Etude de l'émergence comportementale d'un collectif de robots par auto-organisation coopérative

Rapport de DEA
« Représentation des Connaissances et Formalisation du Raisonnement »
Année 2000-2001

PICARD Gauthier

Directeur de Recherche : Jean-Luc SOUBIE
Responsables de Stage : Marie-Pierre GLEIZES et Pierre GLIZE
Equipe d'Accueil : Systèmes Multi-Agents Coopératifs, IRIT

Mots clés : Systèmes Multi-Agents, auto-organisation, coopération, apprentissage, Robotique Collective, émergence.

Résumé : Aujourd'hui, les progrès technologiques en Robotique permettent aux concepteurs d'envisager la résolution de tâches difficiles par des robots. La complexité de conception pour le contrôle de tels robots croît en conséquence et les concepteurs se trouvent face à leurs propres limites de compréhension. C'est à cela que la Robotique Collective essaie de répondre par l'interaction et la coopération au sein de communautés de robots ; encore faut-il disposer de méthodes capables de garantir la maîtrise du comportement collectif.

C'est pourquoi, il semble naturel d'appliquer la théorie des Systèmes Multi-Agents Adaptatifs au domaine de la Robotique Collective dans des exemples de comportements de groupe comme la mise en place d'une circulation dans un collectif de robots transportant des ressources. Les résultats de ce travail montrent que la théorie permet effectivement l'apparition de comportements émergents cohérents. Tout en considérant le cas de laboratoire que constitue mon application, cela ouvre de nombreuses perspectives dans ce domaine.

REMERCIEMENTS

Je tiens à remercier Marie-Pierre Gleizes et Pierre Glize pour m'avoir fait découvrir l'univers des Systèmes Multi-Agents et m'avoir laissé tant de liberté dans mon exploration du domaine et ma découverte du monde de la recherche.

Merci à toute l'équipe SMAC de l'IRIT – Carole Bernon, Christine Régis, Sylvie Trouilhet, Bernard Carpuat, André Machonin, … – et leurs stagiaires, ainsi qu'à Jean-Luc Soubie pour leur accueil chaleureux et leur ouverture d'esprit.

Merci à Jean-Pierre Georgé, doctorant dans l'équipe, pour ses conseils éclairés.

Merci à Davy Capera, étudiant en DEA dans l'équipe, pour les interminables discussions et les nombreuses idées que j'ai échangées avec lui.

Enfin, un grand merci à Michel Cayrol et toute l'équipe pédagogique du DEA RCFR, pour m'avoir transmis un peu de leur savoir.

SOMMAIRE

INTRODUCTION

Les Systèmes Multi-Agents (ou SMA) abordent aujourd'hui la résolution **de problèmes complexes et non linéaires** que des systèmes classiques ne sont pas à même de résoudre efficacement. Ces problèmes, comme la prévision de crues, le courtage en ligne ou bien la résolution de systèmes d'équations, ont été la motivation pour l'équipe SMAC de l'IRIT de mettre en place une théorie pouvant y répondre. La Robotique connaît des problèmes similaires. Que ce soit pour l'établissement de bases spatiales, le déminage de zones à risque, ou le transport de ressources, la difficulté de compréhension ou de conception, commence à être trop grande pour faire résoudre ses tâches par un unique robot. C'est ainsi qu'apparut la Robotique Collective, inspirée des communautés d'insectes sociaux comme les abeilles ou les fourmis.

La résolution de problèmes de haut niveau par des unités de bas niveau et l'apparition de nouvelles fonctionnalités au sein de collectifs sont aussi une des motivations de la théorie des Systèmes Multi-Agents Adaptatifs (ou AMAS). Elle s'inscrit dans un mouvement, appelé **émergence,** qui motive de nombreux chercheurs de par le monde. En effet, divers groupes de réflexion, comme l'atelier PRESCOT à Toulouse, existent actuellement ou se forment pour étudier ce phénomène encore mal connu.

C'est à ce carrefour, entre SMA, Robotique et émergence, qu'une étude de comportements émergents s'avère être intéressante. En effet, l'équipe SMAC se pose **la question de la pertinence de la Théorie des Systèmes Multi-Agents Adaptatifs appliquée** à la Robotique en général et **à la Robotique Collective** en particulier. Pour répondre à cette question, je propose une étude de l'apparition d'un comportement émergent au sein d'un collectif de robots. La tâche affectée aux robots est le transport de ressources entre deux salles séparées par des goulets d'étranglement. De cet exemple devrait émerger un sens de circulation dans les couloirs afin de minimiser les interférences spatiales. Ne disposant pas de robots réels, il s'avère nécessaire de valider le système élaboré, composé des robots et de leur environnement, dans une plate-forme de simulation.

L'évolution de mon travail lors de ce stage s'est faite en trois étapes, correspondant aux trois parties principales de ce rapport :

1. Tout d'abord l'exploration des domaines mis en jeu dans le cadre de cette étude a abouti à l'élaboration d'un **état de l'art** (première partie) portant dans un premier temps sur les Systèmes Multi-Agents en général et sur la théorie des AMAS en particulier. Dans un deuxième temps, nous nous sommes intéressés au domaine de la Robotique Collective et des liens entretenus avec les SMA. Enfin, dans un dernier temps, je présente le concept d'émergence autour duquel s'articule mon étude.

2. Je développe, dans la deuxième partie, **l'exemple d'un comportement collectif émergent : la circulation**. Je présente un moyen d'élaborer un Système Multi-Agent à fonctionnalité émergente, mais je soulève aussi le problème de la nécessité pour les robots de posséder des capacités de mémorisation et d'apprentissage. Je montre que la mise en place d'un sens de circulation est une nécessité pour le collectif sous contrainte d'efficacité.

3. Le **problème de l'apprentissage** est développé dans la troisième partie. Je distingue divers modules pour les robots et les types d'apprentissage associés. Trois types de modules sont explorés.

Après la présentation de mon travail, je conclue sur la portée de cette étude et les perspectives de tels travaux.

Enfin, une brève annexe sur l'outil de simulation utilisé se trouve en fin de ce rapport. Je présente, dans cette section, *oRis*, un langage et une plate-forme de simulation pour Réalité Virtuelle, développée par l'Ecole Nationale d'Ingénieurs de Brest (ou ENIB).

1. ETAT DE L'ART : SYSTEMES MULTI-AGENTS ET EMERGENCE DANS LE CADRE DE LA ROBOTIQUE COLLECTIVE

Dès ses débuts, la Robotique Collective s'est rattachée au domaine des Systèmes Multi-Agents, comme le souligne Maja Matarić dans sa thèse [Matarić, 1994]. Il paraît donc intéressant et nécessaire de se plonger dans ce domaine pour comprendre les choix et les problèmes rencontrés au sein de collectifs de robots. Par ailleurs, nous verrons comment le concept d'émergence est apparu dans ce domaine et comment l'aborder pour l'élaboration de communautés de robots à fonctionnalités et comportements émergents. C'est dans ce cadre des systèmes complexes que se situe la théorie des Systèmes Multi-Agents Adaptatifs, développée par l'équipe SMAC de l'IRIT.

1.1 Les Systèmes Multi-Agents Adaptatifs (AMAS)

Un robot plongé dans un environnement hostile sera souvent exposé à des situations imprévues. Cette problématique n'est pas réservée au domaine de la Robotique, et c'est à ce genre de problèmes que la théorie des AMAS essaie de répondre. Cette approche théorique permet de concevoir des systèmes artificiels, voire physiques comme des robots, destinés à être immergés dans un environnement dynamique.

1.1.1 Historique des Systèmes Multi-Agents et de l'Intelligence Artificielle Distribuée

Face à la complexification des tâches à exécuter - complexité temporelle, géographique ou de traitement - et à l'évolution des technologies, notamment l'apparition des machines multiprocesseurs ou en réseaux, l'Intelligence Artificielle a donné naturellement naissance à *l'Intelligence Artificielle Distribuée* (ou IAD) et aux *Systèmes Multi-Agents* (ou SMA). L'évolution de ce domaine peut être segmentée comme suit [Ferber, 1995] :

- Les débuts de l'Intelligence Artificielle Distribuée (1973)

Cette première génération de *SMA*, née **Intelligence Artificielle Distribuée**, fut dédiée à la **résolution distribuée de problèmes** par distribution des connaissances (plusieurs bases de connaissances) et des traitements (plusieurs systèmes experts), avec **contrôle centralisé** sur ces différents systèmes. Les principaux travaux furent : HEARSAY II de Herman et Lesser (1973-1975), un système de reconnaissance de la parole ; les notions d'Acteurs de Hewitt (1977) et de Beings (ou Êtres) de Lenat (1975).

- L'évolution vers l'Intelligence Artificielle Distribuée décentralisée (1982)

Cette seconde génération fut caractérisée par la **distribution du contrôle des agents** et par le souci de **réutilisabilité** (notion de générateurs de systèmes). Cette distribution décentralisée

implique que les agents n'aient aucun contrôle sur ses pairs ni aucune connaissance globale sur le système. On peut citer en autres travaux : DVMT de Lesser et Corkill (1983), un système complexe de reconnaissance de situations pour le trafic routier, et le célèbre MACE de Gasser (1987), avec sa notion de plate-forme générique.

- La diversité des systèmes d'aujourd'hui (1990)

C'est en partie dans les domaines de *l'éthologie* et de la *vie artificielle* que l'IAD va puiser ses dernières inspirations. C'est ainsi que les notions d'**autonomie** (de contrôle surtout), d'**hétérogénéité**, et d'**interaction** entre agents membres d'une **société**, ont vu le jour dans le domaine des Systèmes Multi-Agents dont une des motivations première est de modéliser des systèmes sociaux ou naturels (Brooks, Steels, Deneubourg, Ferber, Drogoul, …).
D'autres directions, sur un axe plus formel, furent explorées en parallèle : la *Formalisation Logique* avec Castelfranchi (1991) par exemple ; les travaux de Wooldridge et Jennings ; les *Actes de Langages* ; ou bien la *Théorie des Jeux et les SMA* de Rosenschein (1992).

C'est dans la branche « vie artificielle et éthologie », où les notions d'interaction et d'**accointance** prédominent sur la notion d'agent, que le principe d'**émergence** fit son apparition dans le domaine des SMA. Ces systèmes sont capables de réaliser leur fonction globale, sans que leurs agents n'en aient la moindre connaissance, mais via les interactions et la pression de l'environnement dans lequel est plongé le système. C'est dans cette mouvance que la théorie des *Adaptive Multi-Agent System* (ou AMAS) [Camps, 1997], et donc mon étude, vont se situer.

1.1.2 Une définition générale d'un agent

Malgré le caractère imprécis que comporte la notion d'agent, certains, comme Jacques Ferber, ont tenté d'en établir une définition minimale [Ferber, 1995]. De manière générale et assez expressive, pour mon étude, on peut dire qu'un agent est une entité autonome possédant :

- des **compétences** : c'est-à-dire ce qu'il sait faire, son rôle au sein du système ;
- des **connaissances** : partielles a priori, sur lui-même, sur ses pairs et sur son environnement;
- des **accointances** : attitude sociale ou relations avec d'autres agents ;
- des **aptitudes** : c'est-à-dire des capacités de perception, de décision, d'action, et éventuellement d'apprentissage.

Les agents sont destinés à être plongés dans un environnement afin d'effectuer en commun une certaine tâche. La tâche à effectuer sera déterminée par les compétences des agents présents dans le système. La distribution des tâches sera guidée par les accointances de ces agents. L'environnement peut être de plusieurs natures : logiciel, réseau (on parle d'agents mobiles), ou physique (en Robotique ou en automatique).

1.1.3 L'apprentissage dans les Systèmes Multi-Agents

Une des prétentions des Systèmes Multi-Agents est leur adaptabilité. Pourtant, un système ne pourra s'adapter que de manière limitée s'il n'apprend pas sur ses expériences vécues. C'est la raison pour laquelle les Systèmes Multi-Agents sont souvent munis de capacités d'apprentissages à des degrés divers. L'apprentissage des agents peut porter sur quatre sphères différentes et peut être effectué de multiples façons.

1.1.3.1 Les différents types d'apprentissage

Ces différents types définissent sur quoi porte l'apprentissage des agents. Beaune distingue quatre objets d'apprentissage [Topin, 1998].

L'**apprentissage centré agent** concerne ce qu'un agent peut apprendre sur lui-même ou sur les autres agents. Cet apprentissage porte sur le comportement de l'agent, ses stratégies, ses décisions. Il peut par exemple s'effectuer par attribution de récompenses ou de punitions lorsque l'agent agit. Lorsqu'il doit apprendre sur les autres, l'agent doit bien sûr les connaître, mais aussi posséder des croyances en ces agents.

L'**apprentissage centré environnement** se focalise sur ce que l'environnement peut apprendre à l'agent. Cela peut porter sur de nombreux objets tant la diversité des environnements peut être grande. En effet, dans un environnement à forte dynamique, un agent peut apprendre sur des parties lui étant nouvellement apparues. Ce type d'apprentissage se retrouve naturellement dans des systèmes de vie artificielle. Cet apprentissage rejoint souvent l'apprentissage centré agent, compte tenu de la forte interactivité environnement-agent.

L'**apprentissage centré interaction** porte sur les moyens mis en œuvre par les agents pour communiquer ou interagir. Cela peut être l'apprentissage de nouveaux langages d'interactions ou bien de nouveaux thèmes de communication.

Enfin, l'**apprentissage centré organisation** s'occupe de faire évoluer les rôles des agents au sein de leur société, comme par exemple dans une société de fourmis ou d'abeilles, lorsque les agents sont incapables de résoudre une tâche sans partenaires. Lorsque le système est supervisé, les agents sont guidés dans leur démarche, alors que pour des systèmes d'agents autonomes, les agents doivent apprendre à s'auto-organiser. Ce type d'apprentissage apparaît aussi très souvent dans les systèmes de vie artificielle.

1.1.3.2 Les différentes techniques d'apprentissage

Pour effectuer ces différents types d'apprentissages, plusieurs moyens ont été développés. L'apprentissage peut être la conséquence d'attribution directe ou indirecte de récompenses aux agents, on parle alors d'algorithmes par **renforcement**. Il peut aussi être induit par la **coopération et l'interaction** au sein d'un groupe. Je présente dans cette section quelques exemples de techniques utilisées dans les Systèmes Multi-Agents ou en Robotique.

L'**apprentissage par renforcement** se divise en plusieurs courants :

- une **comparaison entre l'action qu'un agent effectue et une action désirée ou théorique** est menée par un système de niveau supérieur, ou superviseur. Ce *feedback* permet à l'agent de modifier sa fonction afin de corriger son action. Les systèmes de classifieurs de John H. Holland en sont un exemple.
- une **mesure de l'utilité** d'un agent au sein du collectif. Le but de l'agent sera alors de maximiser son utilité. Une fois encore, cette technique nécessite la présence d'une entité de supervision d'ordre supérieur.
- pour les systèmes non-supervisés, l'agent doit apprendre par lui-même ce qui est présupposé bon pour le collectif. **L'auto-organisation** en est un exemple très intéressant. Lorsqu'un agent s'aperçoit qu'il a mal agit, soit parce qu'il s'en est rendu compte, soit parce qu'un autre agent l'informe sur la conséquence de son agissement, il remet en question ses croyances sur lui et/ou ses accointances. La théorie des AMAS, que je présenterai dans la section suivante utilise ce type d'apprentissage.

L'**apprentissage par interaction**, consiste en la capacité d'un agent à apprendre grâce aux interactions coopératives qu'il a avec ses congénères. Les agents doivent alors avoir à leur disposition des moyens de communication et des protocoles associés à l'apprentissage. Par exemple, deux robots fourrageurs se rencontrent. L'un d'entre eux revient d'une zone riche en ressources à ramener, et l'autre cherche depuis longtemps des ressources sans en trouver. L'apprentissage par interaction peut être considéré comme la donnée de l'information « il y a des ressources là-bas » par un robot à un autre. Le robot recevant le message a appris qu'une zone fertile en ressources existait et qu'elle se trouvait à un endroit précis.

Pour mettre en place l'apprentissage d'agents, nous disposons de nombreux algorithmes. Je vais en présenter succinctement quelques-uns.

Les **systèmes de classifieurs** : l'agent dispose d'un ensemble de classifieurs, c'est-à-dire des règles de production de faits. Une règle se déclenche en fonction de l'entrée donnée. On associe à chaque règle un poids, lui permettant de s'activer ou non. Plus le poids d'un classifieur est élevé, plus le classifieur a de chance de produire une action. Lorsque l'agent produit une action néfaste, une entité supérieure l'avertit. L'agent va alors baisser les poids des classifieurs ayant produits les actions néfastes. A l'inverse, lorsque l'agent a bien agi, les poids de ses classifieurs augmentent afin qu'ils aient plus de chance de se déclencher dans le futur.

A ces systèmes, on peut ajouter des mécanismes de production de règles, par algorithmes génétiques par exemple.

L'**algorithme du Bucket-Brigade**, est fondé sur les systèmes de classifieurs. Il a aussi été élaboré par Holland. Ici, des classifieurs sont associés à chaque agent. On considère que les agents agissent à tour de rôle, et un système de récompenses au niveau des agents est mis en place de la même façon que précédemment.

L'**algorithme de Q-Learning** associe à chaque paire action-état un poids. Ce poids est mis à jour à chaque action de l'agent. Cet algorithme nécessite encore une fois qu'un superviseur gère la table des paires des agents, en augmentant ou diminuant les poids des paires. Cette mise à jour prend en compte les actions passées que l'agent a effectuées. Ensuite le fonctionnement est simple, lorsqu'un agent veut agir, il consulte la table, et en fonction de son état il déclenchera l'action, correspondant à la paire dont le poids est le plus élevé. Cet algorithme est couramment utilisé en Robotique Collective, notamment par Maja Matarić [Matarić, 1994].

L'apprentissage effectué grâce à un superviseur peut permettre aux robots d'apprendre sur la totalité des expériences vécues par le groupe. L'autonomie des agents nécessite que tous soient capables d'apprendre, ce qui peut ralentir l'apprentissage du collectif car tous les robots devront faire face à de nombreuses situations, alors qu'un superviseur apprendra sur la totalité des expériences de tous les robots. Cet apprentissage centralisé devrait donc faire converger le collectif vers la fonction désirée plus rapidement qu'un apprentissage distribué. La contrepartie de la supervision est la difficulté de mise en œuvre de tels systèmes. Le superviseur doit être une entité complexe ayant de larges connaissances. L'apprentissage par auto-organisation par contre ne nécessite qu'une mise en place simple au niveau des agents. Cette spécification au niveau des agents pour atteindre une fonction globale au niveau du collectif correspond ainsi parfaitement à l'idée d'émergence, concept qui est une des priorités de mon étude.

Ce bref aperçu des techniques d'apprentissages ne peut que renforcer mon intérêt pour la théorie des AMAS où l'apprentissage par auto-organisation ne nécessite aucun superviseur, garantissant l'autonomie des agents.

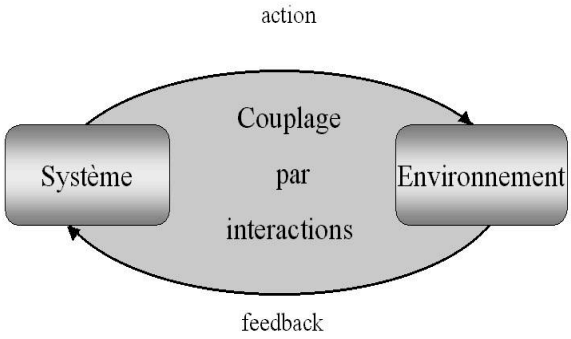
1.1.4 Une théorie Multi-Agent : Adaptive Multi-Agent System

L'équipe SMAC a développé cette théorie pour des Systèmes Multi-Agents destinés à être immergés dans des environnements à forte évolutivité (comme un écosystème). De tels environnements voient apparaître de nouvelles composantes ou disparaître des composantes existantes contrairement à des environnements clos ou à faible évolutivité (comme un terrain de football), où les composantes changent rarement. La théorie repose sur quelques principes de base que je vais exposer.

