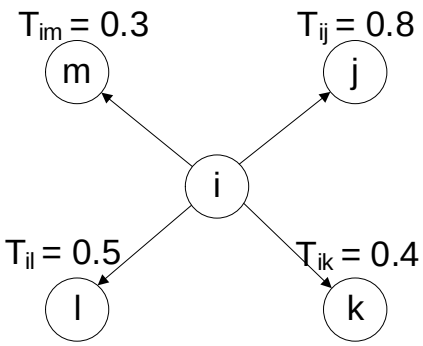


FIGURE 4.7 – Exemple de TrustGraph de l'agent i

Si nous choisissons un seuil bas $Low = 0.25$ et un seuil haut $Upp = 0.8$, i peut calculer le TrustWeight de chaque groupe : $TW(\{m\}) = 1$ et $TW(\{j,k,l\}) = 3*1+1+1 = 5$ si nous choisissons $\alpha = 3$, $\beta = 1$ et $\gamma =$ -1. Ainsi, la fiabilité de l'information F (aucune couleur à la position (x,y)) est $\frac{1}{6}$ et la fiabilité de l'information T (avec une couleur à la position (x,y)) est $\frac{5}{6}$. Le même processus est utilisé pour le niveau de danger. Les résultats sont résumés dans l'arbre de la Figure 4.8. Donc, la probabilité que (x,y) soit un zone de danger de niveau *blue* est $0.83 * 0.4 = 0.33$ (0.4 représentant en effet la probabilité que (x,y) soit un zone de danger de niveau *blue*, une fois prise en compte la possibilité qu'il existe un danger).

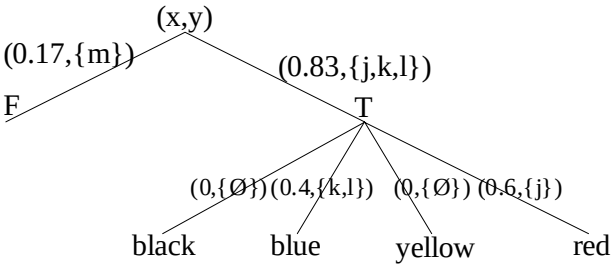


FIGURE 4.8 – Exemple d'arbre de probabilité des informations

Pour l'agent, l'information choisie sera celle qui aura la fiabilité maximale et, à cette information, sera associée sa fiabilité $f(p_x)$. l'Algorithme 5 résume le calcul de la fiabilité de l'information effectué par chaque agent lors de la réception d'informations contradictoires.

Algorithme 5: Algorithme à exécuter lors de la réception d'informations contradictoires

Entrées : une information indirecte $ind_i =< clé, \{< attribut_x, \{a_x\} >\} >$ de $(IND_i)_t$
Sorties : déterminer l'attribut le plus fiable pour un objet donné, $attrMaxClé$

pour tous les *information indirecte reçu* $ind_i =< object, \{< attribute_x, \{a_x\} >\} >| \, ind_i \in (IND_i)_t$ **faire**
  si $taille\,(\{\langle attribut_x, \{a_x\}\rangle\}) > 1$ **alors**
    pour tous les $attribut_x \in \{< attribut_x, \{a_x\} >\}$ **faire**
      $f(< clé, attribut_x >) = \frac{TW(\{a_x\})}{\sum_{k\in[1,n]} TW(\{a_k\})}$
    fin
    $max\,(f(< clé, attribut_x >)) \leftarrow$ coefficient de la fiabilité la plus grande $f\,(< clé, attribut_x >)$ pour tout attribut de la clé
    pour tous les $attribut_x \in \{< attribut_x, \{a_x\} >\}$ **faire**
      si $< clé, attribut_x >= max\,(f(< clé, attribut_x >))$ **alors** $attrMaxClé = attribut_x$
    fin
  fin
fin

4.4.4 Vers la formation de coalitions d'agents fiables

En raison de la mobilité des agents dans le système de collecte d'information distribué que j'étudie, la topologie du réseau physique composé par les agents a un impact significatif sur le système de communication. Comme indiqué dans la figure 4.9, dans un premier temps, les rencontres entre agents vont induire une organisation sociale particulière entre les agents associée aux TrustSets. Une fois cette organisation sociale stabilisée, je propose d'utiliser ces confiances pour induire une auto-organisation spatiale des agents, formant ainsi des « clusters » : les agents qui ont pleinement confiance les uns en les autres vont rester connectés, afin de collaborer plus efficacement. Après sa formation, un cluster peut être considéré comme un réseau ad-hoc ayant émergé de l'organisation sociale. Avec la confiance comme critère de création des clusters et de leur maintien, les agents dissonants seront automatiquement exclus du cluster.

Le contrôle de la communication basée sur le concept de cluster semble être une voie prometteuse en raison des avantages qu'elle apporte au système :

- une meilleure coopération entre les agents : un cluster peut améliorer la stratégie de déplacement des agents par le partage des tâches pour terminer leur travail plus rapidement.
- une minimisation de la redondance des données induite car les agents collaborant au lieu de rivaliser.
- un échange d'information plus rapide et plus sécurisé dans un cluster stable où la perturbation est minimum.

Pour définir le cluster, on introduit les notations suivantes.

Définition 18. (Distance spatiale).
On note $\delta(i, j)$ la distance spatiale entre deux agents i et j.

Définition 19. (Ensemble de voisins).
On note $VN = \{i \,|\, i \in V, \exists j \in V,\, i \neq j,\, \delta(i, j) \leq r_c\}$ l'ensemble des agents qui ont au moins un agent à une distance inférieure au rayon de communication des agents r_c.

Définition 20. (Liens entre voisins).
L'ensemble des liens entre les agents voisins spatialement est noté $EN(V) = \{\overline{ij} \;|\; i, j \in VN$ et $\delta(i, j) \leq r_c\}$.

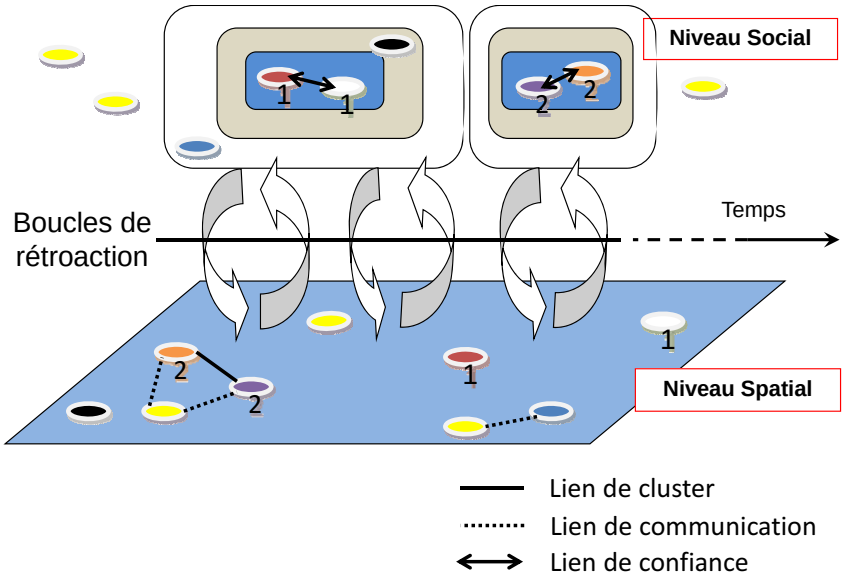


Figure 4.9: Un exemple de cluster émergeant au niveau spatial : une proximité sociale se traduit par une proximité physique ou une proximité de comportements (une proximité permettant l'échange et le partage d'information).

Définition 21. (AGENTS VOISINS).
Deux agents i et j sont dits voisins ssi $\overline{ij} \in EN(V); i, j \in V$.

Définition 22. (CLUSTER).
Un cluster, noté $Cl = (V_{Cl}, E_{Cl})$, est un graphe qui satisfait à un instant t les contraintes ci-dessous :
(1) $V_{Cl} \neq \emptyset$
(2) $E_{Cl} \neq \emptyset$
(3) $V_{Cl} \subseteq VN$
(4) $E_{Cl} \subseteq EN(V)$
(5) $\forall x, y \in V_{Cl}^2$, $\overline{xy} \in E_{Cl}$
(6) $\forall \overline{xy} \in E_{Cl}$, $T_{xy} \geq Upp$ *et* $T_{yx} \geq Upp$, *où* Upp *(*$Upp \in [0,1]$*) est le niveau de confiance au-dessus duquel un agent est considéré comme fiable.*

Dans la suite, Cl_x avec $x \in V$ représentera le cluster dans lequel x est un élément (*i.e.* $x \in V_{Cl_x}$). Je peux montrer, grâce aux contraintes (4) que, dans le cas où il y a un agent dans cluster, il y a toujours au moins un autre agent y dans ce cluster : $\exists y \in V_{Cl_x}, y \neq x, \overline{xy} \in E_{Cl_x}$.

Pour tout groupe d'agents $T \subset V$, Cl_T représentera le cluster dans lequel $V_{Cl_T} = T$.

D'après le point précèdent, il n'est pas possible d'avoir $Cl_{\{x\}}$.

Cl_{xy}, avec $x, y \in V^2$, représentera le cluster composé par les deux agents x et y. $V_{Cl_{xy}} = \{x, y\}$, $E_{Cl_{xy}} = \{\overline{xy}\}$; donc $Cl_{xy} = Cl_{\{x,y\}}$.

Dans la suite, je présente les mécanismes menant à la création du cluster ainsi que ceux gérant son évolution.

4.4.4.1 Création d'un cluster

J'étudie dans cette section comment un cluster peut se former. Un cluster est initialisé par deux agents qui répondent à deux conditions : ils se font mutuellement confiance et ils sont voisins. Le regroupement d'un point de vue spatial va se passer une fois que les agents se feront mutuellement confiance, donc ce processus prendra du temps parce que les valeurs de confiance évoluent progressivement à partir d'une valeur initiale de confiance fixée à $T_{init} = 0.5$. L'algorithme 6 présente la façon dont deux agents voisins

qui n'appartiennent à aucun cluster et qui se font mutuellement confiance vont construire un cluster. Un des deux agents prend l'initiative de proposer à l'autre de créer un cluster. Le second acceptera s'il a suffisamment confiance en le premier.

Algorithme 6: Algorithme pour la formation de clusters

Entrées : Deux agents voisins x, y qui n'appartient à aucun cluster
Sorties : Un nouveau cluster Cl_{xy} est crée avec deux agents en cas de succès. Autrement, rien fait

si $T_{xy} \geq Upp$ **alors**
  L'initiateur x envoie la demande de formation du cluster à son voisin y
  si $T_{yx} \geq Upp$ **alors**
    y répond positivement à la demande de formation d'un cluster de x
    x crée un nouveau cluster Cl_{xy} et informe y de la création du cluster
  fin
fin

On remarque que comme $Cl_{xy} = Cl_{yx}$ par définition d'un cluster, les rôles des agents ne sont pas significatifs : le même cluster est produit s'il est créé par x ou par y.

4.4.4.2 Fusion de clusters

Après un certain temps, nécessaire aux agents pour obtenir une bonne évaluation de la confiance dans les autres agents, les agents commencent à créer des clusters à l'aide de l'algorithme 6. Cet algorithme peut être utilisé dans les cas où les agents ne font pas partie de clusters. Toutefois, lorsqu'au moins un des deux agents appartient à un cluster, ils doivent appliquer un autre algorithme présenté dans l'algorithme 7 qui décrit comment un agent est intégré à un cluster et comment deux clusters peuvent fusionner. Un agent y est intégré dans les cluster Cl_x si et seulement si pour chaque agent $z \in Cl_x$, il existe une valeur de confiance $T_{zy} \geq Upp$ et un cluster Cl_x est intégré au cluster Cl_y si et seulement si pour tout agent $t \in Cl_x$ et $r \in Cl_y$ il existe une valeur de confiance $T_{tr} \geq Upp$.

Algorithme 7: Algorithme pour la fusion de clusters

Entrées : Deux agents voisins x, y avec l'agent x appartenant à Cl_x
Sorties : La mise à jour du cluster Cl_x

si $T_{zy} \geq Upp, \forall z \in Cl_x$ **alors**
  x envoie la demande de formation du cluster à son voisin y
  si $\exists Cl_y \,\&\&\, (T_{kx} \geq Upp, \forall k \in Cl_y)$ **alors**
    y répond positivement à la demande de x de formation d'un cluster
    x met à jour son cluster $Cl_x = Cl_x \oplus Cl_y$ et informe y de l'information du cluster
  fin
  si $\nexists Cl_y \,\&\&\, T_{yx} \geq Upp$ **alors**
    y répond positivement à la demande de x de formation d'un cluster
    $Cl_y = (xy, \overline{xy})$
    x met à jour son cluster $Cl_x = Cl_x \oplus Cl_y$ et informe y de la formation du cluster
  fin
fin

L'opérateur $\oplus$ introduit dans l'algorithme 7 représente l'opérateur de fusion de deux clusters.

Définition 23. (Opérateur de fusion $\oplus$). Considérons $Cl_x = (V_{Cl_x}, E_{Cl_x})$ et $Cl_y = (V_{Cl_y}, E_{Cl_y})$ deux clusters différents. Je définis $Cl_z = Cl_x \oplus Cl_y$ comme $Cl_z = (V_{Cl_z}, E_{Cl_z})$ avec $V_{Cl_z} = V_{Cl_x} \cup V_{Cl_y}$ et $E_{Cl_z} = E_{Cl_x} \cup E_{Cl_y} \cup \{\overline{xy}\}$.

Théorème 1. *Si l'agent x appartient à la fois à Cl_x et à Cl'_x alors $Cl_x = Cl'_x$.*

Démonstration. Si $\exists Cl_x$ tel que $x \in Cl_x$ et $\exists Cl'_x$ tel que $x \in Cl'_x$ alors en appliquant l'opération de fusion $\oplus$ sur les deux clusters, on obtient : $Cl_x = Cl_x \oplus Cl'_x$, $Cl'_x = Cl_x \oplus Cl'_x \Longrightarrow Cl_x = Cl'_x$. □

4.4.4.3 Exclusion d'un élément d'un cluster

Un agent y doit être exclu d'un cluster Cl_x si et seulement s'il existe un élément z de Cl_x tel que la valeur de confiance T_{zy} devient inférieure à Upp.

Définition 24. (Opérateur d'exclusion $\ominus$). Considérons $Cl_x = (V_{Cl_x}, E_{Cl_x})$ un cluster et $y \in V_{Cl_x}$ un agent. Je définis $Cl_z = Cl_x \ominus y$ comme $Cl_z = (V_{Cl_z}, E_{Cl_z})$ avec $V_{Cl_z} = V_{Cl_x} \setminus \{y\}$ et $E_{Cl_z} = E_{Cl_x} \setminus \{\overrightarrow{ky} \mid \forall k \in V_{Cl_x}\}$

4.4.4.4 Cluster TrustGraph

J'aborde dans cette section l'impact des liens entre les TrustGraphs stockés dans chaque agent appartenant à un cluster (rappelons qu'un cluster est un groupe d'agents répondant à deux conditions : ils se font mutuellement confiance et ils sont voisins). J'aspire à construire un TrustGraph partagé stocké dans chaque individu et pouvant être partagé avec d'autres dans le cluster (c.à.d tous les agents dans un cluster pourraient avoir le même TrustGraph). De cette façon, des informations fiables sont échangées au niveau local à travers des interactions individuelles pour éviter de nombreux inconvénients, tels qu'un point unique de défaillance, l'exigence de l'infrastructure et le problème de « *goulot d'étranglement* ».

Définition 25. (Cluster TrustGraph).
Le TrustGraph partagé et stocké dans chaque agent du cluster est appelé Cluster TrustGraph.

Définition 26. (Liens bidirectionnel).
Un double lien $\left(\overrightarrow{ij}, \overleftarrow{ij}\right) \in E'(V)^2$ noté $\overleftrightarrow{ij}$ est appelé un lien bidirectionnel si et seulement si $T_{ij} \geq Upp$ et $T_{ji} \geq Upp$.

Le TrustGraph partagé d'un cluster, appelé **Cluster TrustGraph**, doit être conçu de manière distribuée et adaptée à la dynamique de la topologie du cluster et à ses changements en terme d'agents membres. Pour le construire, on fait les mêmes étapes que lors de la fusion de deux TrustGraphs différents. Mais dans le cas du cluster, on permettra d'avoir un arc bidirectionnel dans le Cluster TrustGraph. Plus précisément, après avoir reçu un TrustGraph, le TrustGraph de l'agent x sera mis à jour quand il reçoit le TrustGraph de l'agent y suivant les trois étapes ci-dessous :

1. x calcule sa confiance T_{xy} en y ou met à jour la confiance existante en comparant ses propres données avec celles reçues. Dans le cas d'un nouveau cluster, Cl_{xy} est créé et un lien de confiance directe $\overrightarrow{xy}$ dans son TrustGraph est remplacé par un lien bidirectionnel $\overleftrightarrow{xy}$.
2. x relie le TrustGraph de y à son propre TrustGraph ;
3. x corrige toutes les incohérences dans les chemins partagés.

Lorsque les agents rencontrent de nouveaux voisins (ou quand un nouveau agent entre dans le cluster), ils échangent et mettent à jour leurs TrustGraphs en utilisant les algorithmes de fusion des TrustGraphs. Par exemple, quand un agent rencontre des anciens voisins, après avoir vérifié les informations de la formation du TrustGraph, il décide de mettre à jour ou non son propre TrustGraph (c.à.d lorsque l'agent détecte qu'un des éléments fondamentaux de son TrustGraph a changé, il mettra à jour son TrustGraph). Pour ce faire, le TrustGraph envoyé par un agent doit contenir des informations sur sa formation (par exemple les identifiants des agents dont les TrustSets ont été fusionnés, le temps de fusions, etc.).

Basé sur son TrustGraph partagé, un agent i peut calculer sa TrustTable privée pour l'estimation de la valeur de confiance attribuée par son cluster à un autre agent y (y n'appartient pas au cluster de i) en utilisant la formule suivante :

$$T_y^{Cl} = \frac{\sum_{x \in V_{Cl}} T_{xy}^i}{Card\,(V_{Cl})} \tag{4.14}$$

où :

- T_y^{Cl} désigne la confiance du cluster Cl en l'agent y
- T_{xy}^i la confiance intrinsèque de l'agent x ($x \in V_{Cl}$) en l'agent y dans la TrustTable de i

4.4.4.5 Dynamique de la topologie du cluster

Cette section décrit trois topologies différentes d'un même cluster correspondant aux niveaux qui sont construits dans les sections précédentes : l'organisation spatiale, l'organisation sociale et l'organisation des confiances intrinsèques. Deux de ces topologies sont virtuelles (la sociale et la confiance intrinsèque), l'autre est réelle et observable dans l'environnement. Elles s'influencent mutuellement, c.à.d que à savoir tout changement au niveau des confiances intrinsèques va produire des changements immédiats dans les organisations sociales et spatiales.

Niveau de l'organisation spatiale (Organisation spatiale du cluster) : Dans l'organisation spatiale, il n'y a qu'une seule et unique organisation physique, la mobilité des nœuds va créer une dynamique de la topologie du cluster. Cette dynamique varie en fonction des déplacements de chaque agent dans le cluster et de la reconfiguration du cluster (qui est induite par les opérations de fusion et d'exclusion diverses). La topologie choisie pour les clusters jouent un rôle important dans la stratégie de communication entre les agents et la performance du système. Par exemple, une topologie dense induit un grand chevauchement dans la domaine de l'exploration et une bonne fiabilité des données, mais une mauvaise performance en temps ; au contraire une topologie clairsemée sera vulnérable aux défaillances de liaison et de partitionnement de réseau mais permettra une meilleure exploration.

En outre, l'organisation spatiale peut être considérée comme un méta-agent ayant une couverture aussi grande que possible du terrain pour collecter autant de données nouvelles que possible. D'autre part, les agents impliqués dans cette organisation spatiale doivent rester en contact les uns avec les autres pour assurer un canal de communication fiable pendant toute la mission.

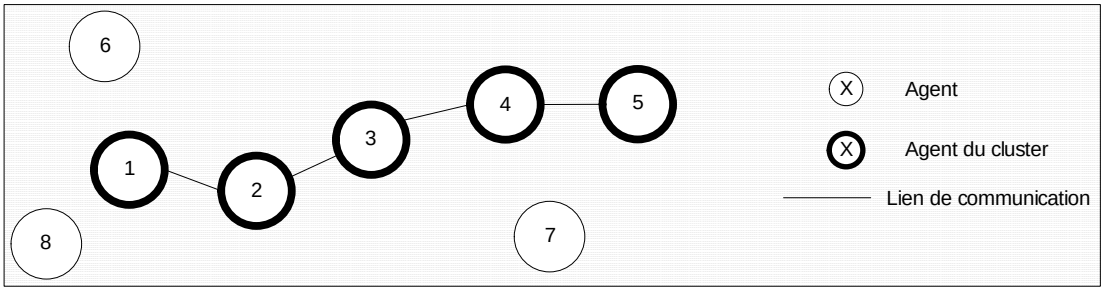


Figure 4.10: Organisation spatiale du cluster

Un exemple de *l'organisation spatiale* est illustré dans la Figure 4.10 qui représente **un graphe non orienté** $G_{spa} = (V_{spa}, E_{spa})$ où $V_{spa} = \{1, 2, 3, 4, 5\}$ est l'ensemble des agents dans le cluster et $E_{spa} = \{\overline{12}, \overline{23}, \overline{34}, \overline{45}\}$ l'ensemble de liens entre eux. L'introduction ou l'exclusion d'agents dans ou à partir de ce cluster peut provoquer des changements importants dans sa topologie. Par exemple, si l'agent 3 est exclu du cluster, les agents 2 et 4 auront pour but de se rapprocher de sorte à ce que la connectivité entre les agents du cluster soit maintenue.

Niveau de l'organisation sociale (Cluster TrustGraph) : L'organisation sociale est liée à la fois aux agents du cluster et aux agents sur lesquelles les agents du cluster ont calculés une confiance. Cette organisation virtuelle est représentée par un « *Cluster TrustGraph* » qui contient à la fois des **liens de confiance fiable** matérialisant les liens entre agents à l'intérieur de cluster et des liens simples de confiance qui matérialisent les liens entre les agents des deux au moins un dont n'appartient pas au cluster. Un **Cluster TrustGraph** a plusieurs origines alors que le TrustGraph d'un agent a une seule origine (l'agent lui-même). Un Cluster TrustGraph peut être considéré comme la fermeture transitive des TrustGraphs de ses membres.

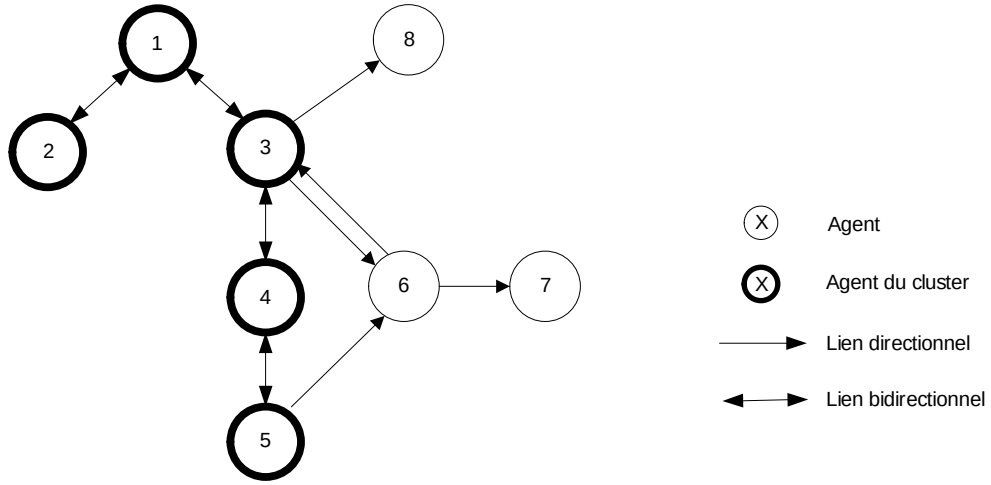


Figure 4.11: Cluster TrustGraph

La figure 4.11 illustre un *Cluster TrustGraph* qui peut être associé à *l'organisation spatiale des clusters* construit dans la figure 4.10. Le *Cluster TrustGraph* est représentée par un graphe $G_{soc} = (V_{soc}, E_{soc})$ où $V_{soc} = \{1, 2, 3, 4, 5, 6, 7, 8\}$ et $E_{soc} = \left\{\overleftrightarrow{12}, \overleftrightarrow{23}, \overleftrightarrow{34}, \overleftrightarrow{45}, \overrightarrow{38}, \overrightarrow{36}, \overrightarrow{63}, \overrightarrow{56}, \overrightarrow{67}\right\}$. Dans E_{soc}, on trouve des liens de confiance fiable et des liens de confiance simple. $\overleftrightarrow{12}$ est un lien fiable qui signifie que l'agent 1 et l'agent 2 ont mutuellement une confiance directe l'un en l'autre, $T_{12} \geq Upp$ et $T_{21} \geq Upp$. $\overrightarrow{36}, \overrightarrow{63}$ est un aussi un lien de confiance bidirectionnelle, mais au moins l'agent 3 ou l'agent 6 a une confiance sur son partenaire de moins de Upp. Sinon, l'agent 6 serait inclus dans le cluster.

Niveau de confiance intrinsèques (TrustGraph intrinsèque du Cluster) : Ce deuxième niveau virtuel prend en compte les valeurs des confiances intrinsèques calculées par chaque agent. Par définition d'un cluster, il doit y avoir un lien fiable entre tous les nœuds du cluster. Cela signifie que n'importe quel agent du cluster a une confiance intrinsèque supérieure au seuil Upp envers tous les agents du cluster. C'est une condition indispensable à l'acceptation d'un agent dans un cluster et à son maintien. Si un de ces liens n'est plus fiable, un des agents doit quitter le cluster.

Notons que ces liens ne sont pas des liens physiques. Cette représentation décrit simplement la qualité des liens de confiance associant les membres du cluster. Ce TrustGraph contient également les connexions intrinsèques simples aux agents en dehors du cluster. Ce niveau est plutôt liée à la TrustTables des agents à partir de laquelle la notion de cluster est dérivée.

Un exemple de TrustGraph intrinsèque du cluster est illustré dans la figure 4.12. Le cluster est représenté par un **graphe complet bidirectionnel** $G_{iTG} = (V_{iTG}, E_{iTG})$ où $V_{iTG} = \{1, 2, 3, 4, 5\}$ et $E_{iTG} = \{\overleftrightarrow{xy} \mid x, y \in V_{iTG}; x \neq y\}$.

4.4.4.6 Propriétés générales du Cluster

Dans cette section, je présente comment le cluster permet d'améliorer les performance du système. Pour cela, je considère le cluster comme un réseau mobile ad-hoc MANET [Drira et al., 2010], et j'applique des algorithmes existant dans ces réseaux pour assurer le maintien de la connectivité entre les agents du cluster, pour fournir un système de communication fiable et pour améliorer la coopération en se basant sur des rôles. Un cluster bien organisé va apporter de nombreux avantages de performance pour le système comme une meilleure coopération des agents, une minimisation des redondances de données et une amélioration de la qualité des informations échangées.

Communication fiable. Dans un système de collecte d'information distribué, les agents doivent communiquer et coopérer efficacement. De nombreuses études ont conclu que même l'échange d'une petite

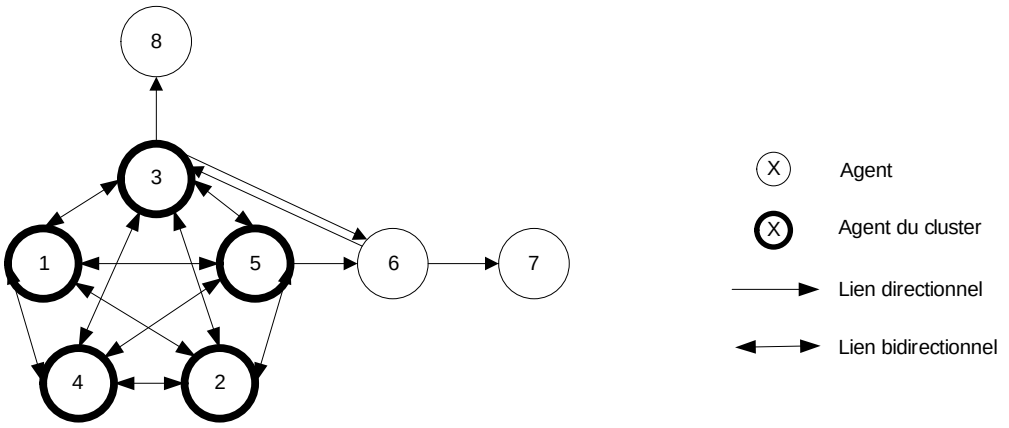


Figure 4.12: Cluster TrustGraph au niveau de confiance intrinsèque

quantité d'information améliore les performances d'un SMA pour certaines tâches [Maclennan, 1991, Balch and Arkin, 1995].

Pour atteindre un degré élevé de flexibilité et d'autonomie, la communication entre les agents devrait être fondée sur les technologies de communication sans fil. En outre, la technologie de communication utilisée doit permettre aux agents de s'auto-organiser pour être opérationnels sans aucune administration centralisée, et doit être capable de s'adapter à la mobilité des agents pendant leur mission. Un réseau avec de telles caractéristiques est connu sous le nom de réseau mobile ad hoc : un **MANET** (A mobile ad-hoc network). Ces caractéristiques font qu'un **MANET** est très flexible et facile à déployer. Pour cette raison, l'utilisation d'un **MANET** pour la communication entre agents appartenant au même cluster où je ne peux raisonnablement supposer l'existence d'une infrastructure de communication robuste, est extrêmement appropriée.

Un agent appartenant à un cluster n'est pas seulement un nœud *"ordinaire"* du réseau, mais aussi un routeur qui relaie les messages de ses voisins. La communication entre les agents qui ne sont pas voisins peut donc avoir lieu au sein des nœuds consécutifs relais intermédiaires. La communication dans les clusters peut être interprétée comme suit en termes de réseaux ad hoc : quand les agents voisins et fiables ont constitué un cluster, ils sont considérés comme un type de réseau ad hoc. Ainsi, ils peuvent communiquer et compter sur le protocole de routage des **MANET** pour la transmission de messages de sorte que dans une période de temps limitée, un message envoyé par un agent est reçu correctement par tous les éléments de son cluster. Comme chaque agent impliqué dans la transmission de l'information (les agents de cluster) est supposé fiable, l'information reçue peut être acceptée par les agents sans défiance.

Maintien de la connectivité d'un réseau de communication. Un des principaux intérêts des clusters est la possibilité de maintenir la connectivité entre leurs éléments pour assurer un canal de communication stable entre les agents fiables. Le problème du maintien de la connectivité consiste à assurer l'existence d'un canal de communication fiable tout au long de la mission. La difficulté dans le maintien de la connectivité entre les agents du cluster est que n'importe quel agent peut potentiellement briser le cluster et provoquer la disjonction de tous les agents du reste du cluster. Je ne peux pas pré-supposer la disponibilité d'une infrastructure de communication opérationnelle et compatible avec les agents. Aussi, les agents doivent constituer et maintenir eux-même un réseau sans fil mobile. Dans notre contexte, j'utilise un algorithme existant pour maintenir la connectivité entre les agents du cluster en utilisant les mécanismes proposées par Le et al. [Le et al., 2009].

Afin de maintenir la connectivité entre les éléments du réseau, j'utilise la notation de «**Nœud de Référence**» comme axe du cluster. Généralement, l'agent avec le plus grand nombre de connexions avec des agents du cluster est choisie. Autrement dit, le choix d'un «**Nœud de Référence**» permet de répartir équitablement la tâche du maintien de la connectivité sur les agents individuels. Puis chaque agent du cluster tout en accomplissant sa tâche, doit rester en contact avec au moins un agent voisin ayant un chemin vers le «**Nœud de Référence**». Si tout agent utilise cette stratégie, alors la connectivité du

système sera assurée. Une connectivité pour un agent donné est matérialisée par une table de connectivité contenant un ensemble de chemins d'accès qui représentent une vue partielle de la connectivité du réseau.

Coopération à base de rôles. Par ailleurs, je vise à créer une coopération basée sur les rôles pour les agents du cluster. Je considère un rôle comme la description d'un comportement bien identifié. Plusieurs rôles sont en général nécessaires pour réaliser une tâche donnée. Je présente l'utilisation des rôles pour décrire les coalitions nécessaires à la réalisation des différentes tâches possibles dans une mission donnée.

L'idée de base de notre approche est de considérer chaque cluster en tant que groupe. Le **« Nœud de Référence »** joue le rôle de gestionnaire du groupe. La deuxième étape consiste à construire une description du système et de la fournir aux agents du cluster afin qu'ils puissent en tenir compte dans leur processus de prise de décision. Ainsi, les agents choisissent indépendamment les rôles qui leur semble les plus appropriés pour eux-mêmes. Le **« Nœud de Référence »** peut aider les agents du cluster en cas de conflit entre les agents (par exemple, de nombreux agents veulent exécuter la même tâche ou le même rôle). Pour résoudre les conflits, un protocole d'attribution des rôles basés sur les enchères est généralement utilisé. Dans un premier temps, chaque agent envoie au **« Nœud de Référence »** son offre sur le prix qu'il demande pour terminer la tâche (le prix prend la forme de notre application d'un niveau d'énergie, une période de temps ...). Ensuite, le nœud de référence sélectionne l'agent proposant le prix le plus faible pour la tâche. Le choix du nœud de référence dépend de l'objectif de l'application et pourrait impliquer de multiples critères comme le niveau d'énergie, le nombre de voisins, les exigences matérielles, etc. Il ne diffère pas beaucoup d'un protocole Contract Net [Smith, 1980].

Le protocole d'attribution des rôles dont j'ai besoin pour un cluster doit permettre des changements dynamiques dans l'organisation globale. Il est inspirée par l'algorithme DMAC (Distributed and Mobility-Adaptive Clustering) proposé par Basagni [Basagni, 1999] pour partitionner un réseau mobile ad hoc **MANET** en clusters.

Dans le projet de cartographie par agents mobiles [Nguyen Vu et al., 2010], j'ai fait la distinction entre deux rôles. Le premier est le **« rôle de noyau »** avec comme objectif le maintien de la réalité physique du cluster qui est fondé sur un « noyau » fiable d'agents. Le second est le **« rôle d'explorateur »** avec comme objectif de recueillir un maximum de données pour le bénéfice du cluster. Ce choix vient de l'idée que la contrainte de rester physiquement connecté à un cluster peut être considérée comme une limite aux capacités d'exploration de l'ensemble de ses agents. Ainsi, certains agents (le nombre est liée à la taille du cluster réel) restent dans le noyau du cluster et les autres peuvent quitter le noyau du cluster pour aller explorer, certains rendez-vous avec le cluster sont prédéfinis et sont stockés dans leur mémoire. Quand ils reviennent au cluster, les explorateurs partagent leurs connaissances avec le noyau (non seulement de nouvelles données fiables recueillies à partir du terrain, mais aussi des nouvelles métadonnées sur les agents inconnus du cluster) et améliorent ainsi le champ d'exploration du cluster.

4.5 Premiers résultats

4.5.1 Résultats sur une application élémentaire

Un modèle simplifié d'un système de collecte d'information a été implémenté sur la plate-forme GAMA [Amouroux et al., 2007] pour tester si l'introduction de la confiance permettait d'améliorer la qualité de la communication dans un SMA perturbé. Cette version simplifiée utilise un environnement représente par une grille sur la quelle des agents peuvent se déplacer et communiquer pour collecter des informations. Lorsqu'ils rencontrent d'autres agents, ils communiquent avec eux. Dans cette version simplifiée, la dangerosité d'une case de l'environnement est codée en utilisant un code de couleur : du rouge au bleu pour représenter des dangers du plus fort au moins fort.

Le simulateur développé pour cet exemple est illustré dans la figure 4.13. Dans ce simulateur, je vais tester les performances du système lorsque les robots utilisent différentes stratégies de communication et comment les différents paramètres de la simulation influencent sur la robustesse du système aux robots défectueux.

Les 4 stratégies de communication qu'on va tester sont les suivantes :

– les robots ne communiquent pas

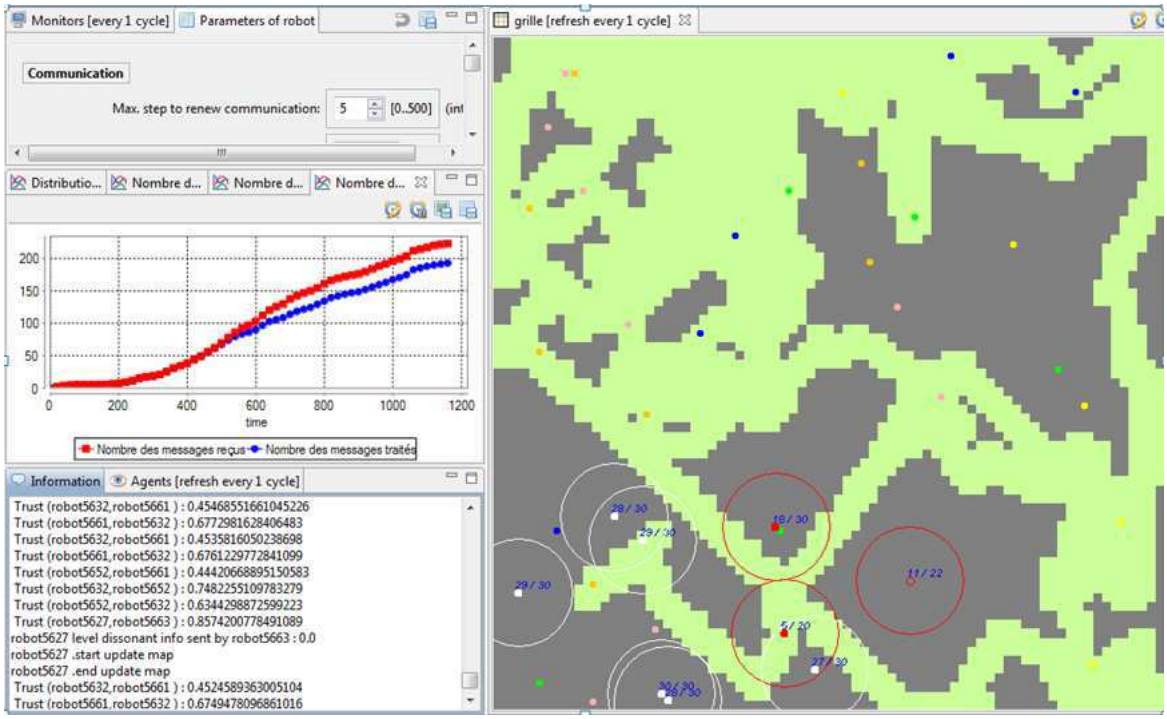


Figure 4.13: Simulateur Danger Mapping en utilisant un environnement représentant par une grille

– les robots ne communiquent que des données
– les robots communiquent des données et les TrustTables
– les robots communiquent des données et les TrustGraphs mais gardent privées les TrustTables

Dans mon approche, on utilise des seuils pour déterminer les agents (et les informations) fiables ou non. Un problème majeur est comment peut-on choisir des seuils convenables. Je vais répondre à cette question dans la prochaine section où des valeurs des seuils sont déterminés par expérimentations.

4.5.2 Seuil de fiabilité de l'information

La première expérimentation que nous avons réalisée avec le simulateur porte sur la valeur du seuil de fiabilité de l'information, et en particulier le minimum nécessaire pour obtenir une information globale fiable, en utilisant les stratégies de communication avec les TrustTable ou TrustSet. Pour obtenir ces résultats, nous avons lancé 50 simulations en faisant varier la valeur du seuil de fiabilité de l'information de $0,5$ à 1, en utilisant un environnement de simulation fixe. Nous pouvons voir que, pour des seuils supérieurs ou égaux à $0,6$, la distance entre la carte obtenue par les agents et la carte réelle est égale à zéro. Si le seuil choisi est de $0,55$ par exemple, le nombre moyen de zones de danger erroné identifié par les agents fiables à la fin de la simulation est de $2,11$ pour la stratégie TrustTable et de $0,64$ pour la stratégie TrustSet. Nous allons donc utiliser la valeur de $0,6$ comme seuil de fiabilité lors des expérimentations suivantes.

4.5.3 Performances du système en utilisant 4 stratégies de communication

La figure 4.14 représente le temps maximal d'exploration nécessaire pour que tous les agents fiables obtiennent une carte exacte avec les 4 stratégies de communication ci-dessus. Toutes les simulations sont lancées dans le même environnement, c'est-à-dire que la position des zones dangereuses et la position initiale des agents sont identiques pour toutes les simulations. Pour chaque simulation, nous recueillons le nombre de pas de temps nécessaire pour atteindre la fin de la simulation, c'est-à-dire quand tous les

agents fiables ont obtenu une carte correcte, identique à la carte réelle (dans laquelle tous les endroits dangereux sont positionnés correctement et avec le bon niveau de danger).

Nous avons constaté que lorsque la proportion d'agents défectueux dans un système est inférieure à 0.75 (15 agents non fiables parmi les 20 agents) (dans les simulations avec 7, 10 et 13 agents défectueux dans la figure 4.14), l'utilisation de la stratégie TrustSet pour détecter les agents défectueux est plus efficace que les autres stratégies. En particulier, le temps pour que tous les agents du système obtiennent la carte réelle est le meilleur en raison des avantages acquis par la communication avec (et seulement avec) les autres agents fiables. Ce résultat semble logique : plus le nombre d'agents fiables dans le système est grand, plus la communication est utile. Par contre, lorsque le système compte de nombreux agents défectueux, les performances des quatre stratégies de communication sont identiques parce que la plupart des informations reçues ne sont pas fiables et que les agents doivent donc vérifier eux-mêmes toute la carte. Ce taux de 0.75 apparaît comme la limite de robustesse du système. Ce nombre est très intéressant car il décrit la proportion d'agents défectueux qu'un système peut supporter. Lors le taux d'agents défectueux est supérieur à cette valeur, il devient préférable pour les agents fiables de ne pas communiquer car il y a trop de perturbations dans le système.

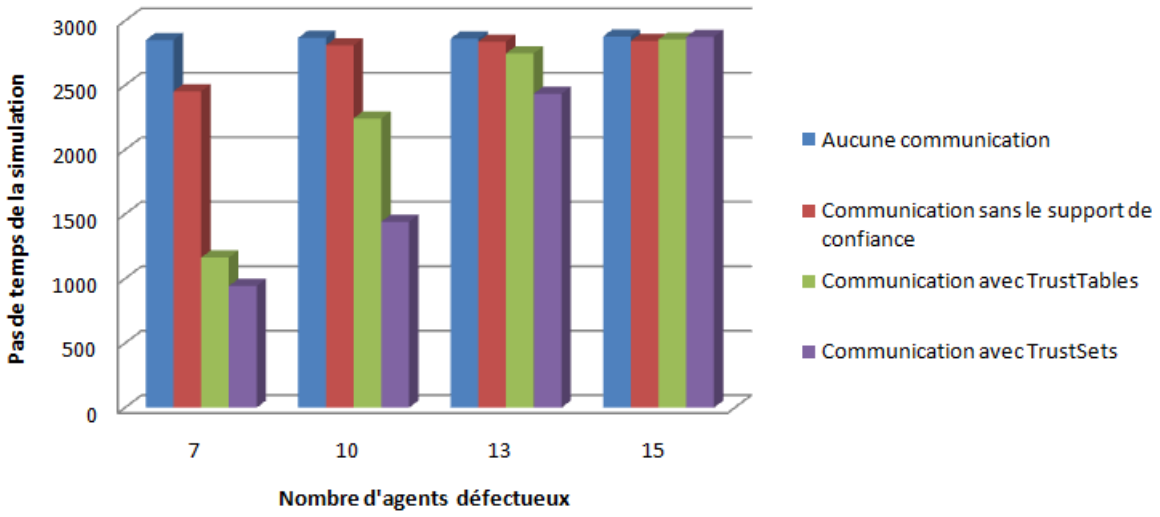


FIGURE 4.14 – Nombre de pas de temps nécessaires pour obtenir une carte réelle avec les 4 stratégies de communication

La figure 4.15 illustre le temps moyen de détection pour toutes les stratégies de communication, c'est-à-dire le temps moyen nécessaire aux agents fiables du système pour détecter correctement la carte. Pour obtenir ce résultat, nous avons lancé 50 fois de simulations pour lesquelles le nombre d'agents défectueux varie de 1 à 15 avec un nombre total d'agents fixe. Le graphique dans la figure 4.14 souligne que nous pouvons utiliser la stratégie TrustSet pour obtenir de bonnes performances aussi longtemps que nous sommes sous la limite du système de 0.75. Plus généralement, lorsque les agents n'ont aucune idée de la proportion d'agents défectueux, il montre que la meilleure stratégie de communication reste la communication avec TrustSets.

