Table des matières

Introduction générale

Dans divers domaines comme ceux des applications industrielles et domestiques, il peut être utile voir même nécessaire (si le milieu est hostile par exemple) de faire coopérer un groupe de robots mobiles dans le but d'accomplir une tâche spécifique, et complexe. Pour mettre en œuvre cette coopération, de nouvelles approches ont vu le jour cette dernière décennie, s'inspirant essentiellement du vivant. La colonie de fourmis capable de réaliser des tâches complexes sans qu'aucune fourmi n'ait conscience de l'œuvre globale accomplie en est un exemple. Les systèmes multi-agents réactifs représentent l'un des modèles de simulation des plus utilisés dans ce cas. Il est à noter que la réactivité des agents ainsi considérés leur permet d'être sensibles aux changements de l'environnement dans lequel ils évoluent, et de réagir directement en conséquence, sans passer par une phase de raisonnement. Il est important de souligner aussi que les robots sont complètement autonomes; en l'absence d'un agent de contrôle central, les colonies de fourmis par exemple, ont la capacité de travailler ensembles au profit de la société toute entière. Chaque fourmi se comporte selon sa perception locale, malgré ceci, des comportements globaux significatifs en résultent. Toutes les fourmis participent à l'émergence d'une tâche à l'échelle macro, en revanche, aucune fourmi n'est indispensable à la dynamique globale, ce qui procure une particularité très intéressante à ces systèmes. Dans ce sens, les chercheurs espèrent, à travers leurs travaux dans le domaine de la robotique collective, atteindre un double but : faciliter la conception des systèmes distribués, et enrichir du coup notre connaissance de la façon dont la collectivité biologique réalise la coopération.

Chaque agent dans la collectivité se comporte selon son propre jeu de règles. Si tous les agents se comportent selon le même jeu de règles, la population est dite homogène ; autrement elle est dite hétérogène. Dans ce type de systèmes, le contrôle décentralisé est préféré au contrôle centralisé, étant donné qu'il est favorable à la mise à l'échelle. Il est à noter que ces entités qui ont l'aptitude d'organiser leurs activités pour réaliser des tâches prédéterminées sans intervention humaine, sont capables de s'auto-organiser. L'étude de la manière dont les systèmes peuvent s'auto-organiser dans leur milieu naturel, est l'un des sujets suscitant le plus d'intérêt actuellement; toute la problématique réside dans la façon de trouver ces règles de communication significatives, conduisant à cette auto-organisation, et faisant émerger une tâche cohérente qui dépasse la somme des parties du système ainsi considéré. Une autre question tout aussi intéressante est de savoir comment asservir le concept de l'auto-organisation au profit de l'humanité en construisant des systèmes artificiels résolvant des problèmes complexes dont on n'arrive pas vraiment à faire face actuellement.

Dans cette perspective, il s'agit dans le cadre du travail mené dans ce mémoire, de rechercher via une simulation multi-agents, l'ensemble des règles sensori-motrices dont on doit doter une collectivité homogène de robots autonomes et mobiles au niveau micro du système cible, afin de faire émerger la tâche de classification d'objets au niveau macro de ce même système, selon une ou plusieurs propriétés caractérisant les objets à classifier. Il est à noter que lorsqu'on parle de « système » on fait plutôt allusion aux objets à classifier, aux robots et à l'espace où opèrent ces robots représentés sous forme d'un tore ou d'un anneau.

A la sortie chaque robot est doté du même jeu de comportements de base, et à l'entrée il est capable de distinguer dans son environnement local (5-voisinages), entre les différentes entités qui peuvent s'y trouver (robots ou objets) et qu'il arrive à reconnaître. L'idée est de rechercher via une approche évolutionniste la plus adéquate des listes de correspondance liant à chaque perception locale possible, l'un des comportements de bases précités. Il faut comprendre le mot « adéquate » dans le sens où cela nous permet de réaliser la classification.

Le travail qui a conduit à l'accomplissement de ceci est organisé à travers ce mémoire comme suit :