1.1.4.1 Présentation de la théorie

Tout d'abord cette théorie est une approche de l'auto-organisation par coopération. Cela signifie que les agents du système vont évoluer vers une bonne répartition des tâches à effectuer, en fonction de leurs compétences propres afin que ce système soit **fonctionnellement adéquat**. Cette auto-organisation est à opposer à des organisations dites statiques, où la place de chaque agent est prédéterminée. Ce genre d'organisation prédéfinie manque de flexibilité et d'adaptation à des milieux fortement dynamiques.

Une des forces des systèmes auto-organisateurs est le couplage par interactions : le système perçoit son environnement et est capable d'agir sur lui. C'est par ce jeu d'interactions entre le système et l'environnement que l'auto-organisation sera possible.



Les interactions entre le système et son environnement, ou entre les agents eux-même peuvent être :

- **coopératives** : l'activité de l'un favorise celle de l'autre (exemple : la symbiose ou les écosystèmes) ;
- **antinomiques** : l'activité de l'un nuit à celle de l'autre (exemple : partage de ressources limitées) ;
- **indifférentes** : l'activité de l'un ne nuit pas à celle de l'autre mais ne la favorise pas non plus (exemple : deux systèmes totalement isolés).

Du point de vue de la théorie des AMAS, un moyen d'aboutir à l'adéquation fonctionnelle est d'avoir des agents coopératifs. Cela signifie que de tels agents vont pouvoir travailler localement au sein de collectifs cohérents pour exécuter la fonction principale du système ou une fonction partielle de cette fonction principale. De tels regroupements peuvent alors être considérés comme des agents de niveau supérieur car leur fonction est inconnue par les agents de niveau inférieur.

Les agents, qui possèdent les mêmes caractéristiques que les agents décrits dans la section 1.1.2, sont en outre coopératifs, c'est-à-dire que l'agent devra avoir les **attitudes sociales** suivantes :

- être **sincère** : toute information communiquée doit lui sembler vraie ;
- être **serviable** : l'agent cherche à faire son travail ;
- être **bien intentionné** : l'agent communique à ses accointances les informations lui semblant utiles;

- être **confiant** : l'agent croit les autres agents comme serviables, sincères et bien intentionnés.

1.1.4.2 L'apprentissage dans les AMAS

En observant de plus près ces attitudes, on peut entrevoir les concepts sur lesquels un agent peut être capable d'apprendre et de s'adapter [Camps, 1997]. En effet la capacité d'apprentissage est parfaitement possible avec de tels agents. On distingue trois niveaux d'apprentissage par auto-organisation:

- Niveau 0 : **adaptation**

L'agent n'apprend pas. Il possède des croyances et des compétences fixes tout au long de sa vie. Un tel agent sait s'il est en situation non-coopérative (voir section 1.1.4.3), peut résoudre cette situation mais n'en apprend rien.

- Niveau 1 : **apprentissage sur les croyances**

L'agent révise ses croyances par auto-organisation. Un tel système est par exemple le réseau de croyance de ABROSE [Athanassiou, 1999] ou ARCADIA [Camps, 1997]. Les croyances sont représentées par un module de croyances qui est lui-même un AMAS dont les agents vont s'organiser en fonction des expériences de l'agent. Lorsque l'agent se trouve face à une situation non-coopérative, il propage ce problème aux AMAS du module de croyance qui s'adaptera en conséquence. Les croyances peuvent être de diverses natures : croyance sur soi, sur les autres agents ou sur son environnement.

- Niveau 2 : **apprentissage sur les croyances et les compétences**

L'agent apprend sur ses croyances et ses compétences par auto-organisation. Comme précédemment l'agent possède un module de croyance plus un module de compétence qui fonctionne de la même manière mais sur les compétences de l'agent.

1.1.4.3 Les situations non-coopératives (ou SNC)

Un agent peut apprendre à la suite d'expériences qu'il aura vécues. Ces expériences seront des violations des règles de coopération que l'on appelle les **situations non-coopératives** (ou SNC). Un agent est en SNC s'il ne satisfait pas au moins une de ces conditions :

1- tout signal perçu peut être interprété sans ambiguïté par l'agent, c'est la **compétence** ;
2- l'information permet d'aboutir à des conclusions, c'est la **productivité** ;
3- les conclusions doivent être utiles à autrui, c'est l'**utilité**.

Ces SNC vont à l'encontre du bon déroulement de la fonction globale, et un agent doit être capable de les percevoir localement, quelle que soit son origine : environnement ou autre agent. La solution consiste en un recensement exhaustif des SNC déterminées par les compétences de l'agent et de leur associer une action permettant de revenir dans un état dit coopératif. Pour un agent, coopérer peut donc se résumer à détecter les SNC puis agir pour revenir dans un état coopératif.

1.1.4.4 Quelques applications de la théorie des AMAS

Diverses applications ont été développées dans le cadre de cette théorie : un **système de prévision de crues**, STAFF [Sontheimer, 1999], et notamment un **système distribué de courtage** en ligne, ABROSE [Athanassiou, 1999]. Ce dernier est un exemple parfait de la capacité d'adaptation de tels systèmes dans un environnement extrêmement dynamique : une communauté de personnes s'échangeant des demandes et des services.

Jean-Pierre Georgé, doctorant dans l'équipe, explore une application originale des AMAS : un **environnement de programmation émergente** [Georgé, 2000]. Dans ce système, les agents sont des instructions du langage qui doivent s'auto-organiser pour effectuer une procédure. L'étude faite lors de son DEA ouvre un large horizon quant à l'application et la conception de systèmes complexes auto-organisateurs.

Devant tant d'applications et de domaines différents, le besoin de définir une **méthodologie de conception** de tels systèmes à fonctionnalités émergentes a conduit à la mise en place du projet ADELFE (Atelier pour le DEveloppement de Logiciels à Fonctionnalité Emergente), pour mettre en place un tel outil de spécification.

Cette théorie est donc un très bon outil pour la réalisation de systèmes complexes, où l'on spécifiera au niveau local (ou micro-niveau) pour s'adapter au niveau global (ou macro-niveau). Cette approche local-global (ou *bottom – up*) ouvre la voie à une notion alternative pour la conception de systèmes non-linéaires : l'**émergence** (voir section 1.3).

1.2 Le domaine de la Robotique Collective

Les Systèmes Multi-Agents ont très vite entretenu d'étroites relations avec le domaine de la Robotique. Que ce soit en *Robotique Cellulaire*, où un robot est considéré comme un Système Multi-Agent dont les agents sont les composantes du robot, ou en *Robotique Collective* à proprement parler, où plusieurs robots-agents doivent accomplir une tâche commune ; la conception orientée agent a toujours trouvé naturellement sa place.

Ce domaine est en plein essor depuis le milieu des années 90, et les groupes de recherches et laboratoires se développent de plus en plus. On peut citer notamment le Laboratoire d'Informatique de Paris VI (LIP6) où œuvre Drogoul, l'Interaction Lab de l'University of South California (USC) ou bien le Mobile Robotics Institute du Georgia Institute of Technology, qui ont été des sources assez complètes pour l'écriture de cette section en particulier et ma culture personnelle en général.

1.2.1 Les motivations

Face à ces évidences, les motivations des recherches en Robotique Collective ont naturellement rejoint celles qui ont poussé les chercheurs à développer les Systèmes Multi-Agents.

Entre autres motivations apparaît la volonté de concevoir des communautés de robots **robustes** aux variations de l'environnement. En effet, l'expérience en exploration spatiale et l'utilisation de robots fonctionnant en solitaires, ont montré la limite de l'utilisation d'entités de forte granularité (robots fortement cognitifs) trop sensibles aux perturbations de l'environnement et aux pannes. De plus le domaine de la Robotique a de grands besoins financiers lorsque l'on parle de robots complexes, alors que l'on peut espérer que plusieurs robots de complexité(s) inférieure(s) seront d'un **coût bien moindre**. Les espoirs se tournent aussi vers la **rapidité** et l'**efficacité** d'un collectif de robots en comparaison à un unique robot de grande complexité qui doit gérer seul toutes les tâches.

1.2.2 Contrôle d'un robot

Cette section traite du contrôle individuel des robots, c'est-à-dire de la manière de contrôler les actions d'un robot. Cette classification que fait Maja J. Matarić [Matarić, 1997],

part de la classique planification pour arriver aux architectures purement réactives ou basées sur le comportement.

La première approche est le **contrôle délibératif**. C'est dans ce type de contrôle que se trouvent les stratégies traditionnelles basées sur un plan, ou délibératives. Ce contrôle consiste en la construction d'un modèle du monde puis en l'élaboration d'un plan, ou séquence d'actions, basé sur la connaissance du monde. La complexité de construction de ce plan rend difficile les actions temps réel, et induit un manque de souplesse face aux changements survenus dans l'environnement (nécessité de reconstruire un nouveau plan). De plus, la mise en place d'un tel contrôle implique une simplification importante de la complexité du monde que le robot perçoit.

La deuxième approche consiste en une vision **purement réactive** du contrôle. Ce genre de contrôle est construit autour d'une collection de paires conditions-actions dont la représentation peut être de types variés (tables, réseaux connexionnistes, classifieurs, …), mais n'ont aucune représentation du monde réel. Les robots réactifs obtenus sont idéaux pour les applications en temps réel, mais nécessitent une spécification complète du problème.

La troisième approche consiste en un compromis entre le contrôle délibératif et le contrôle réactif. Ce **contrôle hybride** peut se diviser en deux sous-contrôles de niveaux différents. Tout d'abord un contrôle de bas niveau, gérant les réflexes de sauvegarde, purement réactifs comme l'évitement d'obstacles par exemple. Un contrôle de haut niveau gère l'exécution des séquences d'actions par planification. Ce type de contrôle est certainement le plus efficace aujourd'hui, bien qu'il ait un manque certain de flexibilité.

Une quatrième approche, issue du contrôle réactif, est le **contrôle basé sur le comportement** (ou behavior-based control) développé par Brooks en 1986 [Brooks, 1986]. Un robot possède un ensemble de comportements simples qui seront activés lorsque le robot se trouvera dans une situation particulière. Le choix du comportement à un instant donné peut être de deux natures différentes : arbitrage (subsomption de Brooks, décision bayesienne, …) ou fusion de comportements (procédures de vote, ensembles flous, …). Ce type de contrôle améliore la flexibilité du contrôle réactif car il est plus modulaire par essence, mais plus complexe car souvent à fonctionnalité émergente (voir section 1.3).

Dans la suite, je placerai mon étude dans une approche basée sur le comportement. Une des difficultés de ce travail sera donc de trouver une bonne méthode d'arbitrage ou de fusion des comportements.

1.2.3 Contrôle d'un collectif de robots

Trois politiques différentes de gestion d'un collectif de robots ont été explorées : une **approche supervisée**, une **approche partiellement distribuée avec « coach »** et enfin une **approche totalement distribuée**. En fin de section, la question de la portée de l'hétérogénéité dans une communauté de robots sera abordée, et sera développée dans la section suivante.

L'idée de **collectif supervisé** est assez simple : l'ensemble des robots est dirigé par un unique contrôleur (ou superviseur). On peut facilement imaginer un ordinateur contrôlant les faits et gestes de robots par radioguidage. Cette vision du collectif est bien sûr très peu populaire car le système possède un point de rupture, le superviseur. En effet, si le superviseur vient à tomber en panne, le collectif n'est plus à même d'effectuer sa tâche, car les robots seront immobilisés. Un autre inconvénient d'une telle approche est la complexité du contrôleur qui croît avec le nombre de robots, ce qui limite fortement le nombre de robots.

Une autre approche, plus robuste, réside en l'idée d'un **collectif « coaché »**. La métaphore du coach (ou entraîneur) que l'on peut découvrir dans l'article de Luc Julia [Drogoul, 1998], est utilisée pour un collectif de robots appliquant des stratégies qui lui sont suggérées par une entité appelée coach. Le coach est un agent (humain ou logiciel) capable d'analyser la situation à un moment donné et d'établir une stratégie adaptée à la situation. Un exemple simple est celui d'une équipe de robots footballeurs guidés par un entraîneur logiciel. Ici les robots sont autonomes mais seront moins efficaces si l'entraîneur venait à tomber en panne. Enfin, le coût de développement d'une telle équipe sera en général supérieur à une approche purement autonome, car l'entraîneur est un agent de très forte granularité, très cognitif et donc difficile à élaborer.

L'approche la plus générique semble être **l'approche totalement distribuée**, où tous les robots sont indépendants. Cette vision trouve son origine dans l'éthologie et l'observation des communautés d'insectes (voir section 1.2.4).

On peut envisager, dans ce cas, des sociétés avec ou sans communication. Pour des robots communicants, il faut dresser des protocoles de communication, ou langages d'interactions, comme dans le domaine des Systèmes Multi-Agents, et bien sûr prévoir le matériel adéquat et le coût associé.

Les robots peuvent être tous identiques (c'est le cas de la grande majorité des travaux dans le domaine) ou bien différents, comme par exemple dans l'acte de Andreas Birk et de Tony Belpaeme [Drogoul, 1998]. Dans le cas où tous les robots sont identiques, logiciellement (du point de vue du contrôle) et physiquement, le coût est bien moindre que dans le cas où il faut développer un organe de contrôle différent pour chacun des agents. C'est ainsi que, naturellement, la question de l'influence de l'hétérogénéité sur le collectif se pose.

C'est dans ce contexte que Tucker Balch [Balch, 1997] introduit de nouvelles notions et les métriques associées afin d'étudier l'hétérogénéité ou l'homogénéité d'une société.

1.2.4 De l'hétérogénéité dans un collectif de robots

Les notions d'hétérogénéité et d'homogénéité qui suivent sont issues de l'article de 1997, « Social Entropy : a New Metric for Learning Multi-Robot Teams », de Tucker Balch [Balch, 1997], alors chercheur au Mobile Robotics Laboratory d'Atlanta.

Terminologie :

- **ROBOTS**
  - R_j est un robot.
  - R est une société de N robots avec $R = \{R_1, R_2, \ldots, R_N\}$.
- **CASTES**
  - C est une partition de R en c sous-ensembles.
  - C_i est un sous-ensemble de C.
  - les sous-ensembles de C sont des castes.

T. Balch définie la notion de **différence comportementale** (notée D) qui est calculée pour toutes les entrées reçues (par exemple voir un objet, tenir une boîte…) et pour les actions observées. Je ne donne pas la formule de cette métrique car sa donnée n'est pas indispensable pour la compréhension des notions importantes. Grâce à cette distance, il peut définir :

Définition *Soient R_a et R_b deux robots de C. On dit que R_a et R_b sont **ε-équivalents** si et seulement si $D(R_a,R_b) < \varepsilon, \ \forall \varepsilon > 0$*

Définition *Une société de robots est **ε-homogène** si et seulement si tous les robots sont ε-équivalents entre eux. De même une société de robots **est ε-hétérogène** si et seulement si il existe deux robots non ε-équivalents.*

Cette différence comportementale permet ainsi de comparer deux robots, et ainsi de construire des classes d'équivalence pour la relation d'équivalence associée à D. L'idée proposée est de regrouper les robots en classes d'équivalence pour former des castes, puis d'étudier la **diversité comportementale** de ces castes, c'est **l'entropie sociale**. Cette métrique doit être capable de prendre en considération les situations suivantes :

- la société la moins diverse est celle dans laquelle tous les agents sont équivalents ;
- la plus grande diversité est atteinte lorsque aucun agent n'est équivalent aux autres ;
- une société dans laquelle un seul agent diffère des autres est un peu plus diverse que celle où tous les agents sont équivalents ;
- si deux sociétés sont composées de groupes de taille uniforme, la plus diverse est celle où il y a le plus de castes ;

Soit Het(R) la fonction de mesure de l'hétérogénéité d'une société R :

- Het(R) = 0 ssi R est ε- homogène ;
- Het(R) sera à son maximum si tous les robots sont différents ;
- Het(R) sera à son minimum excepté 0 si seulement un robot diffère des autres ;
- Si R_a et R_b sont composés de castes de même tailles, Het(R_a) < Het(R_b) quand R_b possède plus de castes que R_a ;

Het est déterminée d'après l'étude de la *théorie de l'entropie de l'information* de Shannon :

$$\text{Het(X)} = -\sum_{i=1}^{c} p_i \log_2(p_i)$$

avec $p_i = \dfrac{|Ci|}{\sum_{j=1}^{c} |Cj|}$ la répartition des robots dans la caste C_i.

T. Balch a donc défini deux métriques facilement exploitables pour comparer la diversité ou la différence comportementale de populations de robots, qui peuvent s'avérer être des outils très utiles pour l'observation de collectifs de robots et répondre à la question suivante : Quelle(s) incidence(s) l'hétérogénéité a-t-elle sur l'efficacité d'un groupe pour effectuer sa tâche ?

On peut remarquer que si les robots disposent d'une capacité d'**apprentissage**, et qu'ils sont confrontés à des expériences totalement différentes, un fossé « comportemental » peut se creuser et **augmenter ainsi la différence comportementale** entre les robots et la diversité comportementale de la société ; et cela même si les robots sont identiques logiciellement (même programme de contrôle) et physiquement (même corps/châssis).

1.2.5 Les insectes sociaux : une source d'inspiration – Exemple de Kube et Zhang

Comme je l'ai suggéré précédemment, la Robotique Collective va puiser ses inspirations dans le domaine de l'éthologie et de l'étude des communautés d'insectes sociaux comme les fourmis, les abeilles ou les guêpes.

Cette approche basée sur le comportement ou purement réactive « caractérisée par un couplage direct de la perception à l'action » [Kube, 1993], a été abordée par de très nombreux chercheurs dans le cadre de la résolution de tâches collectives. Le but est de faire accomplir une tâche complexe à des robots organisés en groupes et mus par un mécanisme de contrôle très simple.

Kube et Zhang [Kube, 1993] proposent, après l'observation de communautés d'insectes, un mécanisme de contrôle en temps réel basé sur **cinq mécanismes de base**, l'idée principale étant de promouvoir la collaboration plutôt que l'effort individuel.

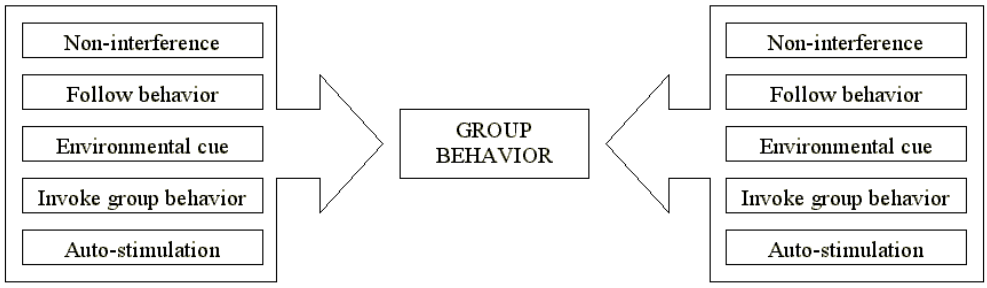


Figure 1 : *Cinq mécanismes pour un comportement de groupe.* [Kube, 1993]

Kube et Zhang appellent le premier de ces cinq mécanismes la **non-interférence**. Ce mécanisme consiste en un but commun que les robots possèdent et peuvent invoquer. C'est une forme simple de coopération où les robots n'interfèrent pas avec les actions des autres robots. Dans leur exemple Kube et Zhang implémentent un comportement de *robot-avoidance* (ou évitement des autres robots) comme mécanisme de non-interférence.