4.5.4 Une stratégie de communication simple

L'utilisation du TrustSet a également été introduite pour influencer le système de communications. Quand un agent sait que d'autres agents vont lui envoyer de mauvaises informations, pourquoi persisterait-il à communiquer avec eux ? Il est préférable pour cet agent d'arrêter de communiquer avec de tels agents dès qu'il est certain qu'ils ne sont pas fiables. La stratégie adoptée par les agents dans notre simulation se présente comme suit. Quand un agent A rencontre un agent B,

1) il peut lui envoyer ou non une information (lorsque sa confiance en B est supérieure ou inférieure à Low),

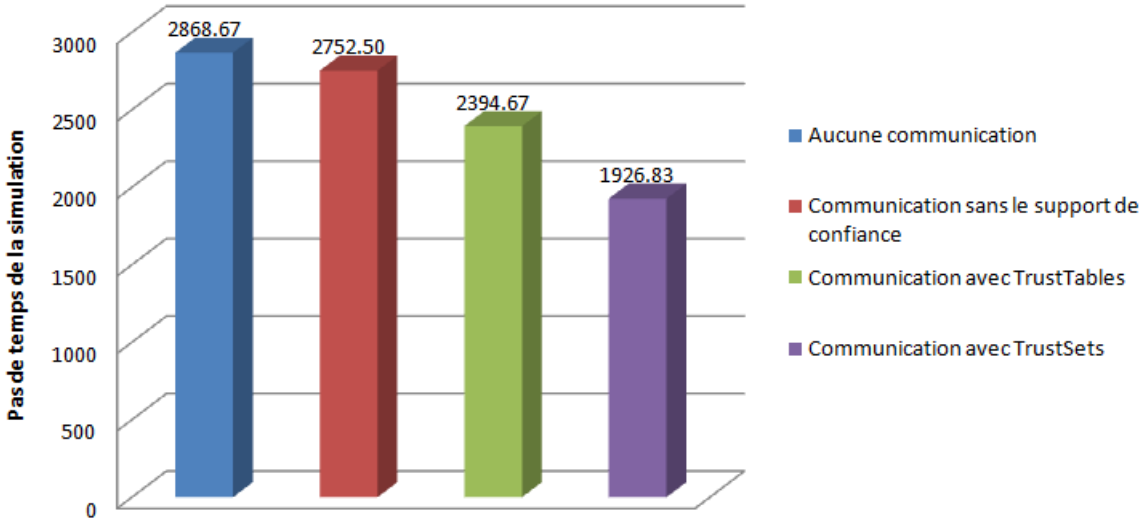


FIGURE 4.15 – Comparaison entre les nombres moyens de pas de temps pour obtenir une carte réelle avec les 4 stratégies de communication

2) il peut recevoir ou non des informations (lorsque la confiance de l'agent B en A est supérieure ou inférieure à Low),

3) il peut examiner ou non les informations reçues (lorsque sa confiance en B est supérieure ou inférieure à Low).

Pour illustrer l'intérêt de cette stratégie, nous avons calculé le nombre de rencontres, le nombre d'échanges d'information, les messages reçus et les messages traités au cours d'une simulation.

Cette stratégie de communication, bien qu'élémentaire, a un impact sur le volume des communications et sur le nombre de pas de temps nécessaires au système pour atteindre l'objectif qui lui a été fixé : obtenir une carte complète et fiable de l'environnement. Dans la figure 4.16, nous comparons le nombre de rencontres qui est égal au nombre de communications dans un SMA sans politique de communications (correspondant au nombre de communications lorsque les agents communiquent sans TrustTables ou TrustSets) et le nombre de messages envoyés dans le cas où chaque agent développe une stratégie de communication envers les agents défectueux (quand chaque agent interrompt la communication avec les agents défectueux), tout ceci sans nuire à la performance du SMA. On remarque que le volume des communications est réduit de moitié (environ 1600 rencontres pour 900 échanges d'information). Comme ce qu'on peut voir sur la figure 4.16, les agents commencent à rejeter les communications des agents défectueux à partir de l'étape de simulation 150.

4.5.5 Limite de l'usage des seuils

En utilisant des seuils, il y a un certain nombre d'inconvénients par exemple liés au fait que le comportement des agents puisse changer brusquement. Par conséquent, un agent peut changer au fil du temps (les agents fiables peut devenir défectueux) mais il ne peut pas faire augmenter la confiance en lui des agents qui le jugent défectueux.

Pour résoudre ce problème, je propose dans le chapitre suivant une nouvelle approche basée sur la tendance. Un agent aura une probabilité de communiquer avec un autre agent proportionnelle à sa confiance en lui. Ainsi même si un agent est jugé non fiables, la probabilité de communiquer avec lui reste non-nulle. Cela permet notamment à un agent défectueux (et jugé non fiable) qui ne deviendra correct de pouvoir faire augmenter la confiance des autres en lui. De plus, cela peut introduire de nouvelles informations même si leur fiabilité est faible.

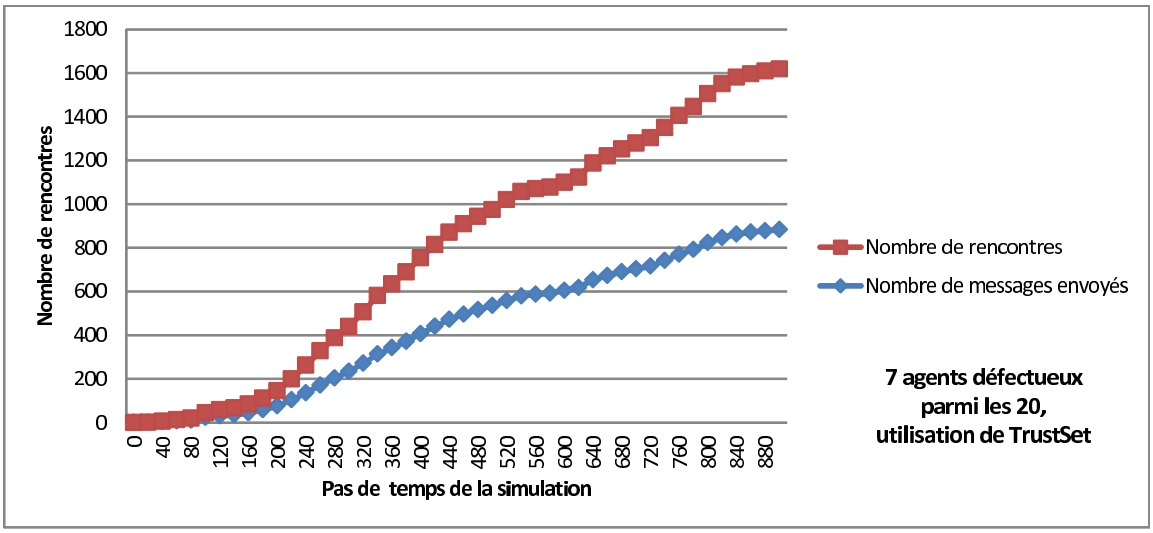


FIGURE 4.16 – Comparaison entre le nombre de messages envoyés et le nombre de rencontres

4.6 Conclusion

Ce chapitre présente un modèle de confiance basé sur le réseau des interactions entre les agents, appelé TrustSet, qui permet de mettre en place des mécanismes pour assurer la cohérence et la robustesse du SMA perturbé où la collecte et la transmission d'information peuvent être altérées par des agents dissonants.

4.6.1 Modèle de confiance TrustSet

Le TrustSet est établi via le réseau des interactions entre les agents. L'avantage de cette approche est que chaque agent peut distinguer les informations directes (collectées dans l'environnement) et les informations indirectes (échangées en communiquant avec les autres agents), pas seulement sur les données stockées, mais également sur les données transmises par les agents qu'il rencontre. Compte tenu de ces informations, plus précises que celles traitées par les modèles traditionnels (TrustNet de Schillo, EigenTrust de Kamvar, modèle de Zia, etc.), les agents peuvent construire et mettre à jour des arbres probabilistes de l'information et des TrustGraphs plus efficaces. J'ai présenté une manière de fusionner les informations directes et indirectes des bases de données stockées par les agents et de calculer les variations de confiance sur les arcs des TrustGraphs en se basant sur la confrontation des informations.

4.6.2 Utilisation du TrustSet

J'ai montré comment chaque agent attribue à la fois une fiabilité aux informations et une confiance aux autres agents afin d'améliorer sa perception du monde. Dans notre modèle les variations de la confiance d'un agent en un autre agent dépendent à la fois de la qualité de la transmission par l'autre agent et de l'importance de la dissonance entre les informations recueillies par les deux agents. Ce processus finalement permet à un agent de renforcer la communication avec les agents fiables et de refuser les communications provenant d'agents jugés non fiables, afin de réduire la dissonance dans le SMA perturbé. De plus, j'ai donné à chaque agent la capacité de procéder à un examen critique de sa situation présente en se basant sur sa réputation. Cela permet à un agent d'éviter de transmettre des informations non fiables. Pour obtenir une meilleure coopération entre les agents, une minimisation des redondances de données et une amélioration de la qualité des informations échangées, on a proposé aussi une approche de la formation de coalitions des agents fiables.

4.6.3 Comparaison avec le modèle de Schillo

La différence principale entre le modèle de Schillo[Schillo et al., 2000] et mon modèle repose sur les données échangées : les agents dans le modèle de Schillo transmettent des données relatives à la confiance d'un agent. Dans mon modèle, je vais encore plus loin : nos agents visent à effectuer une tâche de collecte d'information le plus efficacement possible dans un environnement perturbé. Les agents transmettent des informations sur la confiance des autres agents, mais aussi sur l'environnement (avec la source de ces informations). Ils devraient être en mesure de déterminer si un agent est digne de confiance, si une information est fiable et de choisir entre diverses informations sur le même objet. J'ai présenté des algorithmes pour calculer les confiances à partir des informations reçues, mais aussi des algorithmes pour le calcul de la fiabilité de l'information à partir des valeurs de confiance, c'est à dire basé sur le TrustSet. En outre, dans le modèle de Schillo, les agents communiquent des informations sur les comportements des agents, en revanche, nos agents échangent tout leur TrustGraph. J'ai donc proposé des méthodes pour le calcul des confiances intrinsèques qui sont contenues dans la TrustTable, pour fusionner les TrustGraphs et traiter les incohérences de confiance sur les chemins partagés.

4.6.4 Limite du modèle

La limite du modèle de TrustSet est son utilisation des seuils pour limiter l'influence des agents non fiables et le fait qu'elle ne fournisse pas de moyen efficace pour adapter à la perturbation du système. Je proposerai dans la chapitre 5 une approche auto-organisationnelle basée sur la confiance permettant l'élaboration de systèmes robustes capables de s'adapter aux perturbations. Cette auto-organisation est caractérisée par un couplage dynamique entre les deux composants du système : la collecte d'information directe et la collecte d'information indirecte via la communication. Un des avantages de cette approche est que chaque agent pourra changer son comportement pour s'adapter de la perturbation de l'environnement.

Chapitre 5

Approche auto-organisationnelle basée sur la confiance

Sommaire

Dans ce chapitre, je vais présenter la mise en œuvre d'une démarche d'auto-organisation, basée sur une vision systémique dans laquelle je considère un couplage structurel entre deux composantes du système : une composante associé à la collecte d'information et une autre associé à la communication. Plus précisément, le mécanisme de confiance TrustSet présenté dans le chapitre précédent est utilisé dans des règles locales qui lient les perceptions aux actions et qui régissent les comportements à adopter par les agents et les contraintes de l'environnement qui vont exercer une pression sélective sur les agents pour aboutir à l'émergence de deux organisations qui co-évoluent de façon rétroactive : une au niveau social et l'autre au niveau spatial. L'organisation sociale reflète les relations de confiance établies entre les différents agents. L'organisation spatiale reflète le déploiement des agents dans l'environnement pour favoriser l'exploration des agents. Les règles locales de comportement peuvent être regroupées en les trois catégories : les règles de déplacement, les règles de communication, et les règles de rétroaction entre la communication et le déplacement. Les combinaisons possibles de ces règles de comportement sont ensuite expérimentées dans des simulations pour permettre l'émergence d'organisations et de rôles dans le système.

5.1 Introduction

5.1.1 Description de l'approche

L'auto-organisation est généralement considérée comme la capacité d'un système à passer d'un état désorganisé à un état organisé sans intervention extérieure [Ashby, 1962] ou comme le passage d'une « mauvaise » organisation à une « bonne » organisation. En ce qui concerne les systèmes d'information, l'auto-organisation est la capacité pour le système à fonctionner sans aucune intervention de contrôle explicite de l'extérieur, et l'organisation du système émerge alors à partir des d'interactions et de la coordination entre les éléments du système. Ces actions d'interaction et / ou de coordination sont basées sur des règles locales propres à chaque élément du système. L'auto-organisation vise à limiter l'échange d'informations erronées dans les systèmes perturbés, en particulier dans les systèmes multi-agents perturbés. Je pars du principe que toute structure organisationnelle, qu'elle soit sociale ou spatiale, devra apparaitre par émergence. Elle ne sera codée ni dans l'environnement ni dans les agents. Seul le comportement des agents la fera apparaitre ou non. Il s'agit dans ces travaux de mettre en avant les principes d'une auto-organisation qui nait de et se manifeste par la coévolution de 2 organisations : l'organisation spatiale et l'organisation sociale. Elles s'expriment dans des espaces différents : l'organisation sociale est associée à la fonction de communication dans un espace conceptuel et l'organisation spatiale à la fonction de déplacement dans un espace physique.

La figure 5.1 représente le processus d'auto-organisation d'un système de collecte d'informations distribués S qui est décomposé en 2 systèmes (S_1, S_2) couplés de façon rétroactive et d'un système S_3 contrôlant ce couplage. Le système peut donc s'écrire $S = (S_1, S_2, S_3)$:

- S_1 : Le système de communication représenté par un ensemble de règles de communication R_1 (ce sont des règles régissant la communication et qui sont fonction de la confiance et de la possibilité de communiquer).
- S_2 : Le système de collecte d'informations représenté par un ensemble de règles de déplacement R_2 (ce sont des règles régissant le déplacement et qui sont fonction des positions et de la confiance).
- S_3 : Le système de contrôle qui permet d'agir sur la coévolution du système de communication S_1 et du système de déplacement S_2. Il est représenté par un ensemble de règles contrôlant la rétroaction entre les 2 systèmes.

Un système organisé S' émerge par auto-organisation du système S qui se manifeste par la coévolution de l'organisation spatiale et l'organisation sociale. Ces deux organisations émergent de la dynamique du système, tout en agissant sur cette dynamique qui découle elle-même de l'influence des règles locales et des contraintes de l'environnement.

5.1.2 Notations

Dans le reste de ce chapitre et de ma thèse, j'utiliserai les notations suivantes (qui reprennent et rappellent les notations des chapitres précédents).

- i,j,k,m... : identifiants des agents
- X,Y,Z,... : identifiants des points-cibles visés par les agents pour déterminer leur prochain déplaces.
- $A_{per}(i) = \{i, j, k, m...\}$: ensemble des agents se trouvant dans le zone de perception de l'agent i (depuis le début de la simulation).
- $A_{com}(i) = \{i, j, k, m...\}$: ensemble des agents ayant communiqué avec l'agent i
- $A^t_{com}(i) = \{i, j, k, m...\}$: ensemble des agents ayant communiqué avec l'agent i pendant une étape de simulation t
- $P_{front}(i) = \{X, Y, Z, ...\}$: ensemble des points cibles situés sur la frontière d'exploration de l'agent i
- D_i : ensemble des informations collecté directement par l'agent i
- IND_i : ensemble des informations que l'agent i reçoit d'autres agents
- $D_i(X)$: ensemble des informations directes que l'agent i a collectées sur le point X
- $IND_i(X)$: ensemble des informations indirectes dont l'agent i dispose sur le point X
- $\delta(i, j)$: la distance spatiale entre deux agents i et j
- $\delta(i, X)$: la distance entre la position actuelle de l'agent i et le point cible considéré X

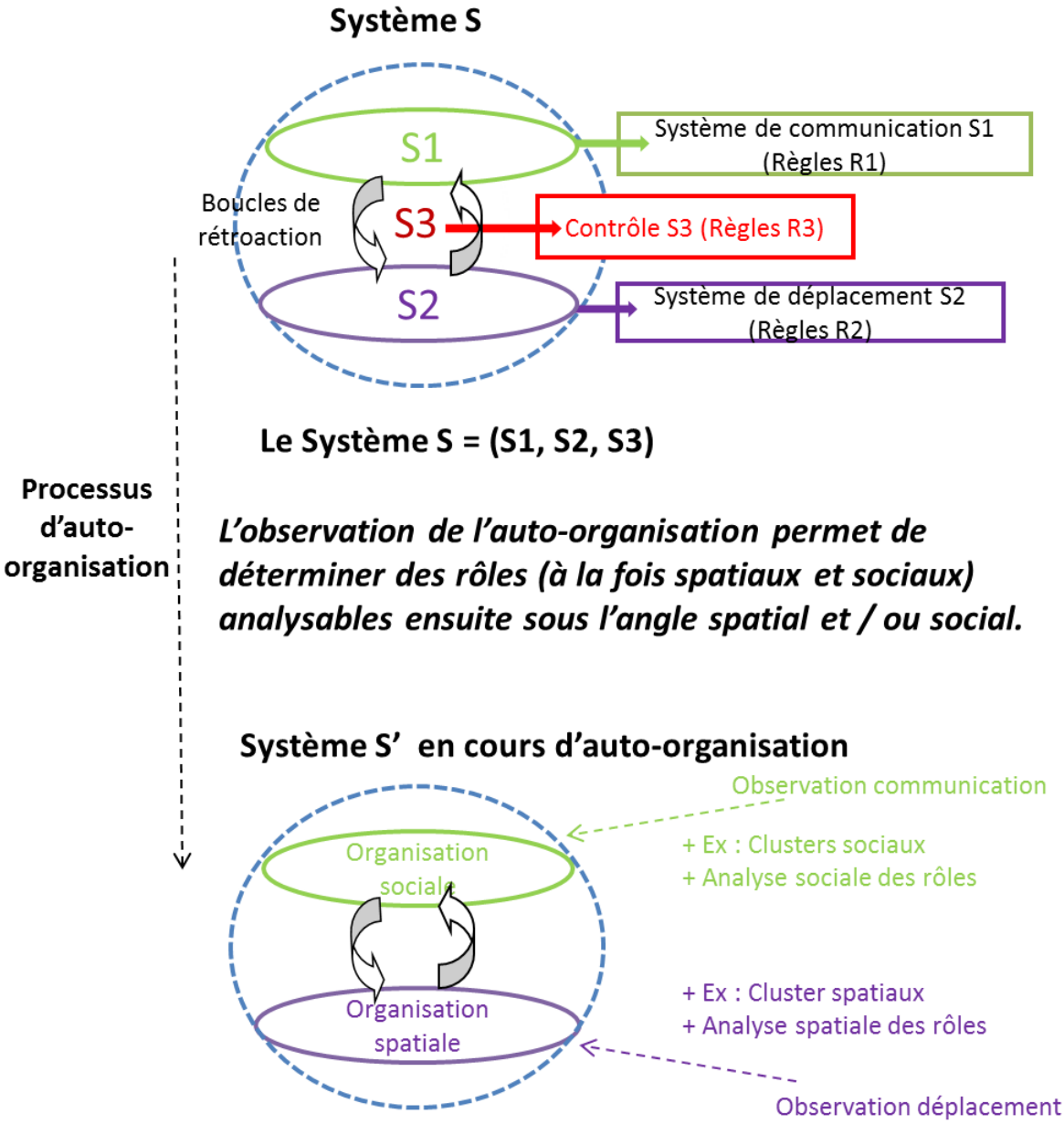


Figure 5.1: Auto-organisation d'un système de collecte d'informations perturbé

- $\delta(X_{curr}, X_{next})$: la distance entre la position actuelle de l'agent i X_{curr} (vers lequel i a commencé à se déplacer) et le point cible considéré X_{next}
- $N(i, X)$: nombre d'informations dont l'agent i dispose sur le point X (le nombre d'agents qui ont communiqué une information sur X à i)
- $f(i, X)$: fiabilité de l'information associée au point X calculée par l'agent i
- $g(i, X)$: coefficient d'attractivité du point cible X pour l'agent i
- $g_{dist}(i, X)$: coefficient d'attractivité du point cible X pour l'agent i basé sur la distance entre i et X
- $g_{inf}(i, X)$: coefficient d'attractivité du point cible X pour l'agent i basé sur le nombre d'informations dont i dispose sur X
- $g_{reli}(i, X)$: coefficient d'attractivité du point cible X pour l'agent i basé sur la fiabilité que l'agent i associe à l'information attachée au point cible X
- $g_{agent}(i, X)$: coefficient d'attractivité du point cible X pour l'agent i basé sur les influences exercées par les agents environnants
- $\overrightarrow{F_{ij}}$: force d'attraction/répulsion exercée par l'agent i sur l'agent j
- $\overrightarrow{F_i}$: force d'attraction/répulsion résultante exercée par l'ensemble des agents perçus par l'agent i
- $\overrightarrow{V_{iX}}$: vecteur reliant l'agent i au point cible X
- $p(i)$: tendance de l'agent i à communiquer
- $p(i, j)$: tendance de l'agent i à communiquer avec l'agent j
- $p_{reli}(i, j)$: tendance de l'agent i à communiquer avec l'agent j basée sur la fiabilité de l'agent j
- $p_{dur}(i, j)$: tendance de l'agent i à communiquer avec l'agent j basée sur la durée de déconnexion entre i et j
- $p_{new}(i, j)$: tendance de l'agent i à communiquer avec l'agent j basée sur les nouvelles informations reçues de l'agent j
- $p(i, j, D_i(X))$: tendance de l'agent i à communiquer à l'agent j les informations directes dont il dispose sur le point X
- $p(i, j, IND_i(X))$: tendance de l'agent i à communiquer à l'agent j les informations indirectes dont il dispose sur le point X

5.2 Système de collecte d'informations

Dans le système de collecte d'informations, le comportement de l'agent est limité à ses choix de déplacement. Afin d'optimiser ses déplacements, et en accord avec les algorithmes de frontière [Yamauchi, 1997] connus qui ont fait leurs preuves dans le domaine de l'exploration [Holz et al., 2010], chaque agent va percevoir un certain nombre de points de la frontière de sa zone connue et sélectionner le meilleur vers lequel s'orienter, compte tenu de ses objectifs. Un exemple d'exploration basé sur la frontière est fourni dans Fig. 5.2. Dans cet exemple, les positions inconnus sont représentées par des petits points, les positions connues sont représentées par un espace blanc. Les positions occupées par un robot sont représentées par des grands points. Les positions sur la frontière entre l'espace inconnu et connu sont colorées. Toutes positions de la même couleur appartient à la même région. La ligne bleue représente la trajectoire du robot (le cercle noir avec une ligne indiquant la direction de robot) à la frontière la plus proche. Le robot tente de suivre cette voie tout en naviguant de manière réactive autour des obstacles inattendus.

Afin d'optimiser sa politique de communication, l'agent doit également tenir compte des agents qu'il perçoit autour de sa position et de leurs caractéristiques. Chaque agent perçu aura donc une influence sur le choix de la cible du déplacement. Pour traduire ces deux contingences, le calcul de la prochaine cible d'un agent, en terme de déplacement, est dirigé par un certain nombre de règles. Les règles permettant à un agent de décider son prochain déplacement sont de trois types : soit génériques, soit associées aux points cibles de sa frontière, soit associées aux agents qui entourent l'agent.

Le nombre de cibles à considérer sur la frontière est entré comme paramètre par l'utilisateur : plus ce nombre est grand, plus la distance entre 2 candidats est petite et plus la simulation est fine. Ce paramètre correspondant à un taux d'échantillonnage de la frontière. A chaque point cible, est associé un coefficient d'attractivité. Ce coefficient sera calculé en utilisant les différentes règles locales présentées ci-dessous.

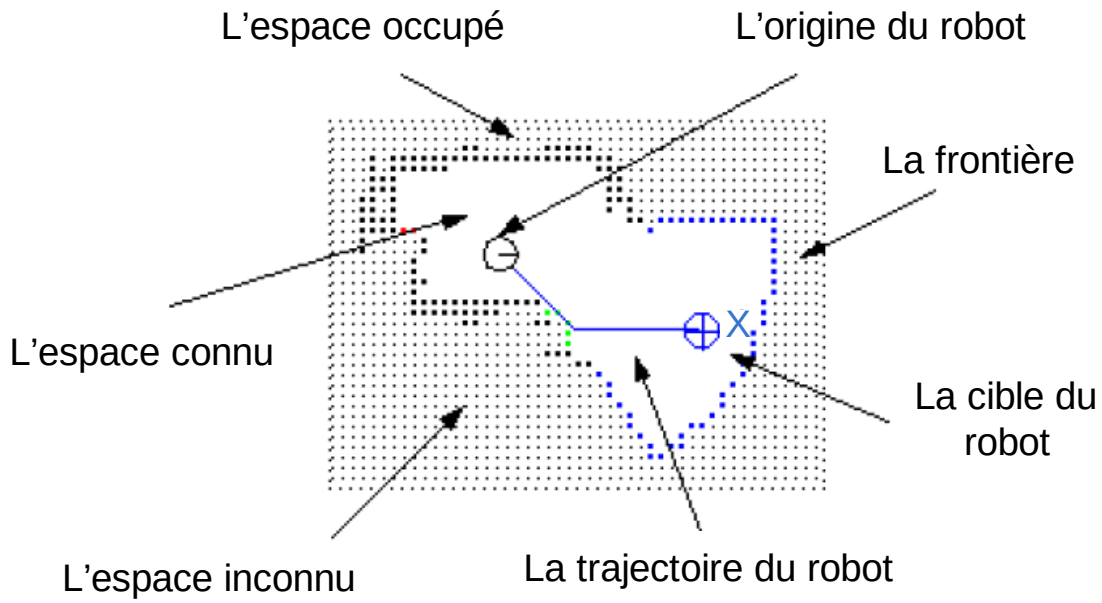


Figure 5.2: Un exemple d'une exploration basé sur la frontière sur la zone connue ([Yamauchi, 1997]).

L'agent choisira pour cible le point dont le coefficient d'attractivité est le plus élevé au moment de son calcul.

5.2.1 Les règles locales

5.2.1.1 Règles génériques :

Règle de déplacement aléatoire (règle par défaut).

- Si l'agent n'a pas de cible, il se déplace au hasard (règle correspondant à l'initialisation de la simulation).
- Si deux (ou plus de deux) cibles ont le même coefficient d'attractivité, l'agent en choisit une au hasard.

5.2.1.2 Règles associées au terrain :

Règle de déplacement vers des points cibles. L'agent a tendance à se déplacer vers le point cible le plus proche possible. L'agent calcule la distance à parcourir $\delta(i, X)$ entre sa position actuelle et le point cible considéré X. Plus la distance est petite, plus l'agent aura tendance à choisir cette position comme cible pour se déplacer.

$$g_{dist}(i, X) = \frac{1}{\sqrt[n]{\delta(i, X)}} \tag{5.1}$$

où n est une constante supérieure à 1 : plus n est grand, plus g_{dist} suit une fonction linéaire (Fig. 5.3 présente des courbes avec 3 n différents.).[1]

Règle de déplacement vers des zones inconnues. Le nombre d'informations $N(i, X)$ d'une position sur la frontière est le nombre d'informations dont l'agent i dispose sur la cible X. Celui-ci est calculé par

1. Une valeur trop élevée de n ralentie le temps d'exécution de la simulation, $n = 2$ est suffisant pour l'application Danger Mapping.

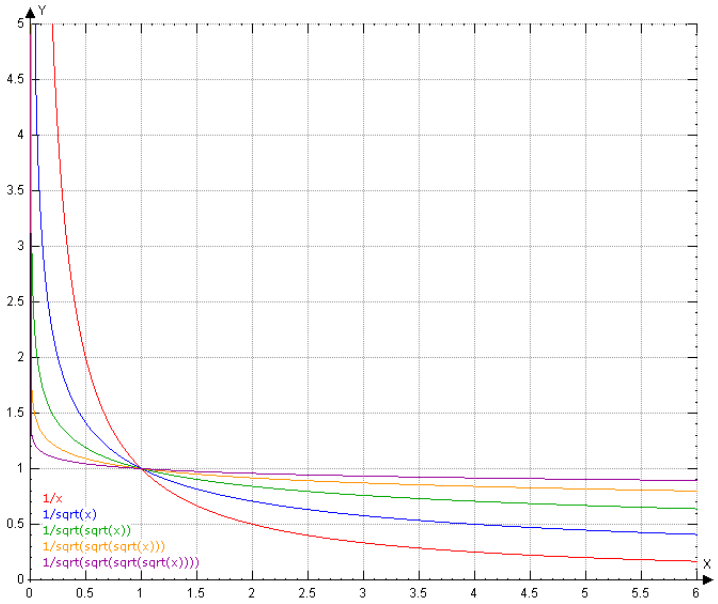


Figure 5.3: Exemple des courbes des fonctions $f(x) = \frac{1}{x}$, $f(x) = \frac{1}{\sqrt{x}}$, $f(x) = \frac{1}{\sqrt[4]{x}}$, $f(x) = \frac{1}{\sqrt[6]{x}}$, $f(x) = \frac{1}{\sqrt[8]{x}}$.

le nombre d'agents qui ont communiqué une information X à i. Moins l'agent a d'informations sur une cible, plus il la convoite. Cela permet d'éviter la concentration d'agents dans les mêmes lieux.

$$g_{inf}(i, X) = \frac{1}{\sqrt[n]{N(i, X) + 1}} \tag{5.2}$$

Règle de déplacement selon la fiabilité de l'information. La fiabilité $f(i, X)$ est calculée pour à chaque position X de la frontière. Moins l'information que l'agent a sur une cible est fiable, plus l'agent la convoite. Cela permet aux agents de vérifier les zones les moins fiables d'abord afin d'améliorer la cohérence de leur représentation de l'environnement.

$$g_{reli}(i, X) = 1 - f(i, X) \tag{5.3}$$

5.2.1.3 Règle associée aux agents

Règle de déplacement vers des agents fiables. Plus un agent est fiable, plus les autres agents se rapprochent de lui. De cette manière, les agents ont tendance à être attirés par des agents fiables et à être repoussés par des agents peu fiables. Cela permet aux agents de recevoir plus d'informations fiables.

Je note $g_{agent}(i, X)$ une valeur comprise entre 0 et 1 qu'on associe à chaque cible en fonction des forces d'attraction et de répulsion exercées par l'ensemble des agents perçus par i et noté $A_{per}(i)$. En effet, la règle ci-dessus détermine un ensemble de forces dont l'agent doit tenir compte pour calculer son futur déplacement. La valeur de $g_{agent}(i, X)$ est calculée en 2 étapes :

Étape 1 : Calcul de la résultante des forces d'attraction ou de répulsion exercées par les agents perçus. Pour chaque agent perçu, on détermine la force d'attraction ou de répulsion exercée grâce aux « règles de déplacement vers des agents fiables ». On calcule ensuite la résultante de toutes ces forces. Considérons 2 agents i et j.

La force d'attraction ou de répulsion exercée par j sur i (celle exercée par i sur j peut être différente) est déterminée par le vecteur $\overrightarrow{F_{ij}}$:

Figure 5.4: La force d'attraction ou de répulsion de j sur i

$$\overrightarrow{F_{ij}} = \frac{(T_{ij} - 0.5)}{0.5} . \frac{\overrightarrow{V_{ij}}}{\delta(i,j)} \tag{5.4}$$

- Si $T_{ij} > 0.5$ alors $\overrightarrow{F_{ij}}$ est une force d'attraction. Par exemple, si $T_{ij} = 1$ alors $\overrightarrow{F_{ij}} = \frac{\overrightarrow{V_{ij}}}{\delta(i,j)}$ et si $T_{ij} = 0.75$ alors $\overrightarrow{F_{ij}} = \frac{1}{2} . \frac{\overrightarrow{V_{ij}}}{\delta(i,j)}$
- Si $T_{ij} < 0.5$ alors $\overrightarrow{F_{ij}}$ est une force de répulsion. Par exemple, si $T_{ij} = 0.25$ alors $\overrightarrow{F_{ij}} = -\frac{1}{2} . \frac{\overrightarrow{V_{ij}}}{\delta(i,j)}$
- Si $T_{ij} = 0.5$ alors $\overrightarrow{F_{ij}} = 0$

Les coordonnées de $\overrightarrow{F_{ij}}$ seront donc :

$$\begin{cases} x_{\overrightarrow{F_{ij}}} = \frac{(T_{ij} - 0.5)}{0.5 * \delta(i,j)} . (x_j - x_i) \\ y_{\overrightarrow{F_{ij}}} = \frac{(T_{ij} - 0.5)}{0.5 * \delta(i,j)} . (y_j - y_i) \end{cases} \tag{5.5}$$

Lorsque l'agent i perçoit plusieurs agents, la force totale $\overrightarrow{F_i}$ qu'exerceront tous les agents perçus sur i sera la résultante des forces de chaque agent sur i :

$$\overrightarrow{F_i} = \sum_{k \in A_{per}(i)} \overrightarrow{F_{ik}} \tag{5.6}$$

Les coordonnées du vecteur $\overrightarrow{F_i}$ seront calculées comme suit :

$$\begin{cases} x_{\overrightarrow{F_i}} = \sum_{k \in A_{per}(i)} \frac{(T_{ik} - 0.5)}{0.5 * \delta(i,j)} . (x_k - x_i) \\ y_{\overrightarrow{F_i}} = \sum_{k \in A_{per}(i)} \frac{(T_{ik} - 0.5)}{0.5 * \delta(i,j)} . (y_k - y_i) \end{cases} \tag{5.7}$$

Étape 2 : Calcul de l'influence de la force résultante $\overrightarrow{F_i}$ sur toutes les cibles. On suppose que $\overrightarrow{F_i}$ est la force finale obtenue. Calculons à présent l'impact de cette force, c'est-à-dire le coefficient d'attractivité lié aux agents, noté $g_{agent}(i, X)$, sur tous les points-cibles de la frontière.

On considère que plus la valeur absolue de l'angle (noté α sur la Fig. 5.5) entre le vecteur $\overrightarrow{F_i}$ et le vecteur $\overrightarrow{V_{iX}}$ (ramenée à 180°) sera grande, moins l'impact de la force résultante exercée par les agents voisins sera important. Le coefficient d'attractivité est donc équivalent au produit scalaire entre $\overrightarrow{F_i}$ et le vecteur $\frac{\overrightarrow{V_{iX}}}{\left|\overrightarrow{V_{iX}}\right|}$. On normalise le vecteur $\overrightarrow{V_{iX}}$ afin que la distance entre l'agent i est le point cible X ne soit pas pris en compte dans le calcul de $g_{agent}(i, X)$. Il intervient en effet déjà dans $g_{dist}(i, X)$.

$$g_{agent}(i, X)) = \frac{\overrightarrow{F_i} . \overrightarrow{V_{iX}}}{\left|\overrightarrow{V_{iX}}\right|} = \frac{x_{\overrightarrow{F_i}} . x_{\overrightarrow{V_{iX}}} + x_{\overrightarrow{V_{iX}}} . x_{\overrightarrow{F_i}}}{\sqrt{x^2_{\overrightarrow{V_{iX}}} + y^2_{\overrightarrow{V_{iX}}}}} \tag{5.8}$$

avec

$$x_{\overrightarrow{V_{iX}}} = x_X - x_i$$

$$y_{\overrightarrow{V_{iX}}} = y_X - y_i$$

$$x_{\overrightarrow{F_i}} = \sum_{k \in A_{per}(i)} \frac{(T_{ik} - 0.5)}{0.5 * \delta(i,j)} . (x_k - x_i)$$

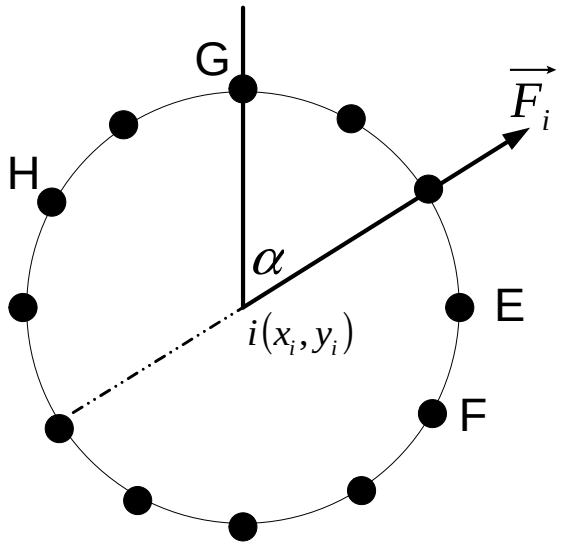


Figure 5.5: L'influence de la force résultante sur toutes les cibles H,G,E,F...

$$y_{\overrightarrow{F_i}} = \sum_{k \in A_{per}(i)} \frac{(T_{ik} - 0.5)}{0.5 * \delta(i,j)} . (y_k - y_i)$$

Le produit scalaire $\overrightarrow{F_i}.\overrightarrow{V_{iX}}$ tient compte à la fois de l'angle entre les deux vecteurs (plus l'angle est petit entre les deux, plus $\overrightarrow{F_i}$ favorisera cette position X), de la norme de $\left|\overrightarrow{F_i}\right|$ (intensité de la force) et de la direction de $\overrightarrow{F_i}$ (distinction entre force d'attraction et de répulsion).

5.2.2 Calcul du coefficient d'attractivité total

Pour calculer le coefficient d'attractivité résultant $g(i,X)$ pour chaque point-cible X, l'agent i utilise une somme pondérée par les poids w_i des différents coefficients d'attractivité normalisés. Plus le poids est élevé, plus l'agent privilégie la règle associé. A chaque position cible de la frontière X sera associée le coefficient d'attractivité $g(i,X)$:

$$g(i,X) = w1.\frac{g_{dist}(i,X)}{max_{Z \in P_{front}(i)}(g_{dist}(i,Z))} + w2.\frac{g_{inf}(i,X)}{max_{Z \in P_{front}(i)}(g_{inf}(i,Z))}$$

$$+ w3.\frac{g_{reli}(i,X)}{max_{Z \in P_{front}(i)}(g_{reli}(i,Z))} + w4.\frac{g_{agent}(i,X)}{max_{Z \in P_{front}(i)}(g_{agent}(i,Z))} \quad (5.9)$$

avec $w_1 + w_2 + w_3 + w_4 = 1$

La prochaine cible du déplacement de l'agent est la cible à laquelle est affecté le coefficient d'attractivité le plus grand :

$$position_{t+1}(i) = M \ : \ g(i,M) = max_{Z \in P_{front}(i)}(g(i,Z)) \quad (5.10)$$

Le choix des coefficients w_i peut être fait manuellement afin de contrôler l'évolution de la simulation et privilégier l'apparition de certains rôles dans l'auto-organisation. Ces coefficients pourraient également être déterminés par l'agent lui-même.

Interprétation. Étudions maintenant l'impact du choix des poids w dans la stratégie de déplacement des agents :

- Plus w_1 est choisi grand, plus l'agent privilégie le rapprochement avec les cibles les plus proches de sa position sur sa frontière. Cela correspond à la stratégie élémentaire utilisée dans les résultats du chapitre 4.
- Plus w_2 est choisi grand, plus l'agent a tendance à se diriger vers les zones les moins explorées et donc à enrichir plus rapidement sa base de données. Cela devrait également permettre d'éviter la concentration d'agents dans des zones contigües.
- Plus w_3 est grand, plus l'agent veut conforter ses connaissances du milieu avant d'en découvrir de nouvelles. L'une des conséquences devrait être un travail plus en profondeur sur la fiabilité de l'information (plus la fiabilité de l'information d'une zone est faible, plus l'agent souhaite affiner ses connaissances la concernant).
- Si w_4 est grand, l'agent serait plutôt enclin à se déplacer en vue de communiquer qu'à se déplacer en vue de découvrir de nouvelles données. En effet, il aura tendance à favoriser pour cibles les agents plutôt que les points de sa frontière. Cela devrait lui permettre de rester près des agents qu'il juge fiables en vue de favoriser une communication plus riche et plus sûre.
- Si le triplet(w_1, w_2, w_3) est favorisé par rapport à w_4, l'attraction de la frontière sera plus forte que celle exercée par les agents proches. Autrement dit, la stratégie d'exploration primera sur les échanges avec les autres agents.

Je retrouve ici, au travers de ces paramètres, les possibilités de conflit pour l'agent entre les deux modes de captation de données dont il dispose : l'exploration et la communication. Ceci de manière indirecte, et après interprétation, puisque la communication n'a jamais été considérée dans les règles ci-dessus.

5.2.3 Algorithme proposé

L'Algorithme 8 décrit d'une manière simplifiée une stratégie de déplacement pour un agent i. L'ordre dans lequel les actions des agents sont exécutées est le suivant : l'agent collecte les données (ligne 1), l'agent vérifie si sa position est égale à la position cible (ligne 2). S'il n'est pas encore arrivé, il continue à se déplacer vers cette position. Sinon, l'agent doit calculer une position vers laquelle se déplacer à partir de l'ensemble des points cibles situés sur la frontière de sa zone explorée. Dans ce cas, l'agent calcule le coefficient d'attractivité pour chaque position considérée de la frontière (lignes 3-8), y compris le coefficient d'attractivité associé à la distance (ligne 4), à la quantité d'informations (ligne 5), à la fiabilité de la position (ligne 6), et à d'autres agents détectés (ligne 10). La position ayant le plus haut coefficient d'attractivité est alors choisie comme la prochaine position sur la frontière vers laquelle se déplacer (ligne 12).

5.2.4 Exemple

Je vais illustrer le calcul du coefficient lié aux agents sur une situation simple. Considérons la situation de la Figure 5.6 où l'agent i doit déterminer une cible parmi les points de la frontière du territoire déjà connu. Sa zone de perception est représentée par le cercle pointillé dans lequel il détecte les agents j, k, l et m. L'agent i est supposé capable de communiquer avec les agents se trouvant dans sa zone de perception.