Dans le chapitre 1, il s'agit de survoler le domaine de la vie artificielle, en montrant les domaines les plus importants qui lui sont connexes. Le chapitre 2 traite des concepts de base sur lesquels s'appuient l'inspiration et la réflexion qui ont guidé ce projet, à savoir : l'auto-organisation, l'adaptation et

l'apprentissage. En ce qui concerne l'apprentissage on présente les approches les plus adoptées dans le domaine de la robotique collective, à savoir : l'approche évolutionniste, celle par renforcement, et la dernière utilisant les réseaux de neurones. Le chapitre 3 nous fait part du domaine des systèmes multi-agents (SMA) utilisés en tant qu'outil de modélisation afin de simuler le système effectif représentant notre problème cible de classification. Aussi nous fait il part du domaine concret de la robotique collective, domaine d'appartenance du système réel à simuler, et essaye d'établir une relation entre les SMAs et la robotique collective. Le chapitre 4 expose le problème de classification d'une manière exhaustive, puis focalise sur le problème de classification d'objets selon un ou plusieurs critères donnés, dans un environnement de robotique collective, et propose trois solutions pour la classification monocritère et une solution pour la classification multicritères. Dans le cinquième et dernier chapitre on montre la mise en œuvre des solutions ainsi proposées, et les résultats de la simulation qui en découlent, programmées dans l'environnement de simulation basé agents « NetLogo ». Une conclusion générale vient pour clôturer ce travail ainsi accompli en donnant une synthèse de tout ce qui a été fait ou acquis, et en ouvrant une parenthèse de perspectives sur ce qu'on projette de faire dans le futur, que ce soit le proche avenir ou à long terme.

1.1. Historique

La première approche du vivant à travers des modèles informatiques est due aux travaux du mathématicien John Von Neumann (1903-1957) à Los Alamos. Von Neumman travaillait particulièrement sur la théorie des automates autoreproducteurs. Il démontra ainsi que l'une des propriétés du vivant parmi les plus importantes : « l'autoreproduction », peut être expliquée sans recourir à aucune « force de vie » mystique. Sa démonstration était basée sur une configuration d'automates cellulaires capables de se reproduire. Le problème réside dans ce cas dans la façon de démontrer cette capacité de reproduction à une époque où la technologie ne permettait pas une mise en pratique de ces idées. Ce problème paraissait insurmontable. La solution vint d'une façon inattendue grâce au mathématicien Stanislas M. Ulam qui mit au point des programmes basés sur quelques règles simples appliquées récursivement. Ainsi Ulam aboutissait après quelques itérations à une figure élégante baptisée « récif de corail »; dont des motifs semblaient se reproduire, dans certaines conditions particulières.

La découverte de l'ADN (Acide Désoxiribo Nucléique) fut postérieure aux travaux de Von Neumann, concernant l'autoreproduction. Il est frappant de noter aujourd'hui les similitudes qui existent entre le modèle établi par Von Neumman et les découvertes biologiques dans ce sens. Notons que les travaux de Von Neumann sur l'autoreproduction, étaient fondés sur une approche numérique à états discrets, à l'inverse, Norbert Wiener proposa en 1948, une approche continue (ou analogique) du contrôle et de la communication dans les domaines des êtres vivants et des machines. Wiener est considéré aujourd'hui comme le fondateur de la cybernétique*.

Après la seconde guerre mondiale, dans les années qui suivirent, James Thatcher, E.F.Codd, Richard Laing et bien d'autres développèrent des automates autoreproducteurs similaires à celui de Von Neumann. Dans un autre domaine, Grey Walter développa en 1950 deux « tortues électroniques » qu'il nomma «Elmer» et « Elsie».

En 1950, L.S. Penrose conçut une série de modèles mécaniques illustrant un principe d'autoreproduction. A partir des années 60, les travaux visant à reproduire artificiellement certaines propriétés des êtres vivants se multiplièrent, pour ne citer que le célèbre « jeu de la vie » de John Horton Conway, ainsi que les recherches dues à John Holland sur les algorithmes génétiques.

Il est important de mentionner aussi l'explosion (à cette même époque) des recherches en Intelligence Artificielle, sur les réseaux de neurones et en robotique, qui ont influencé les travaux actuels. Ce n'est qu'en 1987, sous l'impulsion de Christopher Langton, que se déroula la première conférence sur le thème de la vie artificielle. Qui reste à nos jours une notion difficile à cerner, étant donné que c’est un carrefour où (entre autres) robotique, psychologie, informatique et biologie se sont rencontrées pour dévoiler quelques mystères de la vie jalousement gardés. Dans ce qui suit, nous allons essayer de définir la vie artificielle et de présenter succinctement la plupart des domaines connexes à cette dernière.

1.2. Définition de la vie artificielle (VA)

D'après Jeau-claude Heudin [1], la définition de la vie artificielle pourra s'énoncer comme suit :

Définition 1

La vie artificielle est l'étude des systèmes conçus par l'homme, qui présentent des comportements caractéristiques des systèmes vivants naturels. Elle complète l'approche traditionnelle de la biologie, dont le mode de fonctionnement est l'analyse des êtres vivants, en essayant de synthétiser des comportements « vivants » sur ordinateur et/ou en utilisant d'autres supports artificiels. Par

l'extension des fondations empiriques sur lesquelles la biologie est fondée, au delà des chaînes carbonées des organismes vivants qui ont évolué sur terre, la vie artificielle peut contribuer à la biologie théorique en plaçant la vie « telle que nous la connaissons » dans le contexte plus vaste de la vie « telle qu'elle pourrait être».