Le deuxième mécanisme est un **comportement de suivi** (ou *follow-behavior*). Ce mécanisme regroupe les robots en hordes afin d'effectuer leur tâche en communauté.

Le troisième mécanisme utilise une **réplique environnementale** pour invoquer le comportement de groupe. Kube et Zhang donnent l'exemple de fourmis dont la tâche est de nettoyer leur nid. Les fourmis déposent une substance photosensible sur les parois du nid, l'activité de nettoyage commençant le lendemain lorsque la substance a réagi avec la lumière du soleil.

Les robots doivent avoir la capacité d'**invoquer le comportement de groupe** lorsqu'ils se trouvent dans un collectif, c'est le quatrième mécanisme. Cela consiste en fait en la capacité pour un robot de reconnaître s'il se trouve dans un groupe à l'aide de senseurs appropriés.

Enfin le dernier mécanisme permet d'invoquer le comportement de groupe par **auto-stimulation**. Par exemple un robot, ayant trouvé une tâche à effectuer, va appeler les membres de son groupe à la rescousse pour venir l'aider dans sa tâche qu'il ne peut accomplir seul.

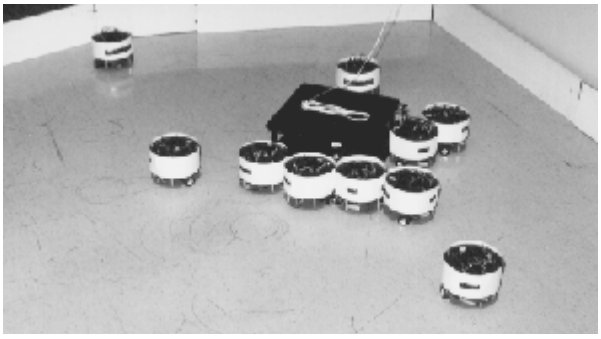


Figure 2 : *Les cinq mécanismes de Kube et Zhang sont utilisés pour effectuer la tâche du « box-pushing ».* [Kube, 1993]

Kube et Zhang ont utilisé ces mécanismes pour étudier le problème du « box-pushing » (fig. 2) qui est proche de l'objet de mon travail.

Ce jeu de mécanismes forme un exemple simple et frappant du souci de coller au plus aux modèles naturels, afin de reproduire des comportements de groupes complexes grâce à des structures de contrôle simples. C'est dans ce contexte d'étude que des concepts forts en Robotique Collective peuvent surgir, comme la **coopération** ou l'**émergence,** thème que je développerai dans la section 1.3 de cet exposé.

Pourtant, on peut soulever quelques questions sur ce jeu de mécanismes :

a- Cet ensemble de mécanismes est-il complet ?

Cette question se pose très souvent en Robotique avec contrôle basé sur le comportement. En effet, comment choisir un ensemble de comportements possibles suffisant pour traiter la tâche demandée ? Il est très difficile de le savoir, et pour certain des problèmes, une spécification, même très poussée, peut **omettre certains comportements**. Cette problématique est la même pour le jeu de mécanismes de Kube et Zhang, mais ce qui est certain c'est que pour les tâches étudiées, il est suffisant étant données les capacités dont disposent les robots.

b- Cet ensemble de mécanismes est-il correct ?

Ce problème reste relatif à l'axe de recherche que l'on désire suivre. Pour ma part, dans le cadre de l'étude de comportement émergent, il me semble dangereux d'indiquer aux agents leur comportement de groupe. Cela signifie que les agents, considérés comme étant au micro-niveau, ne devraient avoir qu'une connaissance limitée, voir nulle, sur le groupe, qui est le macro-niveau. Cette difficulté de caractérisation des systèmes émergents sera développée plus tard dans l'exposé.

1.2.6 La coopération en Robotique Collective

« Pour un agent, interagir avec un autre constitue à la fois la source de sa puissance et l'origine de ses problèmes. » [Ferber, 1995]

Comme dans le cadre des Systèmes Multi-Agents, l'interaction est un des problèmes majeurs de la Robotique Collective, qui sera réduit de manière générale au problème de la **coopération**. Jacques Ferber présente sa vision de la coopération comme une simple équation :

coopération = collaboration + coordination d'actions + résolution de conflits

La collaboration et la coordination représentent des formes particulières d'interactions. La **collaboration** consiste en l'affectation de chacun à des tâches adaptées à ses compétences. Cela peut se traduire par un protocole de communication d'offre et de demande pour des robots cognitifs ou bien par des mécanismes plus simples (voir section précédente) dans le cas de robots réactifs.

La **coordination d'actions** consiste en la bonne mise en œuvre d'une séquence d'actions nécessaire au bon accomplissement de la tâche allouée.

La **résolution de conflits** peut se faire par des mécanismes d'arbitrage ou de négociation. Les conflits peuvent apparaître lorsque les robots doivent partager des ressources communes. C'est souvent la résolution de ces conflits qui va plus nous intéresser dans le sens où elle est

un des principes de base de la théorie des AMAS [Camps, 1998], avec la notion de Situations Non-Coopératives (SNC) :

coopérer = détecter les SNC + résoudre les SNC

Cela signifie que la coopération nécessite que l'on détecte ces situations puis que l'on puisse effectuer des actions afin que l'agent retrouve un état coopératif. J'aurai le loisir de développer ces idées dans les sections 2 et 3.

1.2.7 Un exemple de résolution de conflits : la compétition agressive

Le problème soulevé dans cet article est la résolution des interférences spatiales par un comportement dit « **agressif** » [Vaughan & al, 2000]. Les auteurs exposent le problème d'un ensemble de robots autonomes qui a été conçu sans prendre en compte certains conflits comme celui de l'interférence spatiale.

Les robots peuvent être dans 3 états : **navigation, panique** ou **combat**. L'état de navigation correspond à l'activité de suivit de couloirs et de traversée de portes. L'état de panique est celui qu'un robot doit avoir lorsqu'un obstacle se présente. Enfin l'état de combat correspond à la résolution de conflit au niveau des portes.

La tâche est simple : un ensemble de robots doit amener d'un point A à un point B, des ressources en traversant des couloirs. La navigation se fait simplement en suivant les murs (ce n'est pas le principal problème).

Les observations suivantes sont faites :

- On ne tire aucun bénéfice à augmenter le nombre de robots, bien au contraire.
- Le problème se situe au niveau des portes qui agissent comme des goulets d'étranglement.

Les auteurs proposent alors leur stratégie anti-interférence : l'**agression**. C'est une agression non physique qui ne ralentit pas les robots dans leur tâche. Elle est utilisée lorsque des robots sont face à face dans un goulet. Elle consiste en la comparaison de valeurs détenues par les robots, le robot possédant la plus grande valeur gagnant le combat. Cette résolution des conflits a les caractéristiques suivantes :

- elle ne nécessite aucun senseur supplémentaire et seulement un léger changement du comportement de base,
- elle est totalement décentralisée,
- elle est indépendante de la stratégie de navigation,
- elle ne nécessite aucun travail de calcul complexe,
- elle est utile pour les sociétés de robots hétérogènes.

Les auteurs se sont intéressés à l'impact du choix de la valeur sur l'efficacité des robots. Ils proposent ainsi quatre fonctions de détermination de la valeur :

- *aucune fonction* : pas de combat ;
- *aléatoire* : chaque robot tire au hasard une valeur ;
- *hiérarchique* : chaque robot a une valeur qui lui est propre et qui est différente de toutes les autres
- *espace personnel* : le robot ayant le plus d'espace pour manœuvrer recule.

Pour analyser l'efficacité des différentes stratégies, les auteurs mesurent en fonction du nombre de robots la quantité de ressource ramenée et le temps passé dans chaque état.

Les auteurs observent alors que la **fonction aléatoire** se trouve être la plus **efficace** par rapport à la fonction hiérarchique et à la fonction d'espace personnel. Ceci s'explique car **les robots sont homogènes** (pas de robot physiquement supérieur, plus rapide par exemple) et que l'espace d'expérimentation n'était peut-être pas adapté.

1.3 Emergence comportementale dans un Système Multi-Agent

Nous avons vu que l'une des idées fortes de la conception de systèmes complexes ouverts – i.e. le nombre de leurs composantes est variable dans le temps - est la volonté de restreindre les spécifications d'un système aux entités de micro-niveau dans le dessein d'observer des phénomènes au macro-niveau. Ce passage du bas niveau au haut niveau avec apparition de comportements, de structures ou de propriétés inattendues justifie que l'on s'intéresse au concept d'émergence. Après une brève introduction sur ce thème, je présenterai une caractérisation d'un système émergent afin de justifier son implication dans mon étude.

1.3.1 Introduction au concept d'émergence

C'est en s'intéressant à l'évolution des mentalités sur la notion d'émergence que l'on peut comprendre le besoin de méthodologie de conception de systèmes à fonctionnalité émergente, et l'intérêt que portent certains chercheurs à l'élaboration de tels systèmes.

Les origines du concept d'émergence se perdent dans l'antiquité grecque, comme le montre ce postulat : « le tout est plus que la somme de ses parties ». Ce courant de pensée pré-Socratique ainsi que l'idée d'auto-organisation sont retrouvés dans les écrits de Thalès ou d'Anaximandre, comme le soulignent Ali et Zimmer [Ali, 1997].

Ces idées, ainsi que « le tout avant les parties » d'Aristote, ou le « Gestalt » (forme/silhouette) de Goethe [Goldstein, 1999] donneront naissance plus tard (en 1920), au mouvement du **proto-émergentisme**. Ce courant, le plus important, est appuyé par des philosophes tels que G.H Lewes (1875), C.D. Broad (1925) ou J.C. Smuts (1926). L'idée principale est de considérer le phénomène d'émergence comme une boîte noire (fig.3) possédant des entrées et des sorties multiples de plus haut niveau. Ce concept de boîte noire implique l'absence totale d'explication sur l'émergence mais permet de l'identifier le cas échéant.

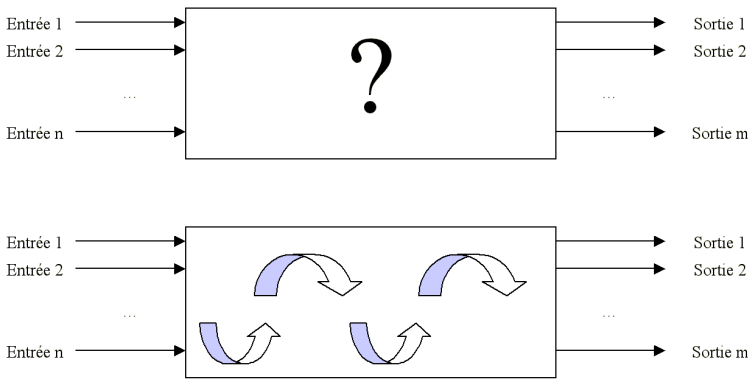


Figure 3*: La métaphore de la boîte noire : le proto-émergentisme (en haut) et le néo-émergentisme (en bas).*

Le courant du **néo-émergentisme** représenté par John H. Holland [Holland, 1997] notamment, cherche à ouvrir cette boîte noire (fig.3) afin de comprendre les phénomènes émergents. Ce courant est intimement lié à la Théorie de la Complexité et se situe aux frontières de différents domaines : Thermodynamique, Synergétique, ou Mathématiques du Chaos. Il consiste en l'élaboration de méthodes visant à démystifier le phénomène d'émergence.

C'est autour de ce courant que je vais baser mon approche en essayant de comprendre le phénomène dans une application en Robotique Collective. Il sera intéressant de voir quels sont les paramètres entrant en jeu dans l'apparition de comportements émergents. Pour cela, une caractérisation plus formelle de ce que constitue un système émergent devient nécessaire.

1.3.2 Caractérisation d'un système émergent

D'après Jeffrey Goldstein [Goldstein, 1999], on peut dégager quatre traits communs à tous les systèmes émergents, basés sur la Théorie de la Complexité :

- la **non-linéarité** : correspond aux interactions système-environnement qui sont non-linéraires par nature ;
- l'**auto-organisation** : correspond au comportement créatif, auto-généré et à la recherche d'adaptivité d'un système complexe de la Théorie de la Complexité ;
- l'**au-delà de l'équilibre** : au lieu de se focaliser sur les points d'équilibre du système, on s'intéresse au voisinage de ces points. En de tels points, l'apparition de phénomènes aléatoires explique le caractère inattendu de l'émergence ;
- les **attracteurs** : contrairement aux systèmes primitifs où il n'y avait qu'un seul type d'attracteur menant à l'état final, les systèmes émergents possèdent différents types : le point fixe, le cycle limite et l'attracteur étrange.

On peut aussi ajouter la caractéristique suivante :

- la **dynamique** : c'est-à-dire la capacité pour le système à se métamorphoser, à s'attribuer de nouveaux attracteurs qui ne sont pas prédéfinis;

Ces cinq traits forment une caractérisation possible des systèmes émergents ; systèmes disposant alors de propriétés permettant de répondre à la question suivante : « Y a-t-il apparition d'un phénomène émergent ou non ? ». Les propriétés sont les suivantes :

- La **nouveauté radicale** : apparition d'un phénomène jamais observé ;
- La **cohérence et la corrélation** : le phénomène a une identité ;
- Le **niveau global** ou macro-niveau : le phénomène émergent apparaît au niveau global, mais pas au niveau local ;
- L'observation d'une **dynamique particulière** ;
- Le phénomène est **ostensible** : l'apparition de l'émergence s'impose à l'observateur.

On a donc vu que pour prétendre qu'un système appartient à la classe des systèmes émergents, il faut le munir des cinq traits précédemment évoqués, et s'assurer qu'il vérifie bien les propriétés énoncées.

1.3.3 Un exemple de méthodologie en Robotique Collective : *Cirta*

Pour parer à la complexité des phénomènes émergents, O. Labbani-Igbida, J.P. Müller et A. Bourjault ont proposé une méthodologie, appelée *Cirta*, ayant pour but de produire des systèmes de micro-robots **réactifs** à comportement collectif émergent [Drogoul, 1998].

Comme souvent dans le domaine, les auteurs proposent leur définition de l'émergence, moins précise que celle de Goldstein (section 1.3.2) :

- un ensemble **d'entités ou agents** interagissant qui peuvent être exprimés dans une théorie ou vocabulaire **micro-niveau** ;
- la dynamique d'interaction produit un **phénomène global** qui peut être une règle, un **état stable** ou un processus ;
- ce phénomène est observé par un observateur externe ou par le système lui-même, et est exprimé au **macro-niveau**.

La méthodologie *Cirta* consiste en trois étapes de développement :

- 1ère étape : **phase de spécification**, du macro-niveau vers le micro-niveau.

Cette phase consiste en la formalisation, en calcul propositionnel du premier ordre, des comportements collectifs comme **modèles spatio-temporels**. Les auteurs introduisent le vocabulaire de base suivant :

- *sensitive(mr,s,i,e)* est vrai si le robot *mr* est sensible au stimulus *s* dans l'intervalle de temps *i* et le sous-espace *e* ;
- *response(mr,s,r)* signifie que le robot *mr* peut avoir la réponse *r* au stimulus *s* (le robot ne répondra pas nécessairement par la réponse *r*);
- un ensemble de propriétés $p_i(c_1,\ldots,c_k)$ de l'environnement et des robots ;
- *hold(prop,i)* est vrai si la propriété *prop* est vraie durant l'intervalle *i* ;
- *produce(prop,s,i,e)* est vrai si *prop* produit le stimulus *s* dans l'intervalle *i* et le sous-espace *e* ;
- $effect_p(r,prop,i)$ est vrai si la réponse *r* produit une propriété *prop* pendant *i* ;
- $effect_s(r,s,i,e)$ est vrai si la réponse peut produire un stimulus direct (de robot à robot) *s* ;
- *effect(r,s,i,e)* permet de parler directement des interactions robot-robot sans exprimer la médiation par l'environnement .

Pour formaliser que la collectivité peut effectuer une tâche T, les auteurs introduisent l'opérateur suivant :

$Can_c\ T \equiv \exists\, mr\ t.q.\ Can_{mr}\ T$

Pour formaliser comment un robot peut effectuer une tâche, on considère les termes *Identify(object,i,e)* et *Localise(object,i,e)* pour décrire les tâches d'identification ou de localisation d'un objet *object* dans un intervalle de temps *i* et un sous-espace *e* (sphère de portée des détecteurs). D'où, par exemple :

$Can_{mr}\ Identify(object,i,e) \equiv \exists\, p,s\ t.q.\ identify(p,object,i_p,e_p) \wedge perceive(mr,p,s,i_s,e_s) \wedge i_s \subseteq (i \subseteq i_p) \wedge e_s \subseteq (e \subseteq e_p)$

Ceci signifie qu'un robot identifie l'objet en un quelconque intervalle ou sous-espace où la propriété d'identification de l'objet est perceptible. Il faut aussi s'assurer que la propriété identify(p,object,i,e) n'est pas localement ambiguë.

- 2ème étape : **phase d'instanciation**.

C'est durant cette phase que le concepteur doit instancier les propriétés des robots et de l'environnement. L'instanciation des capacités collectives sera guidée par **l'analyse du comportement global** en s'appuyant sur la phase de spécification. Cette phase dépend fortement de l'application à modéliser et à concevoir. Un exemple pertinent est donné par les auteurs de cet article.

Leur exemple est celui de fourmis fourageuses. Les robots doivent explorer une zone et ramener des ressources au nid. Pour l'identification collective du nid et des sources, on associe un signal sonar propre (en longueur d'onde) au nid ou aux ressources, comme propriété d'identification non-ambiguë. De même, pour la propriété de localisation collective du nid et des ressources, le sonar permet de déterminer la distance entre un robot et un objet. De plus, les robots ayant trouvé une ressource émettent une piste infrarouge perceptible par les autres mini-robots. Cette piste est de grande importance. Un robot en exploration se détournera de son chemin s'il perçoit une piste et la suivra jusqu'à la source. D'où les stimuli suivants :

- S_n : le robot est dans le nid ;
- S_{ex} : le robot est en exploration ;
- S_{tr} : le robot suit une piste ;
- S_s : le robot voit une ressource.

D'où, les paires stimulus-réponse :

Stimuli	*Etats internes*	*Réponses*
S_n	Repoussé par le nid	Fuir du nid
$(S_{ex} \wedge \neg S_{tr})$	Exploration	Explorer l'environnement
S_{tr}	Poursuite d'une piste	Poursuivre la piste et émettre un signal (S_{tr})
$\neg S_{tr}$	Perte d'une piste	Essayer de retrouver un signal perdu
$(S_s \wedge \neg S_{tr})$	Attiré par une source	Aller jusqu'à la ressource et créer une piste

Les stimuli de ce tableau peuvent être identifiés à des états internes des mini-robots. Ces mêmes états peuvent apparaître dans un graphe de transition d'états représentant la dynamique des robots.

- 2ème étape : **phase d'évaluation**, du micro-niveau vers le macro-niveau.

Cette étape permet d'évaluer les **conditions d'apparition d'un comportement collectif**. La méthodologie utilise des chaînes de Markov pour étudier la dynamique du collectif et l'influence de paramètres comme le nombre de robots par exemple ; ceci sous l'hypothèse très forte que le graphe des états soit fortement connexe et non-périodique.