On considère dans cette exemple que :

- les valeurs de confiance que l'agent i associe aux agents qu'il perçoit sont les suivantes : $T_{ij} = 0.8$, $T_{ik} = 0.9$, $T_{il} = 1$, $T_{im} = 0.1$.
- le nombre d'informations et la fiabilité que l'agent i associe aux points cibles qu'il perçoit soient comme suit (on n'examine que ici les 4 points cibles Q, O, P, R parmi les 24 points cibles candidats dans cet exemple) :
  - $N(i,Q) = 3$; $N(i,O) = 5$; $N(i,P) = 1$; $N(i,R) = 0$
  - $f(i,Q) = 0.4$; $f(i,O) = 0.5$; $f(i,P) = 0.6$; $f(i,R) = 0.1$

On considère que les coordonnées de l'ensemble des agents se trouvant dans la zone de perception de l'agent i sont comme suit : $i(5,5)$; $j(6,4)$; $k(4,7)$; $l(3,4)$; $m(6,6)$

Algorithme 8: Algorithme gérant le déplacement et l'activité de collecte des agents

Entrées : X_{curr} la position actuelle de l'agent, $P_{front}(i)$ l'ensemble des points cibles situés sur la frontière de la zone explorée de l'agent i, X_{next} est la position cible de l'agent i

Sorties : X'_{next} la prochaine position cible de l'agent

i collecte des données autour de sa position actuelle X_{curr}
si X_{curr} *égal à la prochaine position* X_{next} *de l'agent* i **alors**
  pour tous les *points cibles* X *situés sur la frontière de la zone explorée de l'agent* i **faire**
    $g_{dist}(i,X) = \frac{1}{\sqrt[n]{\delta(i,X)}}$
    $g_{inf}(i,X) = \frac{1}{\sqrt[n]{N(i,X)+1}}$
    $g_{reli}(i,X) = 1 - f(i,X)$
    $g_{agent}(i,X) \leftarrow \frac{\vec{F_i}.\vec{V_{iX}}}{|\vec{V_{iX}}|}$
  fin
  pour tous les *point cible* X *située sur la frontière d'exploration de l'agent* i **faire**
    $g(i,X) = w_1 . \frac{g_{dist}(i,X)}{max_{z \in P_{front}(i)}(g_{dist}(i,Z))} + w_2 . \frac{g_{inf}(i,X)}{max_{z_{front}(i)}(g_{inf}(i,Z))} + w_3 . \frac{g_{reli}(i,X)}{max_{z_{front}(i)}(g_{reli}(i,Z))} + w_4 . \frac{g_{agent}(i,X)}{max_{z \in P_{front}(i)}(g_{agent}(i,Z))}$
  fin
  $X'_{next} \leftarrow M : g(i,M) = max_{Z \in P_{front}(i)}(g(i,Z))$
fin

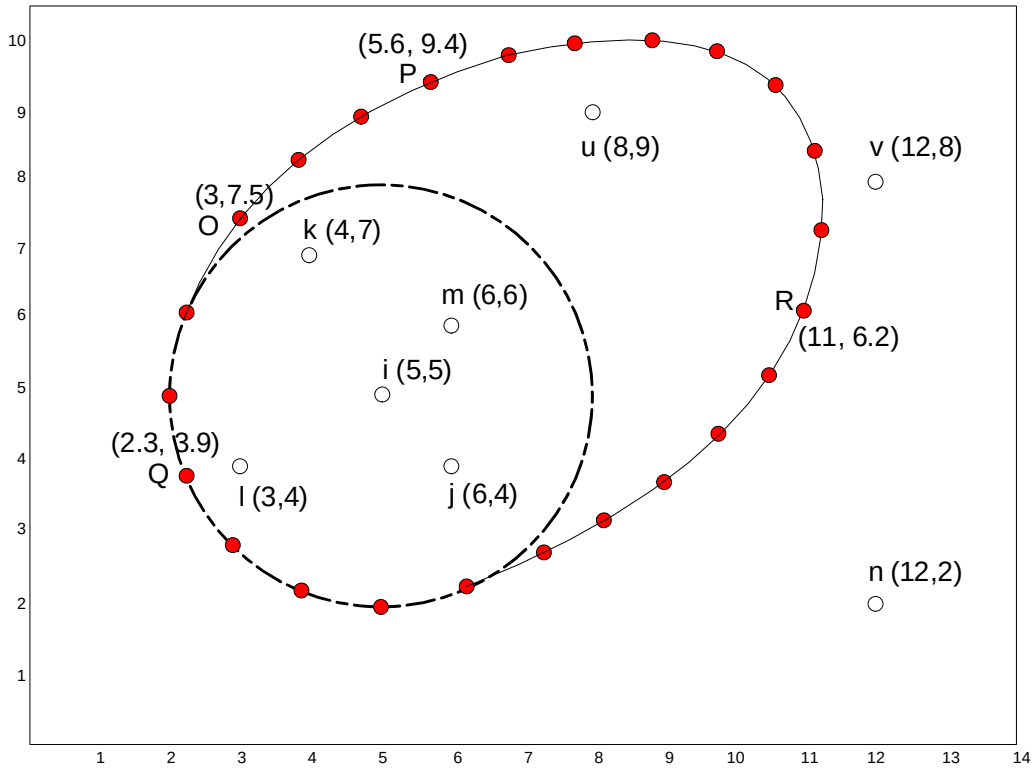


Figure 5.6: Exemple de situation

5.2.4.1 Calcul du coefficient d'attractivité influencé par des règles propres au terrain

Calculons tout d'abord le coefficient d'attractivité qui prend en compte l'influence des règles du terrain, ($g_{dist}(i,X)$,$g_{inf}(i,X)$,$g_{reli}(i,X)$) pour les 4 point cibles Q, O, P et R.

Calcul de $g_{dist}(i,X)$ pour tout X de $\{Q,O,P,R\}$ L'agent i calcule la distance à parcourir $\delta(i,X)$ entre la position actuelle et le point cible considéré. La tendance à se déplacer vers le point cible le plus proche est calculée comme suit :

$$g_{dist}(i,Q) = \frac{1}{\sqrt{\delta(i,Q)}} = 0.58$$

$$g_{dist}(i,O) = \frac{1}{\sqrt{\delta(i,O)}} = 0.55$$

$$g_{dist}(i,P) = \frac{1}{\sqrt{\delta(i,P)}} = 0.47$$

$$g_{dist}(i,R) = \frac{1}{\sqrt{\delta(i,R)}} = 0.4$$

Calcul de $g_{inf}(i,X)$ pour tout X de $\{Q,O,P,R\}$ L'agent i calcule le nombre d'informations $N(i,X)$ dont il dispose sur la cible X. La tendance à se déplacer vers le point cible inconnu est calculée comme suit :

$$g_{inf}(i,Q) = \frac{1}{\sqrt{N(i,Q)+1}} = 0.5$$

$$g_{inf}(i,O) = \frac{1}{\sqrt{N(i,O)+1}} = 0.4$$

$$g_{inf}(i,P) = \frac{1}{\sqrt{N(i,P)+1}} = 0.7$$

$$g_{inf}(i,R) = \frac{1}{\sqrt{N(i,R)+1}} = 1$$

Calcul de $g_{reli}(i,X)$ pour tout X de $\{Q,O,P,R\}$ L'agent i calcule la fiabilité $f(i,X)$ pour à chaque position X de la frontière. La tendance à se déplacer vers le point cible selon la fiabilité de l'information est calculé comme suit :

$$g_{reli}(i,Q) = 1 - f(i,Q) = 0.6$$

$$g_{reli}(i,O) = 1 - f(i,O) = 0.5$$

$$g_{reli}(i,P) = 1 - f(i,P) = 0.4$$

$$g_{reli}(i,R) = 1 - f(i,R) = 0.9$$

5.2.4.2 Calcul du coefficient d'attractivité influencé par des agents.

Calculons ensuite le coefficient d'attractivité qui prend en compte l'influence des autres agents, $g_{agent}(i,X)$ pour chaque point X de la frontière.

Étape 1. Déterminer les forces d'attraction ou de répulsion que les agents exercent sur l'agent i et calculer la résultante de ces forces

$$\overrightarrow{V_{ij}} = (1, -1)$$
$$\overrightarrow{V_{ik}} = (-1, 2)$$
$$\overrightarrow{V_{il}} = (-2, -1)$$
$$\overrightarrow{V_{im}} = (1, 1)$$

On peut donc calculer les forces liées à chaque agent :

$$\overrightarrow{F_{ij}} = \frac{(0.8 - 0.5)}{0.5} . \frac{\overrightarrow{V_{ij}}}{\sqrt{2}} = 0.42 * \overrightarrow{V_{ij}} = (0.42, -0.42)$$

$$\overrightarrow{F_{ik}} = \frac{(0.9 - 0.5)}{0.5} . \frac{\overrightarrow{V_{ik}}}{\sqrt{5}} = 0.35 * \overrightarrow{V_{ik}} = (-0.35, 0.7)$$

$$\overrightarrow{F_{il}} = \frac{(1 - 0.5)}{0.5} . \frac{\overrightarrow{V_{il}}}{\sqrt{5}} = 0.44 * \overrightarrow{V_{il}} = (-0.88, -0.44)$$

$$\overrightarrow{F_{im}} = \frac{(0.1 - 0.5)}{0.5} . \frac{\overrightarrow{V_{im}}}{\sqrt{2}} = -0.56 * \overrightarrow{V_{im}} = (-0.56, -0.56)$$

Ces forces sont illustrées sur la Figure 5.7. La force résultante finale $\overrightarrow{F_i}$ est obtenue par la somme de toutes les forces $\overrightarrow{F_{ij}}$,$\overrightarrow{F_{ik}}$,$\overrightarrow{F_{il}}$,$\overrightarrow{F_{im}}$:
$\overrightarrow{F_i} = \overrightarrow{F_{ij}} + \overrightarrow{F_{ik}} + \overrightarrow{F_{il}} + \overrightarrow{F_{im}} = (-1.37, -0.72)$

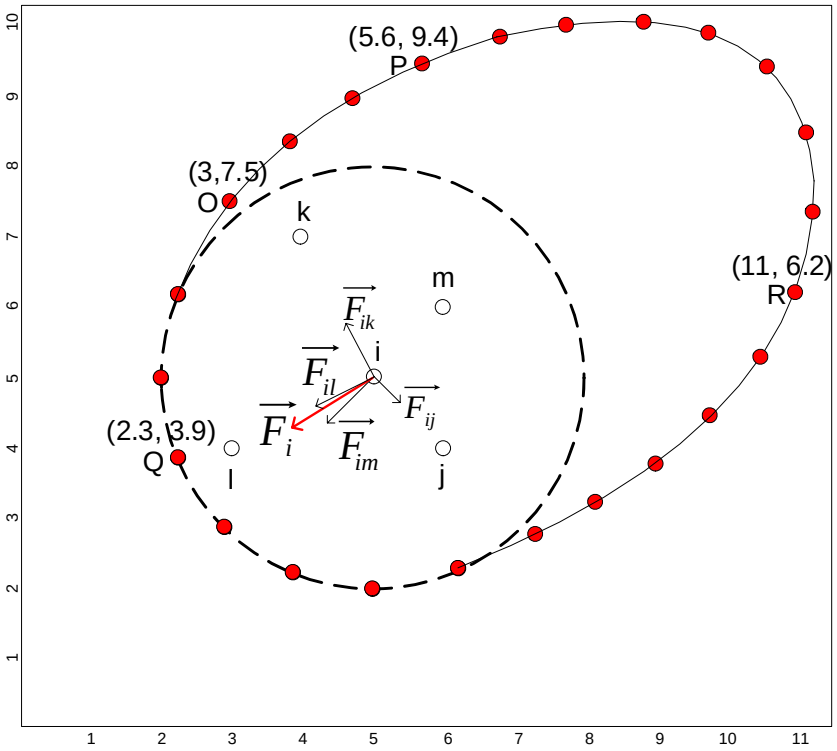


Figure 5.7: Illustration des forces d'attraction et de répulsion que les agents exercent sur l'agent i et de la résultante de ces forces

Étape 2. Calculer l'influence de la force $\overrightarrow{F_i}$ sur toutes les cibles de la frontière On doit déterminer le produit scalaire entre la force finale $\overrightarrow{F_i}$ et les vecteurs $\overrightarrow{V_{iX}}$, X désignant de manière générique une cible de la frontière.

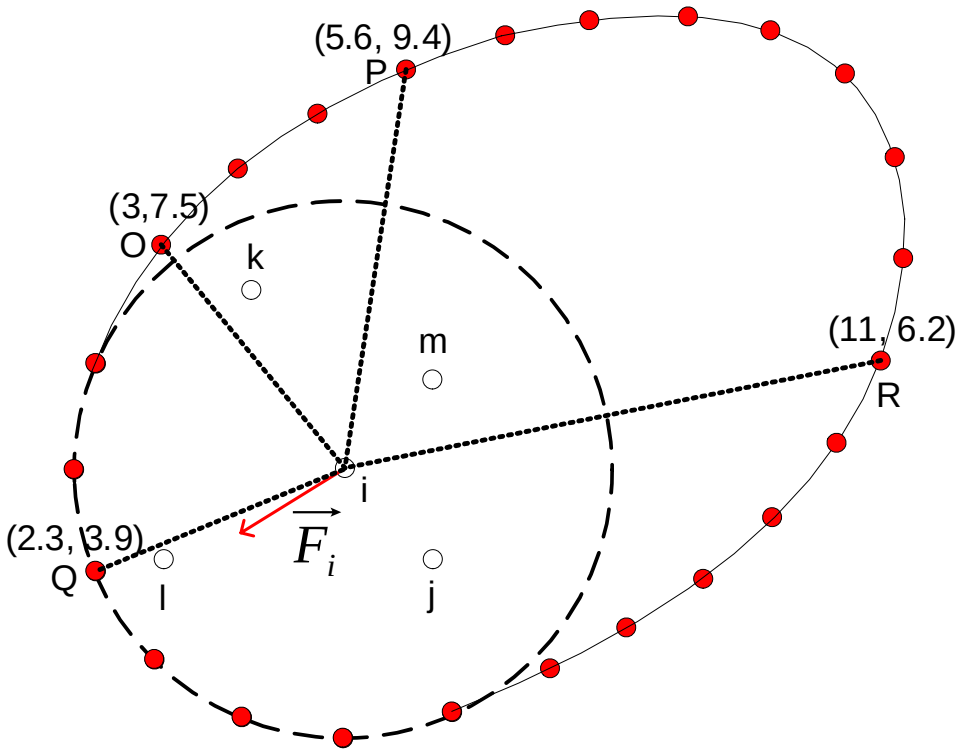


Figure 5.8: Influence de la force résultante $\overrightarrow{F_i}$ sur les positions cibles de la frontière

Les quatre point-cibles O, P, Q, R de la frontière (Figure 5.8) ont pour coordonnées : $O(3, 7.5)$; $Q(2.3, 3.9)$; $P(5.6, 9.4)$; $R(11, 6.2)$.

$$\overrightarrow{V_{iO}} = (x_0 - x_i, y_0 - y_i) = (3 - 5, 7.5 - 5) = (-2, 2.5)$$

$$\overrightarrow{V_{iQ}} = (x_Q - x_i, y_Q - y_i) = (2.3 - 5, 1.3.9 - 5) = (-2.7, -1.1)$$

$$\overrightarrow{V_{iP}} = (x_P - x_i, y_P - y_i) = (5.6 - 5, 7.9.4 - 5) = (0.6, 4.4)$$

$$\overrightarrow{V_{iR}} = (x_R - x_i, y_R - y_i) = (11 - 5, 1.6.2 - 5) = (5.9, 1.2)$$

On peut donc en calculer les coefficients d'attractivité comme suit :

$$g_{agent}(i, O) = \frac{x_{\overrightarrow{F_i}}.x_{\overrightarrow{V_{iO}}} + x_{\overrightarrow{V_{iO}}}.x_{\overrightarrow{F_i}}}{\sqrt{x^2_{\overrightarrow{V_{iO}}} + y^2_{\overrightarrow{V_{iO}}}}} = \frac{(-1.37) * (-2) + 2.5 * (-0.72)}{\sqrt{4 + 6.25}} = 0.29$$

$$g_{agent}(i, Q) = \frac{x_{\overrightarrow{F_i}}.x_{\overrightarrow{V_{iQ}}} + x_{\overrightarrow{V_{iQ}}}.x_{\overrightarrow{F_i}}}{\sqrt{x^2_{\overrightarrow{V_{iQ}}} + y^2_{\overrightarrow{V_{iQ}}}}} = \frac{(-1.37) * (-2.7) + (-1.1) * (-0.72)}{\sqrt{7.29 + 1.21}} = 1.54$$

$$g_{agent}(i,P) = \frac{x_{\overrightarrow{F_i}}.x_{\overrightarrow{V_{iP}}} + x_{\overrightarrow{V_{iP}}}.x_{\overrightarrow{F_i}}}{\sqrt{x^2_{\overrightarrow{V_{iP}}} + y^2_{\overrightarrow{V_{iP}}}}} = \frac{(-1.37) * 0.6 + 4.4 * (-0.72)}{\sqrt{0.36 + 19.36}} = -0.5$$

$$g_{agent}(i,R) = \frac{x_{\overrightarrow{F_i}}.x_{\overrightarrow{V_{iR}}} + x_{\overrightarrow{V_{iR}}}.x_{\overrightarrow{F_i}}}{\sqrt{x^2_{\overrightarrow{V_{iR}}} + y^2_{\overrightarrow{V_{iR}}}}} = \frac{(-1.37) * 11 + 6.2 * (-0.72)}{\sqrt{121 + 38.44}} = -1.55$$

On constate que $g_{agent}(i,Q) > g_{agent}(i,O) > g_{agent}(i,P) > g_{agent}(i,R)$ parce que l'angle $(\overrightarrow{F_i},\overrightarrow{iQ})$ est inférieur à l'angle $(\overrightarrow{F_i},\overrightarrow{iO})$, l'angle $(\overrightarrow{F_i},\overrightarrow{iO})$ est inférieur à l'angle $(\overrightarrow{F_i},\overrightarrow{iP})$ et l'angle $(\overrightarrow{F_i},\overrightarrow{iP})$ est inférieur à l'angle $(\overrightarrow{F_i},\overrightarrow{iR})$.

Les agents entourant l'agent i l'incitent à aller plutôt vers la position Q que vers les autres positions, ou même que vers toute autre position de la frontière, mais le choix de la position dépendra également de tous les autres coefficients.

5.2.4.3 Calcul du coefficient d'attractivité total

Calculons finalement le coefficient d'attractivité total qui prend en compte tous les coefficients d'attractivités influencés par des règles du terrain et par les agents comme suit :

$$g\,(i,Q) = 0.25.\frac{0.58}{0.58} + 0.25.\frac{0.5}{1} + 0.25.\frac{0.6}{0.9} + 0.25.\frac{1.54}{1.54} = 0.785$$

$$g\,(i,O) = 0.25.\frac{0.55}{0.58} + 0.25.\frac{0.4}{1} + 0.25.\frac{0.5}{0.9} + 0.25.\frac{0.29}{1.54} = 0.52$$

$$g\,(i,P) = 0.25.\frac{0.47}{0.58} + 0.25.\frac{0.7}{1} + 0.25.\frac{0.4}{0.9} + 0.25.\frac{(-0.5)}{1.54} = 0.4$$

$$g\,(i,R) = 0.25.\frac{0.4}{0.58} + 0.25.\frac{1}{1} + 0.25.\frac{0.9}{0.9} + 0.25.\frac{(-.1.55)}{1.54} = 0.42$$

avec $w_1 = w_2 = w_3 = w_4 = 0.25$

La prochaine cible du déplacement de l'agent est la cible à laquelle est affecté le coefficient d'attractivité le plus grand :

$$position_{t+1}\,(i) = Q$$

5.3 Système de communication

Après le sous-système régissant le déplacement et la collecte des agents, je m'intéresse maintenant au deuxième sous-système de notre problème : le sous système lié à la communication.

Dans ce sous-système les seules décisions que peut prendre l'agent concernent le choix de communiquer , avec qui communiquer et quoi communiquer. Je prends l'hypothèse que la tendance de communication d'un agent i envers un agent j peut se décomposer en 3 niveaux hiérarchiques que l'on peut organiser comme il suit :

1 Communication globale : i communique-t-il ?

2 Communication ciblée : si i communique, alors avec quel agent communique-t-il ?

3 Communication sériée : Si i communique avec un agent j, quelles informations communique-t-il ?

Chacun de ces niveaux est influencé par une tendance propre, les trois tendances étant régies hiérarchiquement. Elles sont associées aux trois questions fondamentales associées à toute communication : Dois-je communiquer ou non ? Avec qui puis-je communiquer ? Que dois-je communiquer ? L'agent commence donc à calculer sa tendance à communiquer. Supposons que cette tendance soit de 90%, il aura donc 9 chances sur 10 de choisir de communiquer (ou dans cette situation, 9 fois sur 10 il choisira de communiquer). S'il choisit de communiquer, il s'intéressera ensuite de la même manière à savoir avec qui communiquer.

Utiliser la tendance à communiquer permet une évolution lisse du choix de communiquer. L'utilisation la tendance dans la stratégie de communication nous apporte plusieurs avantages : les agents peuvent garder en mémoire certaines informations, même si leur confiance dans l'information est très faible. Cela a également l'avantage de permettre à des agents de pouvoir communiquer à nouveau avec d'autres agents même si leur confiance est devenue très faible et cela leur donne l'occasion de retrouver une valeur acceptable de confiance dès que possible. En effet avec la stratégie utilisant des seuils, une fois qu'un agent n'a plus confiance en un autre agent, il ne communique plus avec lui.

Je présente par la suite les contraintes de l'environnement et les règles locales permettant à l'agent de calculer les trois tendances que l'on vient de présenter.

5.3.1 Les contraintes de l'environnement

Chaque agent ne communique qu'avec des agents qu'il perçoit autour de sa position.

5.3.2 Les règles locales

5.3.2.1 Tendance générale à communiquer (communication globale)

Chaque agent a une tendance générale à communiquer en fonction de sa situation par rapport à la cible vers laquelle il a choisi de se déplacer, en terme de distance. Plus l'agent est loin de sa cible, moins il a tendance à communiquer.

$$p(i) = \frac{1}{\sqrt[n]{\delta(X_{curr}, X_{next})}} \tag{5.11}$$

avec $\delta(X_{curr}, X_{next})$ la distance entre la position actuelle X_{curr} de l'agent i (où i a commencé à se déplacer) à la position frontière sélectionnée X_{next}. Intuitivement, lorsque l'agent i atteint sa cible X, cette possibilité est égale à 1. Donc, l'agent peut communiquer à nouveau. Plus précisément, plus l'agent est proche de la cible, plus la probabilité pour recueillir des informations fiables sur cette position est grande.

Cette règle permet à un agent de se déplacer sans trop communiquer à partir du moment où il a choisi une cible. Cela lui évite de changer de cible trop souvent et économise du temps de calcul. Cette règle incite les agents à chercher à satisfaire leur objectif grâce à l'action plutôt que d'essayer de s'engager dans des communications infructueuses. En revanche, sa tendance à communiquer augmentera dès lors qu'il arrivera dans le voisinage de la cible car la probabilité sera alors plus forte de dialoguer avec des agents qui seront allés sur cette cible et qui lui apporteront des informations utiles (dans le cas favorables, ces informations pourront même lui éviter de se déplacer jusqu'à sa cible).

5.3.2.2 Tendance à communiquer avec tel ou tel agent (communication ciblée)

Si l'agent décide de communiquer (c.à.d. si la communication globale se déclenche), il va calculer une tendance à communiquer avec chacun des agents qu'il perçoit dans son environnement en suivant les règles ci-dessous :

- **Règle favorisant la communication avec des partenaires fiables** : les agents ont tendance à communiquer proportionnellement à leur confiance dans leurs partenaires. Plus les agents se font mutuellement confiance, plus ils ont tendance à communiquer.

$$p_{reli}(i, j) = T_{ij} \tag{5.12}$$

avec T_{ij} est la confiance intrinsèque de i en j.

– **Règle favorisant la communication avec des agents déconnectés** : les agents auront plus tendance à communiquer avec des agents avec lesquels ils n'ont pas communiqué depuis longtemps. En appliquant cette règle, l'agent recherchera donc à renouveler ses informations. Plus le temps de déconnexion avec un agent est long, plus la tendance à communiquer avec lui est grande (et inversement).

$$p_{dur}(i,j) = min\left(\frac{\triangle t_{ij}}{avg\left\{\triangle t'_{ik}\right\}_{k\in A_{com}(i)}}, 1\right) \tag{5.13}$$

avec :

– $\triangle t_{ij} = t - t'_{ij}$ est le temps écoulé depuis la dernière communication entre i et j. t est le temps actuel et t'_{ij} le temps de la dernière communication entre i et j.

– $\triangle t'_{ik} = t'_{ik} - t''_{ik}$ est la dernière valeur du temps de déconnexion enregistré entre i et k, k représentant tout agent ayant communiqué avec i ($k \in A_{com}(i)$). t'_{ik} représente le dernier instant auquel i et k ont communiqué et t''_{ik} l'avant-dernier ($t''_{ik} = 0$ si i a communiqué une seule fois avec k). Je retiens la dernière valeur afin de tenir compte de l'évolution de la simulation dans son ensemble au fil du temps.

– $avg\left\{\triangle t'_{ik}\right\}_{k\in A_{com}(i)}$ est la valeur moyenne des intervalles de déconnexion $\triangle t'_{ij}$ de l'agent i avec tous les agents ayant communiqué avec i.

S'il y a un seul agent j avec lequel i a communiqué,

$$p_{dur}(i,j) = min\left(\frac{t - t'_{ij}}{t'_{ij} - t''_{ij}}, 1\right) \tag{5.14}$$

Exemple 7. Considérons que les délais de déconnexion successifs de i avec deux agents m et n ont été enregistrés dans le tableau 5.1. Je ne travaille que sur la ligne $\triangle t'_{ik}$ car elle représente le dernier délai enregistré, plus à même de traduire la tendance globale de la simulation. Une alternative consisterait à prendre en compte la moyenne des délais de déconnexion pour chaque agent au lieu de la dernière valeur.

	m	n
$\triangle t'_{ik}$	**50**	**4**
$\triangle t_{ik}$	7	10
$avg\left\{\triangle t'_{ik}\right\}$	27	

TABLE 5.1 – Exemple des délais de déconnexion successifs de i avec deux agents m et n

$$p_{dur}(i,m) = min\left(\frac{\triangle t_{im}}{avg\left\{\triangle t'_{ik}\right\}}, 1\right) = min\left(\frac{7}{27}, 1\right) = \frac{7}{27}$$

$$p_{dur}(i,n) = min\left(\frac{\triangle t_{in}}{avg\left\{\triangle t'_{ik}\right\}}, 1\right) = min\left(\frac{10}{27}, 1\right) = \frac{10}{27}$$

Si l'agent i n'avait jamais rencontré l'agent n, la tendance deviendrait :

$$p_{dur}(i,n) = min\left(\frac{\triangle t_{in}}{\triangle t'_{in}}, 1\right) = 1$$

Cette probabilité devrait être 1, car $t > avg\left\{\triangle t'_{ik}\right\}$.

De même, la probabilité de communication tend vers 1 lorsque le temps de déconnexion augmente et déplace le temps de communication moyenne.

– **Règle favorisant le renouvèlement de l'information.** En appliquant cette règle, les agents vont chercher à communiquer avec les agents qui envoient des nouvelles informations. Plus les informations reçues pour un agent sont nouvelles, plus cet agent a tendance à communiquer avec l'émetteur (et inversement).

Le nombre des informations nouvelles transmises par l'agent j sera déterminé à partir des 3 types d'informations suivants : nouvelles positions, positions connues mais avec une information associée différente, nouvelles valeurs de confiance sur les agents. La tendance doit être évaluée non seulement à partir du nombre d'informations nouvelles mais aussi à partir du pourcentage de ces nouvelles informations par rapport au nombre total d'informations reçues.

$$p_{new}(i,j) = \frac{1}{2}.\left(\frac{n_j}{max(n_k)_{k\in A_{com}^{t-1}(i)}} + \frac{n_j}{n'_j}\right) \tag{5.15}$$

avec :
- $p_{new}(i,j)$: tendance de l'agent i à communiquer avec l'agent j, basée sur les nouvelles informations reçues de l'agent j à l'instant t
- n_j est le nombre d'informations nouvelles reçues de l'agent j à l'instant $t-1$
- n'_j est le nombre total d'informations reçues de l'agent j à l'instant $t-1$
- $max(n_k)_{k\in A_{com}^{t-1}(i)}$ est la valeur maximale des nombres d'informations nouvelles reçues des agents avec lesquels i est en contact à l'instant $t-1$

Exemple 8. Supposons que i reçoive des informations des agents j, k, l et m au cours du même pas de la simulation. Pour ces informations, on ne s'intéresse que le nombre d'information nouvelles sur le nombre total d'informations reçues. La tendance de communication peut alors être calculée comme dans le tableau dans le tableau 5.2.

Agent	nombre d'informations nouvelles / nombre total d'informations reçues	$p_{new}(x,y)$
j	$\frac{5}{10}$	$p_{new}(i,j) = \frac{1}{2}.\left(\frac{5}{10} + \frac{5}{10}\right) = 0.5$
k	$\frac{5}{20}$	$p_{new}(i,k) = \frac{1}{2}.\left(\frac{5}{10} + \frac{5}{20}\right) = 0.375$
l	$\frac{10}{10}$	$p_{new}(i,l) = \frac{1}{2}.\left(\frac{10}{10} + \frac{10}{20}\right) = 0.75$
m	$\frac{1}{1}$	$p_{new}(i,m) = \frac{1}{2}.\left(\frac{1}{10} + \frac{1}{1}\right) = 0.55$
$max_{k\in A_{com}^{t-1}(i)}(n_k) = 10$ $A_{com}^{t-1}(i) = \{j,k,l,m\}$		

TABLE 5.2 – Exemple des informations reçues par l'agent i des agents $\{j,k,l,m\}$

Finalement, en considérant simultanément les règles 5.12,5.13 et 5.15 la tendance d'un agent à communiquer avec un autre agent peut se calculer de la manière suivante :

$$p(i,j) = w_5.p_{reli}(i,j) + w_6.p_{dur}(i,j) + w_7.p_{new}(i,j) \tag{5.16}$$

avec $w_5 + w_6 + w_7 = 1$

Interprétation : Je détaille ici l'impact du choix des poids $w5$, $w6$ et $w7$ dans la stratégie de communication ciblée des agents ?
- Plus w_5 est grand, plus l'agent va privilégier la communication avec les agents fiables
- Plus w_6 est grand, plus l'agent souhaite réduire le volume de ses communications pour éviter les informations redondantes qu'il pourrait recevoir en communiquant plusieurs fois de suite avec les mêmes agents.
- Plus w_7 est grand, plus l'agent favorisera la communication avec les agents qu'il juge utiles, autrement dit ceux qui le font progresser plus vite vers son objectif.

5.3.2.3 Tendance à communiquer une partie de ses informations (communication sériée)

En considérant l'hypothèse que les méta-données concernant le réseau de confiance sont toujours transmises, la question se pose de savoir si un agent doit toujours communiquer la totalité de ses informations. J'introduis ci-dessous deux règles destinées à sérier les informations qu'un agent transmet à un autre. Rappelons qu'il existe deux types d'informations : les informations directes collectées par l'agent lui-même sur le terrain et les informations indirectes qu'il a réunies au cours des échanges qu'il a eu avec les autres agents par la communication.

- **Règle de communication des informations directes** : Dans la communication avec des partenaires, plus les informations directes sont fiables, plus l'agent a tendance à les envoyer (et inversement)

$$p\left(i, j, D_i(X)\right) = T_{ii} \tag{5.17}$$

où $D_i(X)$ est l'information directe de l'agent i sur la position X, j est un récepteur, T_{ii} est la confiance de i sur lui-même. Cette règle se base sur l'autoévaluation de l'agent. Sa confiance en lui-même est égale à 1 au début de la simulation. Elle est ensuite affectée par l'opinion que les autres agents le concernant. Si cette confiance baisse, cela signifie que les autres agents ne se fient pas aux informations directes qu'il transmet. L'agent est alors incité à s'interroger sur la fiabilité de ses capteurs et pour ne pas nuire à la communauté à laquelle il appartient, décide de réduire la probabilité de transmission des informations qu'il a collectées.

- **Règle de communication des informations indirectes.** De même, plus les informations indirectes que l'agent a reçues sont fiables, plus l'agent les transmet (et inversement). Cela permet de contribuer à la qualité du système de communication.

$$p\left(i, j, IND_i\left(X\right)\right) = f\left(i, X\right) \tag{5.18}$$

où $IND_i\left(X\right)$ sont les données indirectes que i a sur la position X, $f\left(i, X\right)$ est la fiabilité des données indirectes sur la position X que l'agent i a reçu. De la même façons, afin d'améliorer la qualité des informations qui circulent entre les agents, un agent va favoriser la transmission des informations les plus fiables. Ces deux règles vont permettre de diminuer le volume des communications entre agents tout en assurant une plus grande fiabilité des informations transmises.

5.3.3 Algorithme proposé

L'algorithme 9 décrit d'une manière synthétique la stratégie de communication des agents. L'ordre dans lequel les actions des agents sont exécutées à chaque instant est le suivant : l'agent tout d'abord détermine si il communique (lignes 1-2). Si la réponse est oui, il détermine alors avec lesquels communiquer (lignes 3-9). Une fois que l'agent a trouvé des partenaires pour communiquer, il détermine quelles données il va envoyer à ses partenaires (lignes 10-13 et 14-17) puis les envoie (ligne 12 et 16).

5.4 Interaction entre le système de communication et le système de collecte d'information

Dans les deux sections précédentes, j'ai proposé des règles qui avaient trait soit au déplacement soit à la communication et donc pouvaient relever, à l'interprétation près, de deux systèmes indépendants capables de s'auto-organiser sur la base de ces règles. Or il n'en est rien. Le système de collecte d'informations distribué que j'étudie est la résultante de ces deux systèmes et de l'interaction entre ces systèmes (Fig. 5.9). Chaque agent peut communiquer et se déplacer à la fois. Chaque déplacement d'un agent va interférer avec sa stratégie de communication, étant donné que l'agent est transféré dans un nouvel environnement (fragment perçu de l'environnement global). Chaque acte de communication va de même influencer sa stratégie de déplacement puisqu'il peut disposer alors de nouvelles informations qui vont aider à déterminer sa future cible de déplacement. L'ensemble des règles présentées ensuite est destiné à définir la nature de ces interactions. L'idée étant de modifier les calculs précédents et notamment les forces d'attraction et de répulsion d'un agent sur un autre pour tenir compte de l'interaction entre les deux sous-systèmes. Elles mettent en jeu dans la même règle un élément issu du système associé à la

Algorithme 9: Algorithme décrivant la communication des agents à l'éxécution à l'instant t.

Entrées : X_{curr} la position actuelle de l'agent, X_{next} la position cible vers laquelle se déplace l'agent, $A_{com}^{t-1}(i)$ ensemble des agents ayant communiqué avec l'agent i à l'instant $t-1$, $A_{com}(i)$ ensemble des agents ayant communiqué avec l'agent i.

Sorties : i communique ou non avec agent j se trouvant dans le zone de perception de l'agent i. S'il communique, la probabilité d'envoi des informations directes et indirectes est considérée.

$p(i) \leftarrow \frac{1}{\sqrt[n]{\delta(X_{curr}, X_{next})}}$
si $flip(p(i)) = 1$ **alors**
  $avg(\Delta' t_i) \leftarrow$ la moyenne de $\Delta' t_i$ pour tous les agents k qui ont communiqué avec i
  pour tous les *agents j se trouvant dans le zone de perception de l'agent i* **faire**
    $p_{reli}(i,j) \leftarrow T_{ij}$
    $p_{dur}(i,j) = min\left(\frac{\triangle t_{ij}}{avg\{\triangle t'_{ik}\}_{k \in A_{com}(i)}}, 1\right)$
    $p_{new}(i,j) = \frac{1}{2}.\left(\frac{n_j}{max(n_k)_{k \in A_{com}^{t-1}(i)}} + \frac{n_j}{n'_j}\right)$
    $p(i,j) \leftarrow w_5.p_{reli}(i,j) + w_6.p_{dur}(i,j) + w_7.p_{new}(i,j)$
    si $flip(p(i,j)) = 1$ **alors**
      pour tous les *informations directes portant le point X de l'agent i* **faire**
        si $flip(T_{ii}) = 1$ **alors** envoyer $D_i(X)$ à l'agent j
      fin
      pour tous les *informations indirectes portant le point X de l'agent i* **faire**
        si $flip(f(i,X)) = 1$ **alors** envoyer $IND_i(X)$ à l'agent j
      fin
    fin
  fin
fin

communication et un élément issu du système associé au déplacement. Ainsi le déplacement d'un agent i pourra dépendre de la nature et de la qualité des informations qu'il reçoit des autres agents.

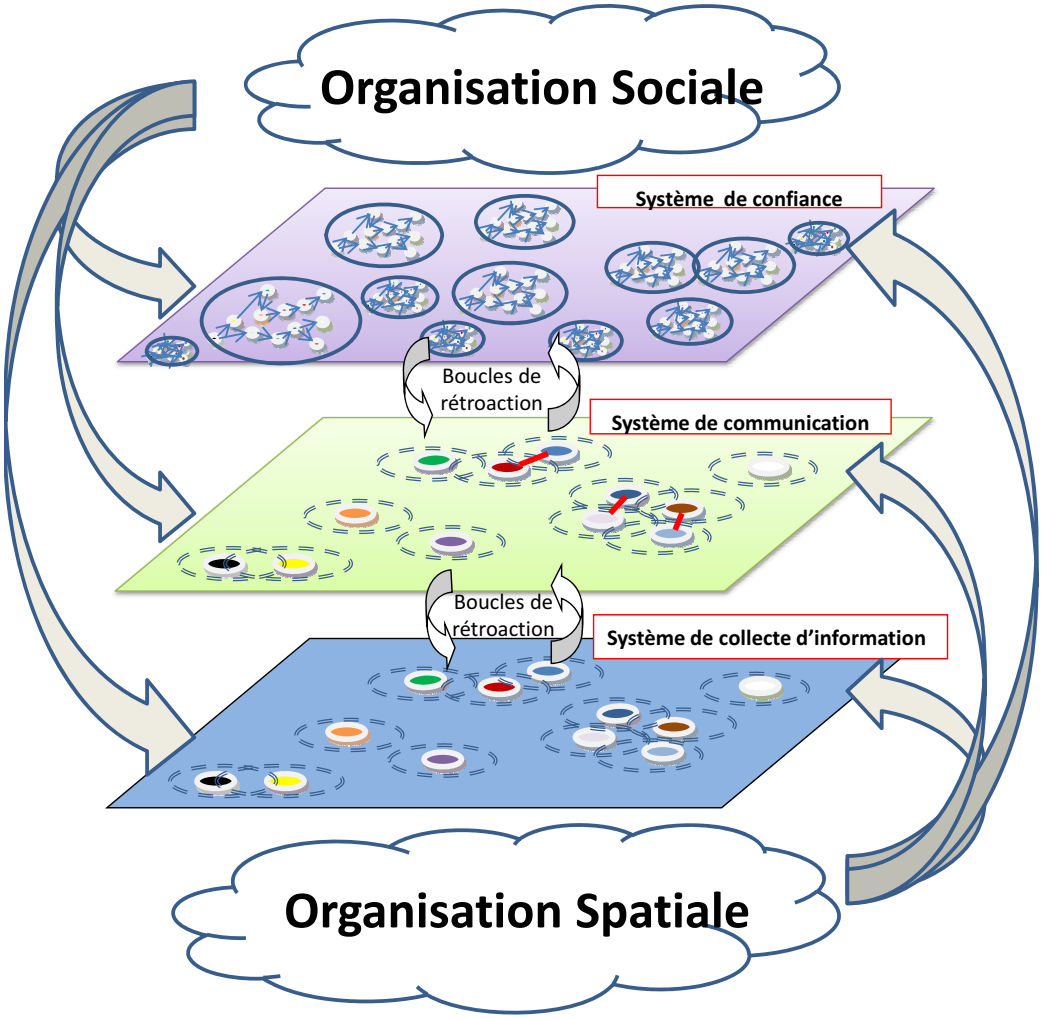


Figure 5.9: Modèle d'un système de collecte d'informations

– **Règle de l'influence mutuelle entre le renforcement et la diversification du déplacement et la fiabilité des informations échangées** : plus les informations reçues d'un agent sont correctes, plus la tendance de l'attractivité (resp. répulsion) entre l'expéditeur et le récepteur est grande.

$$level_{reli} = \frac{\sum_1^n \delta\left(D_i\left(X\right), D_j\left(X\right)\right)}{n} \tag{5.19}$$

où n est le nombre des informations comparables (c'est-à-dire les données portant sur les mêmes objets) ; $\delta\left(D_i\left(X\right), D_j\left(X\right)\right)$ représente le résultat de la comparaison entre l'information directe de l'agent i et l'information directe reçue de l'agent j portant sur la même position X :

$$\delta\left(D_i\left(X\right), D_j\left(X\right)\right) = \left\{ \begin{array}{cc} 1 & D_i\left(X\right) = D_i\left(X\right) \\ -1 & D_i\left(X\right) \neq D_j\left(X\right) \end{array} \right\} \tag{5.20}$$

– **Règle de l'influence mutuelle entre le renforcement et la diversification du déplacement et la nouveauté des informations échangées** : plus les informations reçues d'un agent sont nouvelles, plus la tendance de l'attractivité (resp. répulsion) entre l'expéditeur et le récepteur augmente.

$$level_{new} = p_{new}(i,j) - 0.5 = \frac{1}{2}.\left(\frac{n_j}{max(n_k)_{k\in A^t_{com}(i)}} + \frac{n_j}{n'_j}\right) - 0.5 \tag{5.21}$$

À partir de ces règles, la valeur de la force $\overrightarrow{F_i}$ qui représente la résultante des forces d'attraction ou de répulsion émises par les agents environnants perçus va être modifiée comme suit. Étudions tout d'abord comment les forces qui constituent les composantes de $\overrightarrow{F_i}$ sont elles-mêmes modifiées. Jusqu'à présent, si je ne considère pas les interactions (déplacement / communication / force d'attraction entre i et j). Cette force s'exprime ainsi :

$$\overrightarrow{F_{ij}} = \frac{(T_{ij} - 0.5)}{0.5}.\frac{\overrightarrow{V_{ij}}}{\delta(i,j)}$$

L'agent i va maintenant affiner le calcul de la valeur de force $\overrightarrow{F_{ij}}$ qui le lie à l'agent j pour considérer l'impact de la communication qu'il vient d'avoir avec j et donc des informations qui lui sont transmises par j à l'instant t :

$$\overrightarrow{F_{ij}} = \mu_{ij}.\frac{\overrightarrow{V_{ij}}}{\delta(i,j)} \tag{5.22}$$

où

$$\mu_{ij} = \left(w_8.\frac{(T_{ij} - 0.5)}{0.5} + w_9.level_{reli} + w_{10}.level_{new}\right)$$

$$= \left(w_8.\frac{(T_{ij} - 0.5)}{0.5} + w_9.\frac{\sum_1^n \delta\left(D_i(X), D_j(X)\right)}{n} + w_{10}.\left(\frac{1}{2}.\left(\frac{n_j}{max(n_k)_{k\in A^t_{com}(i)}} + \frac{n_j}{n'_j}\right) - 0.5\right)\right) \tag{5.23}$$

où $w_8 + w_9 + w_{10} = 1$

Le vecteur $\overrightarrow{F_i}$ est ensuite calculé en tant que résultante de ces nouveaux vecteurs $\overrightarrow{F_{ik}}$. Le calcul de la prochaine cible s'effectue comme précédemment : pour chaque point cible X, mise à jour de la valeur de la force résultante puis du coefficient d'attractivité :

$$g_{agent}(i,X) = cos(\overrightarrow{F_i}, \overrightarrow{V_{iX}}).|\overrightarrow{F_i}| = \frac{\overrightarrow{F_i}.\overrightarrow{V_{iX}}}{|\overrightarrow{V_{iX}}|}$$

$$g(i,X) = w1.\frac{g_{dist}(i,X)}{max_{Z\in P_{front}(i)}(g_{dist}(i,Z))} + w2.\frac{g_{inf}(i,X)}{max_{Z\in P_{front}(i)}(g_{inf}(i,Z))}$$

$$+w3.\frac{g_{reli}(i,X)}{max_{Z\in P_{front}(i)}(g_{reli}(i,Z))} + w4.\frac{g_{agent}(i,X)}{max_{Z\in P_{front}(i)}(g_{agent}(i,Z))}$$

La cible $position_{t+1}(i)$ de l'agent i est alors choisie comme étant celle dotée du coefficient d'attractivité le plus grand

$$position_{t+1}(i) = M \quad tel\ que \quad g(i,M) = max_{Z\in P_{front}(i)}(g_{inf}(i,Z)) \tag{5.24}$$

Interprétation : Avant d'aller plus loin, tentons une interprétation des nouveaux paramètres introduits et de leur impact sur le système global.