Définition 2

La vie artificielle a été définie par Langton [13] comme étant l'étude de systèmes construits par l'homme qui présentent des comportements caractéristiques des systèmes vivants naturels.

Définition 3

J. Doyne Farmer et Alleta Belin, ont défini la vie artificielle sous forme d'une liste de propriétés caractérisant un système de vie artificielle [23] : les propriétés minimales que nous retrouvons dans tout système de vie artificielle sont :

1. L'être humain a contribué au processus d'apparition de tout système de vie artificielle ;
2. Un système de vie artificielle est autonome ;
3. Un système de vie artificielle est en interaction avec son environnement ;
4. Il y a émergence de comportements dans un système de vie artificielle ;

Les propriétés qui ne sont pas indispensables mais restent néanmoins très présentes sont :

5. Un système de vie artificielle peut se reproduire lui-même ;
6. Un système de vie artificielle possède une capacité d'adaptation ;
7. Un système de vie artificielle n'est pas une unité. A l'opposé de la vie, un système de vie artificielle peut être réparti en plusieurs endroits : exemple, un robot et un ordinateur pouvant effectuer les calculs reliés par ondes. Même à l'intérieur d'un ordinateur, rien ne garantie que les octets de ce système sont tous regroupés.

Autres définitions

- Tentative de reproduire le « vivant » dans une machine, dans des conditions initiales définies et à partir de mécanismes simples.
- Discipline cousine de l'intelligence artificielle, cherchant à créer des êtres vivants artificiels, en particulier des organismes capables d'évoluer selon des règles qui n'ont pas été rédigées par le concepteur initial. Elle est controversée, surtout parce qu'elle travaille énormément sur les virus, qui sont les formes de vie les plus primitives aussi bien dans la nature que dans les ordinateurs, et qui posent pas mal de problèmes.

De ce qui suit , en arrive à la conclusion que le domaine de la vie artificielle se base sur la théorie de la vie, justifiée en partie, par la connaissance des aspects structuraux des organismes, mais pas satisfaisante dans sa totalité en rapport avec la réalité du vivant. Il faut croire que la nature de la vie n'est pas encore connue; la physique ne peut pas l'expliquer avec les connaissances classiques, ni avec celles modernes et contemporaines. La biologie, non plus. C'est un défi à la science de notre ère qui se prétend mature.

Il est clair dans ce cas que la définition de la vie artificielle est dépendante de la définition de la vie sans équivoque. Or cette dernière étant problématique, il est donc particulièrement difficile de définir la vie artificielle et surtout de la limiter : A partir de quand ne parle-t-on plus de vie, mais de vie artificielle ? Quelle est la limite entre une machine ou un programme, et une forme de vie artificielle ?

Ces questions qui semblent apparemment innocentes sont très épineuses quant à leurs réponses.

Lors de la troisième conférence de Santa Fe, Claus Emmeche [51] évoque les différentes formes de la vie artificielle. Il les a classé en deux groupes principaux : les versions triviales, et les versions non triviales. Pour chacun de ces groupes, nous pouvons y trouver la vie artificielle faible ou forte. La vie artificielle forte correspond à une recherche de création de vie, tandis que l'approche faible correspond plutôt à une imitation de la vie.

a. L'aspect trivial comprend :

- Les simulations et les modèles. Cela correspond aux simulations informatiques fondées sur des modèles mathématiques, conceptuels ou physiques des systèmes biologiques. Elle ne peut aboutir à des formes réelles de vie.
- Les organismes modifiés. Il s'agit d'êtres vivants réels, modifiés par l'homme au moyen de manipulations génétiques par exemple.

b. L'aspect non trivial correspond à/aux :

- Systèmes computationnels. Ces réalisations sont par définition strictement informatiques. Elles ont pour objectif la création d'organismes virtuels considérés comme vivants (du moins par leur auteur). C'est la voie forte la plus contestée de la vie artificielle.
- La robotique évolutive. Cela regroupe les robots autonomes et évolutifs.
- Expériences relevant de la biochimie. Cette dernière catégorie inclut les synthèses de processus pré-biotiques et d'organismes primitifs grâce à des expérimentations physico-chimiques in vitro.

Il semble donc qu'il y ait plusieurs formes de vie artificielle ; et nous voyons qu'elles diffèrent énormément les unes des autres. Il serait donc peut-être plus opportun de ne pas les rassembler sous le même terme, mais de créer de nouveaux noms pour chacune de ces catégories.