La dynamique peut être représentée par ces formules :

$$V_{mr}(t) = P_{mr} \,.\, V_{mr}(t\text{-}1)$$

$$V_{mr}(t = \infty) = P_{mr}^{\infty} \,.\, V_{mr}(t = 0)$$

où $V_{mr}(t)$ représente le vecteur des probabilités que le robot *mr* soit dans un état donné au temps *t*. Par exemple :

$$V_{mr}(t) = \begin{bmatrix} P(S_n, \underline{t}) \\ P((S_{ex} \wedge \overline{S_{tr}}), t) \\ P(\overline{S_{tr}}, t) \\ P(S_{tr}, \underline{t}) \\ P((S_s \wedge S_{tr}), t) \end{bmatrix}$$

P_{mr} définit la probabilité de transition d'un état à un autre. Cette matrice peut être très complexe, et les auteurs montrent que P_{mr^k} peut se stabiliser très rapidement ($k \approx 10$).

C'est par cette technique que les auteurs ont étudié la dynamique des mini-robots ou bien des formations des pistes. D'après les auteurs, « relativement à notre définition de l'émergence, si un tel processus se stabilise, il conduit à une émergence structurelle et dynamique » [Drogoul, 1998]. Ainsi, pour valider le caractère émergent d'une application, il suffit de vérifier la **stabilisation des processus de Markov**. Les auteurs montrent dans leur exemple, l'émergence de formations de chaînes de robots entre le nid et les ressources.

Cette méthodologie est très intéressante dans le sens où elle permet de vérifier si une application est émergente, conformément à la définition des auteurs. Pourtant, rien ne garantit l'apparition de ces phénomènes émergents, car la stabilisation du processus de Markov dépend de la phase d'initialisation des valeurs et de la « forme » du graphe (bien que les auteurs aient restreint le graphe à un graphe fortement connexe et non-périodique). De plus, cette méthodologie ne s'applique qu'à des robots réactifs, ce qui réduit considérablement les possibilités de transitions. Un tel graphe serait inexprimable pour des robots délibératifs. Une autre difficulté provient des phases de spécification et d'instanciation, où l'expression en calcul propositionnel du premier ordre peut s'avérer complexe pour un environnement fortement dynamique et des robots de plus forte granularité.

1.3.4 Justification du concept d'émergence dans notre étude

L'exemple précédent montre bien comment le concept d'émergence peut s'inscrire dans le domaine de la Robotique Collective. Nous avons vu, dans cette section 1.3, que l'idée d'émergence est définie pour des systèmes complexes non-linéaires à forte dynamique. Il est donc tout à fait pertinent que mon étude de comportements collectifs en Robotique s'oriente vers le thème de l'émergence.

Nous verrons ainsi, comment la théorie des AMAS constitue un bon point de départ pour une méthodologie de conception de systèmes à fonctionnalités émergentes qui seront en l'occurrence des communautés de robots devant réaliser une tâche au macro-niveau non-exprimée au micro-niveau.

2. ETUDE D'UN COMPORTEMENT EMERGENT : LA « CIRCULATION »

Après s'être familiarisé avec les concepts de la Robotique Collective, je vais maintenant exposer l'étude que j'ai réalisée dans ce domaine relativement au problème d'apparition d'un comportement émergent. Cette étude met en exergue un problème souvent rencontré, le problème de **l'interférence spatiale** : la mise en place d'une forme de **circulation**, c'est-à-dire une affectation tacite de sens de circulation aux couloirs, par le collectif de robots est un type de comportement émergent.

2.1 Introduction et présentation du problème

Ce problème de transport de ressources par des robots est une tâche classique car, comme nous l'avons déjà vu, le domaine s'inspire largement des communautés d'insectes [Vaughan, 2000][Kube, 1993]. Ici, les robots doivent transporter des ressources d'une zone de retrait jusqu'à une zone de dépôt dans un temps le plus court possible. Ces zones sont situées dans deux salles différentes séparées par des couloirs trop étroits pour que deux robots puissent y passer côte à côte (voir fig. 5). Les robots utilisés pourraient être ceux présentés dans la figure 4.

Ici apparaît un problème d'interférence spatiale, dans le sens où les robots doivent partager une ressource commune : les couloirs. Une fois engagé dans un couloir, que doit faire un robot rencontrant un autre robot circulant en sens inverse ? Ce problème d'interférence a été abordé par l'équipe de l'Interaction Lab [Vaughan & al, 2000], pour des robots circulant dans des couloirs et devant traverser des passages étroits. Leur solution a été de résoudre les conflits par une compétition agressive (voir section 1.2.6).

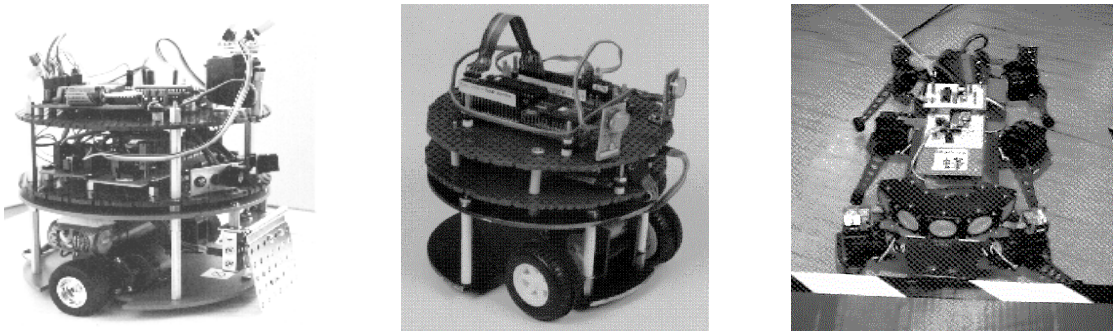


Figure 4 *: Trois exemples de mini- robots à roues ou à pattes, munis de divers types de senseurs.* [Kube, 1993], [Matarić, 1995]

Pour ma part, je modélise ce problème **par la théorie des AMAS**. Je vais ainsi faire une spécification de type AMAS de l'environnement et des robots, puis une étude des SNC rencontrées. Enfin, je présenterai quelques résultats et nous verrons la nécessité de doter nos agents de capacités de mémorisation et d'apprentissage.

2.2 Spécification du problème « circulation »

2.2.1 Le Système Multi-Agent à modéliser

Le système se compose de **plusieurs agents, les robots**, physiquement homogènes. Ce système est plongé dans un **environnement dynamique**, dans lequel se trouvent des objets mobiles (les ressources) et fixes (les murs, le sol, …). Nous allons voir que ce système, auquel nous affectons la tâche de transport de ressources, est modélisable dans la théorie des AMAS.

Cette approche nécessite la définition et l'identification de chaque composante du système. Je vais exhiber une architecture récursive, dans la lignée de certains travaux déjà effectués au sein de l'équipe.

2.2.2 L'environnement

L'environnement se présente sous la forme de **deux salles séparées par plusieurs couloirs étroits**. La configuration de cet environnement aura bien sûr une grande incidence sur l'apparition et la nature des phénomènes émergents.

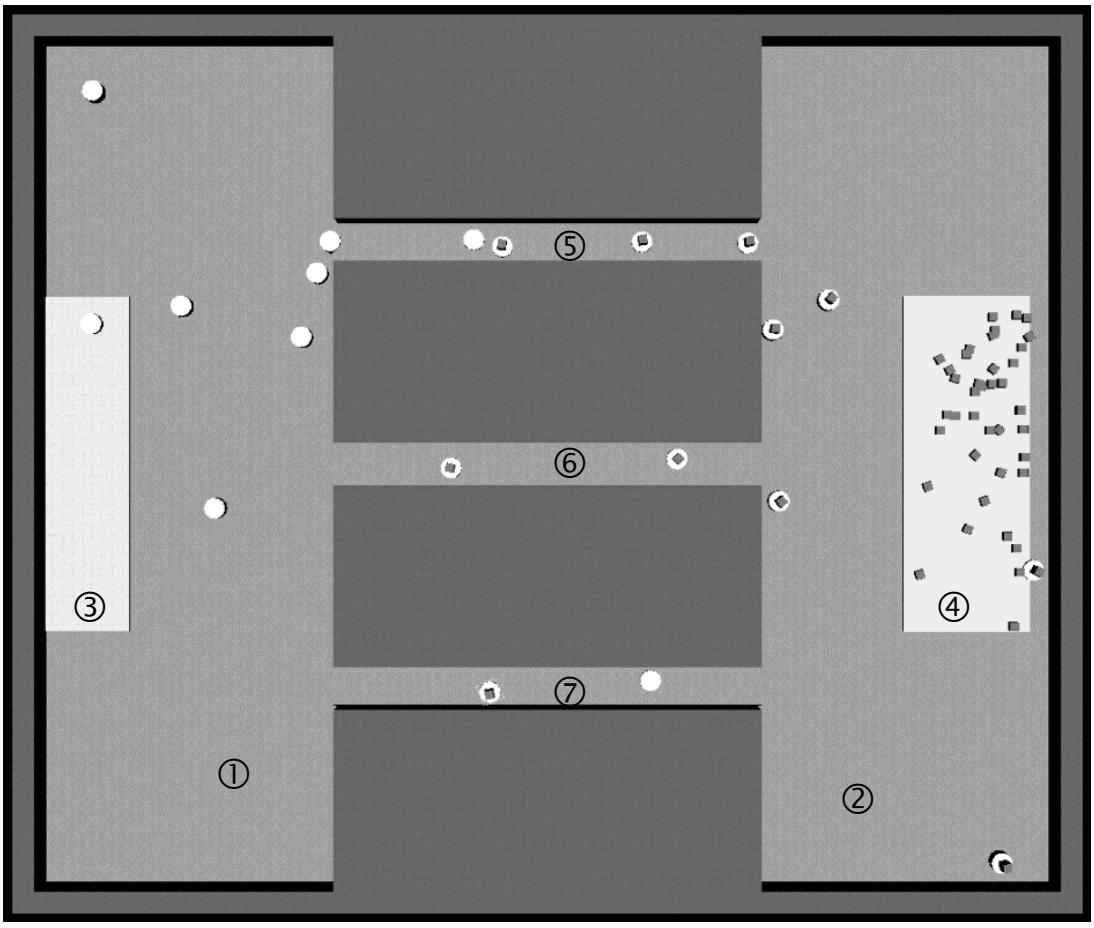


Figure 5 *: exemple d'environnement pour l'étude du comportement émergent de circulation.*

Les salles disposent d'une zone de dépôt ③ (pour la salle de dépôt ①) ou d'une zone de retrait ④ (pour la salle de retrait ②). Ces zones se situent au fond des salles, contre le mur qui fait face aux couloirs, afin de forcer les robots à bien rentrer dans la salle pour prendre ou déposer des ressources.

Les couloirs ⑤, ⑥ et ⑦ sont étroits (environ 150% de la largeur d'un robot), et leur **longueur** peut être paramétrée. En effet, intuitivement , on peut penser que leur longueur aura une grande importance sur l'apparition d'un sens de circulation. Deux cas se présenteront :

- soit la longueur du couloir est inférieure à la distance de perception des robots et un robot pourra voir si un robot est déjà engagé dans le couloir qu'il a choisi ;
- soit la longueur est supérieure à la distance de perception et un robot devra s'engager avant de constater que le couloir est libre ou occupé.

On remarque aussi que **le nombre de couloirs** (3 dans l'exemple) peut être paramétré. Ce nombre a lui aussi une influence sur l'apparition de circulation. En effet, le cas où un seul couloir sépare les deux salles n'est pas à prendre en compte : nous retombons alors sur un problème d'interblocage tout à fait similaire à celui développé par l'Interaction Lab [Vaughan & al, 2000]. Mais il semble intéressant d'étudier l'influence d'un nombre supérieur ou égal à deux.

2.2.3 Les agents

Les agents de ce système multi-agent adaptatif sont les robots qui doivent accomplir la tâche de transport de ressources. Ils sont **autonomes** et de **même type structurel** (même châssis et même programme). Ils ne disposent d'aucun moyen direct pour communiquer entre eux (radio, lumière, …), par contre, ils possèdent des capteurs et des effecteurs afin d'interagir avec leur environnement, et ainsi communiquer indirectement avec leurs pairs.

Pour mon étude, je n'utilise **pas de robots réels**, mais je les simule dans **la plate-forme de simulation de vie artificielle orientée agent *oRis***(voir Annexe).

Les agents comportent quatre parties distinctes:

- des capteurs,
- un module de détection des situations non-coopératives (ou SNC),
- des effecteurs,
- un module de décision.

Ces parties correspondent à un principe que l'on retrouve dans de nombreux travaux en Robotique réactive. Les agents font se succéder en boucle trois phases, dans l'ordre :

- une **phase de perception** pendant laquelle le robot met à jour les registres correspondant à ses capteurs et un **vecteur d'entrées** composé de booléens correspondant à la vision que le robot se fait de son environnement;
- une **phase de décision**, où le robot va choisir en fonction de ses entrées un **état interne** dans lequel le robot devra être à la boucle suivante.
- une **phase d'action**, où le robot va mettre à jour les registres correspondants à ses effecteurs (vitesse de roues, angles de rotation, …), en fonction de la décision prise.

2.2.3.1 Une décomposition récursive des agents

Comme dans certaines application de la théorie des AMAS, le système se décompose lui-même en Systèmes Multi-Agents [Topin, 1999] [Athanassiou, 1999]. Dans le cas qui nous

préoccupe, cette décomposition peut naturellement se faire en plusieurs niveaux définis arbitrairement (fig. 6), comme suit :

- **niveau 3** : l'ensemble formé par l'environnement et les robots est un AMAS dont les agents sont les robots. Ceci est justifié par le fait que les robots sont des entités autonomes évoluant dans un environnement dynamique.

Un robot possède un module de décision capable de désigner l'état interne à suivre :

- **niveau 2** : le module de décision d'un robot est un AMAS dont les agents sont les états internes du robot . Les états sont autonomes les uns des autres et leur agencement est complexe (construction du graphe de transition, voir section 2.3.3) ce qui justifie l'établissement de ce niveau en tant que AMAS.

Un état interne possède un module de décision capable de désigner le bon comportement à suivre :

- **niveau 1** : le module de décision d'un état est un AMAS dont les agents sont des comportements. Le choix de considérer ces comportements comme des agents est justifiable de la même façon que pour les états.

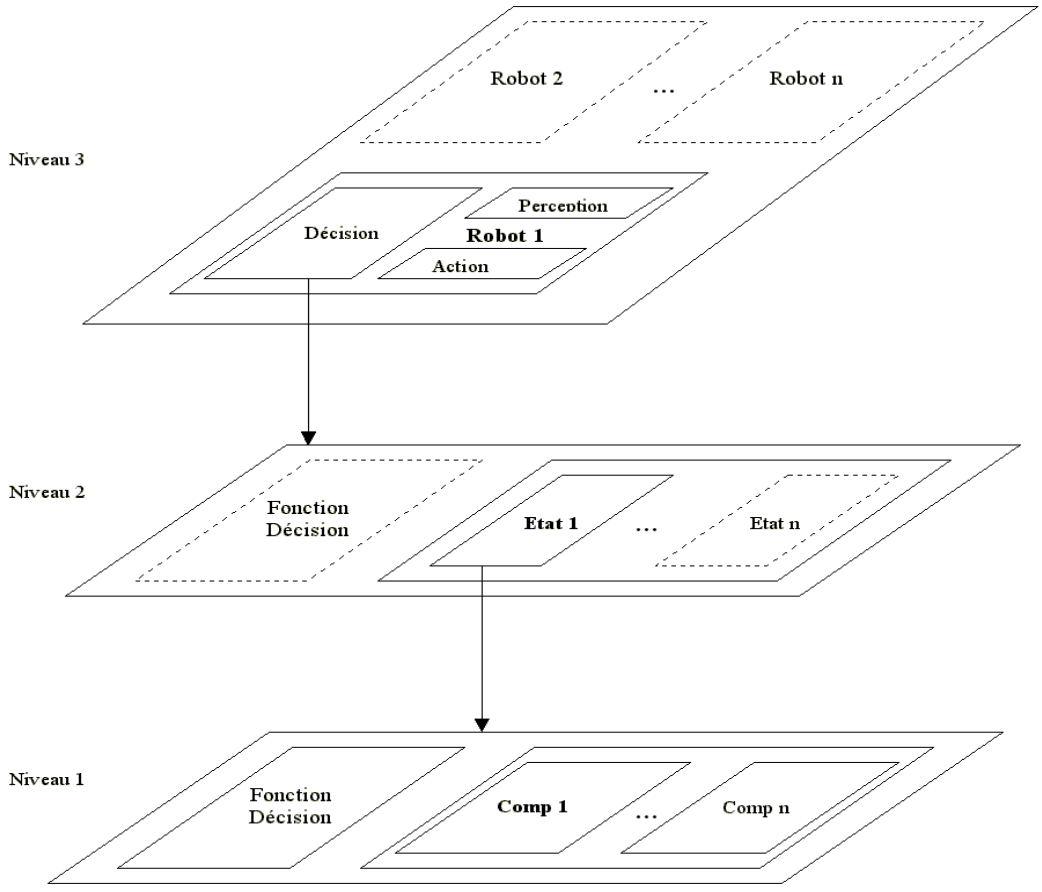


Figure 6 : *les 3 niveaux de décomposition d'un agent-robot*.

Cette architecture peut sembler complexe de prime abord, mais cela va s'éclaircir au fur et à mesure de son développement.

On peut remarquer que cette approche est une **approche basée sur le comportement**, comme celle de Brooks ou Matarić [Brooks, 1986][Matarić, 1994], où **les comportements ou les états sont des agents** du AMAS.

Lors de mon étude je vais, la plupart du temps, me placer aux niveaux 2 et 3 (en ce qui concerne les SNC notamment), bien que le niveau 1 pose de nombreuses questions qui ne

seront que très peu développées. Une des difficultés de mon travail a été de me placer au bon niveau d'abstraction lors de la définition des états ou des comportements notamment.

Le niveau 1 correspond à la Robotique Cellulaire [Capera, 2001]. Davy Capera, étudiant en DEA, s'est intéressé à la Robotique, mais à un niveau différent. Pour lui, le robot est constitué de différents organes (capteurs, roues, …) qu'il faut connecter d'une manière très particulière afin d'obtenir un comportement cohérent du robot (l'atteinte d'une proie par exemple). La connexion entre les différents organes est assurée par un AMAS constitué d'agents opérateurs (Bigger, Difference, Max, Multiplication, Switch et Fusion) et d'agents interfaces (Sensor et Effector) ne manipulant que des données flottantes entre 0 et 1. Le AMAS s'auto-organise par un mécanisme de messages de SNC et de connexions, pour atteindre un état de stabilité jusqu'à ce que l'environnement dans lequel le robot est plongé change. Le comportement « atteinte de proie » correspond donc bien au niveau 1 de mon étude.

2.2.3.2 Les capteurs

Les capteurs constituent le lien entre le robot et son environnement dans le sens **environnement** vers **robot**. Ils permettent au robot de se faire une idée partielle du monde dans lequel il évolue. Ces capteurs peuvent être par exemple un sonar, un système de localisation GPS, un détecteur infrarouge ou une caméra. Pour la plupart d'entre eux, leur portée est limitée, ce qui implique une **distance de perception** réduite pour les robots. Il est possible d'élaborer des senseurs capables de faire la distinction entre un robot, un objet (une ressource par exemple) ou un mur. Nos robots disposent de tels senseurs. Un robot peut aussi être capable de savoir s'il se trouve dans un couloir (par un système de type GPS, ou par reconnaissance d'un spectre de lumière propre au couloir).

Les robots de mon étude disposent des capteurs suivant : deux **capteurs de proximité** pour l'évitement d'obstacles, un **détecteur d'objets** pour reconnaître les robots ou les ressources, et enfin un **détecteur de zones** pour trouver les couloirs ou la zone de dépôt. Ces capteurs permettent de mettre à jour **le vecteur d'entrées** du robot, composé de booléens exclusivement, représentant la vision que se fait le robot de son environnement à un temps donné. Ce vecteur est une donnée logicielle ne correspondant pas à des registres ou un module du robot ; ce n'est qu'un flux de données.