- Plus w_8 est grand, plus l'agent a tendance à rester à proximité des agents fiables
- Plus w_9 est grand, plus l'agent tend à se rapprocher des agents qui viennent de lui transmettre des informations directes fiables. Cette règle se distingue de la précédente en ce qu'elle travaille sur l'instant présent sans présager de la confiance que l'agent peut avoir dans le porteur d'informations. Cette nuance permet de reprendre contact avec des agents dont la fiabilité peut être faible mais qui, à un instant donné, se trouvent capables d'apporter des informations fiables. Je traite ainsi le cas d'agents dont la fiabilité est variable, comme des robots dont les capteurs fonctionneraient correctement par intermittence.

– Plus w_{10} est grand, plus l'agent privilégie la nouveauté des informations qui lui sont apportées et se rapproche des agents capables d'enrichir substantiellement sa base de données.

Algorithme 10 décrit le moteur principal d'un agent. L'ordre dans lequel les actions des agents sont exécutées est la suivante : collecte de données (ligne 1), communication (lignes 2-21), le déplacement (lignes 22-38). Dans le cadre de la communication, l'agent détermine tout d'abord s'il communique (lignes 2-3). Si la réponse est positive, il doit déterminer les agents avec qui communiquer (lignes 4-10). Une fois que l'agent a trouvé des partenaires pour communiquer, il détermine quelles données doivent être envoyées à ses partenaires (lignes 11-14 et 15-18) puis les envoie (ligne 13 et 17). Dans le cadre du déplacement, l'agent vérifie s'il est à la position cible choisie frontière. Si il n'y est pas encore, il se déplace vers la position frontière choisie. Sinon, l'agent doit calculer la prochaine position de la frontière vers laquelle se déplacer. Dans ce cas, l'agent calcule le gain pour chaque position considérée de la frontière (lignes 23-33), y compris le gain associé à la distance (ligne 24), à la quantité d'informations (ligne 25), à la fiabilité de la position (ligne 26), et à d'autres agents détectés (lignes 27-32). La position ayant le plus grand gain est alors choisie comme la prochaine position des frontières vers laquelle il va se déplacer (lignes 34 - 37).

5.5 Contrôle de l'auto-organisation

5.5.1 Introduction

Rappelons tout d'abord le concept d'auto-organisation dans un système complexe. Un système complexe est composé d'un grand nombre d'individus dont le comportement imprévisible. Chaque individu suit différents objectifs. Afin de mieux atteindre un objectif, chaque individu peut modifier de manière dynamique son comportement et apprendre de son expérience pour s'adapter à son environnement réel. La capacité à s'adapter à l'environnement intervient à travers l'émergence du rôle de l'individu(s) ainsi que l'émergence de la structure du nouveau système : le système a la capacité de l'auto-adaptation. Cela conduit un système complexe à une auto-organisation. L'auto-organisation pourraient ainsi être définie comme la capacité d'un système pour atteindre une organisation, par le biais d'un processus interne, sans aucune forme de contrôle ou d'intervention (implicite et explicite).

Afin de s'auto-organiser, chaque individu dans une auto-organisation doit agir en fonction des règles locales. D'un côté, dans le contexte d'un système complexe, de nombreuses règles sont peut-être en conflit. Un individu doit donc choisir les règles les plus appropriées afin d'atteindre son objectif. D'un autre côté, l'environnement du système change toujours, y compris l'environnement physique et celui social. Une règle adaptée à l'instant t peut devenir inadaptée à l'instant $t' > t$. Par conséquent, chaque individu doit être conscient de sa situation actuelle dans l'environnement, puis hiérarchiser les règles à appliquer dans cette situation afin de mieux atteindre son objectif.

Donc, le problème de l'optimisation du comportement individuel dans une auto-organisation devient le problème de l'optimisation de la combinaison des règles locales pour chaque individu. Parmi les nombreuses méthodes proposées pour optimiser le comportement des individus dans l'auto-organisation, l'algorithme génétique est une approche classique très usitée et efficace jusqu'à aujourd'hui [Surmann et al., 1993, Kita et al., 2010]. C'est la raison pour la-quelle j'ai choisit de l'appliquer cette approche dans l'exemple d'application Danger Mapping pour optimiser le comportement des individus suivant un modèle générique pour l'auto-organisation.

Une autre approche consiste à doter les agents de capacités d'apprentissage et de fonctions d'évaluation qui leur permettent après analyse des résultats de leur actions de modifier eux-mêmes les poids qu'ils associent aux règles afin d'optimiser leur comportement en vue de mieux servir la communauté, restent alors à définir les fonctions d'évaluation, leur impact sur le comportement des agents, et donc un ensemble approprié et judicieux de méta règles déclenchables par l'agent qui feront évoluer les différents poids.

Algorithme 10: Algorithme pour le déplacement et la communication de l'agent à chaque instant

Entrées : la position actuelle de l'agent i X_{curr}, des agents que i peut communiquer à l'instant t

Sorties : l'éxecution du déplacement et de la communication de l'agent i selon X_{curr} et des agents que i peut communiquer à l'instant t

i collecte des données à sa position actuelle X_{curr}
$p(i) \leftarrow \frac{1}{\sqrt[n]{\delta(X_{curr}, X_{next})}}$
si $flip(p(i)) = 1$ **alors**
  $avg(\Delta' t_i) \leftarrow$ la moyenne de $\Delta' t_i$ pour tous les agents k qui ont communiqué avec i
  pour tous les *agent j se trouvant dans le zone de perception de l'agent i* **faire**
    $p_{reli}(i,j) \leftarrow T_{ij}$
    $p_{dur}(i,j) \leftarrow min\left(\frac{\triangle t_{ij}}{avg\{\triangle t'_{ik}\}_{k \in A_{com}(i)}}, 1\right)$
    $p_{new}(i,j) \leftarrow \frac{1}{2}.\left(\frac{n_j}{max(n_k)_{k \in A^{t-1}_{com}(i)}} + \frac{n_j}{n'_j}\right)$
    $p(i,j) \leftarrow w_5.p_{reli}(i,j) + w_6.p_{dur}(i,j) + w_7.p_{new}(i,j)$
    si $flip(p(i,j)) = 1$ **alors**
      pour tous les *informations directes portant le point X de l'agent i* **faire**
        si $flip(T_{ii}) = 1$ **alors** envoyer $D_i(X)$ à l'agent j
      fin
      pour tous les *informations indirectes portant le point X de l'agent i* **faire**
        si $flip(f(i,X)) = 1$ **alors** envoyer $IND_i(X)$ à l'agent j
      fin
    fin
  fin
fin
si *X_{curr} égale à la prochaine position X_{next} de l'agent i* **alors**
  pour tous les *points cible X situé sur la frontière d'exploration de l'agent i* **faire**
    $g_{dist}(i,X) \leftarrow \frac{1}{\sqrt[n]{\delta(i,X)}}$
    $g_{inf}(i,X) \leftarrow \frac{1}{\sqrt[n]{N(i,X)+1}}$
    $g_{reli}(i,X) \leftarrow 1 - f(i,X)$
    pour tous les *agents j se trouvant dans la zone de perception de l'agent i* **faire**
      $level_{trust}(i,j) \leftarrow \frac{T_{ij}-0.5}{0.5}$
      $level_{reli}(i,j) \leftarrow \frac{\sum_1^n \delta(D_i(X), D_j(X))}{0.5}$
      $level_{new}(i,j) \leftarrow \frac{1}{2} * \left(\frac{n_j}{max(n_k)} + \frac{n_j}{n'_j}\right) - 0.5$
    fin
    $g_{agent}(i,X) \leftarrow \frac{\vec{F_i}.\overrightarrow{V_{iX}}}{|\overrightarrow{V_{iX}}|}$
  fin
  pour tous les *points cibles X situé sur la frontière d'exploration de l'agent i* **faire**
    $g(i,X) \leftarrow w_1.\frac{g_{dist}(i,X)}{max_{z \in P_{front}(i)}(g_{dist}(i,Z))} + w_2.\frac{g_{inf}(i,X)}{max_{z_{front}(i)}(g_{inf}(i,Z))} +$
    $w_3.\frac{g_{reli}(i,X)}{max_{z_{front}(i)}(g_{reli}(i,Z))} + w_4.\frac{g_{agent}(i,X)}{max_{z \in P_{front}(i)}(g_{agent}(i,Z))}$
  fin
  $X_{next} \leftarrow M : g(i,M) = max_{Z \in P_{front}(i)}(g(i,Z))$
fin

5.5.2 Modélisation générique d'une auto-organisation

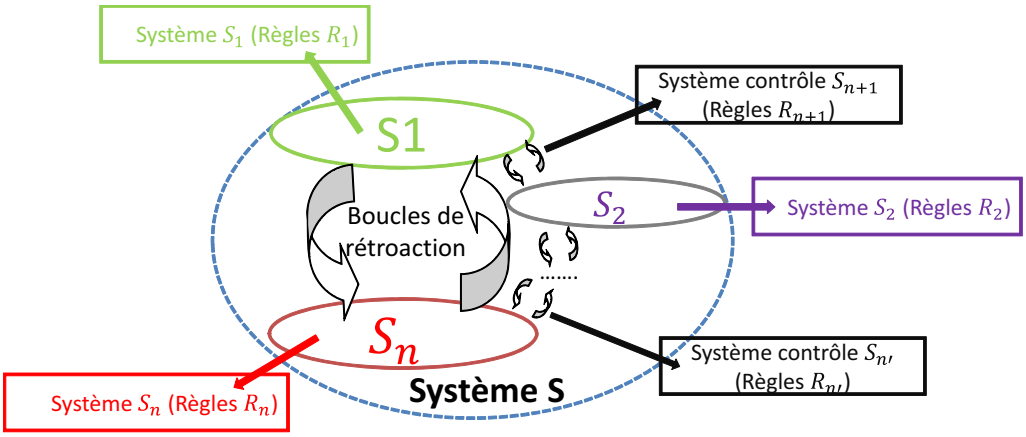


Figure 5.10: Système et règles locales dans suivant modèle auto-organisationnel

Je considère d'un point de vue générale une auto-organisation comme un ensemble de sous-systèmes parmi lesquels je distingue un ensemble de sous-systèmes rétroactifs, dits de contrôle, entre sous-systèmes (illustration Fig. 5.10). On note donc :

- L'ensemble des sous-systèmes $\{S_1, S_2, ..., S_n\}$. Un sous-système représente un point de vue sur le système global.
- L'ensemble des systèmes de contrôle $\{S_{n+1}, S_{n+2}, ..., S_{n'}\}$ entre sous-systèmes. Un système de contrôle gère la rétro-action entre deux sous-systèmes. Le nombre maximum de systèmes de contrôle pour n sous-systèmes est $m = \frac{n(n-1)}{2}$. Par conséquent, l'auto-organisation est un ensemble de sous-systèmes et de systèmes de contrôle : $S = \{S_1, S_2, ..., S_{n'}\}$, $n' = n + \frac{n(n-1)}{2}$.

Chaque sous-système ou système de contrôle S_i a un ensemble de règles locales $R_i = \{r_{i1}, r_{i2}, .., r_{ik}\}$. Une règle locale r_{ij} guidant le comportement des individus dans le sous-système avec un niveau d'importance spécifique. Ce niveau d'importance est caractérisé par un ensemble de poids $w_{ij} \in [0, 1]$. Sans perte de généralité, la somme des poids dans un sous-système S_i est normalisée :

$$\sum_{j=0}^{k} w_{ij} = 1, \forall i : 0 \leq i \leq n' \tag{5.25}$$

L'ensemble des règles locales pour l'auto-organisation est une union des ensembles des règles locales des sous-système $R = \{R_1 \cup R_2 ... \cup R_n ... \cup R'_n\}$

La combinaison des poids des règles locales va faire émerger des caractéristiques et des rôles pour les individus à l'intérieur de l'auto-organisation.

De ce point de vue, on pourrait dire que le comportement d'un individu dans une auto-organisation est contrôlé par un ensemble de règles locales, y compris les règles locales des sous-systèmes et ceux des systèmes de contrôle. Chaque règle locale est associée à un niveau d'importance, représenté par un poids. La combinaison de poids des règles locales affecte le comportement des individus.

Une question intéressante est de savoir comment optimiser la combinaison de ces poids pour influencer le comportement de l'individu pour qu'il atteigne plus facilement son objectif dans une situation particulière ? Considérons une auto-organisation avec n sous-systèmes, m systèmes de contrôle. Dans chaque sous-système ou système de contrôle, il y a k règles locales. Il y a donc $(n + m) * k$ règles locales. Et donc il y a aussi des $(n + m) * k$ poids des règles locales pour chaque individu. Supposons que chaque poids a v valeurs différentes possibles. Alors il y aura $v^{(n+m)*k}$ combinaisons possibles entre les poids.

Par exemple, dans une auto-organisation avec $n = 3$ sous-systèmes, $m = \frac{n(n-1)}{2} = 3$ systèmes de contrôle. Dans chaque sous-système ou système de contrôle, il y a au moins $k = 5$ règles locales. Chaque poids de règle a au moins $v = 100$ valeurs possibles. Il y donc aura $100^{(3+3)*5} = 10^{60}$ combinaisons

possibles. Ainsi, afin de d'optimiser le comportement d'un individu, comment trouver la meilleure combinaison dans un tel ensemble de solutions ? On ne peut bien évidemment pas utiliser une recherche exhaustive. L'application d'un algorithme génétique semble être adapté à cette taille énorme de l'espace des solutions.

5.5.3 Application de l'algorithme génétique pour l'optimisation du comportement dans un système auto-organisé

5.5.3.1 Objectifs

Le comportement des individus dans une auto-organisation est contrôlé par les règles locales $r_1, r_2 ... r_z$, avec z est le nombre total de règles locales. Le comportement d'un individu se caractérise par la valeur d'un ensemble de la poids $\{w_1, w_2, ..., w_z\}$. L'objectif est d'appliquer un algorithme génétique pour approcher la meilleure combinaison de poids des règles locales de w_1 à w_z en regardant ses effets sur l'objectif des individus. L'algorithme génétique nous aide à trouver les meilleures combinaisons de poids pour chaque individu. Les résultats obtenus nous aider à deux niveaux :

- Tout d'abord, nous pouvons utiliser ces résultats pour former les agents pour modifier dynamiquement les poids des règles locales lorsqu'il a des modifications de l'environnement (physique et / ou social), les agents sont conscients de ces modifications. Les résultats de l'algorithme génétique vont aider les agents à choisir la meilleure stratégie dans une nouvelle situation, afin de mieux atteindre leur objectif. Ils aident également l'agent à mieux s'adapter automatiquement à son environnement.
- Deuxièmement, ces résultats permettent d'évaluer si un agent a une bonne ou mauvaise adaptation au changement de son environnement. Un agent est considéré mieux adapté si sa stratégie est proche de la stratégie optimale (au niveau théorique) trouvée par l'algorithme génétique pour chaque situation spécifique.

5.5.3.2 Représentation d'un gène

On considère une combinaison de z-poids $(w_1, w_2, ..., w_z)$ comme un individu solution, par conséquent celle-ci détermine un gène unique. Pour construire un z-tuple $(x_1, x_2, ..., x_z)$ convenable, je pars d'un z-tuple $(x_1, x_2, ..., x_z)$ d'entiers que je construis $(w_1, w_2, ..., w_z)$ en suivant la règle :

$$w_i = \frac{x_i}{\sum_{r_j \in R_k} x_j}, \forall i : r_i \in R_k \tag{5.26}$$

Cette contrainte assure que chaque poids $w_i \in [0, 1]$ et $\sum_{r_j \in R_k} w_j = 1$ pour toutes les sous-systèmes ou le système de contrôle k. Par conséquent, un individu du z-tuple $(w_1, w_2, ..., w_z)$ est représenté par un gène qui est une chaîne de $z * n$ bits où n est le nombre de bits pour représenter un nombre naturel x_i. L'ordre des bits dans le gène correspond à l'ordre de nombre de x_1 à x_z.

5.5.3.3 Fonction d'utilité

La fonction d'utilité devrait dépendre des principaux critères pour évaluer la meilleure stratégie d'un individu. Par exemple, la stratégie d'un individu est considéré comme meilleure si : (i) il atteint son objectif avec une meilleure qualité, (ii) il atteint son objectif plus rapidement que d'autres, ou (iii) il consomme moins d'énergie ou de ressources, etc.

5.5.4 Un cas d'étude : un système de collecte d'information auto-organisé

Comme je l'ai présenté, l'auto-organisation du système de collecte d'information distribué va se faire au moyen des règles que j'ai mises en place et que chaque agent va mettre en application en fonction de l'environnement qu'il percevra à chaque instant. La boucle de rétroaction est en place entre le système gérant le déplacement et celui gérant la communication avec pour outil de calibration. La confiance, qui intervient dans de nombreuses règles. Elle intervient aussi bien au niveau de la fiabilité des informations que de la confiance accordée par un agent à un autre agent.

Il existe deux sortes d'auto-organisation que j'appellerai simple ou dirigée.

1. La première, dite auto-organisation simple, vise à laisser les agents manipuler les règles que l'utilisateur lui a fournies et développer des comportements qui, une fois observés avec les instruments de mesures adéquats et logiquement réunis, pourront dessiner pour chaque agent ou groupe d'agents un rôle identifiable par l'utilisateur qui sera apparu par émergence.
2. La seconde, dite auto-organisation dirigée, prétend que l'utilisateur peut agir sur la manière dont les agents se comportent et influencer de manière significative la démarche du système dans la réalisation de son objectif.

Dans le cas présent, une auto-organisation simple consisterait à avancer que toutes les règles ont une importance comparable pour les agents. Ce qui reviendrait à affecter des poids équivalents à ces règles.

Car je l'ai bien compris, au travers des différentes interprétations que j'ai donné précédemment, c'est en manipulant les poids affectés aux règles que je pourrais inciter les agents à adopter tel ou tel comportement que je voudrais voir apparaitre au fil de la simulation. N'oublions pas, cependant, que face au système complexe auquel je suis confronté, je pourrais être amené à voir apparaitre des rôles différents de ceux que j'ai anticipé, et même nouveaux et non prévus.

Les poids que j'ai affecté aux différentes règles sont donc autant de réglages sur lesquels je vais, en tant qu'utilisateur, pouvoir jouer de manière à «diriger» l'auto-organisation du système. Les poids associés aux règles peuvent être décomposés en trois ensembles, chacun étant tel que la somme de ses poids est égale à 1 :

- (w_1, w_2, w_3, w_4) ensemble associé au déplacement.
- (w_5, w_6, w_7) ensemble associé à la communication.
- (w_8, w_9, w_{10}) ensemble associé à l'interdépendance communication / déplacement.

Le nombre de combinaisons très important montre la richesse du système de guidage de l'auto-organisation. Chaque combinaison représente un comportement possible pour les agents. Le gène de l'agent sera représenté par la suite des poids, avec leurs contraintes liées. La fonction d'évaluation sera proposée au niveau global de la simulation. Dans notre cas, elle mesurera la capacité du système à collecter le plus d'informations dans le temps le plus réduit. Ce calcul aboutira à la détermination de valeurs de poids communes à tous les agents. Les résultats de l'application de l'algorithme génétique est présenté dans la chapitre 6.

Je définis par la suite un sous-ensemble de comportements caractéristiques que j'essaierai ensuite de retrouver sous la forme de rôles dans l'auto-organisation qui se dessinera au cours de simulations.

5.6 Indicateurs pour observer l'organisation émergente

La première tâche sera, pour évaluer le système en cours d'auto-organisation, de détecter les rôles qui émerge dans l'organisation sociale ou dans l'organisation spatiale. Pour cela, il sera nécessaire d'observer tout au long de la simulation, à l'aide d'indicateurs bien choisis, l'évolution des agents. Ce sera cette analyse des indicateurs qui permettra, suite à une interprétation, de fournir la nature des rôles.

Les organisations sociales et spatiales sont définies comme des ensembles de rôles. Elles matérialisent des vues filtrées de l'organisation globale :

- L'organisation sociale est représentée par l'ensemble des rôles sociaux qui permettent de réaliser les tâches de communication et par les relations entre ces rôles (certains agents cessent-ils de communiquer ? certains ne communiquent-ils que des informations indirectes ? certains ne communiquent-ils qu'avec un groupe d'agents ? etc...).
- L'organisation spatiale est aussi exprimée par un ensemble de rôles spatiaux associés au déplacement et par les relations entre ces rôles (certains agents se regroupent-ils ? D'autre s'éloignent-ils des groupes ? etc...).

Les rôles sont le fruit de l'application d'un ensemble de règles assujetties aux contraintes de l'environnement : certains rôles peuvent être pressentis au vu des règles imposées, d'autres non prévus peuvent être observés au cours de la simulation. Les rôles sociaux et les rôles spéciaux sont illustrés dans le tableau 5.3.

Les rôles sociaux sont caractérisés par des comportements différents des agents dans la communication avec les autres. Par exemple basé sur le TrustSet, chaque agent peut évaluer sa tendance à communiquer avec les autres agents. De là, peuvent apparaitre des clusters sociaux, traduisant le fait

Table 5.3: Tableau des rôles spatiaux et des rôles sociaux

	Rôle	Description	Indicateur
Spatial	Noyau spatial	les agents qui se jugent fiables restent proches les uns des autres	il existe au moins un même agent fiable dans sa zone de perception pendant une durée de temps
	Explorateur	les agents veulent explorer le plus d'espace possible	le nombre des zones collectés directes > moyenne du nombre des zones collectés directes de tous les agents
Social	Transmetteur	les agents privilégient l'envoi sur la réception	le nombre des informations envoyées > le nombre des informations reçues
	Récepteur	les agents privilégient la réception sur l'envoi	le nombre des informations reçues > le nombre des informations envoyées.
	Transmetteur d'informations directes	l'agent a confiance en lui mais n'a pas confiance dans les autres	si le pourcentage des informations directes envoyés > le pourcentage des informations indirectes envoyés
	Transmetteur d'informations indirectes	agent se méfie de lui et fait confiance aux autres	si le pourcentage des informations indirectes envoyés > le pourcentage des informations directes envoyés

que les agents qui se jugent fiables communiquent beaucoup entre eux. Selon la tendance de communication estimée et selon l'estimation de l'auto-fiabilité, l'agent pourrait avoir des rôles sociaux différents : récepteur, transmetteur, transmetteur d'informations directes, transmetteur d'informations indirectes, etc. L'organisation spatiale peut également voir l'apparition de clusters spatiaux. **Les rôles spatiaux** pourraient être par exemple : noyau spatial (les agents qui se jugent fiables restent proches les uns des autres), explorateur (les agents veulent explorer le plus d'espace possible), etc. Tout ceci sans oublier qu'un agent aura un rôle qui sera à la fois spatial et social (le social expliquant le spatial qui explique le social etc... - on retrouve la boucle de rétroaction). Ce qui sera intéressant également sera d'analyser le rôle de la confiance dans l'établissement des rôles. Rappelons que la confiance est l'outil générique qui va être partagé à la fois par les systèmes S_1 et S_2.

Pour les rôles spatiaux, on s'intéresse au déplacement d'un agent sous 2 points de vue : individuel et global (comparaison avec d'autres agents). Du point vue d'individuel, on stocke la trajectoire de l'agent avec les détails suivants : nombre de zones connues et inconnues observées à chaque étape, nombre total de zones connues et inconnues observées, le nombre des agents avec la qualité des agents autour de sa position et un nombre des obstacles autour de sa position. Du point vue global, on analyse la trajectoire de l'agent par rapport à celle des autres.

Pour les rôles sociaux, on s'intéresse aux informations sur la communication pour chaque agent à chaque instant de la simulation du point vue individuel. Au niveau de la communication global, on stocke le nombre des rencontres en relation avec le nombre des communications (par exemple, l'agent A a 150 rencontres avec d'autres agents, mais il a 70 communications). Au niveau de la communication ciblée, on stocke le nombre de rencontres avec un agent en lien avec le nombre de communications avec un agent (par exemple, pendant 150 rencontres, l'agent A a 30 rencontres avec l'agent B avec 10 communications). Au niveau de la communication sériée, on s'intéresse au pourcentage des informations directes envoyées (pour voir l'apparition du rôle transmetteur des informations directes), au pourcentage des informations indirectes envoyées (pour voir l'apparition du rôle transmetteur des informations indirectes) et au pourcentage des informations envoyé (pour voir l'apparition du rôle bloqueur d'informations), etc (par exemple, avec 10 communications adressées à l'agent B, l'agent A n'envoie que ses 10 informations

directes (25%) et ses 10 informations indirectes (5%)).

Des indicateurs proposés ci-dessus ont pour but de détecter les rôles qui émergeront soit dans l'organisation sociale, soit dans l'organisation spatiale. L'idée principale est d'observer le changement du comportement de l'agent pour s'adapter à la perturbation de l'environnement (autrement dit **le rôle**) afin d'assurer la cohérence et la robustesse du système.

Les rôles sociaux émergents du système en cours d'auto-organisation permettent de limiter la propagation des information fausses dans l'environnement. Des agents défectueux qui ont tendance à envoyer ses informations indirectes au lieu de ses informations directes (des information incorrectes provenant de capteurs défectueux) sont considérés comme jouant le rôle de **transmetteur d'informations indirectes.** Des agents fiables jouent le rôle de **transmetteur d'informations directes** parce qu'ils ont tendance à envoyer plus d'informations directes que d'informations indirectes.

Les rôles spatiaux émergés du système tels que **noyau de cluster spatial**, **l'explorateur**, etc. permettent une meilleure coopération entre les agents, une minimisation des redondances de données induite par les agents qui collaborent au lieu de rivaliser et un échange d'information plus rapide et plus sécurisé dans une coalition stable où la perturbation est minimale.

5.7 Conclusion

Dans ce chapitre, je définis une approche auto-organisationnelle basée sur la confiance. Elle est utilisée comme un moyen de structurer les échanges au niveau social (interactions et échanges entre agents). Cette structuration a également des incidences sur la structuration spatiale (ou plus généralement physique) des agents. C'est le couplage entre organisation sociale et organisation physique (spatiale) qui définit l'auto-organisation. L'organisation sociale reflète la relation de confiance établie entre les différents agents et l'organisation spatiale reflète le déploiement d'agents dans l'environnement pour favoriser la communication entre les agents.

La contribution majeure de ce chapitre est donc l'étude d'un système de collecte d'information perturbé sous l'angle de l'auto-organisation en analysant le système global sous la forme de :

1. Un système de déplacement basé sur la détermination d'une direction prometteuse selon l'objectif du déplacement. La confiance est utilisée comme un outil de guide. Cette stratégie de déplacement qui suit un modèle attraction / répulsion dans la lignée des travaux de Ferber, Simonin [Simonin and Ferber, 2003], Beurrier[Beurier et al., 2003] mais cela occulte complètement la partie dynamique des couplages entre les différents systèmes.
2. Un système de communication basé sur la tendance est présenté. Grâce à l'utilisation de la tendance, des agents peuvent éviter des effets de seuillage (changements brusques d'attitude) qui est utilisé souvent dans les algorithmes existants.
3. Un système de contrôle permettant la coordination de la co-évolution de la structure et de la fonction du système global. Il est un système émergent de la dynamique des systèmes de collecte d'information et du système de communication, traduisant ainsi leurs couplages.
4. Un contrôle de l'auto-organisation est introduit sur lequel on propose deux pistes intéressantes pour le guidage de l'auto-organisation.

Je vais présenter dans le chapitre suivant des expérimentations qui valident les résultats théoriques des chapitres 4 et 5.

Chapitre 6

Implémentation et évaluation

Sommaire

Ce chapitre présente le simulateur développé pour évaluer et valider l'approche théorique présentée dans les chapitres précédents. Ce chapitre se compose de 4 sections. La première section présente l'implémentation d'une application de collecte d'informations distribuées, appelée « Danger Mapping ». La deuxième section présente une évaluation de la performance de mon approche en terme de cohérence du système par rapport à d'autres modèles de confiance. Dans la troisième section, je présente les expériences que j'ai mené visant à analyser les organisations émergentes dans le système. Ces organisations sont obtenues par auto-organisation sans contrôle. J'analyse ensuite l'impact du contrôle sur l'auto-organisation. Ces expérimentations ont pour but de présenter les émergences qui apparaissent dans les deux sous-systèmes : déplacement, communication. Enfin la dernière partie présente les résultats obtenus par un algorithme génétique concernant l'optimisation de l'auto-organisation que je propose.

6.1 Implémentation

6.1.1 Description de l'application choisie : le « Danger Mapping »

L'exemple que j'ai choisi pour tester l'efficacité de l'utilisation des TrustSets avec l'approche auto-organisationnelle pour améliorer la détection des agents défectueux dans un système de collecte d'informations distribué, que j'appelle "Danger Mapping" par la suite, concerne la cartographie par un essaim de robots mobiles d'une zone sinistrée. Pour cette application, je considère les différents robots mobile comme parfaitement localisés [2]. Ceux-ci patrouillent dans une région inconnue parsemée de zones dangereuses fixes. L'objectif de chaque robot est de réaliser la carte la plus complète, précise et fiable de l'environnement en utilisant le moins de ressources possibles. En particulier, le travail d'un robot consiste à détecter les lieux dangereux et à évaluer leur degré de dangerosité. Pour cartographier la région étudiée, les robots peuvent détecter directement l'état d'une zone par le biais de leurs capteurs. Ils peuvent également communiquer avec d'autres robots pour échanger des connaissances sur l'environnement. Je suppose que chaque robot bénéficie d'une perception locale (dans un certain rayon de perception) et que la communication dans le système est également locale (et limitée à un certain rayon de communication). Parmi ces robots, il s'en trouve un certain nombre qui produisent des informations fausses ou inexactes à cause de capteurs défectueux ou inopérants. Ces informations seront ensuite transmises aux autres agents et introduisent une perturbation dans le système.

6.1.2 La simulation

6.1.2.1 Objectifs

Un simulateur pour cet exemple du "Danger Mapping" a été développé dans le but d'étudier les possibilités d'amélioration de la cohérence et de la robustesse dans ce type de système de collecte d'informations. Les principales questions qui sont posées à la simulation sont les suivantes :

- Quel est l'impact des robots défectueux sur la communauté ?
- Comment les robots défectueux peuvent-ils être détectés ?
- Quelle est l'influence du modèle de confiance sur les performances du système ? Dans la suite, je compare les modèles de confiance suivants :
  - un modèle de confiance basé sur une approche utilisant la logique floue [Kim and Seo, 2008]
  - un modèle de confiance basé sur une approche probabiliste [Zia, 2008]
  - un modèle de confiance basé sur le TrustNet [Schillo et al., 2000]
- Quelle est l'influence de la stratégie de communication sur les performances du système ? J'étudie les différentes stratégies suivantes :
  - les robots ne communiquent pas ;
  - les robots ne communiquent que des données ;
  - les robots communiquent des données et les TrustGraphs mais gardent privées les TrustTables (utilisation de seuils ou de tendances)
  - les robots communiquent des données et les TrustTables
- Comment les différents paramètres de la simulation influencent la robustesse du système ? (par exemple le taux de robot défectueux, etc.)
- Quelles sont les organisations émergentes ? Comment ont-elles été produites ?

6.1.2.2 Modélisation

Le terrain avec ses zones de danger est représenté par une carte vectorielle **SIG** (**S**ystème d'**I**nformation **G**éographique). L'espace est non torique - ce qui signifie que les robots sont contraints par les bordures de l'environnement. Deux cents zones dangereuses sont générées aléatoirement sur la carte avec pour chacune un niveau de dangerosité. Les agents représentent les robots explorateur. À chaque pas de simulation, ils peuvent collecter des données dans leur rayon de perception, communiquer avec les robots situés dans

2. Je ne m'intéresse pas ici au problème de la localisation des robots sur la carte, et ne traite donc pas le problème de la Localisation et de la cartographie simultanée *(Simultaneous Localization And Mapping ou SLAM)* largement étudiée en robotique [Leonard and Durrant-Whyte, 1991].

leur rayon de communication, mettre à jour leurs bases de données et de métadonnées et se déplacer. Des robots défectueux sont définis comme des robots incapables de détecter correctement le niveau de danger d'une zone (représentant un robot ayant des capteurs mal calibrés). La simulation s'arrête soit lorsqu'un robot connait toute la carte réelle, soit lorsque tous les robots fiables connaissent la carte réelle, suivant les choix de l'utilisateur.

La figure 6.1 présente le diagramme de classe du simulateur. Un robot est représenté par la classe **Robot** qui sert de classe mère pour l'ensemble des robots fiables et défectueux. Les classes **Robot fiable** (avec l'attribut de couleur initialisé à bleu) et **Robot défectueux** (avec l'attribut de couleur initialisé à rouge) redéfinissent chacune la méthode **capture_Env()** héritée de la classe **Robot**.

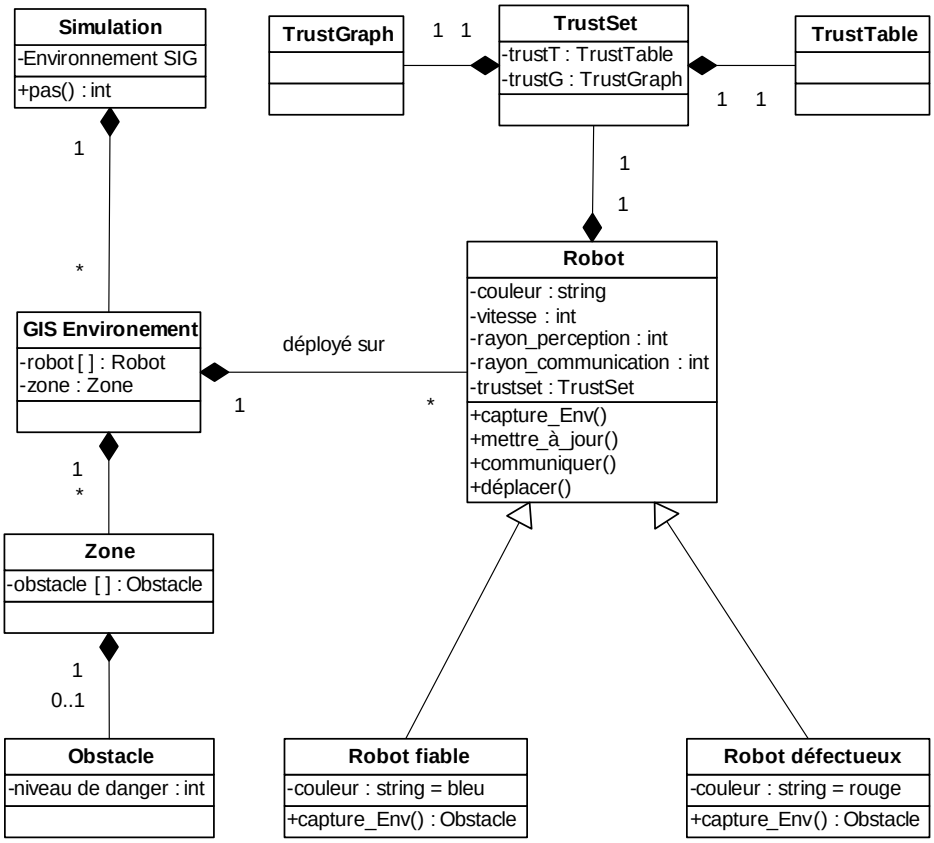


Figure 6.1: Modèle des composantes de simulation

La figure 6.2 représente les flux de données manipulée par un robot. L'ordre selon lequel les actions des robots sont exécutées est le suivant : collecte de données (flèches 1, 2), communication (flèches 3, 4, 5), mise à jour des bases de données (flèche 6) et déplacement (flèches 7, 8). Pour la collecte d'informations, le robot collecte directement l'information de l'environnement avec ses capteurs (flèche 1) et met ensuite sa carte à jour (flèche 2). En ce qui concerne la communication, on rappelle que le robot dispose de deux types d'informations : les informations internes privées sur lesquelles le robot se base pour se déplacer et les informations publiques qui peuvent être communiquées. Le robot peut donc communiquer ses informations publiques avec d'autres robots (flèche 3) et recevoir des informations indirectes (flèche 4). Il met ensuite à jour ses bases de données (flèche 5). La mise à jour sera appliquée pour les informations internes et communicables à partir des informations reçues (flèche 6). Le TrustGraph du robot ne sera recalculé que si l'un des éléments de base du calcul a été modifié, ou si un nouvel élément à considérer dans le calcul a été introduit dans le TrustGraph. Le robot se base enfin sur les informations internes privées pour se déplacer (flèches 7, 8).

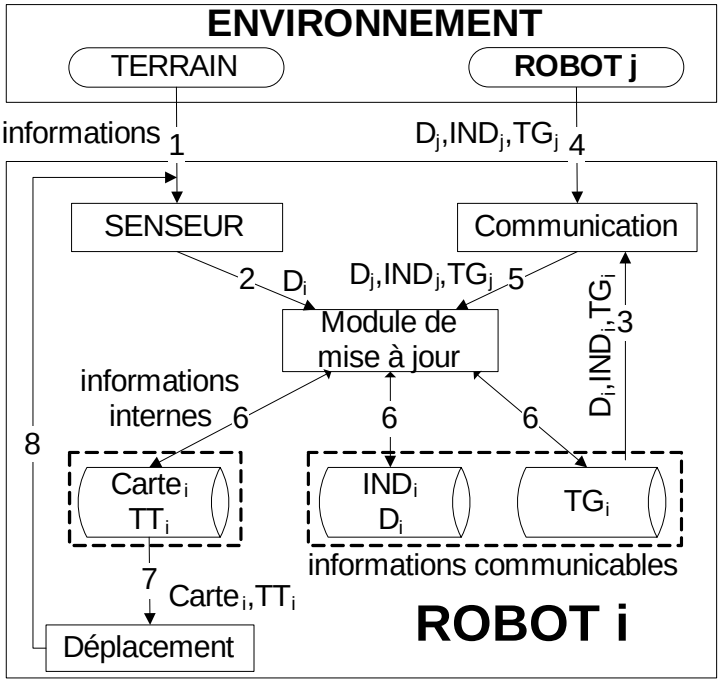


Figure 6.2: Flux des données dans le robot i

6.1.2.3 Simulateur « Danger Mapping »

Le simulateur « Danger Mapping » illustré sur la figure 6.3 a été développé en utilisant la plate-forme de simulation GAMA [Taillandier et al., 2010], dans sa version 1.3.

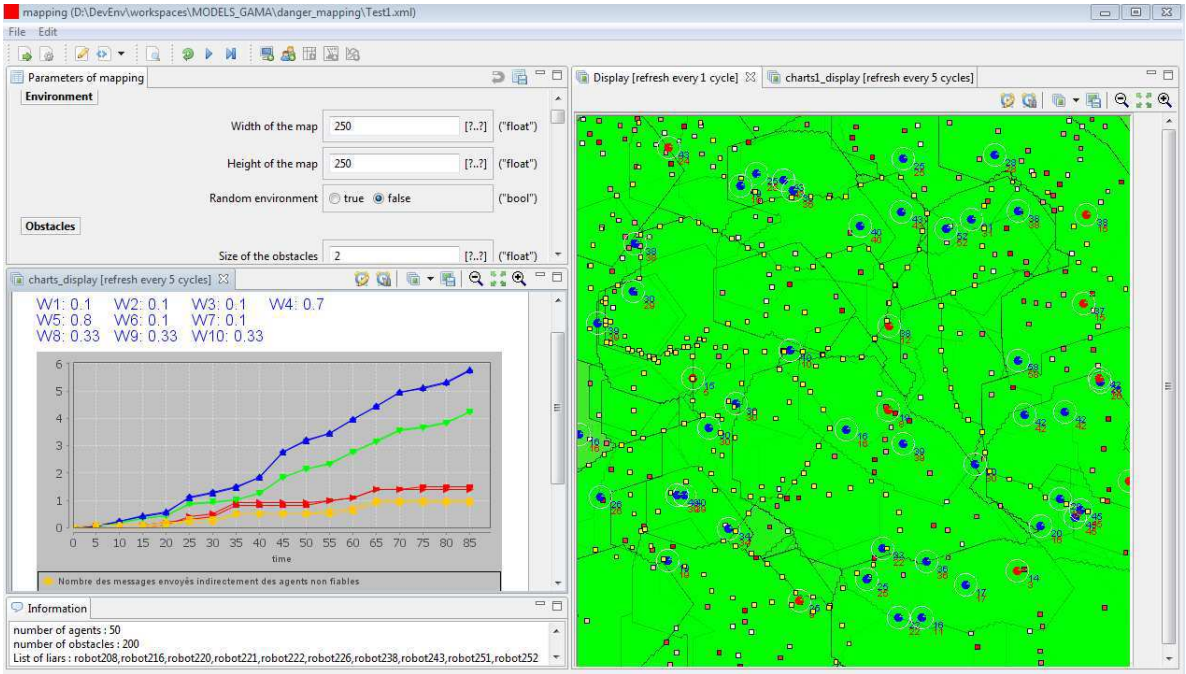


Figure 6.3: Capture d'écran du simulateur Danger Mapping en GAMA

GAMA est une plateforme open source de modélisation et simulation à base d'agents écrite en Java et développée par l'équipe de recherche MSI / IFI depuis 2007. Elle a pour avantages principaux :

1. La généricité : GAMA est générique, elle n'est pas dédiée à un thème.
2. L'assistance à la gestion de l'environnement : GAMA permet une gestion avancée des fichiers SIG (shapefile) pour créer l'environnement, ainsi qu'une manipulation aisée des géométries.
3. Une utilisation simple : la mise en œuvre du modèle est rendue accessible aux experts du domaine grâce au langage de modélisation GAML.
4. L'évolutivité : GAMA permet d'étendre facilement son langage à l'aide de fonctions écrites en Java.
5. La performance : des tests pratiques ont montré que la plate-forme peut supporter des centaines de milliers d'agents sur un ordinateur de bureau.

GAMA a été préféré à d'autres plates-formes de simulation comme NetLogo [Wilensky and Evanston, 1999] car celui-ci ne peut être étendu et est plus limité en terme de performance. Repast [North et al., 2006] et VLE (Virtual Laboratory Experiment) [Quesnel et al., 2009] sont quand à eux moins faciles à manipuler.