On pourrait encore s'interroger une autre fois sur les cas limites, à travers des exemples, comme ceux des prothèses. Avons-nous une vie artificielle, si nous avons un cœur artificiel ou même si nous sommes un transplanté cardiaque ? Cela nous amène à nous interroger sur ce qu'est une prothèse ? Des lunettes de vue sont-elles des prothèses ? Si oui, alors nous pouvons dire que lorsque nous chaussons des skis, il s'agit là aussi de prothèses ; tout comme pourrait l'être finalement, dans le même ordre d'idées, une voiture.

La définition de la vie artificielle est donc bien problématique; difficile de fixer la frontière entre vie et vie artificielle (problème de prothèses par exemple), et difficile de fixer aussi la limite entre vie artificielle et un programme doté d'intelligence artificielle par exemple.

Dans le contexte du travail mené dans ce projet, on peut considérer comme vie artificielle ce que Claus Emmeche appelait « vie artificielle faible », et qui correspondait à une imitation de la vie au sens trivial. C'est à dire : les simulations et les modèles

1.3. Automates cellulaires

Un Automate Cellulaire (AC) est avant tout une machine formelle, c'est-à-dire un objet mathématique abstrait ayant un fonctionnement, c'est-à-dire qui est capable de produire un processus en général spatio-temporel et discret, et qui peut être, selon le cas, déterministe ou stochastique. Ce processus est représenté par une suite finie de situations de l'automate, depuis un état initial, jusqu'à l'état final. Un état (global) de l'automate est constitué du motif formé des états (élémentaires) de ses composants appelés cellules [45].

Il y en a aussi qui définissent les automates cellulaires comme des constructions abstraites aux propriétés très complexes et dont l'abord n'est pas immédiat. On fait généralement remonter l'histoire des automates cellulaires aux années quarante et à Stanislas Ulam. Ce mathématicien s'est intéressé à l'évolution de constructions graphiques engendrées à partir de règles simples. La base en était un espace à deux dimensions divisé en « cellules », soit une sorte de feuille quadrillée. Chacune des cellules pouvait avoir deux états : allumé ou éteint. Partant d'une configuration donnée, la génération suivante était déterminée en fonction de règles de voisinage (voir Fig 1.1) : une cellule reste vivante (allumée) si un nombre déterminé des cellules de son voisinage sont vivantes, sinon elle meurt (elle s'éteint).

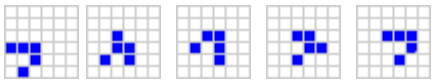


Fig 1.1 Le planeur est une des structures les plus répandues du jeu de la vie. Structure simple constituée de 5 cellules, elle se déplace de manière rectiligne dans l'univers du jeu. Périodique, elle se retrouve identique à elle même au bout de 4 générations

Ulam, qui utilisait l'un des premiers ordinateurs, a rapidement constaté que ce mécanisme permettait de générer des figures complexes et esthétiques et que dans certains cas, ces figures pouvaient se répliquer (voir fig 1.2). Des règles extrêmement simples permettaient de construire des structures très complexes. À partir de là, se posait la question suivante : ces mécanismes récursifs (c'est-à-dire en l'occurrence dépendant de leur propre état antérieur) peuvent-ils expliquer la complexité du réel ? Cette complexité n'est elle qu'apparente, les lois fondamentales étant elles-mêmes simples ?

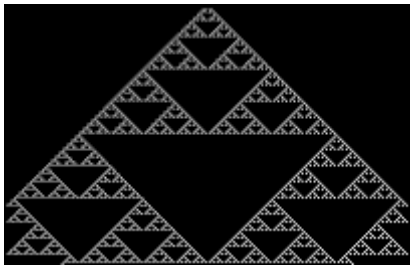


Fig 1.2 Triangle de Pascal

En parallèle, John von Neumann [7] (en s'appuyant sur les travaux de A.Turing) s'intéressait à la théorie des automates autoréplicateurs et travaillait à la conception d'une machine autoréplicatrice le « kinématon ». Une telle machine devait être capable, à partir de matériaux trouvés dans l'environnement, de produire n'importe quelle machine décrite dans son programme, y compris une copie d'elle-même. Von Neumann montrait ici comment résoudre le problème de l'autoréférence de la description.

Pour s'autorépliquer, la machine devrait en effet contenir une description d'elle-même, mais pour être complète, cette description doit également être décrite, etc. La solution réside dans la capacité donnée à la machine d'interpréter sa description à la fois comme un programme, une séquence d'instructions, et comme un composant. La description sera d'abord interprétée pour construire la nouvelle machine, elle sera ensuite simplement copiée afin de donner à la nouvelle machine une description d'elle-même.