2.2.3.3 Le module de détection des SNC

Couplé aux capteurs, ce module analyse le vecteur d'entrées du robot. En fonction de ces entrées et de ce qu'il sait sur le robot lui-même (état interne, état précédent, …), le module détecte les SNC expérimentées par le robot. Lors d'une telle situation, le **module de détection des SNC envoie un message au module de décision** qui devra traiter les informations en conséquence. Une étude plus approfondie des SNC et de leurs conditions de déclenchement sera développée dans la section 2.3.

2.2.3.4 Les effecteurs

Le lien robot-environnement est assuré par les effecteurs. Ils permettent au robot de modifier son environnement. Les effecteurs peuvent permettre au robot de **se déplacer** : chenilles, roues, pattes. Dans le cas des roues ou des chenilles, le robot n'en a qu'une représentation réduite : une vitesse pour l'organe droit et une vitesse pour l'organe gauche. Ces deux vitesses combinées permettent au robot :

- d'avancer : les roues ont la même vitesse positive;

- de reculer : les roues ont la même vitesse négative ;
- de tourner : les roues ont une différence de vitesse ;
- de faire demi-tour : une des roues a une vitesse négative.

Ces organes changeant la position du robot, l'environnement s'en trouve également changé.

Un robot peut aussi posséder des **organes de saisie** : aimant ou pince. Nos robots en sont munis pour pouvoir agripper les ressources.

2.2.3.5 Le module de décision des robots

Ce module doit permettre à un robot de **choisir un état interne en fonction de ses entrées**. Dans le système présent, ce module est lui-même un Système Multi-Agent dont les agents sont les états internes. Un état sera obtenu par agencement d'un SMA dont les agents sont des comportements (fig. 5). La distinction entre états et comportement sera explicitée dans la section 2.3.

Le module doit faire le choix du bon état à un moment donné, en fonction de la perception du robot. Dans la littérature, il existe de nombreux moyens pour faire ce choix : arbitrage, décision bayesienne, etc. Pour ma part, j'ai exploré trois types de décision différents :

- une **décision câblée** : à chaque vecteur d'entrées du robot, le module associe un état. Ce type de décision est efficace mais peu flexible, car il demande une exploration complète de l'espace des vecteurs de booléens lors de la conception (dans mon cas plus de 30 booléens, soit un espace de 10^9 possibilités), bien que certaines factorisations soient possibles. Les agents disposant de ce type de module sont incapables d'apprendre sur leurs compétences ou leurs croyances.
- une **décision arbitraire** : lors de la phase de décision, chaque agent d'état évalue sa propension à agir au temps présent. Les agents d'états possèdent des croyances sur les vecteurs d'entrées et leur affectent un poids. Le module de décision confronte les poids de tous les agents d'états et active l'état ayant le poids le plus fort. Les agents ainsi munis d'un tel organe de décision peuvent apprendre sur leurs croyances.
- une **décision par « réseau logique adaptatif»** : partant sur l'idée que les états forment un graphe à transition [Drogoul, 1998], nous voudrions que ce graphe soit émulé par un réseau d'agents de fonction NAND (ou porte logique). Les agents ayant de tels modules pourront eux aussi apprendre sur leurs croyances.

L'étude de ces types de décision sera approfondie dans la section 3.

On peut remarquer que la question du choix du mode de décision peut aussi se poser au niveau 1. En effet un état doit activer le bon comportement au bon moment afin d'effectuer une tâche cohérente. Je n'ai pas exploité cette voie car j'ai muni les états de modules de décision câblés.

La décision arbitraire et la décision par « réseau logique adaptatif » se trouvent dans la même problématique que la subsomption de Brooks [Brooks, 1986]. En effet, leur rôle est d'agencer des états exclusifs (ne pouvant être activés simultanément) dans le temps.

2.3 Etude des situations non-coopératives

Eviter les situations non-coopératives (ou SNC) est le moyen pour le AMAS de tendre vers l'adéquation fonctionnelle. Pour cela un robot doit être capable de les détecter et de se sortir de telles situations. J'ai aussi souligné la difficulté à se placer au bon niveau

d'abstraction. Une SNC correspond à un niveau précis. Celles que je développerai correspondent aux niveaux 2 et 3. Il paraît donc important de définir les différents niveaux du système avant de développer les SNC.

2.3.1 Définition des états internes et des comportements d'un robot

Les états internes correspondent à des activités que peuvent effectuer les robots. Ces états sont de haut niveau dans le sens où ils ne considèrent que des comportements et n'agissent pas sur les effecteurs directement. Les états suivants ont été choisis :

- **état de retrait (ClaimState)** : correspond à l'activité qu'un robot effectue lorsqu'il doit prendre une ressource. Cet état est composé de trois comportements de plus bas niveau :
  - *recherche de ressource (SeekResource)* : le robot explore les salles à la recherche de ressources ;
  - *atteinte de ressource (ReachRessource)* : le robot se dirige vers une ressource tout en évitant les obstacles ;
  - *prise de ressource (PickResource)* : le robot prend une ressource qui est à sa portée.

- **état de dépôt (LayingState)** : correspond à l'activité qu'un robot effectue lorsqu'il doit déposer une ressource. Il se décompose en trois comportements :
  - *recherche de zone (SeekArea)* : le robot cherche la zone où il doit déposer les ressources ;
  - *atteinte de zone (ReachArea)* : le robot se dirige vers la zone de dépôt tout en évitant les obstacles ;
  - *dépôt de ressource (DropResource)* : le robot dépose la ressource qu'il transporte.

- **état de voyage (TravelStateX)** : il est donc spécifique à un couloir particulier. Il correspond à l'activité que le robot doit effectuer afin de traverser un couloir. Il se décompose aussi en trois comportements :
  - *recherche du couloir (SeekCorridorX)* : le robot cherche un couloir spécifique en explorant les salles ;
  - *atteinte de couloir (ReachCorridorX)* : le robot se dirige vers le couloir tout en évitant les obstacles ;
  - *traversée du couloir (CrossCorridorX)* : le robot traverse effectivement le couloir en évitant les collisions ;

On remarque que les états se composent toujours de comportements similaires, peuvent se résumer en la séquence : *recherche – atteinte – action*. Les états utilisent un jeu de comportements, comme un programme utilise un jeu d'instructions [Georgé, 2000]. Cela nous conforte dans l'idée de l'établissement de différents niveaux de représentation.

2.3.2 Définition des entrées d'un robot

Afin de décider s'il doit activer un comportement ou non, un robot dispose de sa perception du monde. Ces entrées sont la seule vision du monde dont il dispose. Cela est lié au **principe de localité**, que doivent vérifier les agents : ils ne disposent que d'une vision partielle du monde qui les entoure.

Ce vecteur est un ensemble de booléens sur lesquels le robot va raisonner :

Nom	*Description*	*Type*	*Relatif à*
seeResource	Voit une ressource ?	1 booléen	Perception
closeResource	Est proche d'une ressource ?	1 booléen	Perception
carryResource	Porte une ressource ?	1 booléen	Perception
seeRobot	Voit un robot ?	1 booléen	Perception
seeCarryingRobot	Voit un robot portant une ressource ?	1 booléen	Perception
seeImmobileRobot	Voit un robot immobile ?	1 booléen	Perception
inClaim	Est dans la salle de retrait ?	1 booléen	Perception
inLaying	Est dans la salle de dépôt ?	1 booléen	Perception
seeArea	Voit la zone de dépôt ?	1 booléen	Perception
closeArea	Est proche de la zone de dépôt ?	1 booléen	Perception
inArea	Est dans la zone de dépôt ?	1 booléen	Perception
seeCorridor	Voit le couloir ?	1 booléen par couloir	Perception
closeCorridor	Est proche du couloir ?	1 booléen par couloir	Perception
inCorridor	Est dans le couloir ?	1 booléen par couloir	Perception
state	Etat actuel ?	1 booléen par état	Connaissance sur soi
previousState	Etat précédent ?	1 booléen par état	Connaissance sur soi

2.3.3 Le graphe des états internes : le problème du niveau de définition

Nous cherchons à établir que la circulation au sein du collectif dans les couloirs est un comportement émergent. Cela correspond à la construction non dirigée d'un graphe de transition entre les états des robots comme dans la méthodologie *Cirta* [Drogoul, 1998]. J'exposerai plus tard comment cette construction peut se faire dans la section 2.4.

Dans l'absolu nous voudrions qu'un robot : cherche une ressource, la trouve, l'atteigne, la ramasse, cherche un couloir, le trouve, l'atteigne, le traverse, cherche la zone de dépôt, la trouve, l'atteigne, dépose la ressource, cherche un couloir, le trouve, l'atteigne, le traverse et recommence cette procédure complexe. Cela correspond à un graphe dont les nœuds sont des comportements. Nous allons voir les problèmes induits par un tel graphe.

Si nous nous plaçons **au niveau des comportements (niveau 1)** les conditions affectées aux transitions sont complexes (fig.7) car non factorisées, alors que les transitions cond3, cond6, cond9 et cond12, **au niveau des états (niveau 2)**, sont plus simples et permettent de simplifier les transitions de niveau inférieur (entre les comportements). Cette dichotomie, entre état et comportement, est une très bonne motivation pour l'élaboration d'un système récursif (SMA d'états et SMA de comportements) où un robot est dans un état particulier ayant un comportement actif à un moment donné.

En fait, exprimée en terme de niveau 2, l'activité d'un robot devrait être la succession des états : *ClaimState*, *TravelStateX*, *LayingState* et *TravelStateY* (où X et Y désignent des couloirs différents ou non).

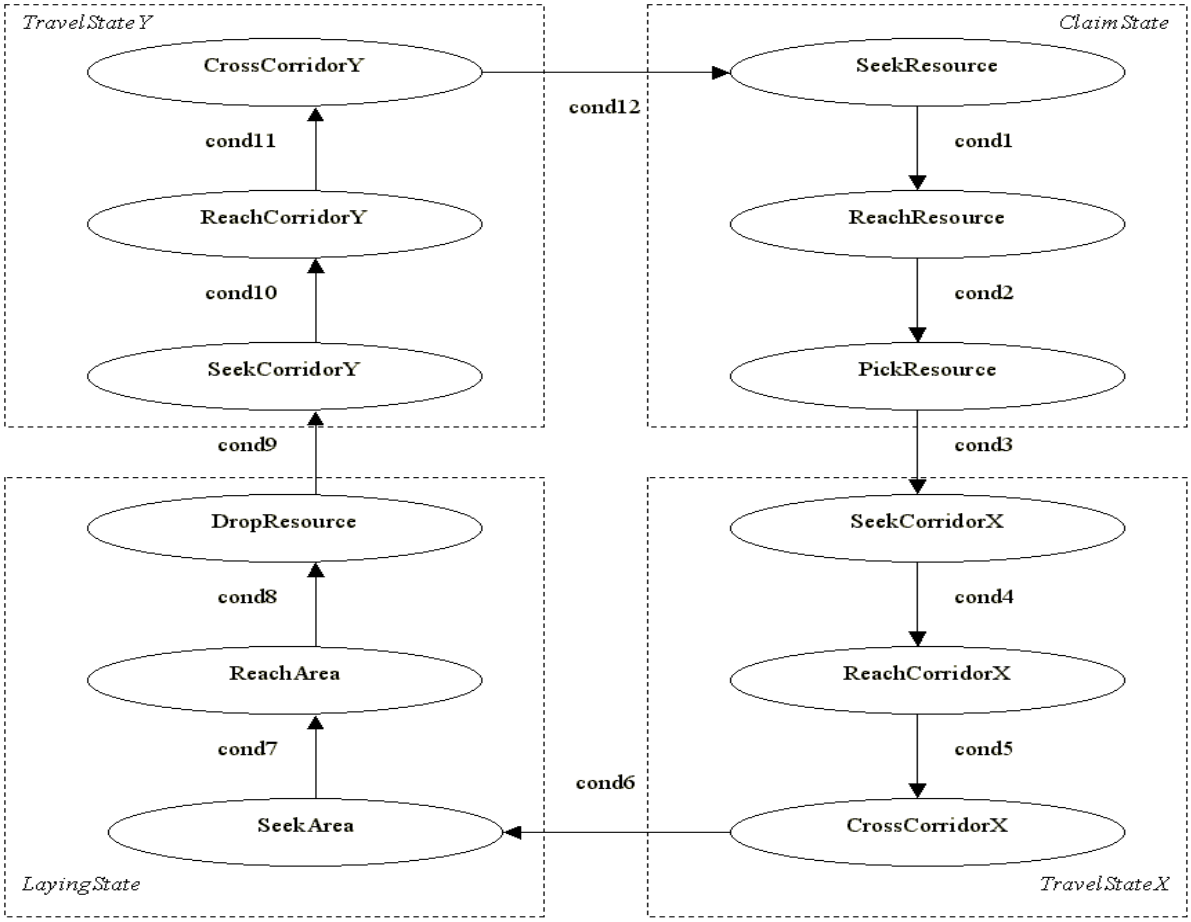


Figure 7: *les niveaux 1 et 2 se retrouvent dans le graphe de transition*.

Une fois cette distinction faite, il est possible de définir les situations non-coopératives que les robots peuvent rencontrer.

2.3.4 Les situations non-coopératives, des robots aux états

Un agent peut se trouver dans une situation dans laquelle il interfère avec un autre agent, ou bien il peut être inutile à la collectivité. Pour être coopératif, un agent doit sortir de telles situations lorsqu'il les détecte. Compte tenu du but de mon étude, c'est-à-dire l'émergence d'un comportement de groupe, **les situations non-coopératives** que je vais présenter **concernent les robots**.

Il faut souligner le problème de la complétude de l'ensemble des situations non-coopératives. En effet, rien ne garantit que les SNC présentées soient nécessaires et suffisantes. Cet établissement subtil des différentes situations reste un problème. L'élaboration d'outils de conception AMAS (comme ADELFE) pourra peut-être y répondre.

2.3.4.1 Les sept SNC liées à la circulation

Lorsqu'un robot détecte une SNC, il envoie au module de décision un message lui signifiant que l'état actif doit remédier à cette situation. J'ai recensé sept situations dans lesquelles les robots ne sont plus coopératifs. Chaque présentation comprend une brève description, les conditions de détection et l'action à effectuer pour sortir de la situation.

❶ Un robot (1) ne portant pas de ressource se dirige vers un couloir et voit un robot (2) portant une ressource. Le robot (1) se dirige vers un couloir apparemment fréquenté par des robots portant des ressources, et va donc les gêner dans leur retour (action antinomique).

Conditions : être dans l'état actif *TravelStateX,* et le comportement actif *ReachCorridorX* ; ne pas porter de ressource, voir un robot mobile transportant une ressource et ne pas être dans le couloir.

Action : le robot (1) fait demi-tour.

❷ Un robot (1) ne portant pas de ressource se dirige vers un couloir et voit un robot (2) ne portant pas de ressource et immobile. Le robot (1) se dirige vers un couloir où un robot du même type que lui a des problèmes à franchir.

Conditions : être dans l'état actif *TravelStateX* et le comportement actif *ReachCorridorX* ; ne pas porter de ressource, voir un robot immobile ne portant pas de ressource et ne pas être dans le couloir.

Action : le robot (1) fait demi-tour.

❸ Un robot (1) portant une ressource se dirige vers un couloir et voit un robot (2) ne portant pas de ressource. Le robot (1) se dirige vers un couloir apparemment fréquenté par des robots ne portant pas de ressource, et va donc les gêner dans leur retour (action antinomique).

Conditions : être dans l'état actif *TravelStateX* et le comportement actif *ReachCorridorX* ; porter une ressource, voir un robot mobile ne portant pas de ressource et ne pas être dans le couloir.

Action : le robot (1) fait demi-tour.

❹ Un robot (1) portant une ressource se dirige vers un couloir et voit un robot (2) portant une ressource et immobile. Le robot (1) se dirige vers un couloir où un robot du même type que lui a des problèmes à franchir.

Conditions : être dans l'état actif *TravelStateX* et le comportement actif *ReachCorridorX* ; porter une ressource, voir un robot immobile portant une ressource et ne pas être dans le couloir.

Action : le robot (1) fait demi-tour.

❺ Un robot (1) ne portant pas de ressource et traversant un couloir aperçoit un robot (2) portant une ressource. Les robots (1) et (2) se trouvent dans un couloir fréquenté par des robots effectuant une activité différente (action antinomique) .

Conditions (pour (1)) : être dans l'état actif *TravelStateX* et le comportement actif *CrossCorridorX* ; ne pas porter de ressource, voir un robot portant une ressource et être dans le couloir.

Action : les robots (1) et/ou (2) vont tirer au hasard entre rester sur place et faire demi-tour (technique utilisée en éthologie) jusqu'à ce qu'un d'entre eux rebrousse chemin..

❻ Un robot (2) portant une ressource et traversant un couloir aperçoit un robot (2) ne portant une ressource. Les robots (2) et (1) se trouvent dans un couloir fréquenté par des robots effectuant une activité différente (action antinomique). C'est la situation symétrique de la situation ❺.

Conditions (pour (2): être dans l'état actif *TravelStateX* et le comportement actif *CrossCorridorX* ; porter une ressource, voir un robot ne portant pas de ressource et être dans le couloir.

Action : les robots (2) et/ou (1) vont tirer au hasard entre rester sur place et faire demi-tour (technique utilisée en éthologie) jusqu'à ce qu'un d'entre eux rebrousse chemin..

❼ Un robot (1) quelconque et traversant un couloir aperçoit un robot (2) immobile. Le robot (1) se trouve dans un couloir obstrué par un ou des robots immobiles. Ce blocage doit être causé en amont par une SNC de type ❺ ou ❻.

Conditions : être dans l'état actif *TravelStateX* et le comportement actif *CrossCorridorX* ; voir un robot immobile et être dans le couloir.

Action : le robot (1) fait demi-tour.

2.3.4.2 Remarques sur les SNC

Tout d'abord, dans ces SNC, les robots sont toujours dans **l'état *TravelStateX*,** où *X* est un numéro de couloir. Cela s'explique dans le sens où mon travail se focalise sur l'apparition d'un phénomène de circulation et non sur l'efficacité du collectif, qui ne devrait apparaître qu'après l'apparition d'un sens de circulation. Dans ce cas, il aurait fallu définir des SNC liées à l'exploitation des ressources par exemple, comme dans le travail de Xavier Topin [Topin, 1998]. Dans mon étude, les ressources partagées sont les couloirs et non pas les objets à transporter.

Ensuite, **ce sont les robots**, par leur module de détection des SNC, **qui ont la compétence de détecter ces SNC** et qui envoient des messages à leur module de décision. Le module de décision relaie l'information jusqu'à l'état actif. Cet état informe ensuite le comportement actif pour qu'il effectue l'action correspondante à la situation. Ce passage de niveau est nécessaire pour doter les robots capacité d'apprentissage, leur permettant de changer leurs attitudes et leurs croyances en fonction des expériences vécues.

2.4 Quelques résultats et réflexions

Le système présenté est séduisant mais il possède de larges limites. Je vais exposer ici les quelques résultats motivant ma volonté d'améliorer le système, notamment en le munissant de capacité d'apprentissage.

2.4.1 La circulation est-elle une nécessité ?

La réponse à cette question peut paraître évidente. Il semble naturel que le collectif devienne plus efficace s'il s'organise pour circuler de manière cohérente dans les couloirs et qu'il affecte un sens de circulation à chacun d'entre eux. Mais n'oublions pas que nous nous plaçons un niveau au-dessus afin d'observer les raisons de l'apparition de cette circulation et, plus généralement, l'émergence de fonctions. **Je ne cherche pas à faire un système efficace**, mais plutôt un **système facilement observable du point de vue de l'émergence**, et nous permettant d'en tirer rapidement des conclusions afin de montrer la pertinence de l'application des AMAS en Robotique Collective d'une part, et d'autre part étudier le très fascinant mais non moins difficile phénomène d'émergence.