6.1.2.4 Paramètres

Les paramètres du simulateur sont résumés dans le tableau 6.1. On peut distinguer 4 catégories :
– les paramètres permettant de configurer l'environnement et le nombre de robots initiaux,
– les paramètres permettant de configurer les stratégies de communication et d'arrêt de la simulation,
– les paramètres relatifs aux comportements des robots (perception, déplacement et communication),
– les paramètres permettant de configurer le modèle de confiance TrustSets.

6.1.2.5 Évolution de la simulation

1. Initialisation : un certain nombre de robots, certains fiables, d'autres défectueux, sont créés de façon aléatoire.
2. Boucle principale (figure 6.4) : chaque robot exécute les actions suivantes dans l'ordre jusqu'à la fin de la simulation.
   (a) Collecter des données du terrain accessibles dans son rayon de perception ;
   (b) Communiquer (ou non) avec les robots présents dans son rayon de communication ;
   (c) Mettre à jour ses bases de données (bases de données du terrain et bases de metadonnées, i.e TrustSets) chaque fois qu'une nouvelle information est ajoutée ;
   (d) Se déplacer.

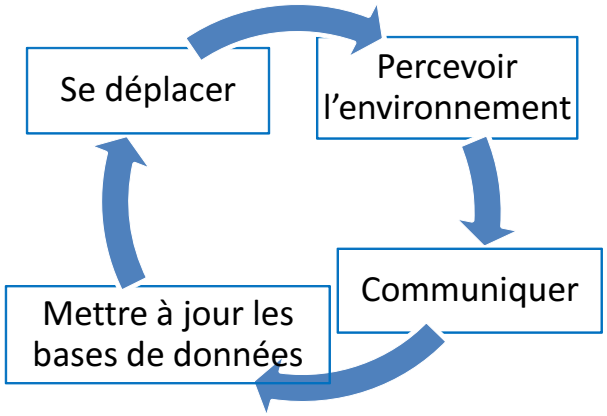


Figure 6.4: Boucle principale de simulation

Nom du paramètre	Description	Valeur par défaut
Environnement		
$width$	Largeur de l'environnement	200
$height$	Hauteur de l'environnement	200
$random$	Environnement inchangé ou non dans les prochaines simulations	$false$
$nRobot$	Nombre de robot fiables initiaux du système	50
$nLiar$	Nombre de robot défectueux initiaux du système	20
$nObstact$	Nombre d'obstacles	200
Simulation		
$stop$	La simulation va s'arrêter soit quand tous les robots fiables ont exploré correctement la carte , soit quand un robot fiable a exploré correctement la carte	All
$strategy$	1. les robots ne communiquent pas	4
	2. les robots ne communiquent que les données	
	3. les robots communiquent les données et les TrustTables	
	4. les robots communiquent les données et les TrustGraphs mais gardent privées les TrustTables	
	5. les robots communiquent les données et la confiance calculée à partir d'une approche utilisant la logique floue [Kim and Seo, 2008]	
	6. les robots communiquent les données et la confiance calculée à partir d'une approche probabiliste [Zia, 2008]	
	7. les robots communiquent les données et la confiance calculée avec le TrustNet [Schillo et al., 2000]	
Robot		
r_p	Le rayon de la région de perception dans laquelle un robot peut reconnaitre des zones dangereuses autour de sa positon	20
r_c	Le rayon de la région de communication dans laquelle un robot peut reconnaitre et communiquer avec d'autres robots autour de sa position.	10
$speed$	La vitesse de déplacement des robots	5
TrustSets		
NI	Nombre d'interactions nécessaires pour qu'un robot puisse avoir une valeur de confiance fiable envers les autres	5
ρ_{stab}	Le taux de stabilisation représentant le pourcentage de la simulation au-delà duquel le concepteur veut qu'une classification entre les robots fiables et défectueux se dégage	0.2
Upp	Le seuil haut de confiance	0.6
Low	Le seuil bas de confiance	0.4
T_{init}	La confiance qu'ont les robots par défaut envers les robots qu'ils n'ont pas encore rencontré.	0.5
μ	Le taux d'informations contradictoires acceptable	0.05

Table 6.1: Tableau des paramètres de la simulation

3. Fin : La simulation s'arrête soit lorsqu'un robot fiable connait la carte réelle, soit lorsque tous les robots fiables connaissent la carte réelle, suivant le choix de l'utilisateur.

6.1.3 Les indicateurs spécifiques

Afin d'observer la performance du système ou des organisations émergentes au niveau du système de collecte d'information et du système de communication, je propose d'introduire les indicateurs spécifiques suivants.

6.1.3.1 Observation de la performance du système

Pour observer la performance du système, on s'intéresse au temps (et temps moyen) d'exploration maximal pour que tous les robots fiables du système obtiennent une carte exacte :

- le temps maximal d'exploration nécessaire pour que tous les robots fiables obtiennent une carte exacte (avec des stratégies différentes en fonction du pourcentage de robots fiables)
- le temps moyen nécessaire aux robots fiables pour détecter correctement la carte (avec des stratégies différentes en fonction du pourcentage de robots fiables)

Plus le temps maximal d'exploration est petit, plus la performance du système est grande.

6.1.3.2 Observation du système de collecte d'information

Pour observer le système de collecte d'information, on stocke la trajectoire de robot avec les détails ci-dessous :

- le nombre de zones connues et inconnues à chaque étape pour chaque robot
- le nombre total de zones connues et inconnues
- la qualité des robots à approximé (fiables, défectueux)
- le nombre d'obstacles dans le périmètre

Plus le nombre de zones perçues de manière directe est supérieur à la moyenne du nombre des zones collectées de manière directe par tous les robots, plus le robot a tendance à explorer (on dira qu'il a plutôt un rôle explorateur). S'il existe au moins un même robot fiable dans sa zone de perception pendant une durée de temps, alors les robots qui s'évaluent fiables restent proches les uns des autres (matérialisation d'un *cluster*).

6.1.3.3 Observation du système de communication

Pour observer le système de communication, on s'intéresse aux informations liées à l'envoi et à la réception d'informations, on mémorise donc pour chaque robot et à chaque instant de la simulation les informations suivantes.

- le nombre d'informations directes qu'envoient les robots fiables
- le nombre d'informations indirectes qu'envoient les robots fiables
- le nombre d'informations directes que reçoivent les robots fiables
- le nombre d'informations indirectes que reçoivent les robots fiables
- le nombre d'informations directes qu'envoient les robots défectueux
- le nombre d'informations indirectes qu'envoient les robots défectueux
- le nombre d'informations directes que reçoivent les robots défectueux
- le nombre d'informations indirectes que reçoivent les robots défectueux

Plus l'écart entre le nombre d'informations envoyées et le nombre d'informations reçues est grand, plus les robots privilégient l'envoi sur la réception (on attribuera à ces robots le rôle de transmetteur par la suite).

Plus l'écart entre le nombre d'informations reçues et le nombre d'informations envoyées est grand, plus les robots privilégient la réception sur l'envoi (rôle de récepteur).

Plus l'écart entre le pourcentage d'informations directes envoyées et le pourcentage d'informations indirectes envoyées est grand, plus le robot a confiance en lui-même et pas en les autres (rôle de transmetteur d'informations directes).

Plus l'écart entre le pourcentage d'informations indirectes envoyées et le pourcentage d'informations directes envoyées est grand, plus le robot se méfie de lui-même et fait confiance aux autres (rôle de transmetteur d'informations indirectes).

6.1.3.4 Observation du niveau de la confiance

Au niveau de confiance, on s'intéresse à l'auto-confiance et à la réputation pour chaque robot à chaque instant de la simulation du point vue local avec les détails ci-dessous :

- la confiance du robot sur lui-même (son auto-confiance)
- la confiance de la communauté sur le robot (sa réputation)

L'auto-confiance permet à chaque robot de procéder à un examen critique de sa situation actuelle, d'auto-diagnostiquer s'il est lui-même fiable ou non. Le robot va changer ainsi son comportement en fonction de sa situation. La réputation d'un robot est calculée par la moyenne des confiances de tous les robots en lui-même. Plus l'auto-confiance et la réputation du robot sont grandes, plus le robot est fiable dans la communication avec les autres.

6.2 Évaluation des performances

Dans un environnement SIG de 400m x 400m, j'ai créé 200 obstacles représentant des zones dangereuses et 50 robots autonomes. Le nombre des robots défectueux est un paramètre de la simulation et variera. Ces chiffres ont été choisis pour que les robots puissent rencontrer un nombre de robots différents suffisant au cours de l'expérimentation pour construire un TrustSet significatif. La stratégie de déplacement utilisée est la même pour tous les robots dans toutes les exécutions présentées : le robot va calculer un ensemble de positions candidates situées à la frontière de l'environnement connu, puis il choisira parmi celles-ci celle qui maximise une fonction d'évaluation. Dans cette section, cette fonction est très simple : plus la position candidate est proche, plus la valeur de la fonction est grande.

6.2.1 Évaluation de mon modèle de confiance TrustSets

Pour tester les performances de l'approche proposée, je teste l'influence de différents paramètres de la simulation sur les résultats et je compare mon approche avec d'autres modèles de la littérature :

- Un modèle utilisant la logique floue proposé de [Kim and Seo, 2008]
- Un modèle probabiliste proposé dans [Zia, 2008]
- Un modèle basé sur le graphe TrustNet proposé dans [Schillo et al., 2000]

6.2.1.1 Présentation des modèles

[Kim and Seo, 2008] a proposé un modèle de confiance utilisant la logique floue et visant à établir des communications fiables entre une source et les nœuds de destination dans un réseau de capteurs sans fil. Plus précisément, ils ont proposé un modèle de capteurs logique floue qui prend en compte à la fois la fiabilité et la non-fiabilité des capteurs pour distinguer les capteurs appropriés et les capteurs anormaux. Les capteurs anormaux peuvent attaquer et contaminer le réseau de capteurs sans fil. La confiance est utilisée comme une agrégation de consensus étant donné un ensemble d'interactions passés entre les capteurs. En utilisant les valeurs de confiance et de méfiance combinées à l'aide de certains opérateurs flous, les nœuds peuvent calculer le niveau d'évaluation de tout réseau de capteurs et décider de communiquer ou non avec un nœud donné (par exemple, si un nœud a une valeur de confiance élevée, les autres nœuds peuvent lui faire confiance pour envoyer et recevoir des données de manière fiable).

[Zia, 2008] a proposé une approche de la gestion de la confiance basée sur la réputation en utilisant des votes de confiance pour établir des confiances entre les nœuds d'un réseau. Tous les nœuds effectuent les trois tâches suivantes : la délivrance d'un vote de confiance, le maintien d'une table de confiance et le report d'un vote non fiable. Au cours de la première tâche, la valeur du vote de confiance est augmentée à chaque transmission d'un message avec succès d'un nœud à l'autre. Plus le vote est haut, plus la valeur de confiance est grande. Cette valeur de confiance est compromise lorsqu'un nœud voisin envoie un vote négatif à un nœud particulier. Si le vote négatif atteint un seuil prédéterminé, ce nœud sera déclaré comme

un nœud non fiable. Dans la deuxième tâche, chaque nœud construit une table de confiance contenant la réputation de tous les nœuds dans un *cluster* (tous les nœuds du réseau sont groupés dans plusieurs *clusters* différents). Les données dans ce tableau contient les identifications des nœuds, le nombre des votes fiables ou défectueux. Les nœuds vont mettre à jour ce tableau à chaque fois qu'ils observent une transmission de message. Dans la tâche finale, une fois que le nombre de votes défectueux atteint un certain seuil, l'information est diffusée à tous les nœuds voisins, afin que le nœud non fiable soit marqué. Lorsque le chef du cluster a des informations sur le nœud non fiable, il va l'écarter de son cluster et rejeter tous les messages envoyés par ce nœud. Le chef du cluster diffuse aussi un message informant que ce nœud a été isolé, de sorte que n'importe quel message envoyé par ce nœud est immédiatement rejeté par ses nœuds voisins. Par conséquent, le nœud non fiable a été effectivement isolé et retiré du réseau.

[Schillo et al., 2000] introduit la notion de TrustNet pour évaluer la confiance sur les autres agents en se basant sur un exemple classique de la théorie des jeux (dilemme du prisonnier). Chaque agent a un caractère pouvant varier entre altruiste et égoïste. Au cours du processus un agent annonce à l'avance le coup qu'il va jouer. Chaque agent possède un TrustNet qui est un graphe dirigé dont l'agent est la racine et les nœuds représentent les autres agents. Deux types d'informations sont collectées pour être associées à tous les nœuds (sauf la racine) : la confiance et le degré d'altruisme. Ce graphe est construit grâce aux témoignages d'autres agents et aux observations de l'agent lui-même. Les agents évaluent les autres à l'aide de réputations représentées par des probabilisés qui sont stockées dans le TrustNet. La valeur finale de confiance d'un agent vers un autre est donc un agrégat d'expériences directes et de témoignages.

J'ai choisi les modèles ci-dessus à comparer avec mon approche pour trois raisons. Premièrement, ces approches peuvent être facilement adaptées à mon exemple d'application Danger Mapping : la confiance est une agrégation de consensus issus d'un ensemble d'interactions passés entre les robots et les robots doivent partager toutes leurs données : celles qu'ils collectent eux-mêmes (directes) et celles qu'ils reçoivent des autres robots (indirectes). De plus, étant dans le contexte d'un système distribué sans contrôle centralisé ou base de données commune des informations, la communication s'effectue essentiellement de manière directe entre deux robots. Un système de réputation centralisé classique ne peut pas être utilisé dans ce contexte. Pour finir, ces trois modèles sont fondés sur des approches différentes (probabilités, logique floue ou graphes) ce qui offre une bonne variété de modèles capables de détecter les robots défectueux.

6.2.1.2 Expérimentations

La figure 6.5 représente le temps maximal d'exploration nécessaire pour que tous les robots fiables obtiennent une carte exacte suivant les 4 modèles de confiance étudiés. Toutes les simulations sont lancées dans le même environnement, c'est-à-dire que la position des zones dangereuses et la position initiale des robots sont identiques pour toutes les simulations. Pour chaque simulation, je mesure le nombre de pas de temps nécessaires pour atteindre la fin de la simulation, c'est-à-dire quand tous les robots fiables ont obtenu une carte identique à la carte réelle (tous les endroits dangereux sont positionnés correctement et avec le bon niveau de danger). J'ai constaté que l'utilisation de la stratégie TrustSet pour détecter les robots défectueux est plus efficace que celles des autres modèles. En particulier, le temps pour que tous les robots du système obtiennent la carte réelle est le meilleur, ce qui montre que mon approche permet de détecter mieux et plus rapidement les agents défectueux. On constate de plus que cette constatation est vraie pour des taux des robots défectueux différents.

On constate que, quand le pourcentage des robots défectueux est inférieur à 0.5, les performances en terme de temps de simulation pour les 4 modèles sont très proches. Par contre à partir de 0.6, quand le nombre des robots défectueux augmente de plus en plus dans système, l'utilisation de mon modèle TrustSet avec des tendances peut contrer mieux la perturbation du système que les autre modèles.

6.2.2 Évaluation de différentes stratégies de communication

La figure 6.6 représente le temps maximal d'exploration nécessaire pour que tous les robots fiables obtiennent une carte exacte. Elle reprend les résultats de la Figure 6.5 en y ajoutant des courbes représentant des stratégies suivantes :

– les robots ne communiquent pas

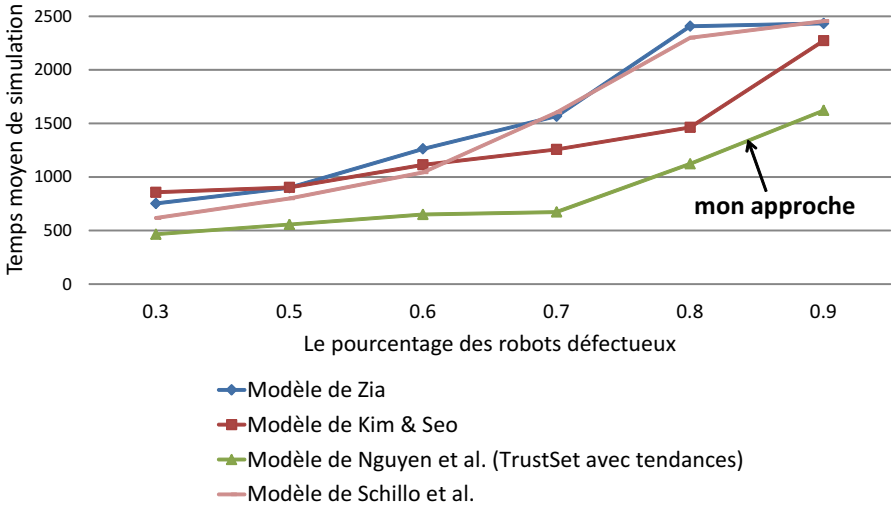


FIGURE 6.5 – Nombre de pas de temps maximal d'exploration nécessaire pour obtenir une carte réelle avec les 4 modèles de confiance.

- les robots ne communiquent que des données
- les robots communiquent des données et les TrustGraphs mais gardent privées les TrustTables (utilisation de seuils ou de tendances)
- les robots communiquent les données et les TrustTables

Dans la stratégie suivant laquelle les robots communiquent les données et les TrustTables, une table simple est utilisée pour stocker des valeurs de confiance sur les autres robots. Ces valeurs sont calculées comme suit : lorsque le robot A rencontre le robot B, il calcule d'abord sa confiance directe en B (DT_{AB}) comme l'étape 1 dans la processus de la fusion des TrustGraphs (4.3.2.3). A met ensuite à jour ses valeurs de confiance propres à tous les robots rencontrés par B ($Agts_B$) par la formule :

$$T_{AX} = \frac{T_{AX}^{old} * T_{AA} + DT_{AB} * T_{BX}}{T_{AA} + DT_{AB}} \tag{6.1}$$

avec $X \in Agts_B$ et T_{AX}^{old} qui représente une confiance ancienne de A en X.

Je constate sur la Figure 6.6 que la stratégie de communication utilisant la confiance calculée en utilisant le TrustSet est la plus efficace.

En particulier elle reste plus adaptée même lorsque le nombre d'agents défectueux et donc d'informations fausses circulant dans le système augmentent.

Sur la figure 6.7, j'ai comparé la quantité d'informations collectées directement par les robots utilisant des stratégies différentes en fonction du pourcentage de robots défectueux. On constate donc que lorsque le système est peu perturbé (30% de robots défectueux), les agents utilisant mon approche n'ont besoin de collecter qu'un tiers de la quantité d'informations qu'ils devraient récolter s'il cartographiaient sans communiquer. Même lorsque le système est très perturbé (70% de robots défectueux), l'utilisation du TrustSet permet aux agents de n'avoir à collecter que 50% de la quantité d'informations nécessaires s'ils ne communiquaient pas, ceci est par ailleurs meilleur que les autres stratégies.

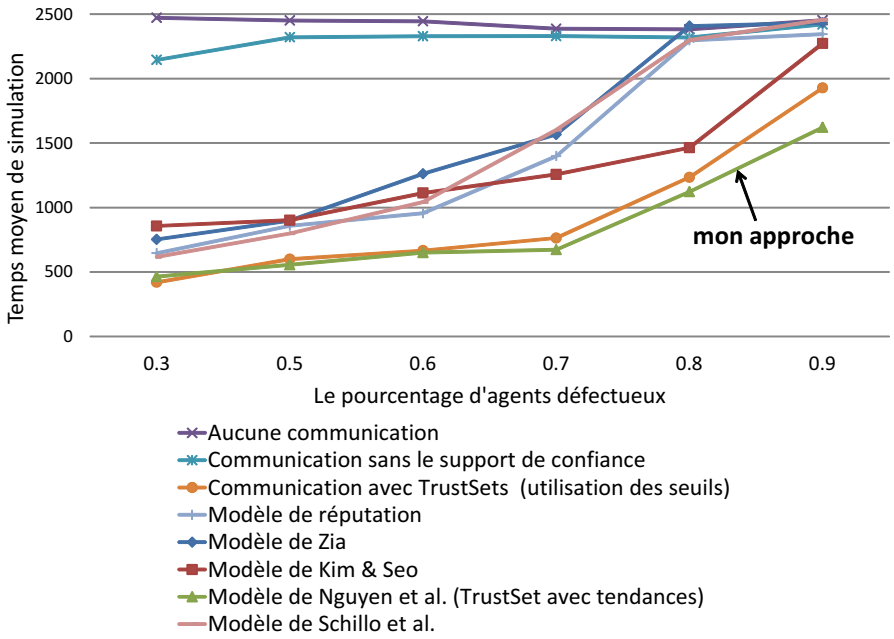


FIGURE 6.6 – Nombre de pas de temps maximal nécessaire pour obtenir une carte réelle avec 8 stratégies de communication.

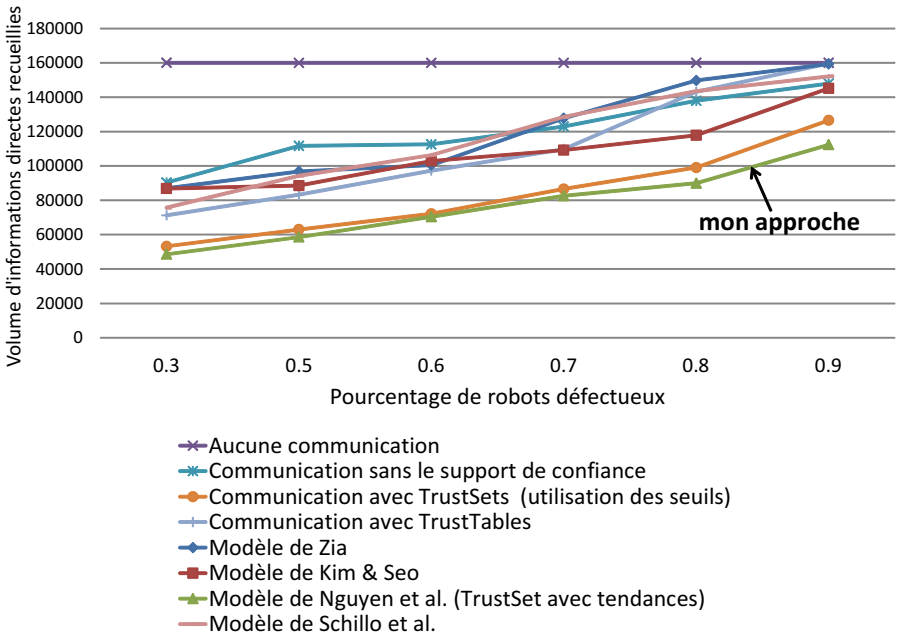


Figure 6.7: Volume d'informations directes recueillies pour obtenir une carte réelle avec 8 stratégies de communication

Dans la figure 6.8, on présente des courbes similaires dans le cas où la fiabilité des robots peut changer au cours de la simulation : tous les robots sont initialement fiables, puis quand certains ont détecté 50% de l'environnement, ils deviennent défectueux jusqu'à la fin de simulation. Ces agents, dits intermittents sont choisis à l'initialisation de la simulation. Dans cette expérience, on fait varier le taux de ces agents intermittents entre 0.3 et 0.9. On constate que mon approche fournit à nouveau de meilleurs résultats

que les autre stratégies.

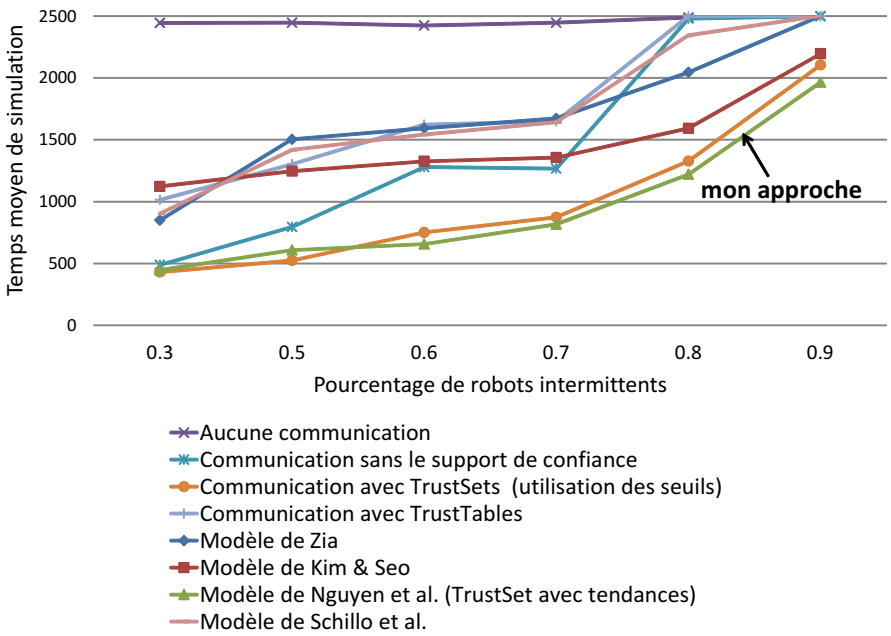


Figure 6.8: Nombre de pas de simulation pour obtenir une carte réelle avec 8 stratégies de communication, lorsque la simulation contient des robots intermittents.

La figure 6.9 illustre le temps moyen de détection pour toutes les stratégies de communication, c'est-à-dire le temps moyen nécessaire aux robots fiables du système pour détecter correctement la carte. Pour obtenir ce résultat, j'ai lancé 50 simulations pour lesquelles le nombre de robots défectueux varie de 1 à 45 pour un nombre total de robots fixe. Le graphique de la figure 6.9 souligne que je peux utiliser la stratégie TrustSets avec les tendances pour obtenir de bonnes performances. Plus généralement, lorsque les robots n'ont aucune idée de la proportion de robots défectueux, ce graphique montre que la meilleure stratégie de communication reste la communication avec TrustSets en utilisant les tendances.

6.2.3 Comparaison de l'utilisation du seuil et de la tendance dans la stratégie de communication

Le système de confiance a des impacts négatifs ou positifs sur la communication du système. Quand un robot sait que d'autres robots lui envoient de fausses informations, pourquoi devrait-il persister dans la communication avec eux ? Il est préférable pour ce robot de cesse de communiquer avec les robots jugés défectueux. A cet effet, de nombreux auteurs ont proposé une stratégie de communication qui utilise un seuil de communication (par exemple un robot décide d'arrêter de communiquer avec un autre robot si la valeur de confiance calculée pour ce robot descent sous une valeur seuil) [Kim and Seo, 2008, Zia, 2008]. Mais l'utilisation du seuil peut être très négative dans le cas où le comportement des robots peut changer au cours de la simulation (dans les cas par exemple où les agents sont dans des situations d'interactions répétées et s'adaptent, ou lorsque leur comportement est intermittent). En utilisant le seuil, la valeur de confiance évolue initialement pour classer le robot dans fiables ou non fiables mais les robots ne peuvent pas regagner la confiance de robots qui ne leur font plus confiance.

Pour éviter cet effet négatif, j'ai proposé d'utiliser des tendances pour la communication : les robots ont tendance à s'attirer ou à se repousser de façon proportionnelle à leurs valeurs de confiance comme indiqué dans la section 5.3. En utilisant la communication de façon proportionnelle à la valeur de confiance du transmetteur de l'information, il est possible de rétablir la confiance dans un robot en qui on n'avait plus confiance, et qui peut changer en cours du temps. Si un robot jugé comme défectueux ne change pas,

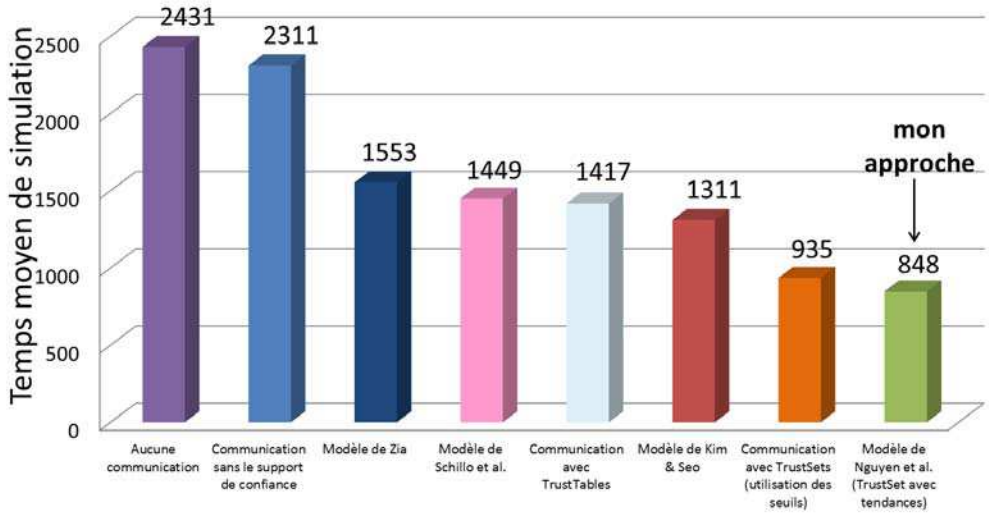
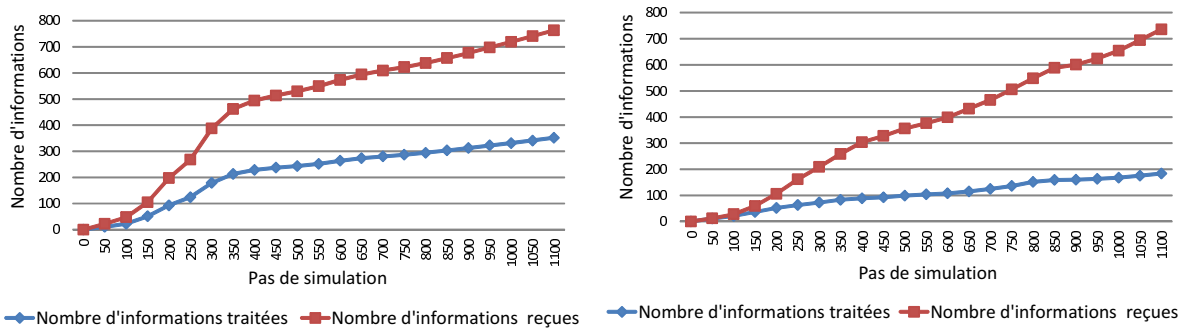


FIGURE 6.9 – Comparaison du nombre moyen de pas de temps pour obtenir une carte réelle avec 8 stratégies de communication

il finira par être écarté. Ceci permet d'avoir une acquisition de nouvelles informations (sur des territoires non visités) même avec peu de confiance.

La figure 6.6 ci-dessus montre que l'utilisation d'une stratégie de communication basée sur la confiance avec TrustSet et utilisant la tendance diminue le temps nécessaire en comparaison avec la stratégie de communication utilisant le seuil, même dans le cas où il y a beaucoup des robots défectueux. En effet, dans un environnement perturbé par de nombreux robots défectueux, la communication basée sur la tendance offre la possibilité de communiquer et d'utiliser des informations correctes envoyées par des robots peu fiables, ce qui n'est pas le cas lorsque les robots utilisent la communication basée sur des seuils.

Cependant, je peux voir sur les Figures 6.10(a) et Figure 6.10(b) (le pourcentage des robots fiables étant égal à $0,8$), que l'utilisation d'une stratégie de communication avec des seuils peut limiter les informations envoyées aux robots fiables plus efficacement qu'avec la stratégie utilisant la tendance. Sur la Figure 6.10(a), on compare le ratio des informations reçues et celui de celles traitées en utilisant la stratégie de communication par seuils. Les informations traitées sont les informations reçues et utilisés pour mettre à jour sa propre carte et calculer la confiance dans les autres robots. Je note que le pourcentage des informations reçues par rapport aux informations traitées avec le seuil est de $0,25$ en comparaison avec une valeur de $0,45$ en utilisant la stratégie de communication avec la tendance. La performance est de $1,8$ fois supérieure en utilisant la stratégie de communication avec la tendance (environ 751 informations reçues pour 350 informations traitées sur la figure 6.10(b) au bout de 1100 pas de simulation). Je considère que cette surcharge est acceptable étant donné les meilleures performances et la flexibilité qu'apporte la stratégie utilisant la tendance.



(a) la stratégie de communication utilisant seuils

(b) la stratégie de communication utilisant tendances

Figure 6.10: Nombre d'informations reçues (en rouge) par rapport au nombre d'informations traitées (bleu), en utilisant la stratégie de communication avec seuils 6.10(a) et avec la tendance 6.10(b).

6.3 Analyse des organisations émergentes

Dans cette partie, je vais analyser et évaluer l'approche auto-organisationnelle présentée au chapitre 5. Je me déplace dans les mêmes conditions expérimentales que précédemment : dans un environnement continu de 400m x 400m, j'ai créé 200 obstacles représentant des zones dangereuses et 50 robots autonomes (avec 20 robots défectueux). Les robots se déplacent et communiquent selon l'approche auto-organisationnelle basée sur la confiance (suivant l'approche proposée dans le chapitre 5). Toutes les simulations sont lancées dans le même environnement, c'est-à-dire que les positions des zones dangereuses et les positions initiales des robots sont identiques pour toutes les simulations. Les résultats de simulation présentés sont réalisés pour une moyenne de 10 simulations différentes afin d'obtenir des valeurs fiables. L'objectif de cette partie est de montrer la capacité d'auto-organisation de mon modèle à travers l'émergence de comportements dynamiques (rôles).

6.3.1 Présentation de l'auto-organisation observée sans contrôle

Comme je l'ai présenté dans la partie 5.5.4, il existe deux sortes d'auto-organisation que j'appellerai simple ou dirigée. Dans cette partie, je présente les résultats associés à l'auto-organisation sans contrôle. Celle-ci laisse les agents manipuler les règles que l'utilisateur lui a fournies et développer des comportements qui, une fois observés avec les instruments de mesures adéquats et logiquement réunis, pourront dessiner pour chaque agent ou groupe d'agents un rôle identifiable par l'utilisateur qui sera apparu par émergence.

6.3.1.1 Évaluation du modèle de confiance dans l'approche auto-organisationnelle

Afin d'évaluer le modèle de confiance dans l'approche auto-organisationnelle, je considère deux indicateurs : la confiance du robot en lui-même et la confiance de la communauté en ce robot (sa réputation). La valeur initiale de la confiance en soi est de 1.0 et celui de la réputation est de 0.5. Plus les robots font confiance à un robot, plus la réputation du robot est augmentée, et vice-versa. Plus un robot reconnait que de nombreux robots lui font confiance, plus son auto confiance est augmentée, et vice-versa.

Les résultats présentés dans la figure 6.11 montrent l'évolution de la réputation et de l'auto confiance pour deux groupes de robots fiables et défectueux. En commençant à la même valeur initiale de 1.0, l'auto-confiance des robots fiables ne varie pas beaucoup tandis que cette valeur pour les robots défectueux diminue rapidement pour atteindre la valeur de leur réputation. De la même manière, la réputation des

robots fiables augmente rapidement pour atteindre la valeur de leur auto-confiance tandis que cette valeur pour les robots défectueux diminue progressivement au cours du temps.

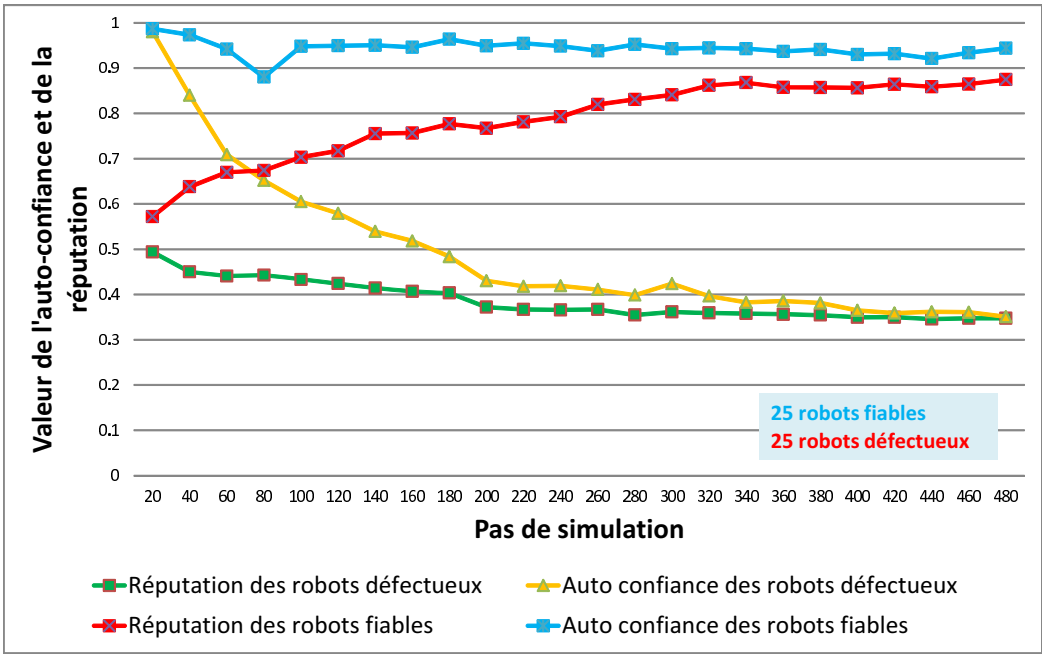


Figure 6.11: Évolution de la réputation et de l'auto-confiance les robots fiables et défectueux en fonction du temps

La figure 6.12 représente pour chaque robot son auto-évaluation (confiance en lui-même) en fonction de sa réputation. On constate que la plupart des robots fiables (resp. robots défectueux) sont représentés par des points sur le coin en haut droite (resp. bas gauche) de la figure. Cela signifie que les robots fiables (resp. robots défectueux) ont une réputation et une auto-confiance élevée (resp. faible).

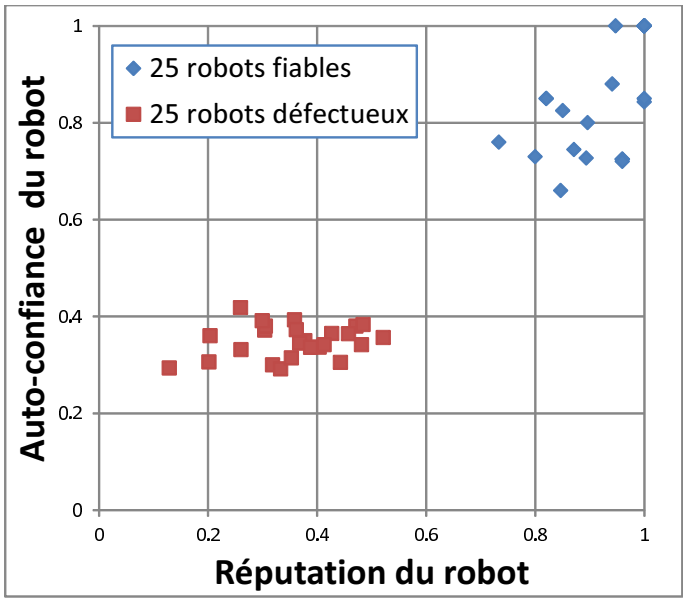
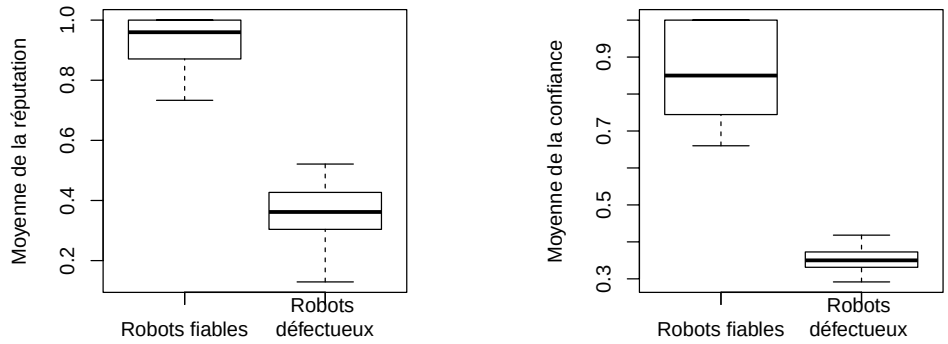


Figure 6.12: Émergence des robots fiables et défectueux

Au niveau de la réputation, la différence à la fin de la simulation entre les deux groupes est significative ($M(robots\ fiables) = 0.9319482$, $M(robots\ défectueux) = 0.3569736$, $p < .001$, voir figure 6.13(a)). On observe le même phénomène pour la différence sur l'auto-confiance entre ces 2 groupes ($M(robots\ fiables) = 0.867583$, $M(robots\ défectueux) = 0.349114$, $p < .001$, voir figure 6.13(b)).



(a) Réputation des robots

(b) Auto-confiance des robots

Figure 6.13: Réputation (figure 6.13(a)) et auto-confiance (figure 6.13(b)) des robots fiables et défectueux à la fin de la simulation.

6.3.1.2 Émergence au niveau du système de collecte d'informations

Parmi les rôles émergents au cours du processus d'auto-organisation, j'ai considéré l'un des rôles les plus simples : le rôle d'explorateur. Je l'utilise comme indicateur du nombre de zones détectées pour chaque robot. Un robot peut être considéré comme un explorateur si le nombre de zones qu'il a détectées est plus élevé que le nombre moyen de zones détectées par tous les robots dans le système.

Les résultats sont présentés sur la Figure 6.14 : chaque point représente le nombre de zones détectées pour chaque agent. On associe aux agents au-dessus de la ligne moyenne. Le nombre de zones détectées par les explorateurs est significativement plus élevé que celui des autres ($M(explorateur) = 52207$, $M(autres) = 22076$, $p < 0.001$, sur la figure6.15). J'utilise la fonction *t-test* pour tester la différence entre des ensembles de valeurs. Par conséquent, si la valeur de probabilité *p-value* est inférieur à *0.05*. Je pourrais conclure que les deux ensembles sont *significativement différents*.

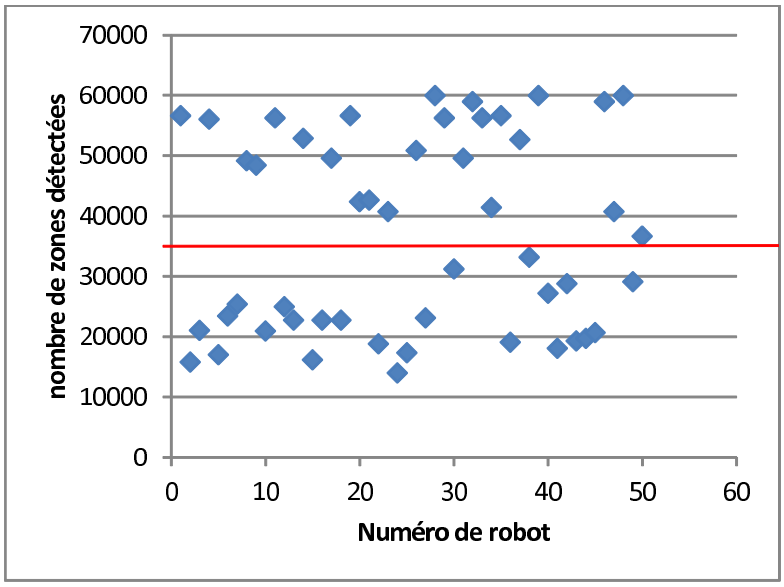


Figure 6.14: Émergence du rôle explorateur

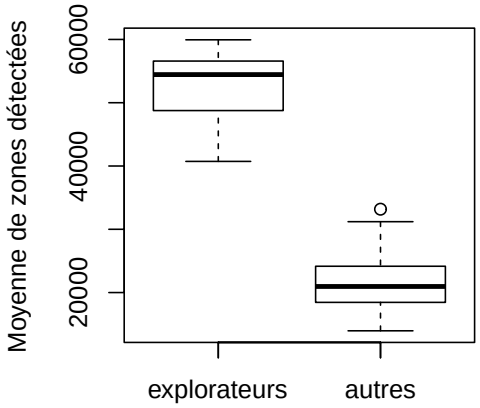


Figure 6.15: Écart entre le nombre de zones détectées par les explorateurs et par les autres robots

La figure 6.16 représente l'émergence de *clusters* dans le système. Ce graphique est construit par les rencontres entre les agents en fonction du temps où l'axe horizontal représente le temps et l'axe vertical représente le numéro de robot. Les robots restent proches les uns des autres qu'ils sont regroupés dans toutes les zones de la même couleur. Les agents défectueux sont représentés par des rouges.