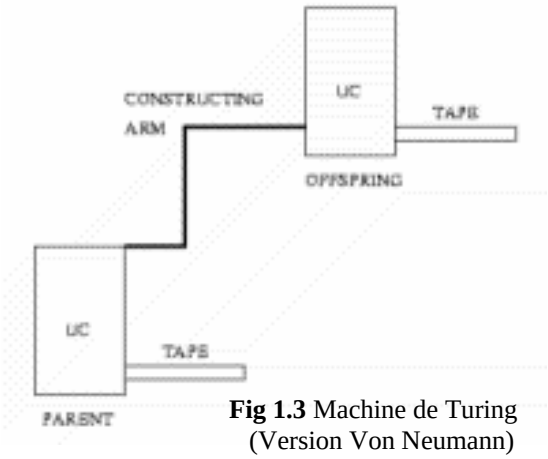
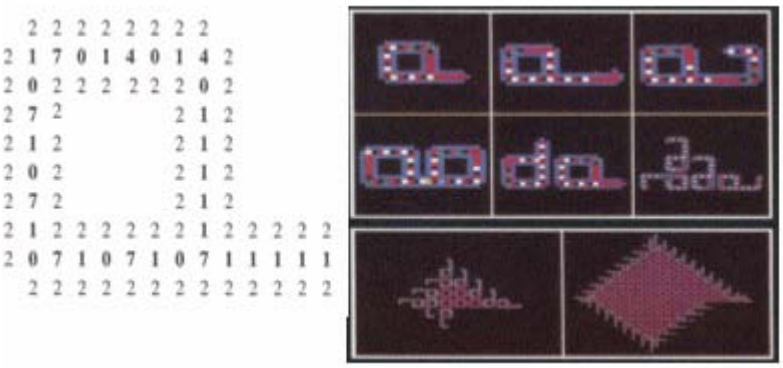


Fig 1.3 Machine de Turing (Version Von Neumann)

Ce mécanisme correspond de fait à l'interprétation actuelle du fonctionnement de la molécule d'ADN découverte après les travaux de von Neumann. C'est S. Ulam qui a suggéré à von Neumann d'utiliser ce qu'il appelait les « espaces cellulaires» pour construire sa machine autoréplicatrice. Il pouvait ainsi s'affranchir des conditions physiques réelles pour travailler dans un univers extrêmement simplifié pourtant apte à engendrer une haute complexité. Les automates cellulaires sont sortis des laboratoires en 1970 avec le désormais fameux « Jeu de la vie » (Life Game) de John Horton Conway qu'on verra un peu plus loin.



Christopher Langton [46] créa en 1984 le premier automate cellulaire. Contrairement à l'entité « cellule » du jeu de la vie possédant uniquement un état « vivant » et un état « mort », cet automate, plus complexe, est composé d'une centaine de cellules comprenant huit états possibles (voir fig 1.4).

Fig1. 4 L'automate de Langton est un automate à 8 états, géré par 29 règles. Il est formé de 86 cellules : une part de ces cellules (bleue « à l'état 2 ») forme une gaine fermée (telle une membrane) repliée en boucle et délimitant un espace de circulation d'information, formé par les cellules restantes. Ce conduit permet la circulation de données sous forme de signaux : couples de cellules (rouge-noire, jaune-noire ou états 0, 1, 4, 7), qui constituent le support « génétique » (l'ADN) de l'automate. La boucle s'achève par une « tige » qui est considérée comme bras de projection où se déroule la duplication.

On remarque dans la figure juste ci-dessus que lorsque les signaux circulants rencontrent la jonction avec le bras de projection, ils sont dupliqués. Une copie est renvoyée dans la boucle, et l'autre copie se propage le long du bras. C'est l'équivalent de la transcription : l'ADN du gène à exprimer est copié sous forme d'une molécule d'ARN. En arrivant au bout du bras, les copies de signaux sont transformées en instructions. C'est l'équivalent de la traduction : à partir des instructions présentées sur l'ARN, la cellule construit une protéine.

Le bras s'étend selon ces instructions pour former une autre structure similaire à la première, dont elle se détachera par la suite, et contenant le même patrimoine informationnel permettant la réplication. L'ajout d'une règle de « stérilisation », qui bloque l'évolution au bout d'un certain nombre de générations, permet la cristallisation des boucles les plus anciennes et amène à la construction d'une sorte de corail.

La propriété émergente de ce type d'automates est la structure que prend une population d'automates reproduits à partir d'un unique automate. Après quelques générations seulement, l'organisme résultant reprend le même développement et la même architecture que les « récifs de coraux ». Les automates cellulaires font apparaître un autre principe de la vie artificielle : Le principe de similitude qui représente l'apparition d'une structure émergente de l'ensemble de l'organisme résultant, possédant une similitude de forme proche de certaines espèces naturelles.