Pour répondre à la question de la nécessité de la circulation, j'ai élaboré deux communautés de robots. L'une est composée des robots ayant un module de décision qui affecte automatiquement un sens unique au couloir : les robots « affectés ». Par exemple, un robot devant transporter une ressource vers la zone de dépôt devra passer par un couloir défini, et pour le retour passer par un autre. L'autre communauté est composée de robots ayant un module de décision tirant au hasard le couloir à traverser à chaque fois que l'occasion se présente : robots « non affectés ».

Les **deux types de robots, « affectés » et « non affectés »**, sont construit sur l'architecture présentée précédemment. Ils détectent les SNC et les résolvent mais n'apprennent rien à la suite d'une telle expérience. Dans les deux cas le **module de décision est de type « câblé »** : à un vecteur d'entrée donné, le robot est affecté à un état précis. Enfin, les résultats présentés ont porté sur des communautés de 50 robots et un environnement comportant 2 couloirs.

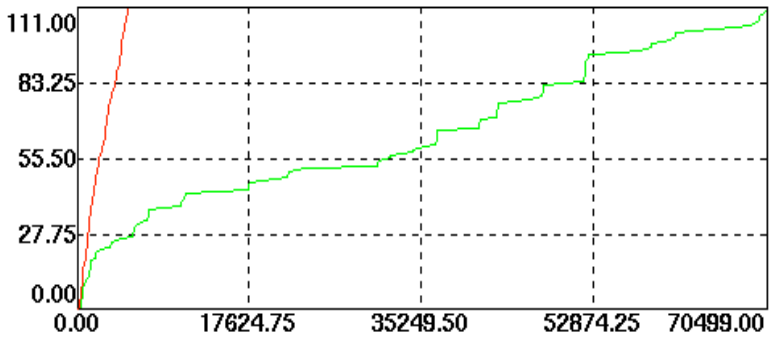


Figure 8: *comparaison du nombre de ressources rapportées pour 50 robots et 2 couloirs.*

Dans la figure 8 apparaît le nombre de ressources rapportées en fonction du nombre de pas écoulés. La courbe ayant la plus forte pente correspond aux robots « affectés », l'autre aux robots non « affectés ». Ces courbes sont explicites : les robots « affectés » sont plus efficaces que les robots « non affectés ». En effet, il faut beaucoup moins de temps à des robots « affectés » pour transporter les ressources : les embouteillages sont limités par l'affectation de sens à des couloirs.

Les courbes suivantes (fig.9 et 10) mesurent l'efficacité du collectif avec l'indice suivant :

$$\frac{nombrederessourcesrapportées}{nombredepasdelasimulation} \times 10^{-2}$$

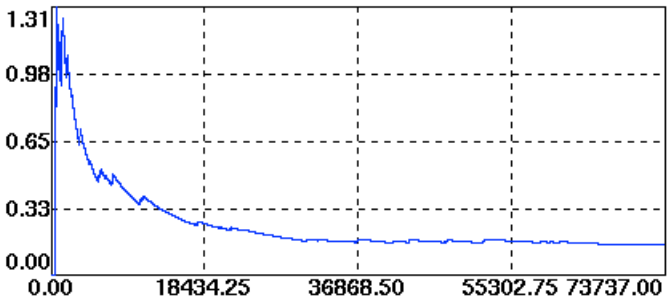


Figure 9 : *efficacité du collectif de robot "non affectés" en fonction du nombre de pas de la simulation.*

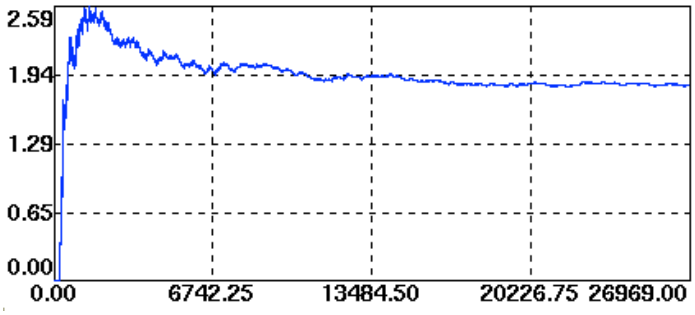


Figure 10 : *efficacité du collectif de robots "affectés" en fonction du nombre de pas de la simulation.*

Les courbes d'efficacité nous montrent encore la différence entre les robots « affectés » et les robots « non affectés ». L'efficacité du collectif de robots « affectés » se stabilise à une valeur proche de 1.90 (fig. 9) alors que celle des robots « non affectés » se stabilise à 0.15 (fig.10).

Une autre remarque peut être faite sur la valeur maximale de ces courbes. Ce maximum est toujours atteint en début de simulation. L'efficacité baisse peu pour les robots « affectés », alors que celle des robots « non affectés » décroît très fortement. Au début de la simulation les robots sont tous dans la salle de retrait. Cela explique la présence d'un pic d'efficacité en début de simulation. En effet, tous les robots prennent une ressource puis vont dans la salle de dépôt sans encombre. Par contre sur le retour, il rencontre les plus attardés et les conflits commencent. Ces conflits sont limités lorsque les sens des couloirs sont affectés. Par contre, ces conflits annihilent toute efficacité lorsque les couloirs sont pris au hasard.

Sous la contrainte de rapporter les ressources le plus vite possible, la **circulation est donc une nécessité**. Bien sûr si on ne se fixait pas cette contrainte, l'émergence de la circulation n'aurait pas de sens. En effet les ressources arriveraient toutes à destination en un temps lointain, mais elles arriveraient. Il faut donc se placer sous l'angle de l'efficacité pour conclure que l'apparition d'une circulation est bien un phénomène émergent. Cette **contrainte est ici posée par l'observateur**, mais elle pourrait très bien être une **pression environnementale** ayant conduit les robots à agir le plus vite possible. Par exemple, si les robots ne disposaient que d'une quantité d'énergie limitée ou alors des ressources nécessaires à la survie d'une colonie. Cela rejoint de nombreux travaux effectués en vie artificielle.

Pour répondre à la question posée, nous savons maintenant que le comportement de circulation doit émerger du système. Pourtant, avec le système présent, pouvons-nous observer cette apparition ?

2.4.2 Résultats pour des robots ne détectant pas les SNC

Afin d'étudier la circulation dans les couloirs, je me suis penché sur la fréquentation des couloirs par les robots portant des ressources ou non. Les résultats suivants ont été effectués avec des **robots de type « câblé » et « non affectés »**. Ils possèdent un module de décision, mais **pas de module de détection de SNC**. Ils ne peuvent donc pas réagir en présence de SNC, et continuent leur tâche comme si de rien n'était. Ici, les simulations ont été paramétrées avec 20 robots, et 2 couloirs.

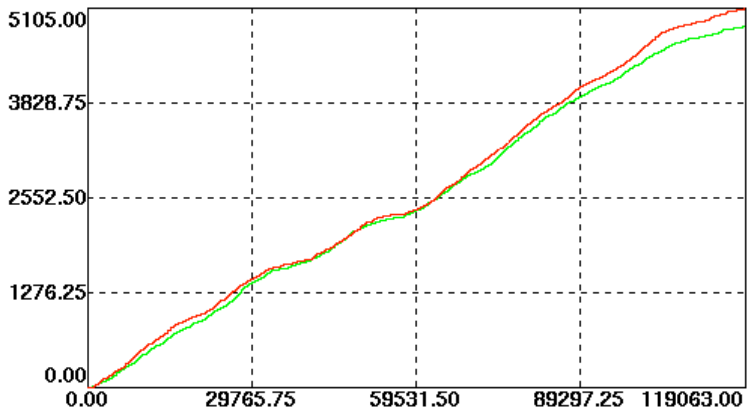


Figure 11 : *Nombre de robots entrés dans le premier couloir en fonction du nombre de pas.*

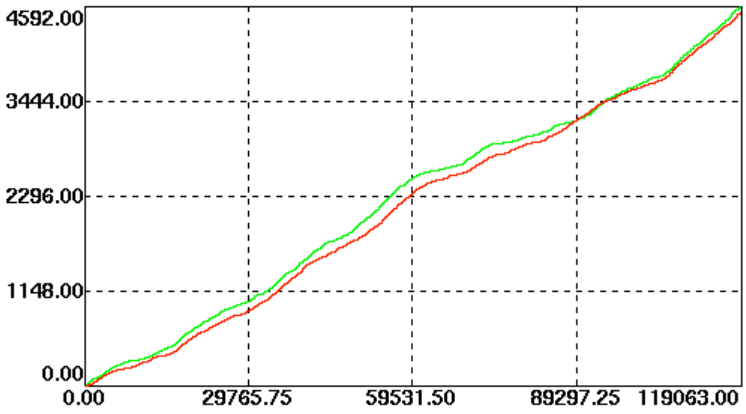


Figure 12 : *Nombre de robots entrés dans le deuxième couloir en fonction du nombre de pas.*

Ces courbes (fig. 11 et 12) montrent que la fréquentation est équivalente pour les deux couloirs – en clair, les robots ne portant pas de ressources, en foncé, les robots portant des ressources – **.** Ainsi aucun phénomène d'émergence n'est apparu.

2.4.3 Résultats pour des robots détectant les SNC

Ici les robots sont de type « non affectés » et sont dotés de **module de décision « câblé »** et de **module de détection des SNC**. Ils peuvent donc réagir à leurs apparitions. Les simulations ont aussi été initialisées pour 20 robots et 2 couloirs. Dans les figures 13 et 14, en clair, apparaissent les robots ne portant pas de ressources et en foncé, les robots portant des ressources.

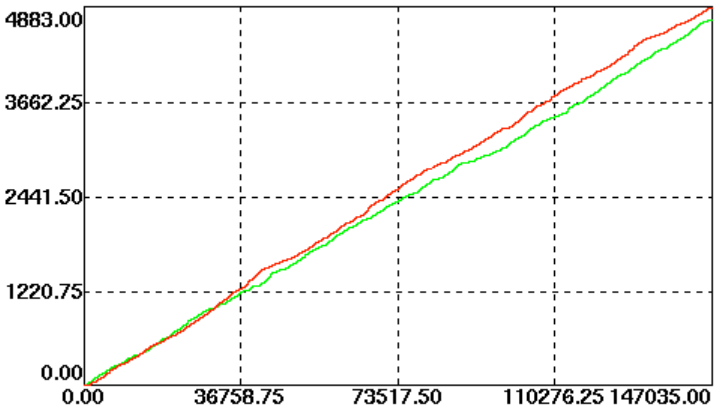


Figure 13 : *Nombre de robots entrés dans le premier couloir en fonction du nombre de pas.*

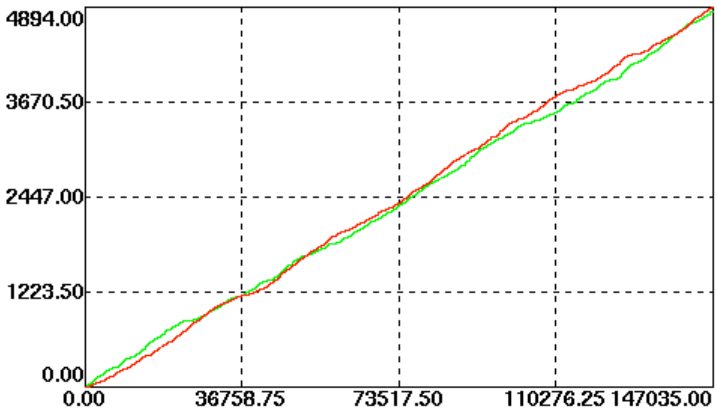


Figure 14 : *Nombre de robots entrés dans le deuxième couloir en fonction du nombre de pas.*

Ici encore (fig. 13 et 14), **chaque couloir est autant fréquenté par des robots portant des ressources que par des robots n'en portant pas**. Cela signifierait-il dire que je me suis trompé depuis le début ? La circulation ne serait-elle donc pas un phénomène émergent, ou bien est-ce mon système et l'architecture de mes agents qui sont en cause ?

2.4.4 Les SNC ne sont pas totalement exploitées

Les résultats précédents peuvent paraître décevants. Pourtant, ils permettent de mettre en exergue le besoin d'apprentissage.

Les résultats ont été obtenus grâce à des robots pouvant ou non détecter des SNC. On pourrait en conclure que les SNC sont inutiles. Une telle conclusion est hâtive et fausse. En effet, la résolution des SNC n'apporte rien dans ce cas. Cela est dû au fait que lorsqu'un robot détecte une SNC dans un couloir par exemple, il est trop tard pour remédier au mal qui a été fait. Les robots concernés sont déjà trop avancés dans le couloir. **Les causes des conflits ne sont pas dues à l'activité actuelle d'un robot**, mais plutôt à une **activité antérieure**. Ici l'activité ayant causée ce conflit est le choix fait par l'organe de décision entre passer en l'état *TravelStateX* ou passer en l'état *TravelStateY*.

Il faut donc que **les robots apprennent** de telles expériences. Les SNC seront alors exploitées pour résoudre les conflits localement, au niveau des robots, mais aussi globalement au niveau du collectif.

3. MEMOIRE, APPRENTISSAGE ET DECISION

Dans la partie précédente, j'ai mis en évidence la nécessité de munir nos robots de capacités d'apprentissage. Cela dans le but de pouvoir observer un phénomène émergent. Dans cette section, je vais présenter trois types de modules de décision différents dont le robot pourra être muni : le module « **câblé** », le module d'« **arbitrage entre comportements** », et le module de type « **réseau logique adaptatif** ». Ces modules ne pourront coexister au sein du robot : ce sont des choix de conception. Mais avant tout un bref rappel de l'apprentissage par auto-organisation s'impose.

3.1 Un apprentissage par auto-organisation

Dans le cadre de la théorie des AMAS, l'apprentissage s'effectue par auto-organisation. Cela signifie que les robots apprennent grâce aux expériences vécues. Un AMAS permet l'apprentissage par la détection des SNC. Ces situations néfastes pour le collectif sont traitées par chaque robot afin de réviser leur comportement futur.

3.1.1 Comment apprendre ?

Les SNC sont un moyen très efficace pour les robots d'apprendre tout au long de leur existence. En effet, lorsqu'un robot se trouve en SNC, il peut apprendre sur cette situation pour éviter son erreur ou au moins limiter au maximum pour ne pas se retrouver dans une telle situation. Si tous les robots agissent de même, le système va converger vers une stabilité qui pourra être remise en cause si l'environnement subit un grand changement. Dans une telle situation, le collectif de robot sera à même de s'adapter à nouveau.

Ce type d'apprentissage se trouve être adéquat pour notre application, d'autant plus qu'il garantit aux robots le **principe de localité** et une **spécification localisée au micro-niveau** seulement (niveau des robots) et non pas au macro-niveau. En effet, ce n'est pas une entité supérieure qui informe les robots sur leurs « bonnes » ou « mauvaises » actions mais bien les robots eux-mêmes grâce à leurs compétences personnelles. Cette contrainte est nécessaire si l'on veut parler d'émergence, en tant que **phénomène ostensible** au macro-niveau. L'idée d'apprentissage supervisé, comme les classifieurs ou le Q-Learning, sont donc éliminés de fait. En outre, les robots de mon étude sont incapables de communiquer directement pour échanger des messages suivant un protocole précis. L'apprentissage par interaction est donc inadéquat (voir section 1.1.3, *L'apprentissage dans les Systèmes Multi-Agents*).

Un système de renforcement par auto-organisation devra donc être mis en place, dans les différents types de modules de décision : au niveau des agents « états », pour le module par

« arbitrage d'états », et au niveau de la fonction de décision pour le module « réseau logique adaptatif ».

3.1.2 Sur quoi apprendre ?

Pour la théorie des AMAS, l'apprentissage est essentiellement **centré agent et environnement**. Cela implique que les agents disposent de moyens de mémorisation des objets de leur environnement, ou de leurs valeurs internes. Il est donc naturel de faire apprendre mes robots sur leur **vecteur d'entrées** qui est la seule vision qu'ils possèdent sur l'environnement et sur eux-mêmes.

Il est important de remarquer que bien que l'on sache, en tant que concepteur, que l'apprentissage portera en majorité sur les couloirs à emprunter, il est dangereux de faire manipuler ce concept par les robots. En effet, cela est un problème classique de conception de systèmes complexes émergents, où le concepteur a souvent tendance à donner trop de connaissances innées aux agents. Cette sur-spécification limitera ainsi l'espace des solutions d'apprentissage des agents et ainsi, violera les principes d'ostentation et d'apparition au niveau global (voir section 1.3.2, *Caractérisation d'un système émergent*) par spécification au niveau local. C'est comme si pour conclure que les chimpanzés préfèrent les bananes, ce qui peut être considéré comme un phénomène émergent, on ne leur donnait que ces fruits à manger…

Cet apprentissage doit permettre aux robots de changer leurs réactions face aux situations rencontrées. Après avoir été en SNC, un robot va donc **changer ses croyances**. En effet, le robot ne déclenchera plus ses états pour les mêmes circonstances. Ce sont des croyances qui portent évidemment sur lui et non sur les autres, car le vecteur d'entrées ne contient aucune trace des autres robots nominalement. Un robot apprendra aussi sur **ses compétences**. Après avoir vécu des échecs dans des couloirs, le robot privilégiera un couloir plus qu'un autre. Sa compétence ne sera donc plus, choisir un couloir, mais plutôt prendre tel couloir de préférence.

3.1.3 Une SNC d'inutilité pour implémenter la compétence des robots

Comme mon centre d'intérêt est le comportement de circulation, je n'ai pas jugé utile de faire directement apprendre aux robots l'état dans lequel ils doivent être à un moment donné. Par exemple, un robot doit savoir que lorsqu'il est dans la salle de retrait et qu'il ne porte pas de ressource, il doit être dans l'état de retrait, *ClaimState*. Pour cela, il peut détecter lorsqu'il n'est pas dans le bon état : c'est **une SNC d'inutilité**. Il compare l'état choisi par son module de décision à un état théorique (généré par exemple par un module de type « câblé », l'informant sur l'état à adopter). Si les états ne correspondent pas, un message de SNC est généré et envoyé au module de décision.

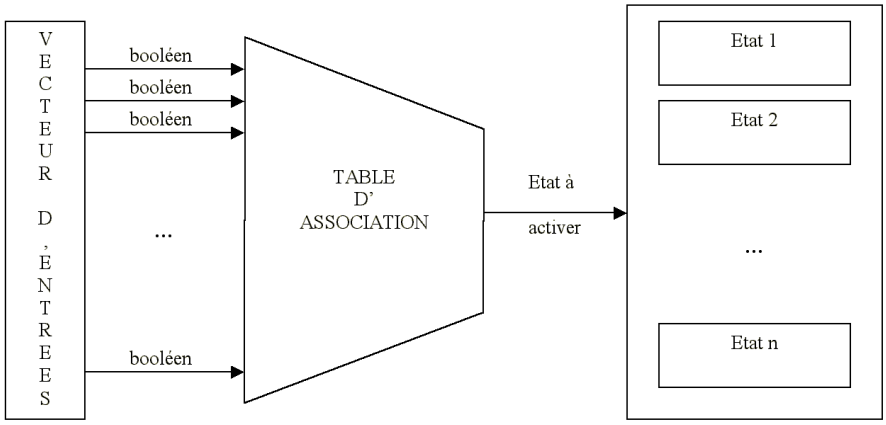
Cette détection ne va pas à contre courant de mes objectifs d'étude, car la comparaison se fait sur le type d'état. Par exemple, *TravelState1* et *TravelState2* seront considérés comme compatibles. Ainsi, le choix du couloir à emprunter n'est pas guidé par cette SNC d'inutilité.