Pour l'apparition des *clusters*, on utilise les paramètres suivants :

– $w1 = 0.1, w2 = 0.1, w3 = 0.1, w4 = 0.7$. Plus $w4$ est grand, plus les robots auront tendance à se déplacer en vue de communiquer plutôt qu'à se déplacer en vue de découvrir de nouvelles données. En effet, chaque robot aura tendance à favoriser la rencontre des autres robots plutôt que le déplacement vers les points de sa frontière. Cela devrait lui permettre à chaque robot de rester près des robots qu'il juge fiables en vue de favoriser une communication plus sure.

- $w5 = 0.8, w6 = 0.1, w7 = 0.1$. Plus $w5$ est grand, plus les robots vont privilégier la communication avec les robots fiables.
- $w8 = w9 = w10 = 1/3$. Rappelons que plus $w8$ est grand, plus les robots ont tendance à rester à proximité des robots fiables. Plus $w9$ est grand, plus les robots tentent à se rapprocher des robots qui viennent de leur transmettre des informations directes fiables. Plus $w10$ est grand, plus les robots privilégient la nouveauté des informations qui leur sont apportées et se rapproche des robots capables d'enrichir substantiellement leur bases de données. La priorité de ces trois tendances est la même avec les poids égaux.

On constate que :

- Les robots fiables ont tendance à privilégier la communication avec les robots fiables ce qui a pour effet de favoriser la formation de coalitions des robots fiables (apparition de *clusters*).
- Les robots défectueux (rouges sur la figure 6.16) ne communiquent pas beaucoup avec les robots fiables, cela permettant de rendre le système plus robuste en écartant ces robots.

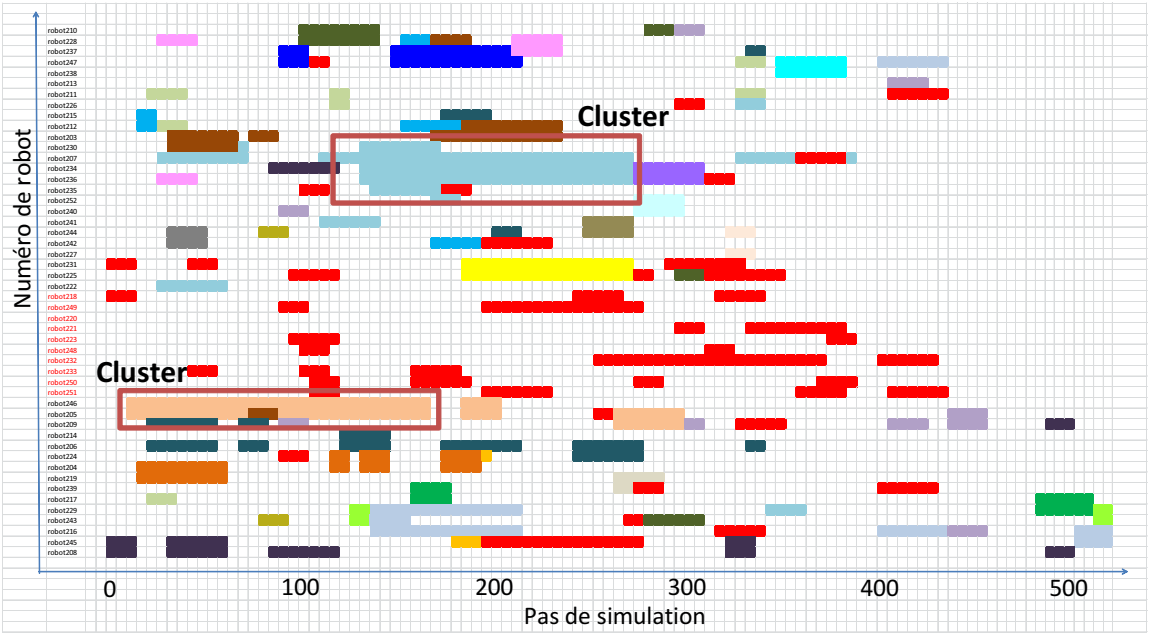


Figure 6.16: Émergence des clusters au niveau du système de déplacement

Il y a plusieurs *clusters* qui apparaissent durant la simulation. On peut observer deux *clusters* émergeants pendant long-temps au niveau spatial : une proximité sociale se traduit par une proximité physique. Il disparaissent plus tard à cause de l'influence des règles favorisant la communication avec des robots déconnectés ou de la règle favorisant le renouvèlement de l'information même si en cas de coefficients sont faibles ($w6 = 0.1, w7 = 0.1$). Un exemple d'émergence d'un cluster au niveau spatial est observé sur la simulation comme dans la Figure 6.17. Dans cette figure, les obstacles sont représentés par les carrés colorés (chaque couleur représentant le différent niveau de danger). Les cercles rouges représentent des agents défectueux, les cercles bleus représentent des agents fiables. Les numéros bleus à côté de l'agent représentent des obstacles collectés par l'agent et les numéros rouges représentent les obstacles corrects vérifiés par le système.

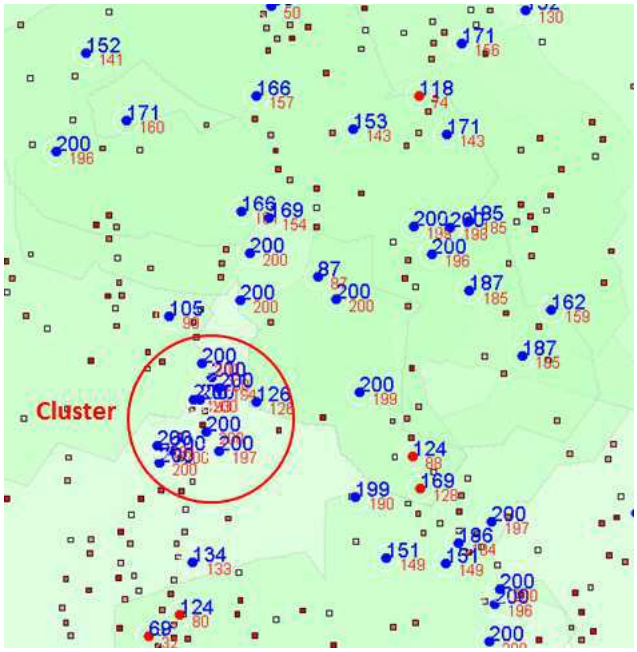


Figure 6.17: Émergence du cluster observé de la simulation.

6.3.1.3 Émergence au niveau du système de communication

Afin d'observer les propriétés émergentes du système de communication, notamment l'émergence de rôles simples tels que *transmetteur* ou *récepteur*, j'ai introduit deux indicateurs : le nombre d'informations envoyées et le nombre d'informations reçues pour chaque robot. Le pourcentage des robots défectueux utilisés dans le système est de 50% (25 robots fiables et 25 robots défectueux). Je vais analyser la tendance à envoyer ou à recevoir des informations pour tous les robots afin de mettre en relief des comportements différents. Ainsi, un robot est considéré comme *transmetteur* s'il a tendance à envoyer plus d'informations qu'il n'en reçoit. Dans le cas contraire, un robot est considéré comme *récepteur* s'il a tendance à recevoir plus d'informations qu'il n'en envoie.

La figure 6.18 représente le nombre d'informations envoyées par rapport au nombre d'informations reçues pour chaque robot à la fin de la simulation. Je peux constater que des robots défectueux ont tendance à envoyer plus d'informations qu'ils n'en reçoivent, jouant le rôle ***transmetteur*** d'informations, tandis que des robots fiables jouent le rôle de ***récepteur*** d'informations puisqu'ils ont tendance à recevoir plus d'informations qu'ils n'en envoient (à cause de l'impact de l'auto-confiance du robot sur la tendance à communiquer des informations). Les transmetteurs (resp. récepteurs) sont représentés par des points qui sont au-dessus (resp. au-dessous) de la ligne médiane rouge. Cependant, globalement les robots défectueux reçoivent moins d'informations que les robots fiables (c'est le cas de tous les robots représentés par des carrés rouges sur la figure 6.18 à gauche de la ligne représentant la moyenne des informations reçues par tous robots). Ce résultat nous montre que les robots fiables ont favorisé la communication avec les robots qu'ils jugent utiles, autrement dit ceux qui le font progresser plus vite vers leurs objectifs. Ces résultats soulignent un comportement assez naturel : plus un robot est fiable (resp. défectueux), plus il a chance d'être impliqué (resp. écarté) de la communication avec d'autres robots.

Afin d'approfondir mon analyse, je compare la différence entre le nombre d'informations envoyées et reçues, suivant la fiabilité des robots. Pour le rôle de transmetteur, le nombre moyen d'informations envoyées est significativement plus élevé que le nombre d'informations reçues (Nombre moyen d'informations envoyées $M(envoi) = 71.657$, Nombre moyen d'informations reçues $M(réception) = 51.300$, $p < .001$, voir figure 6.19(a)). Pour le rôle de récepteur, le nombre moyen d'informations envoyées est significativement plus petit que le nombre d'informations reçues (Nombre moyen d'informations envoyées $M(envoi) = 78.63$, Nombre moyen d'informations reçues $M(réception) = 93.83$, $p < .001$, voir figure

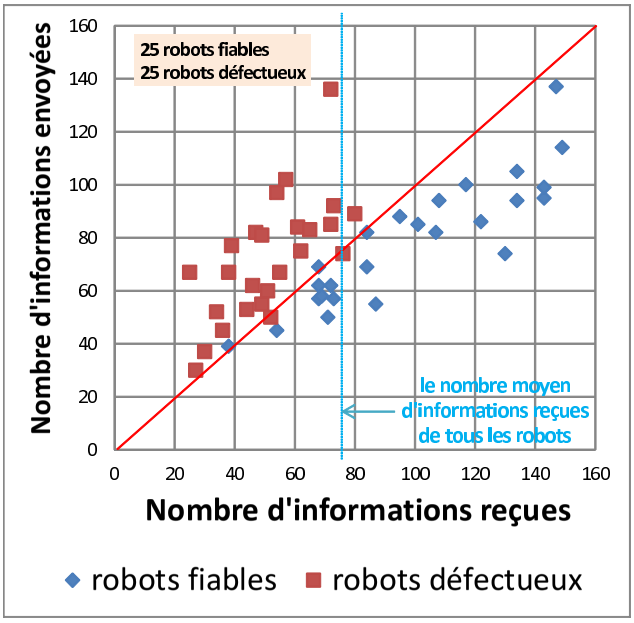
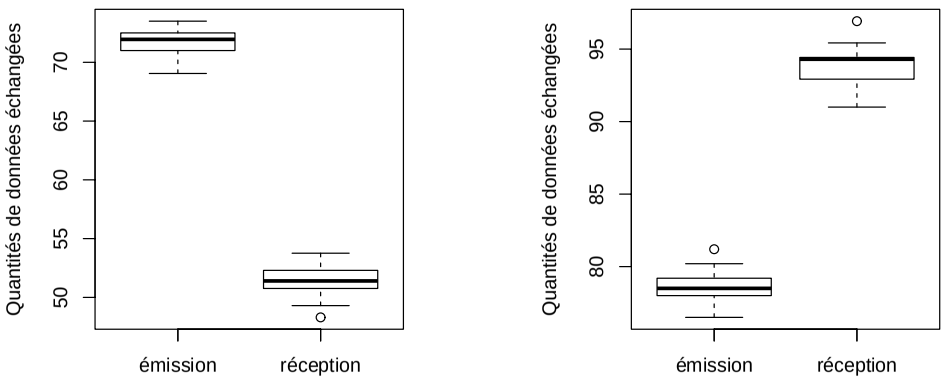


Figure 6.18: Émergence des rôles de transmetteur et d'émetteur

6.19(b)). En d'autres termes, les robots fiables (resp. défectueux) ont tendance à jouer le rôle de transmetteur (resp. de récepteur) rôle qui est un cas significatif de spécialisation émergente.



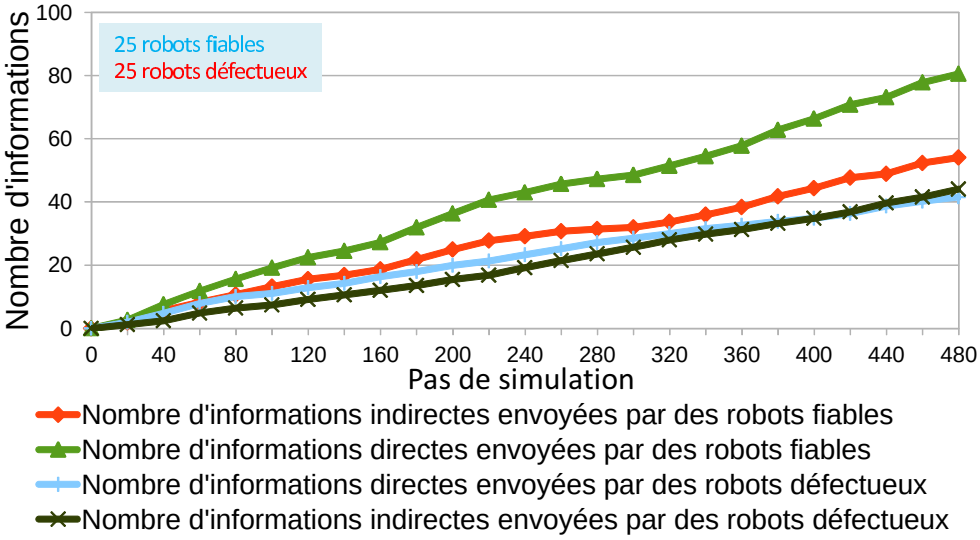
(a) le rôle de transmetteur

(b) le rôle de récepteur

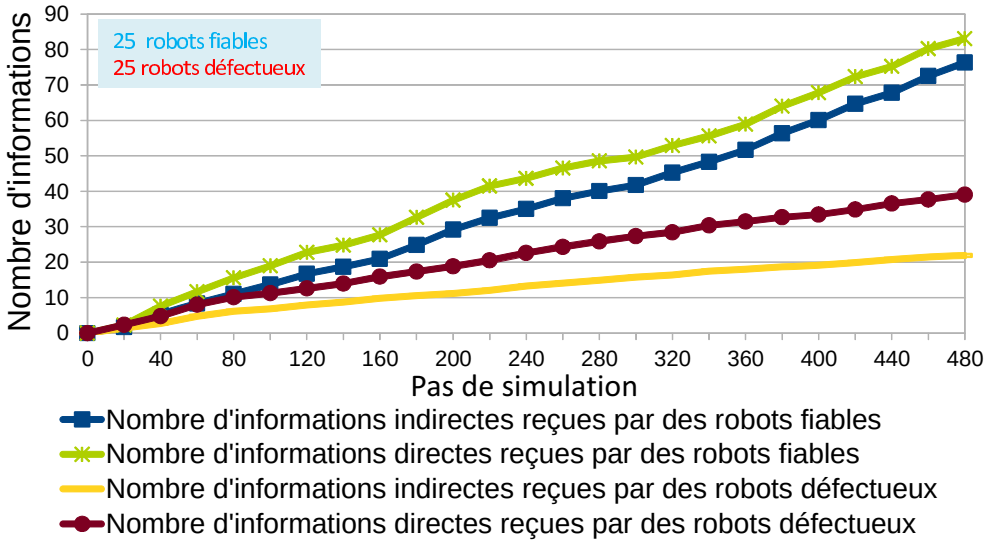
Figure 6.19: Nombre d'informations envoyées et reçues par les rôles de transmetteur 6.19(a) et de récepteur 6.19(b).

La figure 6.20 représente le nombre d'informations directes ou indirectes envoyées et reçues pour 2 groupes de robots : fiables et défectueux.

On constate que les robots fiables ont tendance à envoyer et recevoir plus d'informations directes et indirectes que les robots défectueux. Au niveau de l'envoi des informations, les résultats indiquent que les robots fiables envoient plus d'informations directes que d'informations indirectes. Au contraire les robots



(a) envoi des informations



(b) réception des informations

Figure 6.20: Nombre d'informations directes (et indirectes) envoyées (et reçues) par 2 groupes de robots : fiables et défectueux.

défectueux envoient plus d'informations indirectes que d'informations directes (voir figure 6.20(a)). On peut noter que les robots défectueux envoient plus d'informations directes que d'informations indirectes au début de simulation, puis ensuite envoyer plus d'informations indirectes que d'informations directes. Ce résultat est cohérent avec la règle de communication proposée : plus les informations directes sont fiables, plus le robot a tendance à les envoyer (et inversement).

Au niveau de la réception d'informations, les deux groupes de robots fiables et défectueux reçoivent plutôt des données plus directes que des données indirectes (voir figure 6.20(b)) et les robots fiables reçoivent plus informations directes et indirectes que les robots défectueux.

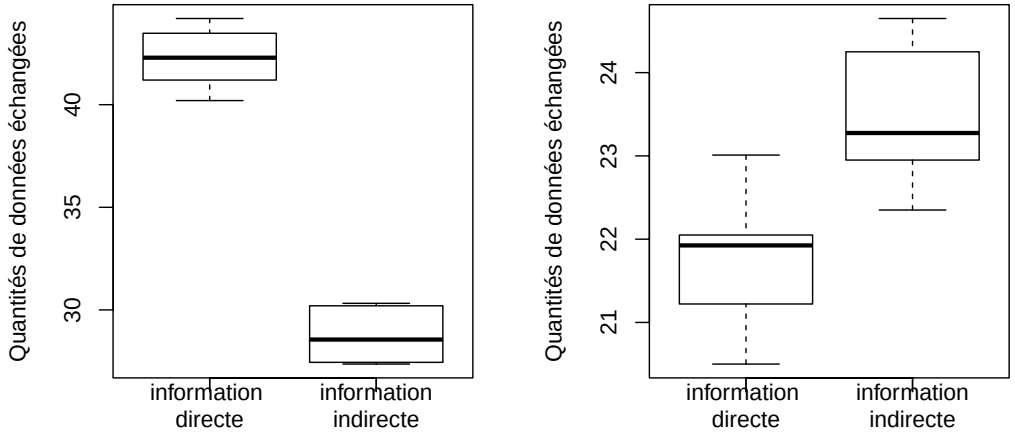
Plus précisément, on peut observer des comportements différents pour chacun des deux types de robots :

– Pour les robots fiables :

Ces robots ont tendance à envoyer et à recevoir plus d'informations directes (courbes bleues) que d'informations indirectes (courbes vertes) car leur confiance en eux-même est plus grande que la confiance envers les autres. On pourrait dire que ces robots jouent le rôle de **transmetteurs d'informations directes** (la moyenne des informations directes envoyées est significativement plus élevée que celle des informations indirectes envoyées, $M(informations \quad directes) = 42.230$, $M(informations \quad indirectes) = 28.814$, $p < 0.001$, voir figure6.21(a)).

– Pour les robot défectueux :

J'ai constaté que ces robots ont tendance à envoyer plus d'informations indirectes que d'informations directes (figure 6.20(a)) grâce au calcul d'auto-évaluation. Les robots défectueux sont capables d'évaluer leur confiance en eux-mêmes correctement, ils ont alors tendance à envoyer des informations indirectes au lieu d'informations directes, qui sont des informations fausses. On pourrait dire que ces robots jouent la rôle de **transmetteur des informations indirectes** (la moyenne des informations directes envoyées est significativement plus petite que celles des informations indirectes envoyées, $M(informations \quad directes) = 21.771$, $M(informations \quad indirectes) = 23.492$, $p < 0.001$, voir figure6.21(b)).



(a) Rôle de transmetteur d'informations directes (b) Rôle de transmetteur d'informations indirectes

Figure 6.21: Nombre d'informations directes et indirectes envoyées. Illustration du rôle de transmetteur d'informations directes 6.21(a) et du rôle de transmetteur d'informations indirectes 6.21(b).

Ce résultat confirme l'idée de la spécialisation dynamiques des robots (et donc de l'émergence de rôle) : les robot fiables ont tendance à explorer et communiquer des informations collectées directement, tandis

que les robots défectueux ont tendance à transmettre des informations reçues plutôt que des informations directes pour limiter la propagation des informations fausses.

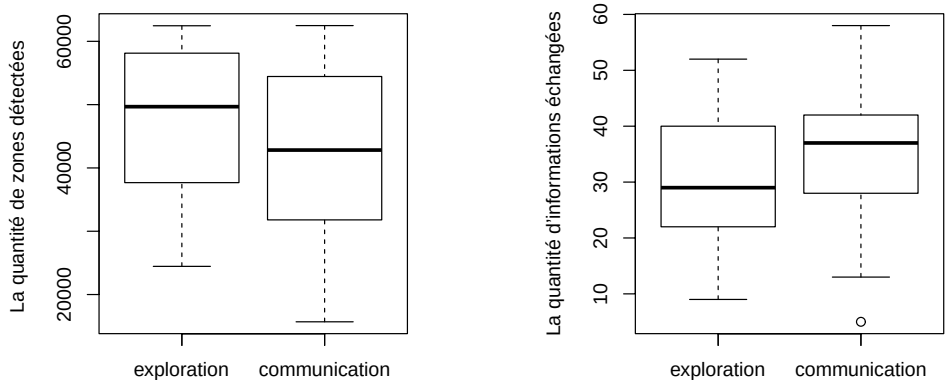
6.3.2 Impact du contrôle sur l'auto-organisation

Afin d'observer l'impact du contrôle des poids $\{w_i\}$ sur les systèmes d'exploration et de communication, je propose les deux scénarios suivants :

J'appelle le premier scénario le scénario d'exploration. Ce scénario utilise $w_1 = w_2 = w_3 = 0.3$ et $w_4 = 0.1$. Plus $w1$, $w2$ et $w3$ sont grands, plus les robots auront tendance à se déplacer afin d'explorer plutôt que de communiquer dans le but de recevoir de nouvelles données. En effet, ceux-ci auront tendance à s'orienter vers les points de la frontière plutôt que de rejoindre d'autres robots. Cela devrait permettre d'éviter la perturbation provenant des informations indirectes corrompues.

J'appelle le deuxième scénario le scénario de communication. Ce scénario utilise $w_1 = w_2 = w_3 = 0.1$ et $w_4 = 0.7$. Plus $w4$ est grand, plus les robots auront tendance à se déplacer pour communiquer plutôt que de se déplacer pour découvrir de nouvelles données. En effet, chaque robot aura tendance à favoriser pour cibles les robots plutôt que les points de la frontière. Cela devrait permettre à chaque robot de rester près des robots qu'il juge fiables afin de favoriser une communication plus riche et plus fiable.

Ces deux scénarios utilisent la même valeur pour les autres poids $w_i = \frac{1}{3}, i = 5..10$. Avec les poids égaux, la priorité des tendances associées sera la même.



(a) Efficacité de la découverte d'information suivant le scénario

(b) Quantité d'informations échangées suivant le scénario

Figure 6.22: Zones détectées 6.22(a) et informations échangées 6.22(b) pour chaque scénario.

Pour évaluer ces scénarios, j'utilise deux indicateurs : le nombre d'informations envoyées et reçues, et le nombre de zones détectées pour chaque robot.

Comme prévu, il y a des différences significatives dans les deux indicateurs qui sont mesurés pour deux scénarios. Le nombre de zones détectées dans le premier scénario est sensiblement plus élevé que dans le second ($M(exploration) = 48102$, $M(comunication) = 42459$, $p < .04$, voir figure6.22(a)). Inversement, le nombre d'informations envoyées et reçues dans le premier scénario est sensiblement plus faible que dans le second ($M(exploration) = 30.34$, $M(communication) = 35.78$, $p < .05$, voir figure6.22(b)). On peut donc en déduire que les poids (w_1, w_2, w_3, w_4) de l'équation (5.9) permettent de réguler et de contrôler les phénomènes émergents de façons appropriés.

6.4 Optimisation de l'auto-organisation

6.4.1 Objectif

On a proposé un modèle d'auto-organisation à base de règles : règles de déplacement, de communication et de rétro-action entre la communication et le déplacement. Chaque règle a une importance différente qui est représentée par un poids. L'objectif de cette section est de proposer une méthode visant à optimiser le fonctionnement du système en jouant sur les poids des règles pour trouver la meilleure (dans la mesure du possible) stratégie de déplacement et de communication pour les robots.

Rappelons que le comportement d'un robot est caractérisé par la valeur d'un ensemble de poids $\{w_1, w_2, .., w_{10}\}$. Dans cette partie, on propose d'appliquer un algorithme génétique [3] pour trouver la meilleure combinaison de poids des règles locales (de w_1 à w_{10}) en fonction de la vitesse de collecte des données et de la quantité de données collectées de manière directe.

6.4.2 Implémentation de l'algorithme génétique

6.4.2.1 Expression d'un gène

Je considère une combinaison des 10 poids sous la forme du 10-uplet (w_1,w_2,...w_{10}) comme représentant un individu. Pour construire un 10-uplet (w_1,w_2,...w_{10}) je pars d'un 10-uplet (x_1,x_2,..x_{10}) avec $x_i \in [0, 15]$ (donc en utilisant 4 bits pour représenter chaque x_i) et je produis les w_i en respectant les contraintes suivantes :

- $w_i = \frac{x_i}{x_1+x_2+x_3+x_4}, i = 1..4$
- $w_j = \frac{x_j}{x_5+x_6+x_7}, j = 5..7$
- $w_k = \frac{x_k}{x_8+x_9+x_{10}}, k = 8..10$

satisfaisant des contraintes $w_1 + w_2 + w_3 + w_4 = 1$, $w_5 + w_6 + w_7 = 1$ et $w_8 + w_9 + w_{10} = 1$ dans les équations étudiées (5.9, 5.16 et 5.23).

Par conséquent, un individu (w_1,w_2,...w_{10}) est représenté par une chaîne de 40 bits (correspondant à 10 nombres x_i de 4 bits). L'ordre des bits dans le gène correspond à l'ordre des nombres x_i. L'espace des combinaisons de 10 nombres x_i est $2^{4*10} > 1$ billard de solutions (à l'exception de certains gènes redondants). Il est donc nécessaire d'utiliser un algorithme génétique pour explorer cet espace.

6.4.2.2 Fonction d'utilité

Un individu est considéré comme bon si : (i) il collecte des informations plus fiables que les autres, et (ii) il collecte des informations fiables plus rapidement que d'autres. Donc, je définis la fonction d'utilité de la façons suivante :

$$f(i) = \frac{Data_i}{t_i} \tag{6.2}$$

où $Data_i$ est la quantité d'informations fiables que le robot i a collectées; t_i est la durée de la simulation.

6.4.2.3 Scénario de simulation

- **Initialisation.** La population est initialisée au hasard avec 68 individus. Chaque individu a un gène différent (donc un 10-uplet de poids différent).
  - Le nombre de robots défectueux est égal à 5 (7%). Comme pour les autres robots, chacun a son propre gène.
  - Pour chaque génération, je lance la simulation 10 fois, puis prends la moyenne de la fonction d'utilité pour chaque individu.

3. Les algorithmes génétiques appartiennent à la famille des algorithmes évolutionnistes. Leur but est d'obtenir une solution approchée à un problème d'optimisation, lorsqu'il n'existe pas de méthode exacte (ou que la solution est inconnue) pour le résoudre en un temps raisonnable [Holland, 1962].

- **Évolution.** Seuls les 30 meilleurs individus sont conservés pour la génération suivante. Ils ont les parents la prochaine génération via des opérations de croisement et de mutation.
  - **Croisement**. Dans la prochaine génération, il y a 30 individus de la génération précédente. Ils se croissent en 15 paires sélectionnées au hasard et ils créent 30 nouveaux individus.
  - **Mutation.** Il y a une certaine mutation des 30 parents avec un taux de mutation environ 10%. Donc, il y a 3 nouveaux individus issus de la mutation. Chaque gène possède 40 bits, de sorte que le nombre de bits mutés est de 10 bits. Ils sont choisis au hasard pour muter.
  - Les 5 défectueux crées à l'initialisation sont les mêmes que ceux de la génération précédente. En conséquence, il y en tout 30 parents + 30 enfants + 3 mutés + 5 défectueux = 68 individus dans la prochaine génération.
- **Arrêt de l'algorithme génétique.** L'algorithme s'arrête lorsque dans deux générations consécutives, les 30 meilleurs individus de la population ne changent pas.

Le taux de mutation choisi α est de 10%. Une simulation s'arrête quand il y a un agent fiable qui a détecté correctement tous les obstacles sur la carte. Une valeur trop élevée des agents fiables dans l'arrêt de l'algorithme ralentie le temps d'exécution de la simulation. J'ai choisi un seul agent pour le temps de calcul de la méthode choisie est plus bas et raisonnable (car il aurait fallu exécuter un scénario pour chaque gêne sinon, ce qui auraient été très long).

6.4.3 Quelques résultats

6.4.3.1 L'évolution à travers les générations

La figure 6.23 représente l'évolution de la moyenne des valeurs d'utilité des individus sélectionnés et de toute la population génération par génération. Les résultats indiquent que la valeur maximale d'utilité est atteinte lors de la 10^e génération. Mais je continue à lancer l'algorithme génétique jusqu'à la 30^e génération afin de vérifier que je ne suis pas sur un maximum local.

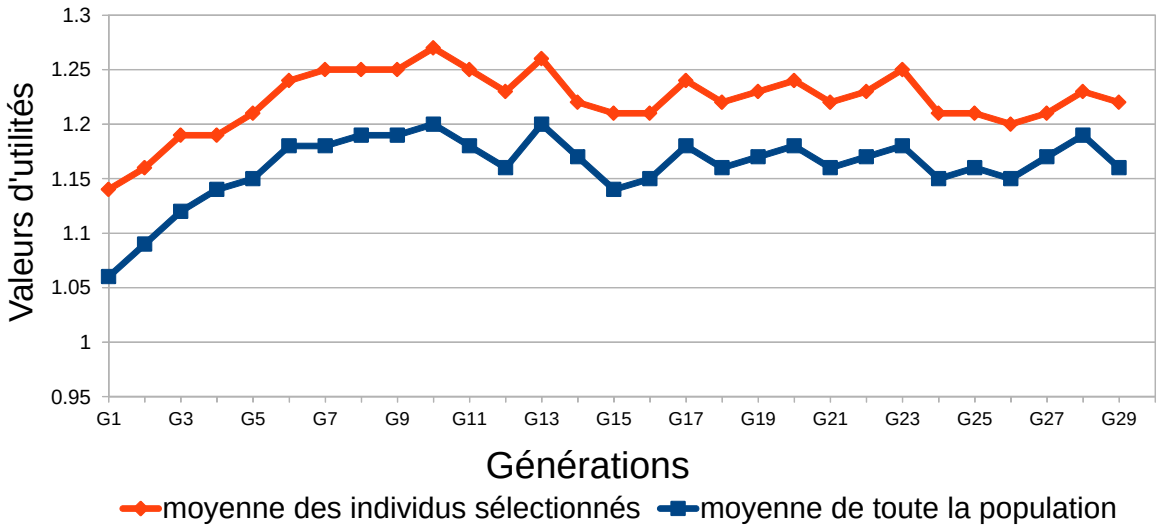
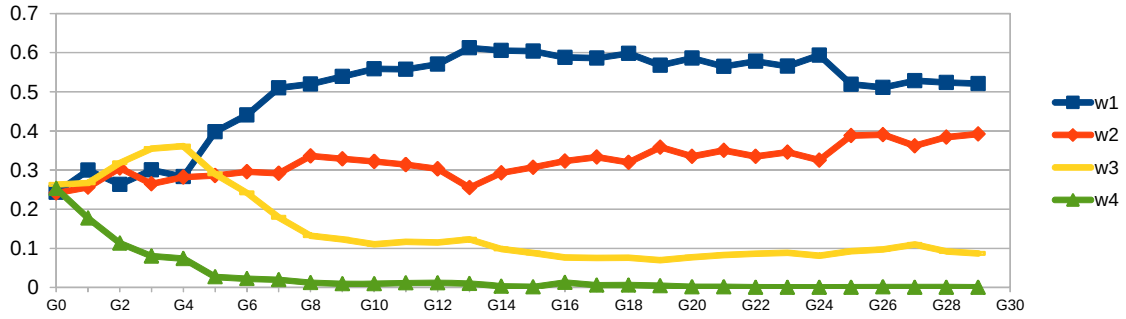


FIGURE 6.23 – Évolution de la moyenne des valeurs d'utilité des individus sélectionnés et de toute la population au cours des générations.

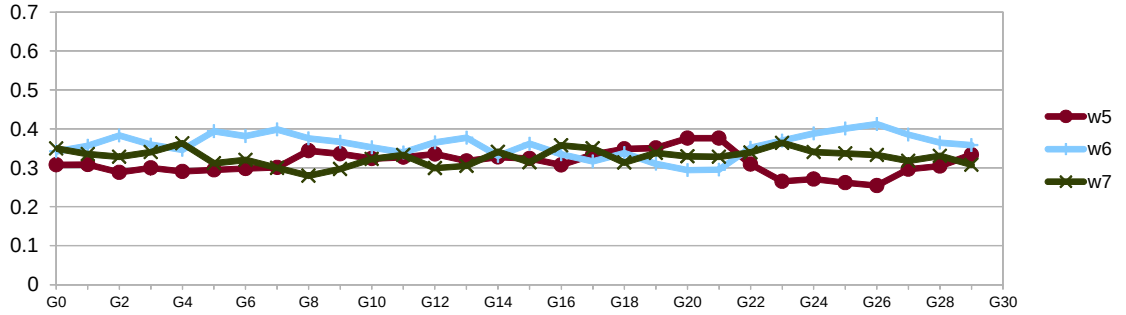
Après l'observation de l'évolution de la valeur de l'utilité, je peux constater l'évolution associée des 10 poids w_i de génération en génération sur les graphiques de la figure 6.24. Sur ces graphiques, la valeur des w_i dans une génération est la moyenne de w_i pour tous les individus choisis pour la prochaine génération (donc pour les 30 meilleurs). On a constaté que :

- Il apparait pour le premier groupe de poids que la meilleur solution est pour $w_1 > w_2 > w_3 > w_4$ (figure 6.24(a)).

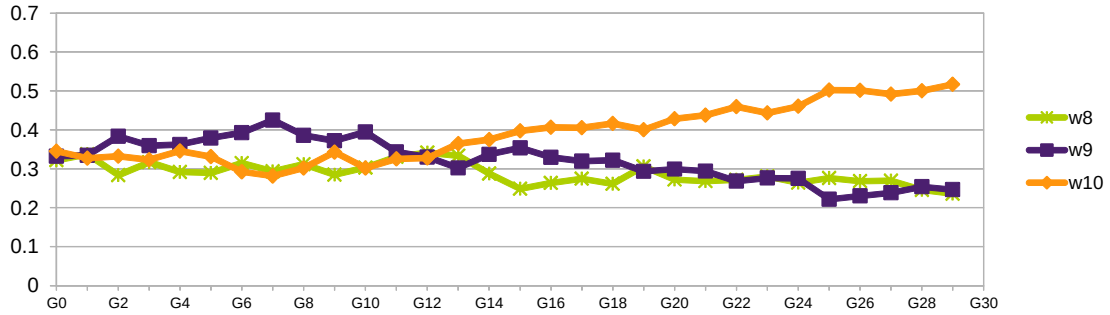
– Dans le deuxième groupe de poids, il n'y a pas de différence significative entre w_5, w_6 et w_7 (figure 6.24(b)).
– Dans le dernier groupe de poids, w_{10} semble être plus important que w_8 et w_9 (figure 6.24(c)).



(a) w1 - w4



(b) w5 - w7



(c) w8 - w10

Figure 6.24: Évolution des poids w_i au cours des générations

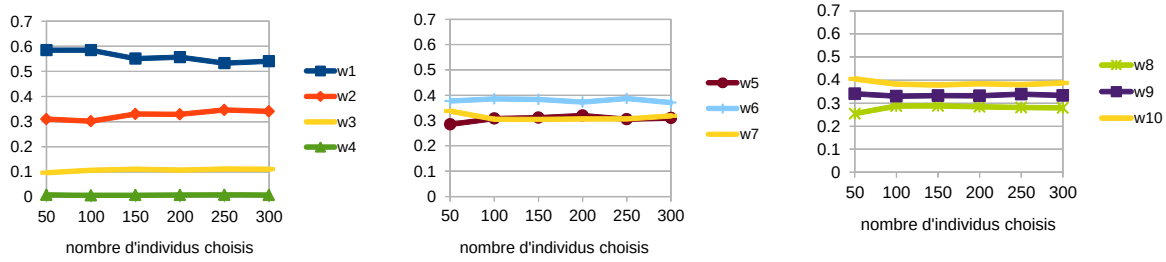
6.4.3.2 Meilleure combinaison de w_i

Afin d'examiner plus en détail sur la combinaison de w_i, je sélectionne tous les meilleurs individus (classés par leur valeur d'utilité) dans toutes les générations, puis je prends la moyenne de chaque poids w_i pour un certain nombre des meilleurs individus. Je considère la moyenne de chaque poids w_i pour les 50, 100, 150, 200, 250 et 300 meilleurs individus de toutes les générations confondues.

Je constate que la valeur de tous les poids reste pratiquement inchangée (figure 6.25). Dans le groupe des règles de déplacement (figure 6.25(a)), w_1 est le plus important avec une valeur variant de $0,55$ à $0,60$, alors que la valeur de w_2 varie de $0,30$ à $0,35$, w_3 a une valeur constante à environ $0,10$ et w_4 semble pas important avec une valeur de $0,01$ (On examinera le cas de w_4 dans la section suivante). En d'autres termes, dans le groupe des règles de déplacement, les résultats suggèrent qu'il est préférable de se déplacer à un point de la frontière en priorité pour ensuite se déplacer vers à un point inconnu, puis passer à un point peu fiable, et finalement s'orienter vers un point qui est près de certains robots fiables.

Dans le groupe des règles de communication, w_6 est un peu plus important que les deux autres poids avec une valeur d'environ $0,40$, w_5 et w_7 n'ont pas de différence significative avec la même valeur d'environ $0,30$ (figure 6.25(b)). Donc, ces résultats suggèrent qu'il est plus efficace de communiquer avec un robot fiable que de communiquer avec un robot déconnecté depuis longtemps ou avec un robot avec de nombreuses nouvelles données.

En ce qui concerne le dernier groupe de règles (figure 6.25(c)), w_{10} dont la valeur est d'environ $0,40$ est un peu plus important que w_9, et w_9 dont la valeur est d'environ $0,35$ est aussi un peu plus important que w_8 dont la valeur est environ $0,30$. Ces résultats suggèrent qu'il est légèrement préférable de se déplacer vers un robot avec des données nouvelles (et de s'éloigner d'un robot sans nouvelle donnée, resp.) que de se déplacer vers un robot avec des données correctes (et de se déplacer loin d'un robot avec des données inexactes, resp.).



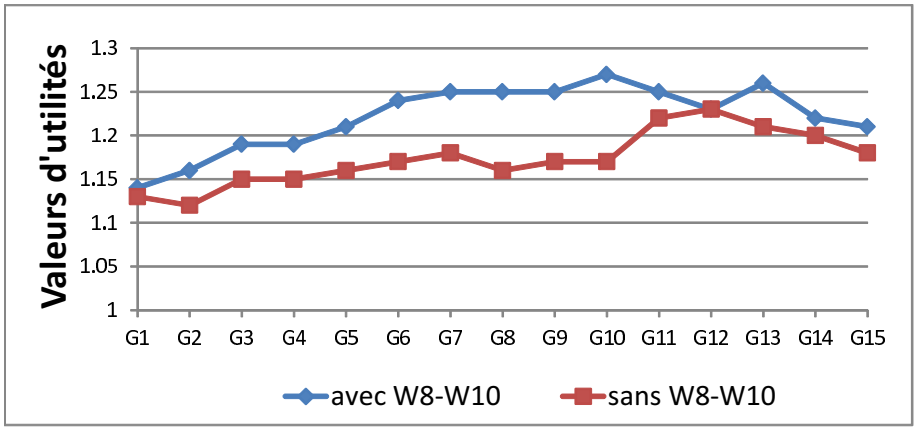
(a) w1 - w4 (b) w5 - w7 (c) w8 - w10

Figure 6.25: Évolution des poids w_i en fonction du nombre de meilleurs individus choisis.

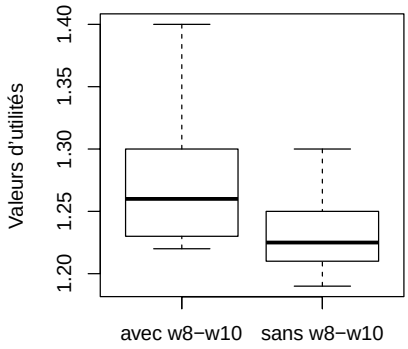
Revenons au poids w_4. Sa valeur est très faible par rapport aux autres poids du groupe des règles de déplacement. w_4 est le poids associé à la règle de déplacement vers un robot fiable (pour communiquer avec lui) ou loin d'un robot défectueux (pour éviter de communiquer avec lui). Par conséquent, l'importance de w_4 implique l'importance des 8^e, 9^e et 10^e règles. Une question se pose donc ici : est-ce que cette règle n'a pas d'importance en elle-même ou est-ce que cette règle n'a pas d'importance en comparaison des autres règles de ce groupe ?

Afin de répondre à cette question, je lance l'algorithme génétique avec le même scénario, sauf que je n'utilise pas les 8^e, 9^e et 10^e règles. Je compare ensuite les résultats avec ceux du cas précédent.

Les résultats sont présentés dans la Figure 6.26. Génération après génération, la moyenne des valeurs d'utilité des individus sélectionnés dans le cas sans $w_8 - w_{10}$ est inférieure à celle dans le cas avec $w_8 - w_{10}$ dans presque toute les générations (figure 6.26(a)). Plus précisément, je compare la moyenne de la valeur d'utilité de la meilleure génération de deux générations (figure 6.26(b)). Les résultats indiquent que la moyenne des valeurs d'utilité des individus sélectionnés de la meilleure génération dans le cas avec $w_8 - w_{10}$ est significativement plus élevée que dans les cas sans $w_8 - w_{10}$: $M(avec) = 1,27$, $M(sans) = 1,23$ avec $p - value < 0,001$. Ces résultats suggèrent que w_4 est peu important par rapport aux autres règles dans le même groupe, mais il est encore nécessaire pour aider les robots à mieux détecter les obstacles.



(a) utilité par génération



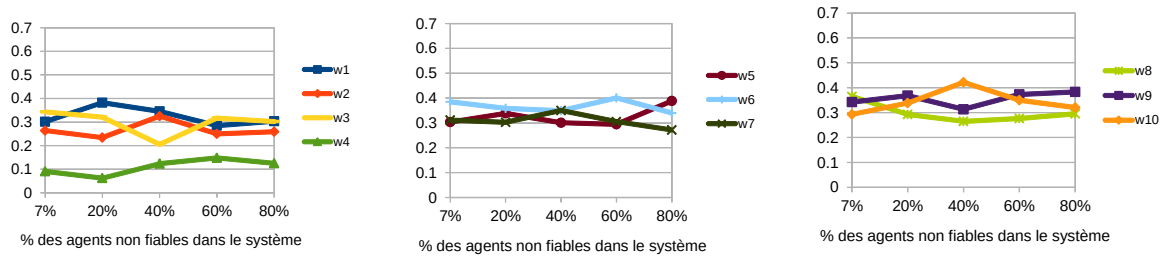
(b) Meilleur utilité suivant les paramètres considérés

Figure 6.26: Différence entre les cases utilisant (ou non) les poids ($w_8 - w_{10}$) : avec (6.26(a)), et sans (6.26(b)).