1. 3.1. Classes d'automates

Compte tenu de la diversité des types de comportements possibles à partir d'automates cellulaires, il est utile de prendre en compte les travaux de Stephen Wolfram [8]. Celui-ci après avoir étudié de nombreux types d'automates, a établi une classification des comportements :

- **Classe I :** Les automates cellulaires évoluant vers des états fixes et homogènes, comme les structures cristallines, les récifs de coraux ou encore celles du type « beehive » du jeu de la vie.
- **Classe II :** Les automates cellulaires aboutissant à des structures périodiques simples, comme des mouvements d'oscillations ou les traffic lights du jeu de la vie.
- **Classe III :** Les automates cellulaires évoluant vers des comportements chaotiques, caractérisés par des attracteurs étranges et des structures apériodiques, comme certaines formes apparaissant dans le jeu de la vie.
- **Classe IV :** Les automates cellulaires aboutissant à l'émergence de structures globales complexes, comme les planeurs (gliders) du jeu de la vie.

Les automates cellulaires de la classe I aboutissent à des structures uniformes et homogènes, similaires aux cristaux. Ceux de la seconde classe sont également caractérisés par des structures ordonnées, mais pas aussi accomplies que dans la première classe puisqu'elles oscillent généralement entre plusieurs configurations. Les automates de la classe III montrent des comportements totalement désordonnés, tant au niveau local que global. La classe IV est de loin la plus intéressante pour la Vie Artificielle, mais également la moins connue. Les automates de cette catégorie présentent des motifs locaux ordonnés, mais très peu d'organisation apparente au niveau global. Le « jeu de la vie » de Conway en est le plus représentatif.

1.3.2. Automate élémentaire de Wolfram (à 1 Dimension)

Au début des années 80 Stephen Wolfram [8] utilisait des automates cellulaires mettant en jeu des règles simples afin d'étudier la complexité dans la nature. Il a systématisé cette approche de la complexité dans son livre « A New Kind of Science » (2002) et a contribué au domaine des automates cellulaires en automatisant l'exploration.

a. Architecture de l'automate :

Cet automate est l'un des plus simples et pourtant son étude permet d'envisager la complexité que peut engendrer ce type de systèmes. Il est le seul dont on puisse étudier de manière exhaustive l'ensemble des comportements possibles, en appliquant « à la main » les différentes fonctions de transitions, et en observant l'évolution de l'automate. C'est en ce sens que son étude est particulièrement intéressante.

Caractéristiques:
- 1 dimension (1 ligne de cellules).
- k = 2 états (actif (1) / passif (0)).
- Voisinage « r »: les deux cellules adjacentes + la cellule cible.
- Pour k = 2 et r = 3, il y a 256 règles de transition différentes.

b. Exemple d'évolution :

On sélectionne 8 règles de transition parmi 256: (cellule cible en mauve et cellules adjacentes en noir)

8 configurations de voisinage

possible: **111 110 101 100 011 010 001 000**

Le nouvel état de la cellule

centrale: **101 110 101 110 011 010 011 000**

Temps ↓

Automate 1D, k = 2, r = 3
- en abscisse : la dimension de l'automate.
- en ordonnée : e temps.

La représentation des automates à une dimension (une ligne), utilise la seconde dimension pour représenter le temps. À chaque génération, une nouvelle ligne est ajoutée au-dessous de la précédente, on peut visualiser ainsi la dynamique de ce type d'automates. L'univers initial de ces automates (ligne de cellules du haut) est aléatoire, avec équiprobabilité pour chacun des états possibles.

Les réseaux d'automates cellulaires sont des matrices régulières de machines très simples. Ces machines sont parmi les plus élémentaires que l'on puisse imaginer, car elles ne peuvent se trouver à un instant donné que dans un seul état choisi parmi un nombre restreint d'états possibles. Les règles de transition permettent de définir exactement l'état d'un automate en fonction de celui de ses voisins. Ils sont, à plus d'un titre, une approche intéressante pour la Vie Artificielle:

- Les automates cellulaires sont des systèmes dynamiques non linéaires,

- Leurs comportements couvrent l'ensemble des dynamiques possibles, des structures simples jusqu'au chaos total,
- Les automates cellulaires sont capables de supporter la plus universelle des fonctions de calcul, c'est à dire la machine de Turing;
- Ils permettent la modélisation des systèmes physiques sous la forme d'une « matière artificielle » dont nous contrôlons finement le comportement des « molécules » et des « atomes » qui la constituent,
- Ils sont d'une mise en oeuvre relativement simple sur un ordinateur.