3.2 Différents types de modules de décision

Dans cette section je vais présenter les différents types de modules de décision que j'ai envisagés pour mon application : module « câblé », module par « arbitrage d'états » et module

avec « réseau logique adaptatif ». Un robot ne pourra avoir qu'un seul type de module pour gérer les états. Ces trois types ont été introduits dans le paragraphe 2.2.3.5, « *Le module de décision des robots* », et je vais détailler leur fonctionnement.

3.2.1 Le module de décision de type « câblé »



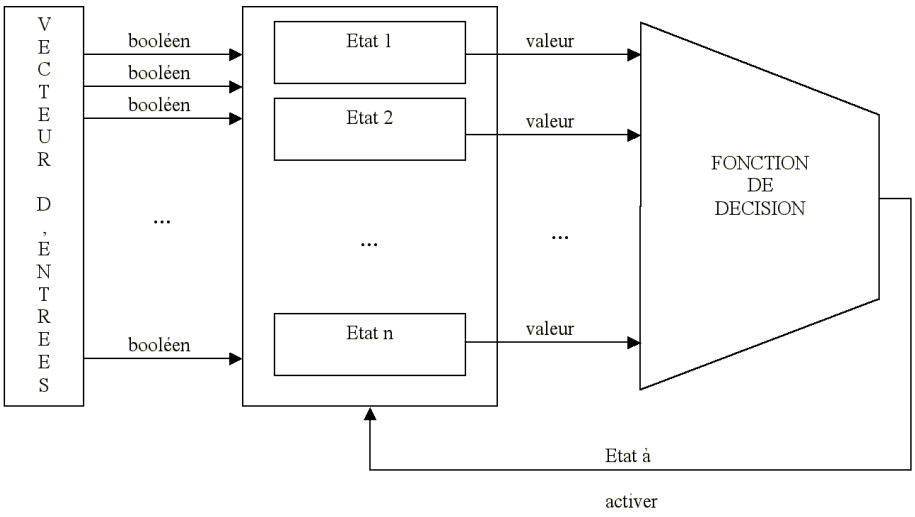
Ce type de module peut s'avérer difficile à concevoir à cause de la **difficulté à trouver les bonnes associations vecteur-état**.

Je ne me suis pas posé le problème de l'apprentissage sur ce module. Il m'a essentiellement servi à tester mon environnement graphique, car ici les associations n'étaient pas trop complexes.

3.2.2 Le module de décision de type « arbitrage d'états »

Ce module se place directement dans l'approche du contrôle basé sur le comportement. De nombreux travaux ont été faits sur le sujet, et montrent l'efficacité de cette approche [Brooks, 1986][Matarić, 1994].

3.2.2.1 Architecture du module



3.2.2.2 Idée principale

Le principe général est de considérer **les états comme des agents indépendants**. Ces agents se proposent d'agir à chaque fois que le module de décision le demande. Pour un

vecteur d'entrées, les agents états vont émettre une valeur. La fonction de décision doit en fonction de ces valeurs **choisir l'état à activer**.

3.2.2.3 Les agents états

Les agents état assurent deux fonctions au sein du module et du robot au niveau supérieur. Ils doivent gérer l'exécution des comportements de niveau inférieur, et fournir des valeurs à la fonction de décision en temps voulu.

La première tâche des agents est de **choisir, parmi les comportements à disposition, le plus approprié à la situation**. Pour cela, ils disposent d'un module de décision leur permettant ce choix. Cela ressemble très fortement au choix qu'un robot doit faire entre plusieurs états. Nous avons mis en évidence une architecture récursive très intéressante. Nous allons seulement nous focaliser sur le niveau 2, c'est-à-dire les états.

Un état devra **fournir une valeur à la fonction de décision** chaque fois que le module de décision est sollicité par le robot pour choisir un état. Cette valeur correspond à la propension que l'état a à se déclencher pour une situation donnée. Pour cela, l'agent possède des croyances sur les entrées booléennes fournies (vecteur d'entrées). Il effectue alors un produit entre le vecteur d'entrées modifiées (e_i), et le vecteur des poids croyances (c_j). Les entrées modifiées correspondent à un 1 pour une entrée vraie et à –1 pour une entrée fausse.

$$\begin{bmatrix} e_1 & e_2 & \dots & e_{n-1} & e_n \end{bmatrix} \bullet \begin{bmatrix} c_1 \\ c_2 \\ \dots \\ c_{n-1} \\ c_n \end{bmatrix} = \textit{valeur de retour}$$

Il est facile d'imaginer comment l'**apprentissage** va s'opérer, par attribution de **récompenses et de punitions**. Lorsque le robot détecte une SNC, il envoie un message au module de décision. Ce message est relayé à tous les agents d'états, qui modifient les poids de leurs croyances en conséquence, afin d'affaiblir la valeur de retour correspondante. A contrario, à chaque fois qu'un état est choisi pour être actif par la fonction de décision, l'agent modifie ses poids afin d'augmenter la valeur de retour correspondante.

3.2.2.4 La fonction de décision : le *max*

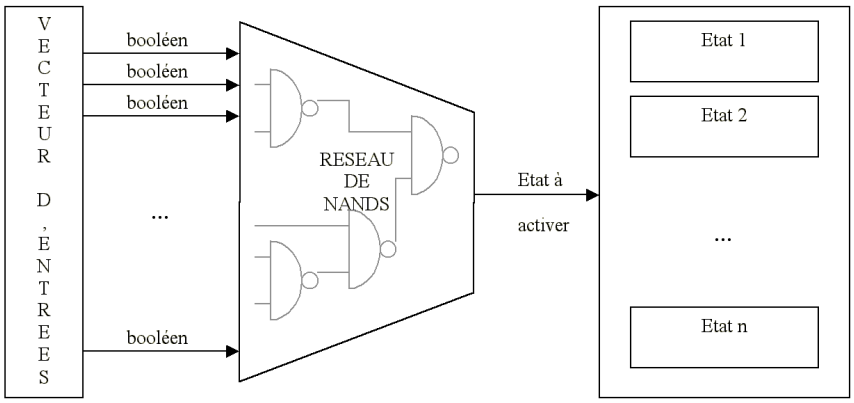
Le choix entre les états à activer est à la charge de cette fonction. Dans la littérature de nombreuses fonctions de décision apparaissent. Pour ma part, j'utilise la très simple fonction *max*, qui choisit **l'état ayant la plus haute valeur de retour**.

On peut imaginer des méthodes d'arbitrage plus fines, comme la subsomption [Brooks, 1986] ou bien même des fusions d'états [Matarić, 1994], mais je ne me suis pas attardé sur le sujet, qui relève plus spécifiquement du contrôle robotique et non pas de la Robotique Collective qui est le domaine dans lequel je me place.

3.2.3 Le module de décision de type « réseau logique adaptatif »

Ce module est une nouvelle application intéressante de la théorie des AMAS. Ici, la fonction de décision sera assurée par un AMAS d'agents ayant pour tâche de fournir un état pour un vecteur donné et de se ré-organiser lorsqu'une SNC lui est envoyée.

3.2.3.1 Architecture du module



3.2.3.2 Idée principale : apprentissage d'une fonction de transition

Nous avons vu que dans un module « câblé », la fonction de décision se place entre le vecteur d'entrées et les états. Cette fonction était déterministe et consistait en une table d'associations. Cette table peut être vue comme la représentation d'un graphe. L'idée est d'émuler cette fonction par un « réseau logique adaptatif », **comme les portes logiques peuvent assurer les fonctions logiques dans un circuit imprimé**. Le problème de la complétude du réseau est justifiable de fait. En effet, les nands sont suffisants pour coder toute fonction logique, donc un réseau uniquement composé d'agents réalisant la même fonction nand doit être capable de réaliser la fonction logique correspondant aux transitions du graphe.

3.2.3.3 La fonction de décision : le réseau logique adaptatif

Le réseau logique adaptatif reçoit en entrée un vecteur de booléens. En sortie, il fournit un état, codé sous forme de chaîne de bits, puisque le réseau ne manipule que des 0 et des 1.

Le réseau possède trois couches d'agents nands :

- les agents d'entrées : directement connectés au vecteur d'entrées ;
- les agents internes : réalisant la fonction ;
- les agents de sortie : fournissant la valeur correspondant à un état.

Lorsque le robot détecte une SNC, il informe le module de décision, qui informe à son tour le réseau logique adaptatif. D'abord, les agents nands de sortie du réseau sont informés, car se sont eux qui fournissent une valeur d'état. Puis, l'information se propage dans le réseau (agents interne et agents d'entrées).

Le réseau, qui est un AMAS, s'auto-organise pour retourner la bonne valeur. Après avoir reçu de nombreuses SNC, le réseau va se stabiliser et converger vers le réseau émulant le plus fidèlement possible le graphe de transition d'états. Le système ne peut parvenir à une telle stabilité, que si le nombre d'agents nands est suffisant pour émuler la fonction et explorer la totalité de l'espace des solution.

3.2.3.4 Les agents « nands »

Le AMAS, constitué d'agents nands, reçoit des messages de SNC provenant du niveau supérieur. Les agents disposent de deux entrées et d'une sortie. Leur compétence est de

calculer la valeur de leur sortie en fonction des valeurs en entrée, comme la fonction *nand*. Ils possèdent des **croyances sur les autres agents, pour chacune de leurs entrées**. Ces croyances seront révisées à la réception de messages de SNC. De manière générale, un agent connectera une des ses entrées à l'agent à qui il accorde le plus de confiance.

A leur niveau, les nands ne peuvent détecter qu'**un seul type de SNC : « L'agent *n* a besoin de la valeur *x* »**. x est toujours un booléen. Les messages proviennent toujours d'agents connectés en sortie, ou depuis le niveau supérieur pour les agents de sortie du réseau. Face à un tel message, l'agent peut adopter des attitudes différentes :

- relayer le message aux agents connectés à ses entrées : c'est la relaxation;
- créer un message ;
- connecter directement un des agents connectés en entrée, ayant la bonne valeur à l'agent ayant besoin de cette valeur.

Le choix entre ces attitudes devra se faire en fonction de ses croyances et de sa compétence. En effet, lorsqu'un agent est en SNC , il ne renvoie pas la bonne valeur. Deux cas se présentent, car il ne manipule que des 0 et des 1 : soit il renvoie un 1 et il devrait renvoyer un 0, soit il renvoie un 0 et devrait renvoyer un 1. Si un agent lui demande une valeur, l'agent peut, dans tous les cas, trouver cette valeur dans ces entrées (propriété de la fonction nand) et donc proposer un branchement entre l'agent entrée concerné et l'agent sortie demandeur de valeur.

❶ Le nand (1) connecté en entrée aux nands (2) et (3) ne renvoie pas la bonne valeur au nand (4) connecté en sortie.

Conditions : (1) envoie une valeur différente de celle voulue par (4).

Action : si l'agent est connecté en sortie à peu d'agents alors il relaxe la SNC afin que ce soient les agents connectés en entrée qui changent leurs valeurs. Si l'agent est connecté à plusieurs agents en sortie, s'il dispose en entrée de la valeur voulue par la sortie alors il connecte les agents en conséquence, sinon il génère lui-même un message de SNC à l'intention de ses entrées.

Les agents devraient être ainsi capables de s'auto-organiser pour répondre aux exigences du module de décision. L'apprentissage se fera par les variations des taux de croyance que les agents modifient au cours du temps.

3.3 Expérimentation et résultats

Nous avons vu la nécessité de munir les robots de capacité d'apprentissage pour atteindre notre but : l'émergence du comportement de groupe de circulation. Nous avons aussi défini différents modules susceptibles de remplir cette fonction. Ainsi, je présente dans cette section des résultats montrant les conséquences de la mise en place de tels modules.

3.3.1 Résultats avec le module de décision de type « arbitrage d'états »

Ici les résultats correspondent à des expériences avec module de type « arbitrage d'états » dans des simulations avec 20 robots et 2 couloirs.

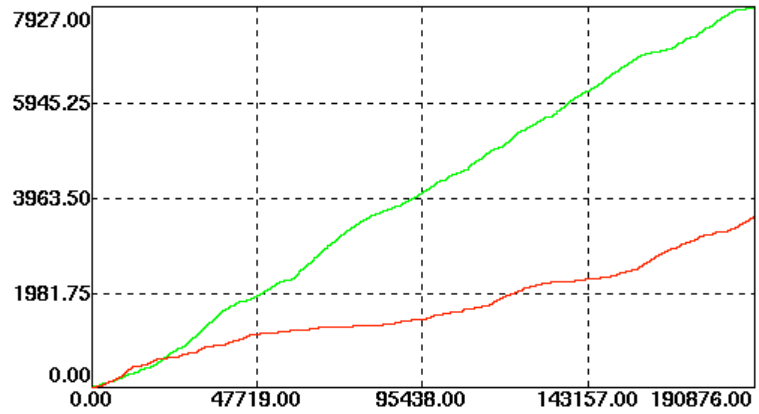


Figure 15 : *Nombre de robots entrant dans le premier couloir en fonction du nombre de pas avec apprentissage au niveau du module de décision par "arbitrage d'état".*

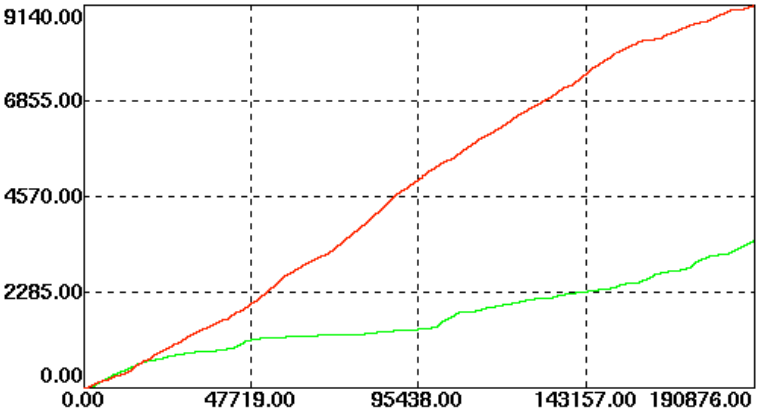


Figure 16 : *Nombre de robots entrant dans le deuxième couloir en fonction du nombre de pas avec apprentissage au niveau du module de décision par "arbitrage d'état".*

Dans les figures précédentes (fig. 15 et 16), la courbe claire correspond aux robots ne portant pas de ressources, et la courbe foncée correspond aux robots portant des ressources.

Les résultats montrent bien la dissociation entre les classes de robots (portant une ressource et ne portant pas de ressource) fréquentant les couloirs. Les robots portant les ressources se sont appropriés le deuxième couloir. Ce phénomène apparaît grâce à la capacité d'apprentissage dont les robots sont maintenant munis. En effet, en début de simulation, les robots rencontrent de nombreuses SNC car ils se gênent au niveau des couloirs. Au fur et à mesure, ils vont remettre en cause leurs croyances et préférer un graphe de transition à un autre. Par exemple, après avoir été en *ClaimState* les robots préféreront, passer en *TravelState2* plutôt qu'en *TravelState1* dans l'exemple ci-dessus. Bien sûr, pour des simulations différentes, ces préférences peuvent changer car les robots apprennent pour une simulation.

La dissociation au niveau des couloirs ne correspond pas à une formation de castes [Balch, 1997]. En effet, ici tous les robots suivent le même circuit : salle de retrait, deuxième couloir, salle de dépôt, premier couloir, puis salle de retrait, … Etant donné que l'environnement ne comporte que deux couloirs cela semble naturel. Mais quant serait-il s'il y avait plus de deux couloirs ? Il serait intéressant alors de se pencher sur la formation de castes. En effet, on peut imaginer que si l'environnement disposait de quatre couloirs, plusieurs circuit pourraient émerger.

Dans les deux cas, le couloir semble être encore fréquenté par les deux «classes » de robots, car les deux courbes d'un même graphe continuent de monter. Cela est tout

simplement la conséquence de la métrique utilisée. Pour avoir ces résultats, ce sont les robots entrant dans un couloir qui sont comptabilisés et non pas les robots traversant entièrement les couloirs. Ainsi des robots qui entrent par hasard dans un couloir (pour éviter un obstacle, par exemple), seront comptabilisés alors qu'ils n'ont pas traversé le couloir d'une salle à une autre.

Dans un deuxième temps, on peut remarquer que cette dissociation se fait en un temps réduit, en début de simulation (environ 20000 pas). Cela montre l'efficacité de l'apprentissage par auto-organisation pour un espace de solutions qui est de l'ordre de 10^9 pour un vecteur de 30 booléens.

3.3.2 L'émergence est bien au rendez-vous

Les robots se sont auto-organisés pour transporter les ressources en passant par des couloirs spécifiques. Pourtant, en aucun cas, je n'ai codé un algorithme les conduisant à ce choix. La **spécification au micro-niveau** aboutit à l'apparition d'un comportement totalement nouveau pour le collectif : la circulation. Tout le long de mon étude, je me suis attaché à **une ligne de conduite de conception** me permettant d'affirmer que ce comportement est bien émergent, par définition. Un des buts de mon travail est donc atteint.

L'autre but de mon étude est de montrer que l'utilisation de la théorie des AMAS est pertinente pour la conception de collectivités robotiques. D'une certaine façon, je considère que ce travail a été en partie accompli. Bien sûr, les simulations ne sont pas la réalité et c'est pour cela que je nourris l'espoir de voir un jour des robots réels conçu de la même manière que les miens.

CONCLUSION

La première partie de mon travail consiste en une exploration des domaines suivants. Tout d'abord, les Systèmes Multi-Agents, de leurs origines aux concepts et théories développées actuellement, comme par exemple la théorie des **Systèmes Multi-Agents Adaptatifs**, ou AMAS. Cette partie met en évidence l'intérêt des SMA en ce qui concerne les problèmes complexes et non linéaires qui sont mis en exergue depuis plusieurs années. L'étude des SMA nous permet alors de nous intéresser au domaine, très proche, de la **Robotique Collective**. Les problèmes sont sensiblement les mêmes que ceux posés aux SMA. La Robotique Collective est en perpétuel changement, comme le témoignent les différents exemples présentés. Elle s'avère être aussi un formidable outil d'étude des phénomènes de masse, et de l'apparition de comportements collectifs. C'est pour cette raison, que le dernier sujet abordé lors de l'état de l'art est l'**émergence**. Ce concept, évoqué depuis la Grèce antique, est l'objet d'un renouveau et de l'intérêt de nombreux chercheurs. C'est sous cet angle, où le collectif émane de l'individuel, que l'étude s'est fixée.

Les résultats

Dans ce contexte, le développement d'un exemple de problème de Robotique Collective, le transport de ressources à travers des couloirs, est abordé dans l'idée **d'observer un comportement émergent** du collectif de robot. Après une spécification et une étude des situations non-coopératives, « orientées émergence », **quelques résultats** sont émis. Primo**,** la mise en place d'une circulation au sein du collectif est une nécessité pour que le groupe achève son travail dans le plus court délai. Secundo, les robots doivent être capables de se remettre en question et d'apprendre en fonction de leurs expériences. Ces conclusions sont la motivation pour le développement de capacités d'apprentissages. Une **étude de trois modules de décision** permettant aux robots de s'adapter aux situations est faite. Cette étude, et la mise en place de certains de ces modules dans les robots, permettent des expérimentations sous forme de simulations. Le résultat attendu se manifeste : le **comportement de circulation émerge du collectif**. Ce résultat est encourageant pour deux raisons. Tout d'abord, la méthode utilisée pour la spécification et la conception du système s'avère être efficace pour l'étude de systèmes complexes à fonctionnalité émergente. Ceci montre bien l'intérêt de développer des outils et des méthodologies comme ADELFE. Enfin, tout en considérant le cas de laboratoire que constitue mon application, la théorie des AMAS semble être pertinente en Robotique Collective.