6.4.3.3 Variation du taux des robots défectueux

Afin d'observer comment les poids évoluent lorsque le taux de perturbation change, j'ai lancé une série d'expériences dans lesquelles le taux de robots défectueux varie. J'ai lancé l'algorithme génétique avec le même scénario que dans la partie précédente à l'exception du nombre des robots défectueux que j'ai fait varier. Je considère trois cas avec 20%, 40% et 60% de robots défectueux 7% dans la partie précédente : 5 robots défectueux parmi 68 robots. L'analyse s'effectue comme précédemment : je prends la valeur moyenne pour chaque poids w_i des valeurs des 30 meilleures individus de toutes les générations, puis je les compare.

Il est intéressant de constater que la valeur des poids w_i change lorsque le taux de robots défectueux dans le système change (voir figure 6.27). Dans le groupe des règles de déplacement, w_1 est toujours plus élevé et w_4 est toujours plus bas. Les poids w_2 et w_3 suivent une tendance inverse : plus le taux de robots défectueux est élevé, plus la valeur de w_2 est faible et plus la valeur de w_3 est élevée (figure 6.27(a)). Ces résultats suggèrent que lorsque le taux de robots défectueux dans le système augmente, il est préférable de se déplacer vers un point de l'environnement peu fiable plutôt que de se déplacer vers à un point inconnu : les informations sur les points fiables ayant une grande probabilité d'être faux.



(a) w1 - w4 (b) w5 - w7 (c) w8 - w10

Figure 6.27: Évolution des poids w_i en fonction du taux de robots défectueux dans le système

Dans le groupe des règles liées à la communication, il n'y a pas beaucoup de changement concernant w_5, w_6 et w_7. Dans la plupart des cas avec le taux de robots défectueux variant de 7% à 60%, w_6 est plus grand que w_5 et w_7, c.à.d que le robot réduisent le volume de leurs communications pour éviter les informations redondantes qu'il pourrait recevoir en communiquant plusieurs fois de suite avec les mêmes robots. En particulier lorsque le taux de robots défectueux est supérieur à 60%, w_5 augmente rapidement (figure 6.27(b)). Rappelons que w_5 encourage le robot à choisir les partenaires en se basant sur la confiance. Ces résultats indiquent que ce critère pour choisir le partenaire est utile lorsque les robots défectueux sont majoritaires dans le système, la communication des robots doivent alors se focaliser sur les partenaires les plus fiables pour éviter d'accumuler trop d'informations indirectes inexactes transmises par les robots défectueux.

Dans le groupe des règles de contrôle, il n'y a pas beaucoup de changement de w_8 , de w_9 et de w_{10}. En particulier lorsque le taux de robots défectueux est supérieur à 40% : w_9 augmente et w_{10} diminue (figure 6.27(c)). Ces résultats suggèrent que, lorsque le taux de robots défectueux dans le système est élevé, il est préférable de se déplacer vers un robot qui vient de lui transmettre des informations directes fiables plutôt que de se déplacer vers les robots qui sont suggérés fiables par les autres ou se déplacer vers les robots portant de nombreuses nouvelles données.

Ces résultats nous aident à permettre aux robots de faire évoluer eux-mêmes et dynamiquement les poids $\{w_i\}$ des règles en considérant le taux de robots défectueux observés. Ainsi, quand le taux de robots défectueux dans le système augmente d'environ 40% à environ 60%, alors le robot devrait changer dynamiquement sa stratégie de la manière suivante : se déplacer plutôt vers un point peu fiable que de se déplacer vers un point inconnu (augmenter $w3$ et diminuer $w2$) ; se déplacer vers un robot avec des données fiables (vérifiées directement) plutôt que de se déplacer vers un robot qui est suggéré fiable ou vers un robot portant de nombreuses nouvelles données (augmenter $w9$). Quand le taux de robots défectueux dans le système est trop important (environ 80%), il est préférable de communiquer avec des partenaires fiables plutôt qu'avec des partenaires avec lesquels il a être déconnecté depuis longtemps (augmenter $w5$ et diminuer $w6$).

6.5 Conclusion du chapitre

Dans ce chapitre, j'ai présenté une implémentation des concepts présentés précédemment. Au moyen du simulateur Danger Mapping, j'ai pu tester les différentes solutions proposées dans cette thèse. Les premiers résultats de simulation sont assez prometteurs. Après avoir analysé les résultats expérimentaux, je résume les caractéristiques suivantes de mon approche.

1. Concernant l'assurance de la cohérence du système, j'ai testé l'influence de différents paramètres et de 8 différentes stratégies de communication. J'ai constaté que l'utilisation de la stratégie TrustSet pour détecter les robots défectueux est plus efficace que les autres stratégies. En particulier, le temps pour que tous les robots du système obtiennent la carte réelle est le meilleur en raison des avantages acquis par la communication avec (et seulement avec) les autres robots fiables.
2. Concernant le modèle de confiance dans l'approche auto-organisationnelle, on a vérifié que mon modèle conserve de bonnes propriétés. En particulier, on a constaté que pour les robots fiables, leur confiance en eux-même est stable et leur réputation est augmentée au cours du temps à partir de la première interaction. Pour les robots défectueux, leur confiance en eux-mêmes et leur réputation diminuent pour atteindre une même valeur.
3. Concernant l'émergence au niveau du système de collecte d'information, on a mis en évidence l'émergence du rôle d'explorateur. On a également constaté que les robots fiables ont tendance à privilégier la communication avec ceux qui sont fiables pour la formation de coalitions de robots fiables. De plus, les robots défectueux ne communiquent pas longtemps avec les robots fiables, cela permettant de rendre le système plus robuste en écartant les robots défectueux.
4. Concernant l'émergence au niveau du système de communication, deux rôles d'envoi d'informations sont observés : les robots défectueux ont tendance à envoyer plus d'informations qu'ils n'en reçoivent, ce que j'appelle le rôle d'émetteur d'informations, tandis que les robots fiables jouent le rôle de récepteur d'informations puisqu'ils ont tendance à recevoir plus d'informations qu'ils n'en envoient. De plus, les robots fiables ont tendance à envoyer et à recevoir plus d'informations directes que d'informations indirectes, tandis que les robots défectueux ont tendance à envoyer plus d'informations indirectes que d'informations directes.
5. Concernant l'impact des contrôles sur l'auto-organisation, on a testé des changements de poids pour vérifier leur impact sur la communication et sur le déplacement des robots.
6. Concernant l'optimisation de l'auto-organisation, j'ai appliqué un algorithme génétique pour trouver la meilleure combinaison de poids des règles locales en fonction de la vitesse de collecte des données et de la quantité de données collectées. Les valeurs obtenues (et leur évolution en fonction du nombre de robots défectueux) pourront être ajoutées aux connaissances du robot pour qu'il s'adapte mieux à son environnement.

Troisième partie

Conclusion

Chapitre 7

Conclusion

Sommaire

L'objectif principal de cette thèse a été de proposer une solution au problème de la résilience dans les SMA perturbés (quand certains agents transmettant des informations fausses ou inexactes sur l'environnement). Un tel système doit se développer de manière autonome (auto-organisation) et faire évoluer dynamique ses stratégies d'échange d'information (communication), de déplacement, et les comportements des agents, permettant de garantir le maintien de la cohérence de l'information globale recueillie par les agents.

Je me place dans le cadre applicatif de la cartographie collaborative d'un environnement ayant subie une catastrophe et représentant des zones dangereuses à détecter. Un essaim de robots mobiles capables de détecter les dangers et de communiquer avec d'autres robots est utilisé pour cette détection. Les solutions à mon problème de résilience doivent donc satisfaire les exigences suivantes :

- Cohérence : la perception du monde par les différent robots doit rester correcte.
- Robustesse : dans le cas d'une panne complète ou partielle d'un robot, le système doit être capable de reprendre pour continuer à partir de l'état actuel
- Auto-organisation : cela fait référence à la capacité du système à fonctionner sans aucun changement de contrôle explicite de l'extérieur, et à l'émergence d'organisations elles-mêmes construites à partir de l'interaction et la coordination entre les éléments du système
- Hétérogénéité : les agents dans le système peuvent varier en terme de la capacité physique (panne ou non)
- Calcul distribué : les calculs sont répartis entre les charges en cours de simulation.

J'ai identifié et étudié les deux sous-problèmes suivants :

- Aide au maintien de la cohérence dans un SMA perturbé en proposant un modèle de confiance (le TrustSet) et en utilisant dans les choix des agents.
- Renforcement de la robustesse du SMA perturbé en développant une auto-organisation basée sur la confiance.

Dans ce chapitre, je résume tout d'abord mes contributions. Ensuite, j'analyse les limitations de l'approche proposée et discutons les résultats obtenus. Enfin, je discute différentes perspectives de ce travail.

7.1 Résumé des contributions

La première contribution de cette thèse est un **modèle de confiance appelé TrustSet.** Le modèle utilise trois types d'informations : les informations sur l'environnement directement collectées par l'agent, les informations sur l'environnement reçues et les informations sur les agents. Des méthodes pour l'échange d'informations entre agents sont présentées dans ce modèle. Le TrustSet nous fournit quelques avantages.

- Premièrement, il permet à chaque agent de procéder à un examen critique de sa situation actuelle , de s'auto-diagnostiquer pour savoir s'il est fiable ou non. Le comportement des agents sera changé en fonction des résultats de l'auto-diagnostique.
- Deuxièmement, il met en place une stratégie de robustesse consistant à séparer les agents peu fiables et des agents fiables. Ce processus permet à un agent de renforcer la communication avec les agents fiables et de refuser les communications provenant d'agents jugés non fiables, afin de réduire la dissonance dans le SMA perturbé.
- Troisièmement, il donne une méthode du calcul de la fiabilité de l'information dans le cas où les agents reçoivent beaucoup d'informations dont certaines sont contradictoires.
- Enfin, il vise à la formation de coalitions d'agents fiables. Ce processus permet de faire une meilleure coopération des agents, une minimisation des redondances de données et une amélioration de la qualité des informations échangées.

La deuxième contribution est **une démarche d'auto-organisation à base de confiance**, qui se base sur le modèle de confiance proposé et sur une vision systémique dans laquelle je considère l'auto-organisation que couplage structurel entre les deux composantes du système : une composante de collecte d'informations directes et une autre de la communication. Plus précisément, elle est due au mécanisme de confiance TrustSet développée combinée avec un ensemble de règles locales. Ces règles locales lient les perceptions aux actions et régissent les comportements à adopter par les agents en fonction des contraintes de l'environnement. Ces règles vont ensuite exercer une pression sélective sur les agents pour aboutir à l'émergence de deux organisations qui co-évoluent de façon rétroactive : une au niveau social et l'autre au niveau spatial. L'organisation sociale reflète la relation de confiance établie entre les différents agents. L'organisation spatiale reflète le déploiement des agents dans l'environnement pour favoriser l'exploration des agents. Les règles locales de comportement sont composées de trois catégories de règles : déplacement, communication, et rétroaction entre la communication et de déplacement. La combinaison de ces règles de comportement est alors expérimentée et laisse apparaitre dans des simulations l'émergence d'organisations et de rôles dans le système. Grâce à cette démarche, les agents peuvent optimiser l'exploration de leur environnement et mieux communiquer lors du partage des informations et mieux ajuster leurs comportements en fonction de l'objectif.

7.2 Discussions

7.2.1 Le modèle de confiance - TrustSet

Je traite du problème d'une communauté d'agents perturbés où la collecte et la transmission d'informations peuvent être altérées par des agents non fiables. En associant une fiabilité à l'information et une confiance aux autres membres de la communauté, chaque membre améliore sa perception du monde. L'avantage de mon approche est que chaque agent distingue les informations directes (collectées sur l'environnement) et les informations indirectes (échangées en communiquant avec les autres agents), non seulement sur les données stockées, mais également sur les données transmises par les agents qu'il rencontre. Compte-tenu de ces informations, plus précises que celles traitées par les modèles existants, les agents peuvent construire et mettre à jour des structures de données plus efficaces pour représenter les données (arbres probabilistes) et les métadonnées (TrustSets). Ceci est l'une des originalités de mon approche. Le calcul de la confiance d'un agent envers un autre agent prend en compte à la fois la qualité des métadonnées transmises par l'autre agent et la distance entre les données recueillies par les deux agents. Ce processus permet finalement à un agent de renforcer la communication avec les agents fiables et de refuser les communications provenant d'agents jugés non fiables, afin de réduire l'impact de la perturbation dans le SMA.

L'extension de la notion de TrustGraph à celle de TrustNet de Schillo, dans la quelle on distingue les valeurs publiques et privées, et les nouveaux algorithmes de fusion entre TrustSets nous permettent d'obtenir de meilleurs résultats que les stratégies traditionnelles utilisant des TrustTables. Ainsi, je montre que parfois dans une communauté perturbée, il est préférable de ne pas donner toutes les informations que je reçois d'autres membres lorsque ces membres ne sont peut-être pas fiables.

7.2.2 La démarche d'auto-organisation à base de confiance

En développant dynamiquement des organisations grâce à une approche auto-organisationnelle, chaque agent du SMA est capable de développer dynamiquement des stratégies efficaces pour l'exploitation et l'exploration de l'information car la performance globale du système ne dépend pas uniquement de leur performance mais aussi du couplage de leurs dynamiques respectives. L'avantage de cette démarche est de combiner le modèle de confiance TrustSet développé avec un ensemble de règles locales, qui lient les perceptions aux actions et qui régissent les comportements à adopter par les agents et les contraintes de l'environnement qui vont exercer une pression sélective sur les agents pour aboutir à l'émergence de deux organisations qui co-évoluent de façon rétroactive : une au niveau social et l'autre au niveau spatial. Ce couplage est réalisé par auto-organisation. L'organisation sociale reflète la relation de confiance établie entre les différents agents. L'organisation spatiale reflète le déploiement des agents dans l'environnement pour favoriser l'exploration des agents. Les règles locales de comportement sont composées par les trois catégories de règles : déplacement, communication, et rétroaction entre la communication et de déplacement.

L'organisation sociale est exprimée par des rôles sociaux. Ils sont représentés par des comportements différents des agents dans la communication avec les autres. Par exemple, en se basant sur le TrustSet, chaque agent peut évaluer la tendance de communication avec l'autre agent. Via la tendance de communication estimée, l'agent pourrait avoir des comportements différents : envoyer toutes ses informations collectées, envoyer une partie de ses informations collectées, n'envoyer aucune information.

L'organisation spatiale est aussi exprimée par des rôles spatiaux. Elle permet d'exprimer la réalisation de l'organisation sociale (des tâches de la communication à réaliser) dans l'environnement spatial pour tenir compte par exemple des contraintes de l'environnement physique. L'apparition de clusters et des comportements observés des agents mènent à une spécialisation des rôles spatiaux, comme par exemple : transmetteur - l'agent qui se détecte lui-même comme non fiable (capteur en panne) va changer son comportement (de transmetteur d'information direct à transmetteur d'information indirecte). Un autre rôle spatial concerne les agents qui gardent la connexion avec les agents du cluster pour le maintien de la cohérence de l'information, ou les éclaireurs qui sont des agents qui permettent de rafraichir les informations du système en apportant des nouvelles informations.

7.3 Limitations

L'approche proposée possède cependant un certain nombre de limitations.

7.3.1 Problème de la latence de l'information

Dans mon modèle, pour limiter la latence de l'information, j'émette l'hypothèse que l'environnement est clos et statique, c.à.d des informations sur l'environnement restent inchangés pendant la simulation. Cela n'est pas toujours vrai en réalité, par exemple à l'instant t, l'agent A circule dans le terrain et il découvre un danger à la position (x, y). A envoie alors cette information à l'agent B, à l'instant $t + 1$ l'agent C reçoit cette info de l'agent B mais à ce moment-là le niveau du danger dans la réalité peut avoir changé.

7.3.2 Problème de l'extensibilité du système

Le problème ici est de déterminer dans quelle mesure le modèle est extensible à l'autre application. Tout d'abord j'ai considéré l'environnement comme clos, il n'y a pas de nouveaux agents. De plus, la

méthode de représentation et d'échange des informations est simple. Comment dans ce cas appliquer ce modèle aux situations où l'information à plusieurs attributs différents.

7.3.3 Problème de la vitesse de simulation

Les résultats obtenus à partir de la plate-forme GAMA [Amouroux et al., 2007] ont nécessité un temps de simulation très longues. En effet l'échange et la mise à jour du TrustSet demande beaucoup de temps de calcul. Cette limitation doit être abordée lorsqu'on travaille avec de grandes bases des agents acquises et avec un grand environnement.

7.3.4 Problème du contrôle de l'auto-organisation

On a proposé un modèle d'auto-organisation à base de règles (avec notamment les règles de déplacement, règles de communication et règles sur la rétro-action communication-déplacement). Chaque règle a un niveau d'importance différent qui est représenté par un poids. Les poids que j'ai affectés aux différentes règles sont donc autant de réglages sur lesquels je vais, en tant qu'utilisateur, pouvoir jouer de manière à «diriger» l'auto-organisation du système. Il pourrait être intéressant de donner aux agents la capacité de pourvoir auto-configurer ces poids pour choisir leur meilleure stratégie de communication et de collecter d'information.

7.3.5 Problème de limite dans le système

L'approche proposée est évaluée sur un ensemble de simulations particulières mais plusieurs aspects comme la complexité de l'algorithme ou le temps d'exécution doivent être testés. De plus, il n'est pas facile pour les utilisateurs novices de comprendre les modèles proposés.

7.4 Perspectives

Cette section présente quelques pistes à suivre pour remédier aux limitations que j'ai indiquées dans la section7.3, ainsi que des extensions possibles de mon travail.

7.4.1 Perspectives à court terme

7.4.1.1 Latence de l'information

Pour limiter la latence de l'information, on travaille sur des informations inchangées, ce qui n'est pas toujours vrai en réalité. Pour surmonter ce problème et travailler dans un environnement dynamique, il pourrait être intéressant d'attribuer aux informations collectées un intervalle de temps sur lequel l'information est sensée rester valide.

7.4.1.2 Extensibilité du système

Il pourrait être intéressant d'appliquer mon approche sur une deuxième application de recherche d'informations fiables dans un environnement ouvert. Le Web est très riche en informations de toutes sortes. Il n'en reste pas moins que toutes les informations qu'il abrite ne sont pas fiables. Ainsi tout utilisateur du Net ayant besoin d'une information précise se retrouve un jour ou l'autre confronté à un ensemble d'informations contradictoires ou différentes les unes des autres.

7.4.1.3 Optimisation de la vitesse de simulation

Je présente une méthode pour s'échanger des informations entre les agents. Cependant la performance de la simulation n'est pas très bonne. Deux pistes semblent intéressantes pour trouver des solutions à ce problème. D'une part, l'optimisation du calcul peut être fait en ajoutant des règles permettant de faire émerger un autre type de cluster où la communication entre les agents est optimal. D'autre part, l'utilisation de grille de calcul permettrait d'augmenter la vitesse de simulation.

7.4.2 Perspectives à long terme

7.4.2.1 Apprentissage dans le système

Dans le cadre de l'auto-organisation du système, les données et métadonnées permettent à chaque entité de choisir à chaque instant une stratégie de déplacement et de communication qu'elle juge adaptée à l'environnement qu'elle perçoit. On a proposé un modèle d'auto-organisation à base des règles dont les règles de déplacement, règles de communication, et règles sur le rétro-action communication-déplacement. Chaque règle a un niveau d'importance différent qui est représenté par un poids. L'apprentissage pourra être utilisé sur ce problème pour essayer de trouver de manière automatique une solution tendant vers un optimum.

7.4.2.2 Limite du système

Les travaux futurs se concentreront sur les limites de la perturbation dans un système : à partir de quel niveau, la perturbation empêche-t-elle le système de converger (atteindre son objectif) ? Deux pistes semblent intéressantes pour trouver des solutions à ce problème. D'une part, en utilisant la théorie de la propagation des rumeurs pour étudier comment l'information se déforme lorsqu'elle 'traverse' plusieurs agents. D'autre part, en utilisant la théorie de la percolation pour étudier les limites du système.

Annexe A

Présentation de la plate-forme de simulation GAMA

La première étape de la thèse a consisté à proposer deux modèles (un modèle de la confiance et celui de l'auto-organisation). La deuxième étape consiste à développer un modèle de simulation basé sur des modèles proposés au sein de la plate-forme GAMA (**Gis & Agent-based Modeling Architecture**) [Amouroux et al., 2007, Taillandier et al., 2010]. Dans ce modèle, je concentre plutôt sur les aspects haut niveau d'un système de collecte d'information, le point important est que les robots doivent être capable d'échanger les informations. L'environnement est représenté par une carte SIG (**Système d'Information Géographique**).

A.1 Introduction de GAMA

Il existe de nombreuses plates-formes de modélisation et simulation multi-agents comme Swarm [Minar et al., 1996], NetLogo [Wilensky and Evanston, 1999], Repast[North et al., 2006], VLE (Virtual Laboratory Experiment) [Quesnel et al., 2009]. Mais ces plates-formes sont souvent assez spécifiques en termes d'application. Certains plates-formes ont suffisamment de maturité comme Swarm et NetLogo mais il leur manque certains éléments importants, par exemple la capacité à utiliser des données de SIG, tandis que les nouvelles plate-formes qui sont développées pour répondre à ces besoins n'ont pas encore la maturité suffisante. En outre, de nombreuses plates multi-agents, comme Jade [Bellifemine et al., 2001] par exemple, n'offrent pas de support particulier pour décrire l'environnement des agents.

Ces différentes raisons ont poussé les membres de l'équipe MSI (qui est une équipe de recherche IRD/AUF (IFI) et qui est appartient à l'UMI UMMISCO qui est une unité IRD/UPMC située à Hanoi, Vietnam) à développer, sur la base de la boite à outil Repast, la plate-forme GAMA. Il s'agit d'un environnement de développement et d'expérimentation générique qui a pour but de fournir aux experts du domaine, aux modélisateurs et aux informaticiens tous les outils nécessaires à la modélisation et à la simulation de systèmes multi-agent spatialement explicites.

Les spécificités de GAMA sont :

1. une capacité d'utilisation transparente des données complexes provenant de SIG comme environnement pour les agents,
2. la capacité à gérer un grand nombre d'agents (hétérogènes)
3. la capacité d'offrir une plate-forme pour des expériences automatisées et contrôlées (par variation automatique des paramètres, enregistrement des statistiques, etc),
4. la possibilité pour des non informaticiens de concevoir des modèles ou d'interagir avec les agents pendant l'exécution des simulations (langage de modélisation intuitif adapté à la modélisation agent) ;
5. la possibilité, enfin, pour des modélisateurs, de pouvoir facilement étendre les modèles de comportement disponibles grâce à la définition d'un méta-modèle générique.

GAMA (version 1.3) est organisé comme présenté dans la Figure A.1. Il est principalement constitué d'un noyau, un méta-modèle et différents contrôleurs de la simulation liés à plusieurs outils : Eclipse pour l'interface l'utilisateur, GeoTools pour des primitives SIG. Le noyau exécute la simulation qui peut être commandée de différentes manières : en mode Batch (l'exploration des paramètres, par exemple), à distance (par exemple pour la collaboration) ou de façon interactive (ce qui permet de modifier les paramètres et d'observer leurs influences). Les classes du méta-modèle définissent les briques de base du langage de modélisation. En particulier, les classes de Skills fournissent des supplémentaires primitives (Mouvement pour l'exemple) pour les agents et le classes dites de « décision » fournissent différentes architectures de comportement des agents : machine à états finis, réflexes...

A.2 Entités et représentation de l'environnement

Le méta-modèle utilisé dans la plate-forme GAMA est présenté dans la Figure A.2. Les trois éléments centraux sont *agent*, *place* et *world*. *Agent* correspond aux entités du système, les agents. De même, *places* correspondent aux zones de l'environnement du système de référence (un champ ou un bâtiment, par exemple). *World* est le gestionnaire de la simulation, il contient les paramètres (entrées et dynamiques), les variables globales. Il contient également les sorties (visualisation, les graphiques et les sorties numériques).

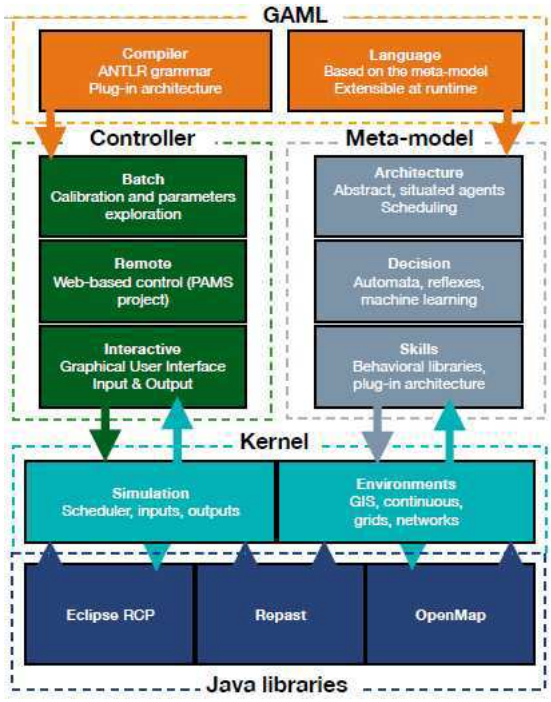


FIGURE A.1 – Structure de la plate-forme GAMA [Taillandier et al., 2010]

Place et *world* sont des spécialisations d'agent, afin de leur permettre d'avoir des caractéristiques similaires en termes de dynamique et d'état interne.

La classe *Agent* est responsable pour la tenue de l'état interne des individus et de leur processus de décision. Il est couplé à la classe *Body* qui met en oeuvre toutes les actions et interactions décidées dans la classe *Agent*. Le *Body* à l'agent permet d'interagir et de communiquer avec d'autres agents, de se localiser dans l'environnement et de le percevoir.

Avec un sens similaire à la notion de classe en programmation-objet, le concept de species est introduit dans le méta-modèle de GAMA. Il représente la généralisation d'une classe d'entités présentes dans le système de référence. Par exemple, dans un modèle de trafic, un *species* véhicule contient ce qui est commun à tous les agents véhicules. Il définit des individus le comportement et l'état interne. Il simplifie la mise en œuvre car il fournit l'architecture de contrôle. C'est aussi dans cette classe que les *Skills* disponibles seront déclarés. En GAMA, un *skill* est un ensemble de primitives définies qui peuvent réutiliser dans tous les modèles. Dans mon modèle « Danger Mapping », j'ai développé un skill s'appelle « TrustingSkill » pour les opérations sur les confiances.

A.3 GAML, langage de modélisation basé sur XML

La langage de modélisation GAML est un langage de modélisation générique basé sur XML créé pour développer des modèles dans GAMA. Il s'inspire notamment du langage de NetLogo. Il a été conçu pour permettant un apprentissage rapide et facile par des thématiciens.

Un modèle en GAML est constitué de quatre sections :

1. Global : les variables globales et les commandes concernant le simulateur (paramètres, séries temporelles, type de contrôleur, etc).
2. Environnement : l'environnement sa dynamique et son état interne.
3. Entités : les espèces avec leur dynamique et leur état interne.
4. Output : Les sorties avec la visualisation, les cartes, les diagrammes ...

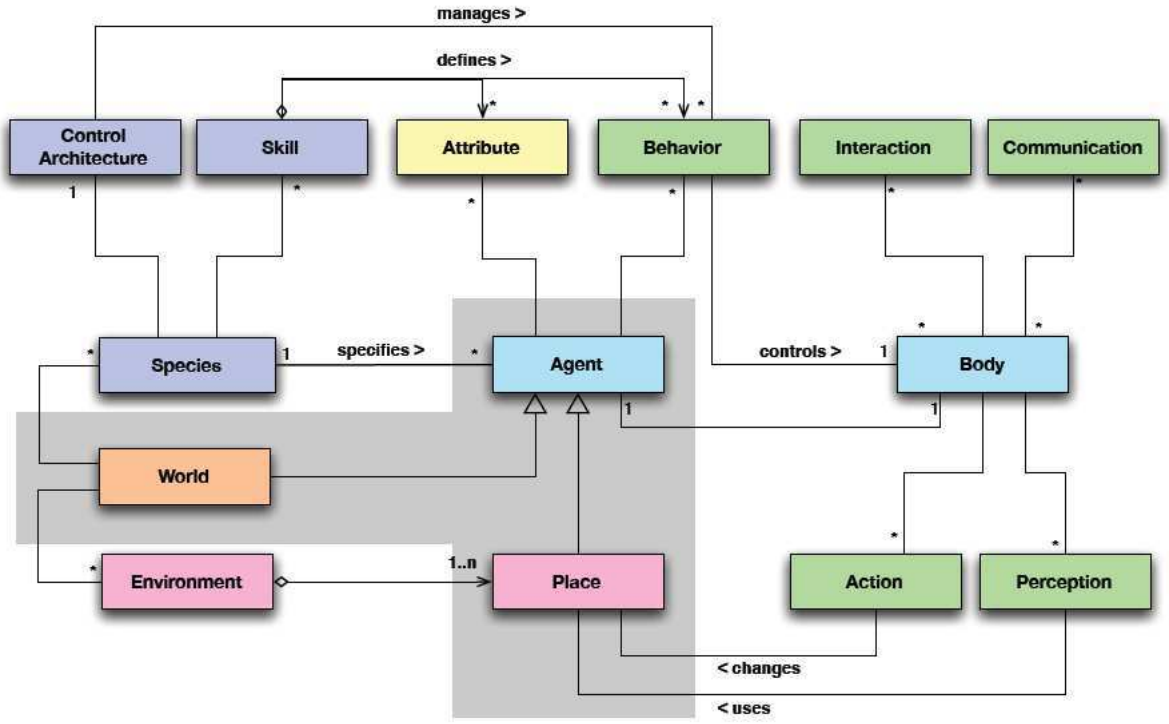


FIGURE A.2 – Méta modèle de GAMA 1.3 [Taillandier et al., 2010]

La figure A.3 montre un exemple de *species*. Il définit une *species* appellé « robot » et qui est capable de se déplacer (moving skill). Son état interne est défini par une variable nommée « color » and une variable nommée « size ». Elle a seul comportement « wander » en fonction de deux paramètres : « amplitude » (combien il change sa direction à chaque étape) et « speed » (vitesse de l'agent).

```
<species name="robot" skills="moving">
    <rgb name="color" init="rgb 'blue'" const="true" />
    <var type="float" name="size" value="(rnd 100) / 50" min="1" />
    <reflex>
        <do action="wander">
            <arg name="amplitude" value="120" />
            <arg name="speed" value="(rnd 200) / 50" />
        </do>
    </reflex>
</species>
```

FIGURE A.3 – Exemple d'une description GAML d'un robot agent

Annexe B

Implémentation de l'approche proposée

En dépit du fait que GAMA est encore dans sa phase d'implémentation, la plate-forme est actuellement utilisée pour plusieurs applications. Je détaille ci-après une implémentation de la simulation « Danger Mapping » proposée dans la chapitre 6.

B.1 Implémentation un nouveau comportement TrustingSkill

J'ai construit dans GAMA version 1.3 un modèle de simulation dans lequel j'ai développé un *skill* appelé « TrustingSkill » pour des opérations sur les confiances. Pour utiliser des opérations concernant la confiance, il faut tout d'abord initialiser les valeurs des paramètres nécessaires. Pour ce faire, j'ai défini une action permettant l'initialisation des paramètres avec six valeurs des paramètres :

- *threshold_reliable_agent* : seuil haut de confiance,
- *threshold_defective_agent* : seuil bas de confiance,
- *number_of_interactions* : nombre d'interactions nécessaires pour que l'agent qu'un puisse calculer la confiance envers les autres,
- *rate_stabilisation* : le taux de stabilisation représentant le pourcentage de la simulation après lequel le concepteur veut qu'une classification entre agents fiables et non fiables se dégage,
- *threshold_dissonant* : le niveau d'informations contradictoires acceptable,
- *choix_max_value* : la confiance d'un groupe est calculé soit par la moyenne les confiances des membres ou par la confiance la plus grande.

```
<!-- initialization trust parameters -->
<do action ="initTrustParameters">
 <arg name ="threshold_reliable_agent" value ="threshold_reliable_agent"/>
 <arg name ="threshold_defective_agent" value ="threshold_defective_agent"/>
 <arg name ="number_of_interactions" value ="number_of_interactions"/>
 <arg name ="rate_stabilisation" value ="rate_stabilisation"/>
 <arg name ="threshold_dissonant" value ="threshold_dissonant"/>
 <arg name ="choix_max_value" value ="choix_max_value"/>
</do>
```

J'ai également défini une action pour mettre à jour le TrustGraph en fonction de cinq paramètres :

- *receiver* : nom de récepteur,
- *sender* : nom de transmetteur,
- *levelDissonant* : le niveau d'incohérence total entre les informations du transmetteur et de récepteur,
- *tgReceiver* : le TrustGraph du récepteur,
- *time* : le temps d'envoi.

```
<!-- update trust graph -->
<do action ="updateTrustGraph" return="tgUpdated">
        <arg name ="receiver" value ="self.name"/>
        <arg name ="sender" value ="new_agent_id"/>
        <arg name ="levelDissonant" value ="levelInfoDissonant"/>
        <arg name ="tgReceiver" value ="new_trust_graph"/>
        <arg name ="time" value ="time" />
</do>
```

Une autre action permettant de calculer la fiabilité d'une information en fonction de deux paramètres est proposé :

- *list_level_src* : une liste des données différents portent la même information,
- *strategy_comm* : stratégie de communication.

```
<!-- get the reliability of information -->
<do action ="getInfoReliability" return="level_reliability">
        <arg name ="list_level_src" value ="new_list_obst_@_newObst" />
        <arg name ="strategy_comm" value ="strategy_simulation" />
</do>
```

Le code Java à la dernière action est reproduit ci-dessous :

```
/**
* To get information reliability from different source path of information
*
* @param args
* @return a list which contains 4 components (level, reliability) with 2
*         options of trust computation (mean and max final value)
*
*/

@action("getInfoReliability")
@args({ "list_level_src", "strategy_comm" })
synchronized public  CommandStatus primGetInfoReliability( final  Arguments args) {

String lstReturn = "";
GamaMap<Integer,List<String>> lst_level_src = args.mapValue("list_level_src");
int strategy = args.intValue("strategy_comm");
Double reliability_overage = 0.0;
Integer level_overage = 0;
Double reliability_max = 0.0;
Integer level_max = 0;

for (Integer level : lst_level_src.keySet()) {
 List<String> group = lst_level_src.get(level);
 List<String> group_temp = new  ArrayList<String>();
 group_temp.addAll(group);
 List<Object> tw_group = getTrustWeight(group_temp, TRUST_TABLE);
 double mean_trust_group = (Double) tw_group.get(1);
 double max_trust_group = (Double) tw_group.get(2);
 if (strategy == 1)
        mean_trust_group = max_trust_group = group_temp.size();
 if (reliability_overage <= mean_trust_group) {
        reliability_overage = mean_trust_group;
        level_overage = level.intValue(); }
 if (reliability_max <= max_trust_group) {
        reliability_max = max_trust_group;
    level_max = level.intValue(); }
}

if (choix_max_value) {
 lstReturn = String.valueOf(level_max).concat(comma)
        } else  {
                lstReturn = String.valueOf(level_overage).concat(comma)
}

setReturn(lstReturn);
return CommandStatus.success;
}
```

B.2 Implémentation du modèle Danger Mapping

B.2.1 Environnement

Je définis un *species* s'appelle « background » qui est situé dans (au moins) un environnement (situated skill). Il sera affiché comme un rectangle rose.

```
<species name ="background" skills="situated">
 <aspect name ="base">
  <draw shape ="geometry" color ="rgb 'pink'" />
 </aspect>
</species>
```

Un environnement d'un modèle Danger Mapping est un agent background, dont la géométrie est un rectangle, dans lequel seront crées des obstacles.

a. Création du font

```
<!-- create the back ground -->
<create species="background" number="1" return="a_bg">
        <set name ="geometry" value ="[{0,0},{width,0},
            {width,height}, {0,height}, {0,0}]"/>
</create>
<set name ="the_bg" value ="a_bg" />
<let type ="list" of="point" name ="bg_tmp"
     value="the_bg interior [buffer_in::size_obstacle]"/>
```

b. Création des agents et des obstacles à des positions aléatoires

```
<!-- create position of agents / obstacles randomly -->
<create species="obstacle" number="number">
  <do action ="build_basic_geometry">
        <arg name ="location" value ="self place_in [geometry::bg_tmp]" />
        <arg name ="geometry_type" value ="'square'" />
        <arg name ="side_size" value ="size_obstacle" />
  </do>
  <let type ="list" of="point" name ="geom_tmp"
          value="self buffer [buffer_size::2*size_obstacle]" />
  <set name ="bg_tmp" value ="self difference
         [geometry1::bg_tmp, geometry2::geom_tmp]"/>
</create>
<create species="robot" number="number_robots + number_liars" />
<ask target="number_liars among (list robot)">
        <set name ="color" value ="rgb 'red'" />
        <add item="self.name" to="list_liars" />
</ask>
```

c. Création des agents et des obstacles à des positions fixées

```
<!-- load robots / obstacles position from textfile -->
<var type ="matrix" name ="robot_data"
     init="file 'data/Robot_location.Data'" const="true" />
<var type ="matrix" name ="obs_data"
     init="file 'data/Obstacle_location.Data'" const="true" />
<loop from="1" to="(rows_number obs_data) - 1" var ="i">
 <let type ="float" name ="ind_i" value ="obs_data at {0,i}" />
 <let type ="float" name ="ind_j" value ="obs_data at {1,i}" />
```

```
 <let type ="int" name ="ind_k" value ="obs_data at {2,i}" />
 <create species="obstacle">
  <set name ="danger_lvl" value ="ind_k" />
  <set name ="color"
       value="rgb [255,255 * (1- (float ind_k/max_danger_lvl))
                  ,255* (1- (float ind_k/max_danger_lvl))]" />
  <do action ="build_basic_geometry">
        <arg name ="location" value ="{ind_i, ind_j}" />
        <arg name ="geometry_type" value ="'square'" />
        <arg name ="side_size" value ="size_obstacle" />
  </do>
  <let type ="list" of="point" name ="geom_tmp"
       value="self buffer [buffer_size::2*size_obstacle]" />
  <set name ="bg_tmp"
    value="self difference [geometry1::bg_tmp, geometry2::geom_tmp]" />
 </create>
</loop>
```

B.2.2 Obstacle

Un *species* **obstacle,** situé dans (au moins) un environnement (situated skill), est défini ci-dessous. Son état interne est définie par une couleur représentant le niveau de danger.

```
<species name ="obstacle" skills="situated">
 <var type ="rgb" name ="color"
     init="rgb [255, 255 * (1- (float danger_lvl/max_danger_lvl)) ,
                  255* (1- (float danger_lvl/max_danger_lvl))]" />
 <aspect name ="base">
  <draw shape ="geometry" color ="color" />
 </aspect>
</species>
```

B.2.3 Robot

J'ai construit la *species* robot qui est capable de se déplacer (moving skill) et d'utiliser les opérations sur la confiance (trusting skill). Il comporte plusieurs variables caractérisant les 4 comportements principaux : se déplacer, percevoir l'environnement, communiquer et mettre à jour la base de données. Le code ci-dessous est une version simplifiée de la species dans lequel je ne présente que les reflex (omettant notamment les variables).

```
<species name ="robot" skills="moving, trusting">
 <var type ="rgb" name ="color" init="rgb 'blue'" />
 <reflex name ="move"/>
 <reflex name ="update_the_known"/>
 <reflex name ="communicate"/>
 <reflex name ="update"/>
 <aspect name ="base">
  <draw shape ="circle" size =2*size_obstacle" color ="color" />
  <draw shape ="circle" color ="'white'" size ="range" empty="true" />
 </aspect>
</species>
```

L'implémentation des comportements : se déplacer, percevoir l'environnement, communiquer et mettre à jour d'un robot sont présentées dans la suite :

a. Se déplacer

Un robot se déplace vers un but (target_loc) qui est déterminé par la fonction « choose_target ». L'objectif de cette fonction est de choisir le point cible dont le coefficient d'attractivité est le plus élevé au moment de son calcul (comme indiqué dans l'algorithme 8).

```
<!-- reflex moving -->
<reflex name ="move">
 <if condition="location = target_loc">
  <set name ="target_loc" value ="nil" />
 </if>
 <if condition="target_loc = nil">
  <set name ="init_loc" value ="self.location" />
  <do action ="choose_target" />
 </if>
 <do action ="goto">
  <arg name ="target" value ="target_loc" />
 </do>
</reflex>
```

b. Percevoir l'environnement

Un robot se déplace et détecte des obstacles et des zones non visités par le biais de ses capteurs. L'implémentation de ce comportement est détaillée comme suit :

```
<!-- reflex capture environment -->
<reflex name ="update_the_known">
 <let type ="boolean" name ="liar" value ="self.color = rgb 'red'" />
 <let type ="list" of="point" name ="percept"
      value="self percieved_area [agent::the_bg,
      precision::percept_prec, range::percept_dist]" />
 <let type ="list" name ="myObsts" value ="keys list_obst_direct" />
 <!-- update obstacles captured direct -->
 <ask target="list obstacle">
  <if condition="self overlaps [geometry::percept]">
   <let name ="level_obst" value ="liar ? rnd danger_lvl : danger_lvl" />
   <set name ="myself.color_srcs" value ="level_obst :: [myself.name]" />
   <add item="self.location :: level_obst" to="myself.list_obst_direct" />
   <add item="self.location :: level_obst" to="myself.
       list_reliable_obstacle" />
   <add item="self.location :: myself.color_srcs" to="myself.list_obst" />
  </if>
 </ask>
 <!-- update zone captured direct -->
 <ask target="the_known_agent">
  <set name ="geometry" value ="self union [geometry::percept]" />
  <do action ="simplification">
       <arg name ="dp_dist" value ="dp_dist" />
  </do>
 </ask>
</reflex>
```

c. Communiquer avec d'autres robots

Une version simplifiée du comportement de la communication de l'agent i est montrée ici où a_com représe une liste d'agents voisins de l'agent i. L'agent i va appeler la fonction « receive_map » de l'agent j en fonction de la probabilité représentée par la confiance de i en j.

```
<!-- communicating -->
<reflex name ="communicate">
 <ask target="a_com">
  <let name ="pij" value ="self.name in (keys myself.pij_acom) ?
                  (myself.pij_acom @ self.name) : 0" />
  <if condition="flip pij">
   <do action ="receive_map">
    <arg name ="new_agent_id" value ="myself.name" />
    <arg name ="new_the_known_agent" value ="myself.the_known_agent_send" />
    <arg name ="new_the_known_agent_direct"
        value="myself.the_known_agent_direct_send" />
    <arg name ="new_the_known_agent_indirect"
        value="myself.the_known_agent_indirect_send"/>
    <arg name ="new_list_obst" value ="myself.list_obst_send"/>
    <arg name ="new_list_obst_direct" value ="myself.list_obst_direct_send"/>
    <arg name ="new_list_obst_indirect" value ="myself.
       list_obst_indirect_send"/>
    <arg name ="new_trust_graph" value ="myself.trust_graph" />
   </do>
   <let name ="last_time" value ="myself.myself.map_t0_ik @ self.name" />
   <put item="last_time" at="self.name" in="myself.map_t1_ik" />
   <put item="time" at="self.name" in="myself.map_t0_ik" />
  </if>
 </ask>
</reflex>
```

d. Mettre à jour la base de données

La mise a jour de la base de données de l'agent sera appelée lors qu'il reçoit des informations nouvelles. Elle est composée par 2 actions principales : mettre à jour les zones et les obstacles détectés, et mettre à jour le TrustSet.

```
<!-- updating database -->
<reflex name ="update">
 <!-- update the known and obstacles dectected -->
 <do action ="update_map">
 <arg name ="new_agent_id" value ="new_agent_id" />
 <arg name ="new_list_obst" value ="new_list_obst" />
 <arg name ="new_list_obst_indirect" value ="new_list_obst_indirect" />
 <arg name ="new_the_known_agent" value ="new_the_known_agent" />
 <arg name ="new_the_known_agent_direct"
     value="new_the_known_agent_direct" />
 <arg name ="new_the_known_agent_indirect"
     value="new_the_known_agent_indirect" />
 </do>
 <!-- compute direct trust -->
 <if condition="(new_list_obst_direct != nil) and !(empty
     new_list_obst_direct)">
  <do action ="compute_direct_trust" return="levelInfoDissonant">
   <arg name ="new_list_obst_direct" value ="new_list_obst_direct" />
   <arg name ="new_agent_id" value ="new_agent_id" />
  </do>
  <!-- update trust graph -->
  <do action ="updateTrustGraph" return="tgUpdated">
   <arg name ="receiverA" value ="self.name" />
```

```
    <arg name ="senderB" value ="new_agent_id" />
    <arg name ="levelDissonant" value ="levelInfoDissonant" />
    <arg name ="tgReceiver" value ="new_trust_graph" />
    <arg name ="time" value ="time" />
   </do>
   <set name ="trust_graph" value ="tgUpdated" />
  </if>
</reflex>
```

Bibliographie

[Abdallah and Lesser, 2007] Abdallah, S. and Lesser, V. (2007). Multiagent reinforcement learning and self-organization in a network of agents. In *AAMAS '07 : Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, pages 1–8, New York, NY, USA. ACM.