Cette description montre donc l'étendue des comportements envisageables d'automates cellulaires possédant toujours des règles simples. Elle met en évidence un autre principe de la vie artificielle : Le principe de diversité représente l'étendue des dynamiques comportementales émergentes d'agents pourvus de certaines propriétés du vivant.

1.4. Jeu de la vie

À l'origine, le Jeu de la vie fut présenté comme un jeu mathématique. Sa description va nous permettre de matérialiser et mieux comprendre ce que sont les automates cellulaires. C'est en 1970, que le mathématicien John Conway [55], eu l'idée de créer un jeu, avec un minimum de règles, où des cellules naissent, vivent et disparaissent, dans un espace discret toroïdal à deux dimensions. Les cases de cet espace représentent les cellules du jeu et possèdent deux états possibles : vivant ou mort, c'est à dire allumé ou éteint. Après une initialisation aléatoire de l'environnement, l'état des cellules à un instant t+1est déterminé par son état à l'instant t et l'application des deux lois suivantes:

- Une cellule naît si elle possède trois cellules voisines vivantes.
- Une cellule meurt si elle possède moins de deux cellules voisines vivantes (mort par isolement) ou plus de trois cellules voisines vivantes (mort par sur-population).

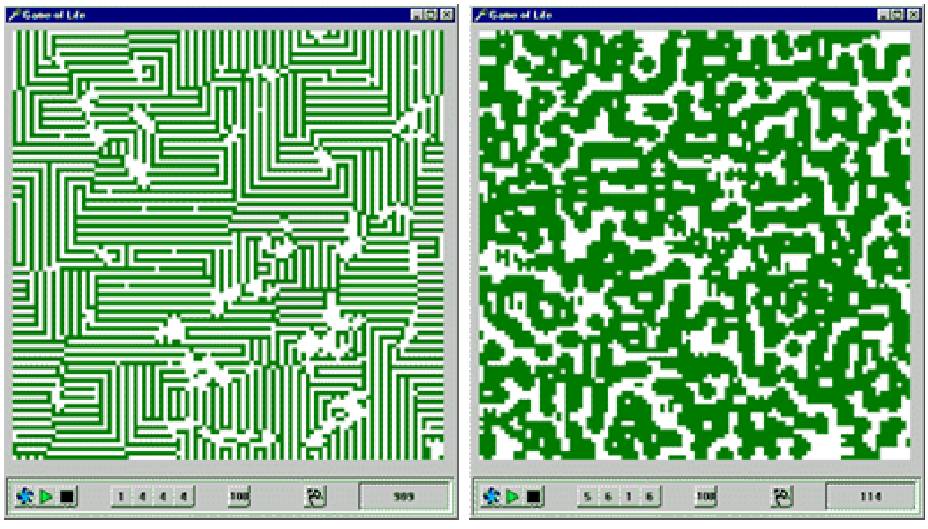


Fig 1.5 Les deux formes du Jeux de la Vie ci-dessus sont « Life 1444 » et « Life 5616 ». Ces formes présentent des oscillations périodiques après un certain nombre de générations.

Après quelques générations, des comportements surprenant apparaissent de la part du système. A partir de ces deux lois simples, certains types de formation de cellules émergent de la structure grouillante du départ (voir fig 1.5).

Cependant, il existe d'autres formes susceptibles d'apparaître dans le jeu de la vie, et à ce jour, aucune étude suffisamment poussée n'est parvenue à établir une liste exhaustive des structures réalisables par ce jeu.

A partir de 1987, Carter Bays, [57, 58, 59, 60] a proposé dans un ensemble d'articles une approche plus formelle du Jeu de la Vie. Chaque type de jeu peut être écrit avec une règle de la forme :

EbEhFbFh où Eb représente le nombre minimum de voisins d'une cellule vivante pour lui garantir sa survie, Fb définit le nombre minimum de voisins nécessaires à une cellule morte pour la faire passer à l'état vivant. Les valeurs Eh et Fh sont les bornes supérieures correspondantes. Ainsi, le Jeu de la Vie standard de J. Conway se note dans ce formalisme : « Life 2333 » (voir Fig1.5)

Les règles de transition d'un état t vers un état t+1 sont alors les suivantes :

Si [(St=0) **et** (Nt>=Fb) **et** (Nt<=Fh)] **ou** [(St=1) **et** (Nt>=Eb) **et** (Nt<=Eh)] **alors** St+1:=1
Sinon St+1:= 0

Où St représente l'état de la cellule à l'instant t et Nt le nombre de cellules voisines vivantes.

Des variations sur les valeurs **EbEhFbFh** aboutissent à des résultats très divers, d'états très stables à des états très chaotiques en passant par un large panel d'états périodiques.

Il est à noter que les automates cellulaires représentent l'outil par excellence pour illustrer le concept des systèmes complexes adaptatifs.