Les perspectives

La plate-forme développée reste incomplète, mais offre de larges perspectives d'études futures :

- **Optimisation des résolutions de SNC** : car elles entraînent parfois des blocages de courte durée à l'entrée des couloirs ;
- **Expérimentation du « réseau logique adaptatif »** : en cours de développement. La structure de base est implémentée (classes et agents). Un **algorithme de traitement des SNC** met en œuvre l'auto-organisation ;
- **Etude d'autres comportements émergents** : avec ma mise en place de nouveaux types de robots et de comportements. Le problème du *« box-pushing »* semble intéressant si l'on dote les robots de capacités de mémorisation sur les autres robots

nominalement pour étudier la **formation éventuelle d'équipes** ou de hiérarchisations. On pourra aussi s'appuyer sur les travaux de Balch pour étudier la dynamique des castes et des formations de groupes [Balch, 1997].

- **Etude de la stabilisation du système** : comment se comporte le système face à des situations très déstabilisantes ? Par exemple quelle sera la réaction des robots s'il n'y a qu'un seul couloir ? On pourrait imaginer que si les robots manipulent la notion de temporalité, ils puissent mettre en place une circulation alternée. Ou bien encore, que faire si un couloir se bouche (fermeture de porte) ?
- **apprentissage par coopération** : les robots que j'ai conçu ne disposent volontairement d'aucun moyen de communication. On peut facilement imaginer des protocoles de communication simples afin que les robots partagent leurs connaissances. Par exemple, si deux robots se rencontrent et que l'un d'entre eux est en SNC d'inutilité, on peut concevoir un **moyen d'échanger ou de « fusionner » les croyances** afin de rendre plus efficace le robot non coopératif.

Les premiers pas vers une thèse

C'est la deuxième année que j'ai l'occasion d'effectuer un stage dans l'équipe SMAC. Notre première rencontre fut à l'occasion d'un stage de Maîtrise où je découvris le monde des Systèmes Multi-Agents et de l'éthologie. Aujourd'hui, mon année de DEA touchant à sa fin, et me retournant vers le travail effectué, je me rends compte du chemin parcouru. Découvrir le domaine des systèmes complexes ou de la Robotique Collective, comprendre des concepts métaphysiques ou philosophiques comme l'émergence, ou bien développer un environnement d'expérimentation : tant d'idées abordées et pourtant…

Pourtant, il reste tant à faire et à découvrir. Autant le travail accompli, que les résultats encourageants obtenus lors de cette étude, me forcent à croire qu'il faut continuer à chercher dans ce domaine. A mon sens, mes travaux constituent de bonnes fondations pour une thèse. L'émergence restera certainement mon domaine de prédilection, bien que la Robotique me fascine tout autant. L'idée de méthodologie de conception de systèmes complexes à fonctionnalités émergentes ou de la mise en place de plate-formes génériques d'agents auto-organisateurs et adaptatifs me paraissent être des phases essentielles afin de construire une base solide pour des applications futures que j'espère nombreuses.

ANNEXE : IMPLEMENTATION D'UN ENVIRONNEMENT DE SIMULATION

L'étude effectuée dans les paragraphes 2 et 3 reste très théorique. Par contre, les résultats obtenus sont le fruit de simulations dans un environnement de programmation pour la Réalité Virtuelle, *oRis*. Je vais brièvement présenter cet environnement, puis décrire en quelques lignes le programme conçu pour obtenir ces simulations.

Un langage de programmation orienté agent : *oRis*

L'Ecole Nationale d'Ingénieurs de Brest (ou ENIB) a mis au point un langage de programmation couplé à une plate-forme de simulation destinée à des applications de Réalité Virtuelle. Le problème de la visualisation des résultats et de simulation s'étant posé à de maintes reprises, l'équipe SMAC a décidé de développer quelques applications en langage *oRis*.

Un langage objet destiné aux Systèmes Multi-Agents

Ce langage est un langage objet qui s'inspire directement de langages tels que C++ ou Java de *Sun MicroSystems*. Les objets ont beaucoup de points communs avec les agents, il est donc naturel que la plupart des Systèmes Multi-Agents soient programmés en objet. Mais attention, généralement un objet n'est pas un agent. Un objet n'est pas capable naturellement de s'exécuter indéfiniment et indépendamment ou bien d'envoyer des messages aux autres objets**. Du point de vue d'*oRis***, par contre, **tout objet est agent**. Tous les objets implémentés peuvent recevoir, envoyer ou stocker des messages, sans programmation supplémentaire. De plus, un objet dispose toujours d'une méthode virtuelle *main*, qu'il peut définir ou non. A chaque pas de la simulation, tous les objets possédant une méthode *main* définie l'exécutent. L'ordre d'exécution lancé par le contrôleur de la simulation est peut être déterminé ou aléatoire. Ainsi un objet est capable d'évoluer seul, et indépendamment des autres.

Le langage *oRis* est un langage interprété, ce qui donne la possibilité de modifier des paramètres en cours de simulation.

Construction aisée de monde virtuel

Il est toujours difficile de faire des mondes virtuels, à la fois agréables à voir évoluer, et facile à concevoir. Grâce à *oRis*, cette tâche s'est avérée plutôt aisée. Que ce soit en 2 dimensions ou 3 dimensions, les objets sont faciles à construire (fig. 3).

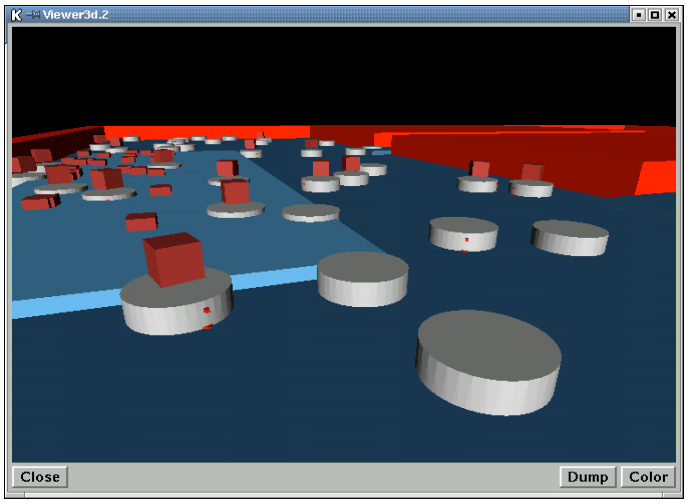


Figure 17 : *des robots virtuels évoluant dans un monde virtuel.*

Des résultats facilement observables

En ce qui concerne, l'expérimentation et l'exploitation de résultats, *oRis* dispose d'outils pratiques. Des navigateurs permettant de consulter en direct les différentes instances de classes, ainsi que leurs attributs, sont accessibles très facilement. Des constructions de courbes permettent d'éditer en direct des résultats.

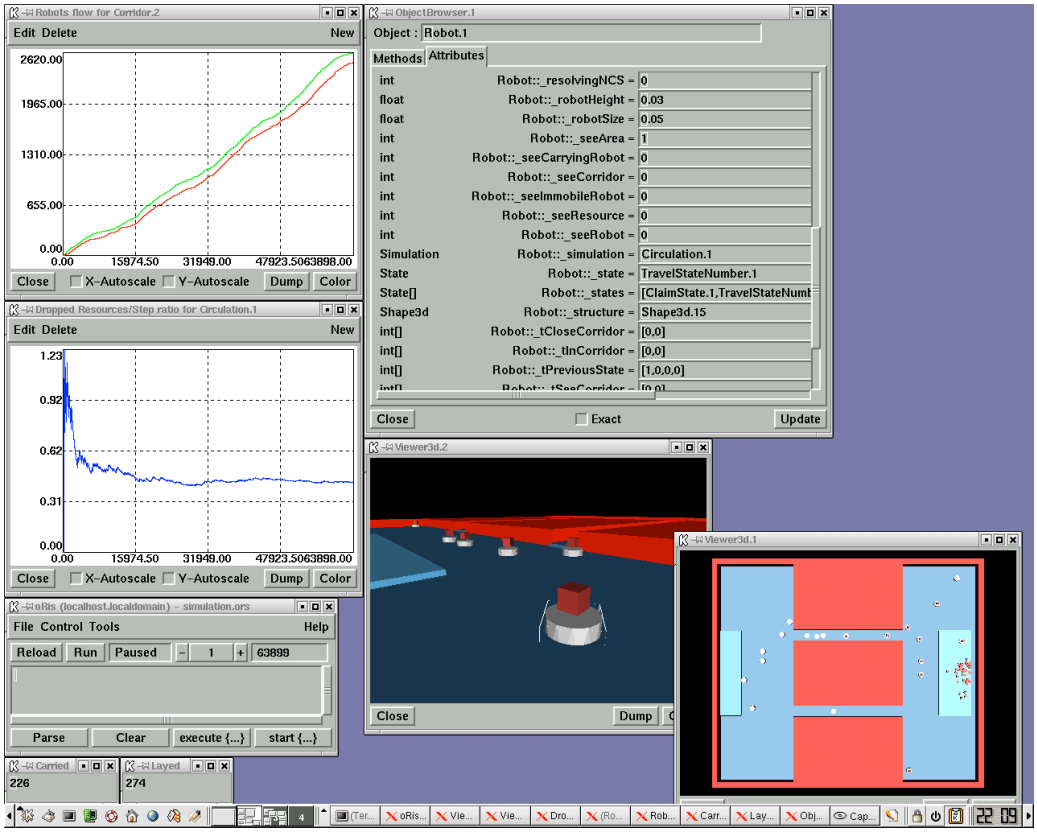


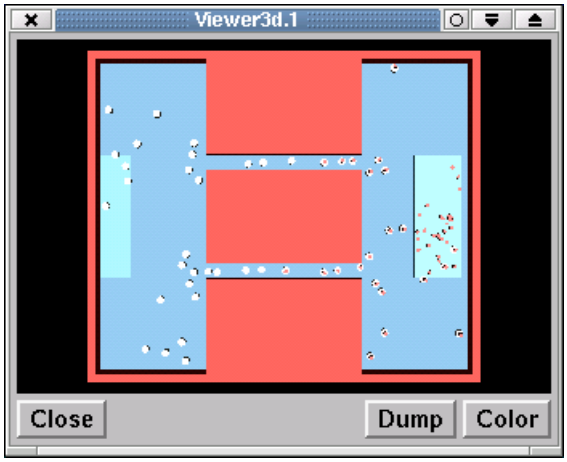
Figure 18 : *les nombreux outils d'oRis.*

Description du système implémenté

Bien qu'une majeure partie de mon travail lors de mon stage de DEA fut d'établir un état de l'art du domaine et ainsi pouvoir spécifier le « problème de la circulation », il a fallu passer à une phase de conception pour pouvoir vérifier mes hypothèses. Dans cette section je présenterai très brièvement le système implémenté.

Description générale

La conception du système a été décomposée en deux phases. La première phase a été de mettre en place un environnement de simulation en langage *oRis*. Cet environnement correspond à l'environnement décrit dans la section 2.2.3 (*L'environnement*). Il est totalement configurable, du nombre de robots et de couloirs, à la hauteur des murs en passant par la taille des salles.



Description des agents de base

Chaque agent du système est défini de la manière la plus générale possible dans un souci de réutilisabilité et d'affectation à d'autres tâches.

Les agents implémentés sont (voir fig. 19):

- **les robots** : disposant de senseurs divers, d'un module de décision et d'effecteurs. Ce sont des « *threads* » répétant en boucle la procédure suivante : *perceive – decide – act*. Ces méthodes font appel aux méthodes correspondantes de l'état interne actif. Chaque agent robot possède un jeu d'états internes auxquels il pourra faire appel.

- **les comportements** : disposant des trois méthodes principales, différentes en fonction des comportements. La méthode *perception* met à jour un vecteur d'entrées. La méthode *decision* calcule les valeurs des effecteurs (la vitesse et le sens de rotation en général). Enfin, la méthode *action* fait agir le robot dans la simulation. Tous les comportements décrits dans la section 2.3.1 héritent directement ou indirectement de la classe *Behavior* que j'ai définie.

- **Les états** : considérés comme des comportements particuliers. Un état pointe sur un comportement actif. Ses méthodes *perception*, *decision* et *action* appellent les méthodes correspondantes du comportement actif. Chaque agent état possède un jeu de comportement sur lesquels il pourra pointer. Tous les états décrits dans la section 2.3.1 héritent directement de la classe *State* que j'ai aussi définie.

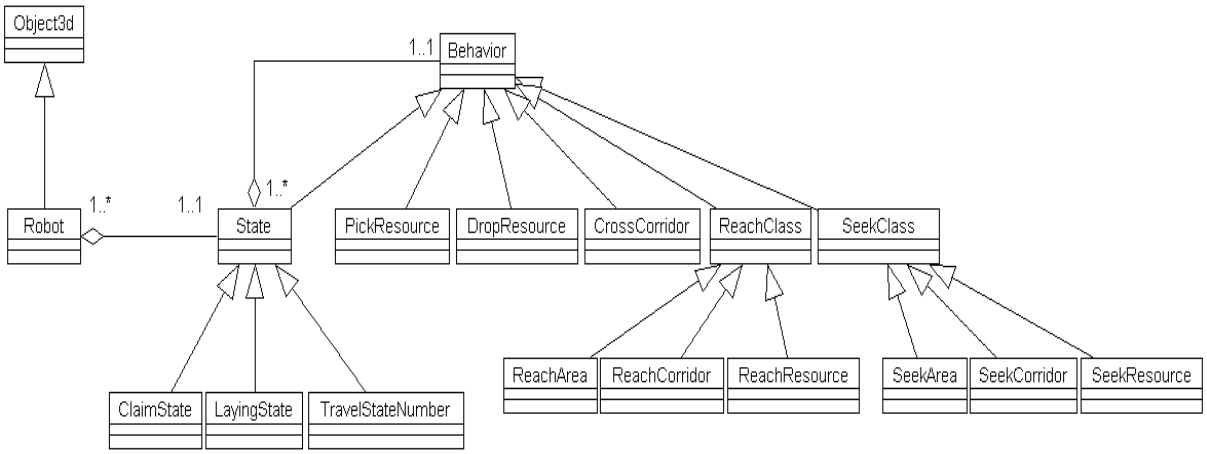


Figure 19: *diagramme de classes des agents*.

Cette spécification permet de rajouter facilement de nouveaux types de comportements, d'agents, ou bien de robots afin d'étudier d'autres tâches de collectifs comme par exemple le « *box-pushing* » [Kube, 1993] [Matarić, 1995].

REFERENCES BIBLIOGRAPHIQUES

Emergence

[Ali, 1997]
S.M. ALI, R.M. ZIMMER – The Question Concerning Emergence – CASYS'97, Abstract Book, 1st International Conference on Computing Anticipatory Systems, CHAOS asbl, 1997

[Georgé, 2000]
Jean-Pierre GEORGE – Etudes des caractéristiques d'un environnement de programmation émergente utilisant le principe des systèmes multi-agents auto-organisateurs – Rapport DEA, IRIT, 2000

[Goldstein, 1999]
Jeffrey GOLDSTEIN – Emergence as a Construct : History and Issues – Emergence : a Journal of Complexity Issues in Organizations and Management, the New England Complex Systems Institute, Vol. 1, Issue 1, 1999

[Holland, 1997]
John H. HOLLAND – Emergence : From Chaos to Order – Addison-Wesley, 1997

Robotique Collective

[Balch, 1997]
Tucker BALCH – Social Entropy : a New Metric for Learning Multi-Robot Teams – Proceedings, 10th International FLAIRS Conference (FLAIRS-97), 1997

[Brooks, 1986]
Rodney A. BROOKS – A Robust Layered Control System for a Mobile Robot – IEEE Journal of Robotics and Automation, RA-2, 14-23, 1986

[Drogoul, 1998]
Alexis DROGOUL, Milind TAMBE, Toshio FUKUDA – Collective Robotics – Lecture Notes in Artificial Intelligence 1456, Proceedings, 1st International Workshop, CRW'98, Paris , France, July 1998

[Goldberg, 2000]
Dani GOLDBERG, Maja J. MATARIĆ – Robust Behavior Based Control for Distributed Multi-Robot Collection Tasks – USC Institute for Robotics and Systems Technical Report IRIS-00-387, July 2000

[Kube, 1993]
C. Ronald KUBE, Hong ZHANG – Collective Robotics Intelligence : From Social Insects to Robots – Proceedings, 3rd International Conference on Simulation of Adaptive Behavior, 1993

[Kube, 1996]
C. Ronald KUBE, Hong ZHANG – The Use of Perceptual Cues in Multi-Robot Box-Pushing – Proceedings, IEEE International Conference on Robotics and Automation, 1996

[Matarić, 1994]
Maja J. MATARIĆ – Interaction and Intelligent Behavior – MIT EECS PhD Thesis AITR-1495, MIT AI Lab, 1994

[Matarić, 1995]
Maja J. MATARIĆ, Martin NILSSON, Kristian T. SIMSARIAN – Cooperative Multi-Robot Box-Pushing – IEEE/RSJ International Conference on Intelligent Robots and Systems, Pittsburg, PA, 1995

[Matarić, 1997]
Maja J. MATARIĆ – Behavior Based Control : Examples from Navigation, Learning and Group Behavior – Journal of Experimental and Theorical Artificial Intelligence, Vol.9, 1997

[Vaughan, 2000]
Richard T. VAUGHAN, Kasper STØY, Gaurav S. SUKHATME, Maja J. MATARIĆ – Blazing a trail : Insect-inspired resource transportation by a robot team – Proceedings, 5th International Symposium on Distributed Autonomous Robotic Systems (DARS), Knoxville, TN, Oct 4-6, 2000

[Vaughan & al, 2000]
Richard T. VAUGHAN, Kasper STØY, Gaurav S. SUKHATME, Maja J. MATARIĆ – Go ahead, make my day : Robot Conflict Resolution by Aggressive Competition – Proceedings, 6th International Conference on the Simulation Adaptive Behavior (SAB-2000), Paris, France, Sep 11-15 2000

Systèmes Multi-Agents

[Camps, 1997]
Valérie CAMPS, Bernard CARPUAT, Marie-Pierre GLEIZES, Pierre GLIZE, André MACHONIN, Jo PEZET, Christine REGIS, Sylvie TROUILHET – Une théorie de la coopération pour l'auto-organisation des systèmes artificiels - Rapport interne n°97-58-R, IRIT, 1997

[Camps, 1998]
Valérie CAMPS – Vers une théorie de l'organisation dans les systèmes multi-agents basés sur la coopération –Thèse IRIT, n° 2890, 1998

[Ferber, 1995]
Jacques FERBER – Les Systèmes Multi-Agents : vers une Intelligence Collective – Interéditions, 1995

[Athanassiou, 1999]
Eleutherios ATHANASSIOU, Delia CHIRICHESCU, Marie-Pierre GLEIZES, Pierre GLIZE, Nick LAKOUMENTAS, Hans SCHLENKER, Alain LEGER, José MORENO – Abrose : A Cooperative Multi-Agent Based Framework for Marketplace – IATA, Stockholm, Sweden, August 1999

[Sontheimer, 1999]
Thomas SONTHEIMER – Modéle adaptatif de prévision de crues par systèmes multi-agents auto-organisateurs – Stage IUP, 1999

[Topin, 1999]
Xavier TOPIN – Etude de l'auto-organisation par coopération appliquée à l'apprentissage comportemental de robots-fourmis – Rapport DEA, IRIT, 1999

INDEX

NOTES