[Abdul-Rahman and Hailes, 1999] Abdul-Rahman, A. and Hailes, S. (1999). Relying on trust to find reliable information. In *Proceedings 1999 International Symposium on Database, Web and Cooperative Systems (DWACOS99)*.

[Abdul-Rahman and Hailes, 2000] Abdul-Rahman, A. and Hailes, S. (2000). Supporting trust in virtual communities. *Hawaii International Conference on System Sciences*.

[Alan Galan, 1999] Alan Galan, A. B. (1999). Multi-agent communication in jafmas. In *Working Notes of the Workshop on Specifying and Implementing Conversation Policies*, pages 67–70.

[Ali et al., 2003] Ali, S., Maciejewski, A. A., Siegel, H. J., and Kim, J.-K. (2003). Definition of a robustness metric for resource allocation. In *Proceedings of the 17th International Symposium on Parallel and Distributed Processing*, IPDPS '03, Washington, DC, USA. IEEE Computer Society.

[Amouroux et al., 2007] Amouroux, E., Quang, C., Boucher, A., and Drogoul, A. (2007). GAMA : an environment for implementing and running spatially explicit multi-agent simulations. In *10th Pacific Rim International Workshop on Multi-Agents (PRIMA)*, Thailand.

[Amza, 2011] Amza, C. (2011). First steps toward a fault-tolerance multi-agent systems. *Computer Science Master Research*, 1(2).

[Anderson, 2002] Anderson, C. (2002). Self-Organization in Relation to Several Similar Concepts : Are the Boundaries to Self-Organization Indistinct ? *The Biological Bulletin*, 202(3) :247–255.

[Arpaci-dusseau et al., 1999] Arpaci-dusseau, R. H., Anderson, E., Treuhaft, N., Culler, D. E., Hellerstein, J. M., Patterson, D., and Yelick, K. (1999). Cluster I/O with river : Making the fast case common. In *Proceedings of the Sixth Workshop on Input/Output in Parallel and Distributed Systems*, pages 10–22. ACM Press.

[Ashby, 1965] Ashby, W. (1965). *Design for a brain : the origin of adaptive behaviour*. Science paperbacks. Chapman & Hall and Science Paperbacks.

[Ashby, 1947] Ashby, W. R. (1947). Principles of the self-organizing dynamic system. *The Journal of General Psychology*, 37(2) :125–128.

[Ashby, 1962] Ashby, W. R. (1962). Principles of the self-organizing systemvv. In v. Foerster, H. and Zopf, G. W., editors, *Principles of Self-Organization : Transactions of the University of Illinois Symposium*, pages 255–278. Pergamon, London.

[Balch and Arkin, 1995] Balch, T. and Arkin, R. C. (1995). Communication in reactive multiagent robotic systems. *Auton. Robots*, 1(1) :27–52.

[Balter, 1985] Balter, R. (1985). *Maintien de la cohérence dans les systèmes d'information répartis*. These, Institut National Polytechnique de Grenoble - INPG ; Université Joseph-Fourier - Grenoble I.

[Banzhaf, 2009] Banzhaf, W. (2009). Self-organizing systems. In Meyers, R. A., editor, *Encyclopedia of Complexity and Systems Science*, pages 8040–8050. Springer New York.

[Bar-Yam, 1997] Bar-Yam, Y. (1997). *Dynamics of Complex Systems*. Studies in Nonlinearity. Addison-Wesley.

[Barber et al., 1999] Barber, K. S., Liu, T. H., Goel, A., and Martin, C. E. (1999). Conflict representation and classification in a domain-independent conflict management framework. In *Proceedings of the third annual conference on Autonomous Agents*, pages 346–347. ACM Press.

[Barto, 1975] Barto, A. G. (1975). *Cellular automata as models of natural systems.* PhD thesis, Ann Arbor, MI, USA. AAI7609336.

[Basagni, 1999] Basagni, S. (1999). Distributed clustering for ad hoc networks. In *ISPAN '99 : Proceedings of the 1999 International Symposium on Parallel Architectures, Algorithms and Networks*, Washington, DC, USA. IEEE Computer Society.

[Bellifemine et al., 2001] Bellifemine, F., Poggi, A., and Rimassa, G. (2001). Jade : a fipa2000 compliant agent development environment. In *Proceedings of the fifth international conference on Autonomous agents*, AGENTS '01, pages 216–217, New York, NY, USA. ACM.

[Bernon et al., 2006] Bernon, C., Chevrier, V., Hilaire, V., and Marrow, P. (2006). Applications of self-organising multi-agent systems : An initial framework for comparison. *Informatica (Slovenia)*, 30(1) :73–82.

[Bertalanffy and Ludwig, 1973] Bertalanffy, V. and Ludwig (1973). *Théorie générale des systèmes.* Paris Dunod.

[Beurier, 2007] Beurier, G. (2007). *Codage indirect de la forme dans les systèmes multi-agents : émergence multi-niveaux, évolution et morphogénèse.* PhD thesis, Université des sciences et techniques de Montpellier 2.

[Beurier et al., 2003] Beurier, G., Simonin, O., and Ferber, J. (2003). Un modèle de système multi-agents pour l'émergence multi-niveaux. In *11eme journées Francophones sur les Systèmes Multi-Agents*, Revue des Sciences et Technologies de l'Information, pages 235–247. Hermès.

[Bonabeau et al., 1999] Bonabeau, E., Dorigo, M., and Theraulaz, G. (1999). *From Natural to Artificial Swarm Intelligence.* Oxford University Press.

[Bond and Gasser, 1988] Bond, A. and Gasser, L. (1988). *Readings in distributed artificial intelligence.* M. Kaufmann.

[Bongaerts, 1998] Bongaerts, L. (1998). *Integration of Scheduling and Control in Holonic Manufacturing Systems.* Katholieke Unuversiteit te Leuven.

[Boucher et al., 2009] Boucher, A., Canal, R., Chu, T. Q., Drogoul, A., Gaudou, B., Le, V. T., Moraru, V., Van Nguyen, N., Vu, Q. A. N., Taillandier, P., Sempé, F., and Stinckwich, S. (2009). The AROUND project : Adapting robotic disaster response to developing countries. In *Proceedings of 2009 IEEE International Workshop on Safety, Security, and Rescue Robotics*. IEEE Computer Society.

[Bourjot et al., 2003] Bourjot, C., Chevrier, V., and Thomas, V. (2003). A new swarm mechanism based on social spiders colonies : from web weaving to region detection. *Web Intelli. and Agent Sys.*, 1(1) :47–64.

[Browning et al., 2002] Browning, B., Kaminka, G. A., and Veloso, M. M. (2002). Principled monitoring of distributed agents for detection of coordination failure. In *Proceedings of Distributed Autonomous Robotic Systems 6*, pages 319–328. Springer-Verlag.

[Buchegger and Le Boudec, 2004] Buchegger, S. and Le Boudec, J. Y. (2004). A Robust Reputation System for P2P and Mobile Ad-hoc Networks. In *Proceedings of the Second Workshop on the Economics of Peer-to-Peer Systems.*

[Burgin, 2009] Burgin, M. (2009). Robustness of information systems and technologies. In *Proceedings of the 8th WSEAS international conference on Data networks, communications, computers*, DNCOCO'09, pages 67–72, Stevens Point, Wisconsin, USA. World Scientific and Engineering Academy and Society (WSEAS).

[Caballero et al., 2009] Caballero, A., Garcia-Valverde, T., Botia, J. A., and Gomez-Skarmeta, A. F. (2009). A trust and reputation model as adaptive mechanism for multi-agent systems. *Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial*, pages 3–11.

[Camazine et al., 2003] Camazine, S., Deneubourg, J., Franks, N., Sneyd, J., Theraula, G., and Bonabeau, E. (2003). *Self-Organization in Biological Systems*. Princeton Studies in Complexity. Princeton University Press.

[Campo, 2002] Campo, C. (2002). Directory Facilitator and Service Discovery Agent. Technical report, Foundation for Intelligent Physical Assets (FIPA).

[Capera et al., 2003] Capera, D., Georgé, J.-P., Gleizes, M.-P., and Glize, P. (2003). The AMAS theory for complex problem solving based on self-organizing cooperative agents. In *Proceedings of the Twelfth International Workshop on Enabling Technologies : Infrastructure for Collaborative Enterprises*, WETICE '03, Washington, DC, USA. IEEE Computer Society.

[Castanet R., 2003] Castanet R., W. H. (2003). Techniques avancées de test de système complexes : test de robustesse. Technical report, CNRS-AS23.

[Castelfranchi, 1992] Castelfranchi, C. (1992). *Communication from an artificial intelligence perspective*. Springer Verlag.

[Castelfranchi and Falcone, 1998] Castelfranchi, C. and Falcone, R. (1998). Social trust : cognitive anatomy, social importance, quantification and dynamics. In *Autonomous Agents '98 Workshop on "Deception, Fraud and Trust in Agent Societes"*, pages 35–49. Minneapolis, USA.

[Castelfranchi and Paglieri, 2007] Castelfranchi, C. and Paglieri, F. (2007). The role of beliefs in goal dynamics : prolegomena to a constructive theory of intentions. *Synthese*, 155(2) :237–263.

[Conte and Paolucci, 2002] Conte, R. and Paolucci, M. (2002). *Reputation in Artificial Societies : Social Beliefs for Social Order*. Kluwer Academic Publishers, Norwell, MA, USA.

[Dasgupta, 2000] Dasgupta, P. (2000). *Trust as a Commodity*. Department of Sociology, University of Oxford, electronic edition.

[De Wolf and Holvoet, 2004] De Wolf, T. and Holvoet, T. (2004). Emergence and self-organisation : a statement of similarities and differences. In *Lecture Notes in Artificial Intelligence*, pages 96–110. SpringerVerlag.

[De Wolf and Holvoet, 2005] De Wolf, T. and Holvoet, T. (2005). Towards a methodology for engineering Self-Organising emergent systems. *Self-Organization and Autonomic Informatics*, 135 :18–34.

[Demazeau, 1995] Demazeau, Y. (1995). From interactions to collective behaviors in agent-based systems. In *Proceeding of First European Conference on Cognitive Science*, Saint-Malo, France.

[Dempster, 1998] Dempster, M. (1998). *A Self-Organising Systems Perspective on Planning for Sustainability*. PhD thesis, University of Waterloo, School of Urban and Regional Planning.

[Dimitrakos and Bicarregui, 2001] Dimitrakos, T. and Bicarregui, J. (2001). Towards modelling e-trust. In *3rd Panhellenic Logic Symposium 2001*.

[Dowling and Cahill, 2001] Dowling, J. and Cahill, V. (2001). The k-component architecture meta-model for self-adaptive software. In *Proceedings of the Third International Conference on Metalevel Architectures and Separation of Crosscutting Concerns*, REFLECTION '01, pages 81–88, London, UK, UK. Springer-Verlag.

[Drira et al., 2010] Drira, K., Seba, H., and Kheddouci, H. (2010). ECGK : An efficient clustering scheme for group key management in manets. *Comput. Commun.*, 33(9) :1094–1107.

[Druschel and Banga, 1996] Druschel, P. and Banga, G. (1996). Lazy receiver processing (LRP) : a network subsystem architecture for server systems. In *Proceedings of the second USENIX symposium on Operating systems design and implementation*, OSDI '96, pages 261–275, New York, NY, USA. ACM.

[Durand, 2002] Durand, D. (impr. 2002). *La systémique*. Que sais-je ? Presses universitaires de France, Paris, 9e édition edition.

[Edgar, 1981] Edgar, M. (DL 1981). *La méthode 1, La nature de la nature*. Points. Seuil, Paris.

[Ephrati and Rosenschein, 1991] Ephrati, E. and Rosenschein, J. S. (1991). The clarke tax as a consensus mechanism among automated agents. In *Proceedings of the ninth National conference on Artificial intelligence - Volume 1*, AAAI'91, pages 173–178. AAAI Press.

[Farley and Clark, 1954] Farley, B. and Clark, W. (1954). Simulation of self-organizing systems by digital computer. *IRE Transactions on Information Theory*, PGIT-4 :76–84.

[Festinger, 1957] Festinger, L. (1957). *A Theory of Cognitive Dissonance*. Stanford University Press.

[Fiorino, 1998] Fiorino, H. (1998). Elaboration de conjectures par des agents cooperants. Master's thesis, ENSAE.

[Floyd and Jacobson, 1994] Floyd, S. and Jacobson, V. (1994). The synchronization of periodic routing messages. *IEEE/ACM Trans. Netw.*, 2(2) :122–136.

[Focardi et al., 2002] Focardi, S., Cincotti, S., and Marchesi, M. (2002). Self-organization and market crashes. *Journal of Economic Behavior & Organization*, 49(2) :241–267.

[Foukia, 2005] Foukia, N. (2005). *IDReAM - Intrusion Detection and Response Executed with Agent Mobility - a Distributed Approach Inspired from Natural Life Systems*. Université de Genève Faculté des Sciences Département d'informatique.

[Gardner, 1970] Gardner, M. (1970). Mathematical games : The fantastic combinations of John Conway's new solitaire game 'Life'. *Scientific American*, 223(4) :120–123.

[Geard et al., 2005] Geard, N., Willadsen, K., and Wiles, J. (2005). Perturbation analysis : A complex systems pattern. In *Recent Advances in Artificial Life, volume 3 of Advances in Natural Computation*, pages 69–84. World Scientific Publishing.

[Gershenson, 2007] Gershenson, C. (2007). *Design and Control of Self-organizing Systems*. PhD thesis, Vrije Universiteit Brussel.

[Gleizes, 2004] Gleizes, M.-P. (2004). *Vers la résolution de problèmes par émergence*. Habilitation à diriger des recherches, Université Paul Sabatier, Toulouse, France.

[Gleizes and Camps, 1999] Gleizes, M. P. and Camps, V. (1999). A theory of emergent computation based on cooperative Self-Organization for adaptive artificial systems. In *4th European Congress of Systems Science*.

[Goldstein, 1999] Goldstein, J. (1999). Emergence as a construct : History and issues. *Emergence*, 1(1) :49–72.

[Gribble, 2001a] Gribble, S. (2001a). Robustness in complex systems. In *8th Workshop on Hot Topics in Operating Systems (HotOS-VIII)*, pages 21–26.

[Gribble, 2001b] Gribble, S. D. (2001b). Robustness in complex systems. In *Proceedings of the Eighth Workshop on Hot Topics in Operating Systems*, HOTOS '01, Washington, DC, USA. IEEE Computer Society.

[Guibert, 1999] Guibert, N. (1999). *La confiance en marketing : fondements et applications*. Presses universitaires de Grenoble, Grenoble, FRANCE (1986) (Revue).

[Haken, 1977] Haken, H. (1977). *Synergetics : an introduction : nonequilibrium phase transitions and self-organization in physics, chemistry and biology*. Springer.

[Halsall, 1992] Halsall, F. (1992). *Data communications, computer networks, and open systems*. Electronic systems engineering series. Addison-Wesley.

[Hannebauer, 2002] Hannebauer, M. (2002). Modeling and verifying agent interaction protocols with algebraic petri nets. In *Proceedings of the Sixth International Conference on Integrated Design and Process Technology*.

[Hassas, 2003] Hassas, S. (2003). Systèmes complexes à base de multi-agents situés. Mémoire d'habilitation à diriger les recherches, Université Claude Bernard-Lyon 1.

[Havlin et al., 2010] Havlin, S., Araujo, N. A. M., Buldyrev, S. V., Dias, C. S., Parshani, R., Paul, G., and Stanley, H. E. (2010). Catastrophic cascade of failures in interdependent networks. *CoRR*, abs/1012.0206.

[Hewitt, 1986] Hewitt, C. (1986). Offices are open systems. *ACM Trans. Inf. Syst.*, 4(3) :271–287.

[Heylighen, 1999] Heylighen, F. (1999). The science of self-organization and adaptivity. In *in : Knowledge Management, Organizational Intelligence and Learning, and Complexity, in : The Encyclopedia of Life Support Systems, EOLSS*, pages 253–280. Publishers Co. Ltd.

[Holland, 1962] Holland, J. H. (1962). Outline for a logical theory of adaptive systems. *Journal of the ACM JACM*, 9(3) :297–314.

[Holland and Melhuish, 1999] Holland, O. and Melhuish, C. (1999). Stigmergy, self-organization, and sorting in collective robotics. *Artif. Life*, 5(2) :173–202.

[Holz et al., 2010] Holz, D., Basilico, N., Amigoni, F., and Behnke, S. (2010). Evaluating the efficiency of frontier-based exploration strategies. In *ISR/ROBOTIK*, pages 1–8.

[Hortensius et al., 2006] Hortensius, P. D., McLeod, R. D., Pries, W., Miller, D. M., and Card, H. C. (2006). Cellular automata-based pseudorandom number generators for built-in self-test. *Trans. Comp.-Aided Des. Integ. Cir. Sys.*, 8(8) :842–859.

[Huynh, 2006] Huynh, T. D. (2006). *Trust and Reputation in Open Multi-Agent Systems*. PhD thesis, University of Southampton.

[Huynh et al., 2006] Huynh, T. D., Jennings, N. R., and Shadbolt, N. R. (2006). An integrated trust and reputation model for open multi-agent systems. *Journal of Autonomous Agents and Multi-Agent Systems*, 13 :119–154.

[Ioannidis and Sellis, 1989] Ioannidis, Y. E. and Sellis, T. K. (1989). Conflict resolution of rules assigning values to virtual attributes. In *Proceedings of the 1989 ACM SIGMOD international conference on Management of data*, SIGMOD '89, pages 205–214, New York, NY, USA. ACM.

[Ishida et al., 1992] Ishida, T., Gasser, L., and Yokoo, M. (1992). Organization self-design of distributed production systems. *IEEE Trans. on Knowl. and Data Eng.*, 4(2) :123–134.

[Jensen, 2001] Jensen, M. T. (2001). Improving robustness and flexibility of tardiness and total flowtime job shops using robustness measures. *Journal of Applied Soft Computing*, 1 :35–52.

[Jiang and Baras, 2006] Jiang, T. and Baras, J. S. (2006). Trust evaluation in anarchy : A case study on autonomous networks. In *INFOCOM*.

[Jiang et al., 2005] Jiang, Y. C., Xia, Z. Y., Zhong, Y. P., and Zhang, S. Y. (2005). Autonomous trust construction in multi-agent systems : a graph theory methodology. volume 36, pages 59–66, Oxford, UK, UK. Elsevier Science Ltd.

[Kaminka et al., 2001] Kaminka, G. A., Pynadath, D. V., and Tambe, M. (2001). Monitoring deployed agent teams. In *Proceedings of the fifth international conference on Autonomous agents*, AGENTS '01, pages 308–315, New York, NY, USA. ACM.

[Kamvar et al., 2003] Kamvar, S. D., Schlosser, M. T., and Garcia-Molina, H. (2003). The Eigentrust algorithm for reputation management in P2P networks. In *WWW '03 : Proc. of the 12th Int. Conference on World Wide Web*, pages 640–651, New York, NY, USA. ACM.

[Karuna et al., 2005] Karuna, H., Valckenaers, P., Saint-Germain, B., Verstraete, P., Zamfirescu, C., and Van Brussel, H. (2005). Emergent forecasting using a stigmergy approach in manufacturing coordination and control. In Brueckner, S., Di Marzo Serugendo, G., Karageorgos, A., and Nagpal, R., editors, *Engineering Self-Organising Systems*, volume 3464 of *Lecture Notes in Computer Science*, pages 67–75. Springer Berlin / Heidelberg.

[Kim and Seo, 2008] Kim, T. K. and Seo, H. S. (2008). A trust model using fuzzy logic in wireless sensor network. In *Proceedings of world academy of science, engineering and technology*, volume 32.

[Kita et al., 2010] Kita, E., Kan, S., and Fei, Z. (2010). Investigation of self-organizing map for genetic algorithm. *Adv. Eng. Softw.*, 41(2) :148–153.

[Klein and Dellarocas, 1999] Klein, M. and Dellarocas, C. (1999). Exception handling in agent systems. In *proc. of the 3rd int. conference on autonomous agents*.

[Kumar and Cohen, 2000] Kumar, S. and Cohen, P. R. (2000). Towards a fault-tolerant multi-agent system architecture. In *Proceedings of the fourth international conference on Autonomous agents*, AGENTS '00, pages 459–466, New York, NY, USA. ACM.

[Langton, 1997] Langton, C. (1997). *Artificial Life : An Overview*. Complex Adaptive Systems. Mit Press.

[Le et al., 2009] Le, V. T., Bouraqadi, N., Stinckwich, S., Moraru, V., and Doniec, A. (2009). Making networked robots connectivity-aware. In *ICRA'09 : Proceedings of the 2009 IEEE international conference on Robotics and Automation*, pages 1835–1840, Piscataway, NJ, USA. IEEE Press.

[Le Moigne, 1977] Le Moigne, J. L. (1977). *La théorie du système général. Théorie de la modélisation*. Presse Universitaires de France.

[Lefevre et al., 2009] Lefevre, O., Armetta, F., Clair, G., and Hassas, S. (2009). Mana : A new multi-agent approach for complex assignment problems. In *Proceedings of the 2009 Computation World : Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns*, COMPUTATIONWORLD '09, pages 167–172, Washington, DC, USA. IEEE Computer Society.

[Leonard and Durrant-Whyte, 1991] Leonard, J. and Durrant-Whyte, H. (1991). Simultaneous map building and localisation for an autonomous mobile robot. In *Proc. IEEE Int. Workshop on Intelligent Robots and Systems (IROS)*, pages 1442–1447, Osaka.

[Lesser and Corkill, 1981] Lesser, V. R. and Corkill, D. D. (1981). Functionally accurate, cooperative distributed systems. *IEEE Transactions on Systems, Man, and Cybernetics*, 11 :81–96.

[Lewicki and Bunker, 1995] Lewicki, R. J. and Bunker, B. B. (1995). *Conflict, Cooperation and Justice : Essays Inspired by the Work of Morton Deutsch*, chapter Trust in Relationships : A Model of Development and Decline. Jossey-Bass Publishers, San Francisco, CA, United States of America.

[Lorenz, 1995] Lorenz, E. (1995). *The essence of chaos*. University of Washington Press.

[Loui, 1987] Loui, R. P. (1987). Defeat among arguments : a system of defeasible inference. *Computational Intelligence*, 3(3) :100–106.

[Luhmann et al., 1996] Luhmann, N., Bednarz, J., and Baecker, D. (1996). *Social Systems*. Writing Science. Stanford University Press.

[Maclennan, 1991] Maclennan, B. (1991). Synthetic ethology : An approach to the study of communication. In *Artificial Life II*, pages 631–658. Addison-Wesley, Redwood City, CA.

[Mamei et al., 2005] Mamei, M., Vasirani, M., and Zambonelli, F. (2005). Engineering self-organising systems. chapter Self-organizing spatial shapes in mobile particles : the TOTA approach, pages 138–153. Springer-Verlag, Berlin, Heidelberg.

[Marsh, 1994] Marsh, S. (1994). *Formalising Trust as a Computational Concept*. PhD thesis, University of Sterling.

[Marti and Garcia-Molina, 2004] Marti, S. and Garcia-Molina, H. (2004). Limited reputation sharing in p2p systems. In *EC'04 : Proceedings of the 5th ACM conference on Electronic commerce*, pages 91–101, New York, NY, USA. ACM Press.

[Marzo Serugendo, 2009] Marzo Serugendo, G. (2009). Robustness and dependability of self-organizing systems - a safety engineering perspective. In *Proceedings of the 11th International Symposium on Stabilization, Safety, and Security of Distributed Systems*, SSS '09, pages 254–268, Berlin, Heidelberg. Springer-Verlag.

[Marzo Serugendo et al., 2005] Marzo Serugendo, G., Gleizes, M.-P., and Karageorgos, A. (2005). Self-organization in multi-agent systems. *Knowl. Eng. Rev.*, 20 :165–189.

[Marzo Serugendo et al., 2006] Marzo Serugendo, G., Marie-pierre, G., and Anthony, K. (2006). Self-organisation and emergence in mas : An overview. *Informatica (Slovenia)*, 30 :45–54.

[Mason and Johnson, 1990] Mason, C. L. and Johnson, R. R. (1990). Distributed artificial intelligence : vol. 2. chapter DATMS : a framework for distributed assumption based reasoning, pages 293–317. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[Maturana and Norrie, 1996] Maturana, F. P. and Norrie, D. H. (1996). Multi-agent mediator architecture for distributed manufacturing. *Journal of Intelligent Manufacturing*, 7 :257–270. 10.1007/BF00124828.

[Mena et al., 2000] Mena, E., Illarramendi, A., and Goñi, A. (2000). Automatic ontology construction for a multiagent-based software gathering service. In *Proceedings of the 4th International Workshop on Cooperative Information Agents IV, The Future of Information Agents in Cyberspace*, CIA '00, pages 232–243, London, UK, UK. Springer-Verlag.

[Minar et al., 1996] Minar, N., Burkhart, R., Langton, C., and Askenazi, M. (1996). The swarm simulation system : A toolkit for building multi-agent simulations. Santa Fe Institute.

[Mitra, 2006] Mitra, S. (2006). Introduction to robust systems. In *Robust Systems Seminar : EE 392U Autumn 2005-2006*.

[Mui et al., 2002a] Mui, L., Mohtashemi, M., and Halberstadt, A. (2002a). A computational model of trust and reputation for e-businesses. In *Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS'02)-Volume 7 - Volume 7*, HICSS '02, pages 188–, Washington, DC, USA. IEEE Computer Society.

[Mui et al., 2002b] Mui, L., Mohtashemi, M., and Halberstadt, A. (2002b). Notions of reputation in multi-agent systems : A review. In *Proceedings of AAMAS'02*, pages 280–287, Bologna, Italy. ACM.

[Nagi, 2001a] Nagi, K. (2001a). Scalability of a transactional infrastructure for multi-agent systems. In *Revised Papers from the International Workshop on Infrastructure for Multi-Agent Systems : Infrastructure for Agents, Multi-Agent Systems, and Scalable Multi-Agent Systems*, pages 266–278, London, UK, UK. Springer-Verlag.

[Nagi, 2001b] Nagi, K. (2001b). *Transactional agents : towards a robust multi-agent system*. Springer-Verlag, Berlin, Heidelberg.

[Nguyen Vu et al., 2012a] Nguyen Vu, Q.-A., Canal, R., Gaudou, B., Hassas, S., and Armetta, F. (2012a). Trustsets : using trust to detect deceitful agents in a distributed information collecting system. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–13.

[Nguyen Vu et al., 2012b] Nguyen Vu, Q.-A., Gaudou, B., Canal, R., Hassas, S., and Armetta, F. (2012b). A cluster-based approach for disturbed, spatialized, distributed information gathering systems. In Desai, N., Liu, A., and Winikoff, M., editors, *Principles and Practice of Multi-Agent Systems*, volume 7057 of *Lecture Notes in Computer Science*, pages 588–603. Springer Berlin / Heidelberg.

[Nguyen Vu et al., 2010] Nguyen Vu, Q. A., Gaudou, B., Canal, R., Hassas, S., Armetta, F., and Stinckwich, S. (2010). Using trust and cluster organisation to improve robot swarm mapping. In *Workshop on Robots and Sensors integration in future rescue INformation system (ROSIN) in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Taipei Taïwan, Province De Chine.

[NguyenVu et al., 2009] NguyenVu, Q. A., Gaudou, B., Canal, R., Hassas, S., and Armetta, F. (2009). Stratégie de communication dans un système de collecte d'information à base d'agents perturbés. In *Systèmes Multi-Agents*, pages 207–217, France.

[NguyenVu et al., 2010] NguyenVu, Q.-A., Gaudou, B., Canal, R., Hassas, S., and Armetta, F. (2010). Trustsets - using trust to detect deceitful agents in a distributed information collecting system. In *RIVF*, pages 1–6.

[NguyenVu et al., 2011] NguyenVu, Q.-A., Hassas, S., Armetta, F., Gaudou, B., and Canal, R. (2011). Combining trust and self-organization for robust maintaining of information coherence in disturbed mas. In *SASO*, pages 178–187.

[Nimis and Lockemann, 2004] Nimis, J. and Lockemann, P. C. (2004). P. : Robust multi-agent systems : The transactional conversation approach. In *In : 1st International Workshop on Safety and Security in Multiagent Systems (SASEMAS04)*.

[Nodine and Unruh, 2000] Nodine, M. H. and Unruh, A. (2000). Constructing robust conversation policies in dynamic agent communities. In *Issues in Agent Communication*, pages 205–219, London, UK, UK. Springer-Verlag.

[North et al., 2006] North, M. J., Collier, N. T., and Vos, J. R. (2006). Experiences creating three implementations of the repast agent modeling toolkit. *ACM Trans. Model. Comput. Simul.*, 16(1) :1–25.

[Parsons and Klein, 2004] Parsons, S. and Klein, M. (2004). Towards robust multi-agent systems : Handling communication exceptions in double auctions. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 3*, AAMAS '04, pages 1482–1483, Washington, DC, USA. IEEE Computer Society.

[Pasquier and Chaib-draa, 2002] Pasquier, P. and Chaib-draa, B. (2002). Cohérence et conversations entre agents : vers un modèle basé sur la consonance cognitive. In *Actes des JFIADSMA'02*, pages 188–203.

[Paurobally et al., 2003] Paurobally, S., Cunningham, J., and Jennings, N. R. (2003). Ensuring consistency in the joint beliefs of interacting agents. In *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, AAMAS '03, pages 662–669, New York, NY, USA. ACM.

[Picard et al., 2009] Picard, G., Hübner, J., Boissier, O., and Gleizes, M.-P. (2009). Reorganisation and self-organisation in multi-agent systems. In *International Workshop on Organizational Modeling (OrgMod'09)*, pages 66–80.

[Pinyol et al., 2012] Pinyol, I., Sabater-Mir, J., Dellunde, P., and Paolucci, M. (2012). Reputation-based decisions for logic-based cognitive agents. *Autonomous Agents and Multi-Agent Systems*, 24(1) :175–216.

[Pontisso, 2009] Pontisso, N. (2009). *Association cohérente de données dans les systèmes temps réel à base de composants - Application aux logiciels spatiaux.* Thèse de doctorat, Institut National Polytechnique de Toulouse, Toulouse, France.

[Poslad et al., 2000] Poslad, S., Buckle, P., and Hadingham, R. (2000). The FIPA-OS agent platform : Open source for open standards. *Technology.*

[Quesnel et al., 2009] Quesnel, G., Duboz, R., and Ramat, E. (2009). The Virtual Laboratory Environment – An operational framework for multi-modelling, simulation and analysis of complex dynamical systems. *Simulation Modelling Practice and Theory*, 17 :641–653.

[Reitbauer et al., 2005] Reitbauer, A., Battino, A., Germain, B., Karageorgos, A., Mehandjiev, N., and Valckenaers, P. (2005). The mabe middleware. In Camarinha-Matos, L., editor, *Emerging Solutions for Future Manufacturing Systems*, volume 159 of *IFIP International Federation for Information Processing*, pages 53–60. Springer US.

[Reix, 1999] Reix, R. (1999). *Dictionnaire des systèmes d'information.* Vuibert.

[Richardson et al., 2003] Richardson, M., Agrawal, R., and Domingos, P. (2003). Trust management for the semantic web. In *Proceedings of the second international semantic web conference*, pages 351–368.

[Ricordel, 2001] Ricordel, P.-M. (2001). *Programmation Orientée Multi-Agents : Développement et Déploiement de Systèmes Multi-Agents Voyelles.* PhD thesis, Institut national polytechnique de Grenoble.

[Rosnay, 1975] Rosnay, J. d. (1975). *Le macroscope : vers une vision globale.* Editions du Seuil, Paris.

[Saad Khorchef et al., 2006] Saad Khorchef, F., Rollet, A., Castanet, R., and Berrada, I. (2006). Cadre formel pour le test de robustesse. Application au protocole SSL. In *Colloque Francophone sur l'Ingénierie des Protocoles - CFIP 2006*, page 12, Tozeur, Tunisie. Eric Fleury and Farouk Kamoun, Hermès.

[Sabater and Sierra, 2002] Sabater, J. and Sierra, C. (2002). Social ReGreT, a reputation model based on social relations. *SIGecom Exch.*, 3(1) :44–56.

[Sabater-Mir, 2002] Sabater-Mir, J. (2002). *Trust and Reputation for Agent Societies.* PhD thesis, Universitat Autonoma de Barcelona.

[Sabater-Mir et al., 2006] Sabater-Mir, J., Paolucci, M., and Conte, R. (2006). Repage : Reputation and image among limited autonomous partners. *Journal of Artificial Societies and Social Simulation*, 9(2) :3.

[Sandholm and Lesser, 2002] Sandholm, T. and Lesser, V. R. (2002). Leveled-commitment contracting : A backtracking instrument for multiagent systems. *AI Magazine*, 23(3) :89–100.

[Sansores and Pavón, 2008] Sansores, C. and Pavón, J. (2008). An adaptive agent model for self-organizing mas. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems - Volume 3*, AAMAS '08, pages 1639–1642, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

[Schillo et al., 2001] Schillo, M., Bürckert, H.-J., Fischer, K., and Klusch, M. (2001). Towards a definition of robustness for market-style open multi-agent systems. In *Proceedings of the fifth international conference on Autonomous agents*, AGENTS '01, pages 75–76, New York, NY, USA. ACM.

[Schillo et al., 1999] Schillo, M., Funk, P., and Rovatsos, M. (1999). Who can you trust : Dealing with deception. In *Proceedings of Autonomous Agents '99 Workshop on "Deception, Fraud, and Trust in Agent Societies"*, pages 81–94, Seattle, USA.

[Schillo et al., 2000] Schillo, M., Funk, P., and Rovatsos, M. (2000). Using trust for detecting deceitful agents in artificial societies. *Applied Artificial Intelligence, Special Issue on Trust, Deception and Fraud in Agent Societies*, 14(8) :825–848.

[Sen and Sajja, 2002] Sen, S. and Sajja, N. (2002). Robustness of reputation-based trust : Boolean case. In *Proceedings of the 1st International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 288–293. ACM Press.

[Silva et al., 2009] Silva, V. T., Hermoso, R., and Centeno, R. (2009). Coordination, organizations, institutions and norms in agent systems iv. chapter A Hybrid Reputation Model Based on the Use of Organizations, pages 111–125. Springer-Verlag, Berlin, Heidelberg.

[Simonin and Ferber, 2003] Simonin, O. and Ferber, J. (2003). Un modèle multi-agent de résolution collective de problèmes situés multi-échelles. In *JFSMA 2003, RSTI/hors série*.

[Smith, 1980] Smith (1980). The Contract Net Protocol : High-Level Communication and Control in a Distributed Problem Solver. *IEEE Transactions on Computers*, C-29(12) :1104–1113.

[Sole and Bascompte, 2011] Sole, R. and Bascompte, J. (2011). *Self-Organization in Complex Ecosystems. (MPB-42)*. Monographs in Population Biology. Princeton University Press.

[Steeb et al., 1988] Steeb, R., Cammarata, S., Roth, H. F. A., Thorndyke, P. W., and Wesson (1988). Distributed intelligence for air fleet control : Architectures for distributed air traffic control. *Readings in Distributed Artificial Intelligence*, 101(3) :90–101.

[Surmann et al., 1993] Surmann, H., Kanstein, A., and Goser, K. (1993). Self-organizing and genetic algorithms for an automatic design of fuzzy control and decision systems. In *Proc. EUFIT'93*, pages 1097–1104.

[Sycara, 1988] Sycara, K. (1988). Utility in conflict resolution. *Annals of Operation Research*, 12.

[Sycara et al., 1991] Sycara, K., Roth, S., Sadeh, N., and Fox, M. (1991). Distributed constrained heuristic search. *Ieee Transactions On Systems Man And Cybernetics*, 21(6) :1446–1461.

[Sycara, 1990] Sycara, K. P. (1990). Persuasive argumentation in negotiation. *Theory and Decision*, 28(3) :203–242.

[Taillandier et al., 2010] Taillandier, P., Vo, D.-A., Amouroux, E., and Drogoul, A. (2010). Gama : A simulation platform that integrates geographical information data, agent-based modeling and multi-scale control. In Desai, N., Liu, A., and Winikoff, M., editors, *PRIMA*, volume 7057 of *Lecture Notes in Computer Science*, pages 242–258. Springer.

[Teacy et al., 2006] Teacy, W. T., Patel, J., Jennings, N. R., and Luck, M. (2006). Travos : Trust and reputation in the context of inaccurate information sources. *Autonomous Agents and Multi-Agent Systems*, 12 :183–198.

[Toffoli, 1984] Toffoli, T. (1984). Cellular automata as an alternative to (rather than an approximation of) differential equations in modeling physics. *Physica Nonlinear Phenomena*, 10 :117 – 127.

[Ulam, 1962] Ulam, S. (1962). On Some Mathematical Problems Connected with Patterns of Growth and Figures. In *American Mathematical Society Proceedings of Symposia in Applied Mathematics*, volume 14, pages 215–224. American Mathematical Society.

[Urbano et al., 2009] Urbano, J., Rocha, A. P., and Oliveira, E. (2009). Computing confidence values : Does trust dynamics matter ? In *Proceedings of the 14th Portuguese Conference on Artificial Intelligence : Progress in Artificial Intelligence*, EPIA '09, pages 520–531, Berlin, Heidelberg. Springer-Verlag.

[Van Dyke ParunaK et al., 2002] Van Dyke ParunaK, H., Brueckner, S., and Sauter, J. (2002). Digital pheromone mechanisms for coordination of unmanned vehicles. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems : part 1*, AAMAS '02, pages 449–450, New York, NY, USA. ACM.

[Vercouter and Muller, 2010] Vercouter, L. and Muller, G. (2010). L.I.A.R. : Achieving social control in open and decentralized multiagent systems. *Appl. Artif. Intell.*, 24 :723–768.

[von Foerster, 1960] von Foerster, H. (1960). *Self-Organizing Systems*, volume On Self-Organizing Systems and Their Environments. Pergamon Press, London.

[Wagner, 1995] Wagner, P. (1995). Traffic simulations using cellular automata : Comparison with reality. In *Traffic and granular flow*. World Scientific.

[Weinberg, 2001] Weinberg, G. M. (2001). *An Introduction to General Systems Thinking*. Dorset House Publishing, New York, USA, Silver Anniversary edition.

[Weiss, 1999] Weiss, G. (1999). *Multiagent systems : a modern approach to distributed artificial intelligence*. MIT Press, Cambridge, MA, USA.

[Weyns et al., 2004] Weyns, D., Schelfthout, K., Holvoet, T., Glorieux, O., and Distrinet, A. (2004). A role based model for adaptive agents. In *Fourth Symposium on Adaptive Agents and Multiagent Systems at the AISB '04 Convention*.

[White and Engelen, 1993] White, R. and Engelen, G. (1993). Cellular Automata and Fractal Urban Form : a Cellular Modelling Approach to the Evolution of Urban Land use Patterns. *Environment and Planning A*, 25 :1175–1199.

[Wilensky and Evanston, 1999] Wilensky, U. and Evanston, I. (1999). Netlogo. Center for connected learning and computer based modeling. Technical report, Northwestern University.

[Wolfram, 1986] Wolfram, S. (1986). *Theory and Applications of Cellular Automata : Including Selected Papers, 1983-1986*. Advanced Series on Complex Systems. World Scientific.

[Wooldridge and Jennings, 1994] Wooldridge, M. and Jennings, N. R. (1994). Intelligent agents : Theory and practice. *Knowledge Engineering Review*. Submitted to Revised.

[Xiong and Liu, 2004] Xiong, L. and Liu, L. (2004). PeerTrust : Supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Transactions on Knowledge and Data Engineering*, 16 :843–857.

[y López et al., 2002] y López, F. L., Luck, M., and d'Inverno, M. (2002). Constraining autonomy through norms. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems : part 2*, AAMAS '02, pages 674–681, New York, NY, USA. ACM.

[Yamauchi, 1997] Yamauchi, B. (1997). A frontier-based approach for autonomous exploration. In *In Proceedings of the IEEE International Symposium on Computational Intelligence, Robotics and Automation*, pages 146–151.

[Ye et al., 2010] Ye, D., Zhang, M., and Sutanto, D. (2010). Self-organisation in an agent network via learning. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems : Volume 1*, AAMAS '10, pages 1495–1496, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

[Yu and Singh, 2000] Yu, B. and Singh, M. P. (2000). A social mechanism of reputation management in electronic communities. In *Proc. of CIA '00*, pages 154–165. Springer-Verlag.

[Yu and Singh, 2002] Yu, B. and Singh, M. P. (2002). Emergence of agent-based referral networks. In *Proceedings of the first international joint conference on Autonomous agents and multiagent systems : part 3*, AAMAS '02, pages 1268–1269, New York, NY, USA. ACM.

[Zacharia and Maes, 2000] Zacharia, G. and Maes, P. (2000). Trust management through reputation mechanisms. *Applied Artificial Intelligence*, 14(9) :881–907.

[Zhang et al., 2009] Zhang, C., Abdallah, S., and Lesser, V. (2009). Integrating organizational control into multi-agent learning. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '09, pages 757–764, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

[Zia, 2008] Zia, T. A. (2008). Reputation-based trust management in wireless sensor networks. In *2008 International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, pages 163–166. IEEE.