1.5. Les insectes sociaux

Les insectes sociaux comme les fourmis, les termites ou encore les abeilles présentent au niveau collectif des capacités supérieures à la somme des capacités individuelles. Cette performance est le résultat des interactions directes ou indirectes entre les individus. La compréhension de ces phénomènes est indispensable pour la compréhension de la vie artificielle et de la vie biologique. Voyons cela par quelques exemples sur les essaims de particules et les fourmis.

1.5.1. Les essaims de particules

Les algorithmes « d'optimisation par essaim* de particules » PSO trouvent leur origine dans les boids de l'infographiste Craig Reynolds [61, 62]. Analysant les nuées d'oiseaux, Reynolds a proposé de les simuler à partir de trois règles simples d'application locale :

1. Chaque individu doit éviter de heurter ses voisins.
2. Chaque individu tend à s'approcher des vitesses et directions générales du groupe local, c'est-à-dire des voisins immédiats.
3. Chaque individu cherche à s'approcher du centre de gravité du groupe local. Il est possible d'y ajouter une règle de propension à rejoindre un point donné dans l'espace (Perchoir).

L'application de ces règles simples a permis de construire des simulations graphiques d'un réalisme étonnant. Des algorithmes de ce type ont été utilisés dans des films aussi célèbres que « Le retour de Batman » ou « Le Bossu de Notre Dame ».

Les rassemblements animaux présentent divers avantages adaptatifs relevant au moins pour partie de l'échange d'informations. De la même manière que les algorithmes évolutionnaires s'inspirent des mécanismes évolutifs pour mettre en oeuvre des procédures d'optimisation, PSO s'inspire des phénomènes de rassemblement et de nuée, considérant qu'en tant que processus adaptatifs, ils sont potentiellement porteurs de capacités d'optimisation. PSO repose sur deux règles simples :

1. Chaque individu se souvient du meilleur point (le plus proche de l'objectif) par lequel il est passé au cours de ses évolutions et tend à y retourner.
2. Chaque individu est informé du meilleur point connu au sein de la population prise dans son

ensemble et tend à s'y rendre.

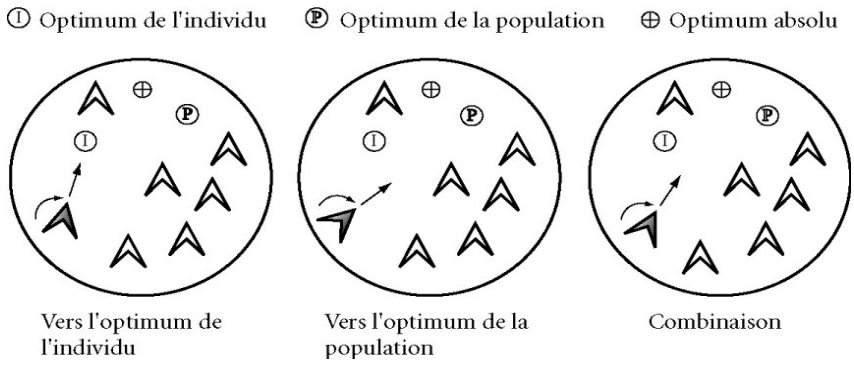


Fig 1.6 Essaim de particules

1.5.2. Les fourmis

Les fourmis sont l'exemple même des animaux très simples ayant un comportement collectif complexe. En effet, chaque fourmi prise individuellement n'est pas un animal extrêmement complexe. Cependant, les fourmilières sont très bien organisées et sont des constructions assez élaborées. En fait, ce sont les phéromones émises par les fourmis qui sont essentiellement à l'origine de ces comportements compliqués.

Chaque fourmi est spécialisée dans une tâche particulière. Ainsi certaine fourmi sont-elles chargées de s'occuper des larves, d'autres de ramener de la nourriture, d'autres encore des matériaux pour la construction ou la réparation de la fourmilière, etc. Cependant, on observe que, si pour une raison quelconque (interne ou externe) il manque des fourmis pour une tâche particulière, d'autres fourmis, « spécialisées » dans d'autres tâches, viendront effectuer ce travail.

En fait, chaque fourmi possède un seuil de phéromones* pour chaque « spécialité » et n'effectuent une tâche donnée qu'à partir du moment où le taux de phéromones correspondant à cette tâche dépasse ce seuil. Par exemple, les larves ont besoin de fourmis pour les nourrir. Quand on ne les nourrit pas, elles émettent une phéromone A. Tant que les fourmis chargées de les nourrir sont en nombre suffisant tout se passe normalement, le taux de phéromones A dans la fourmilière reste constant. Mais si elles viennent à manquer, ce taux augmente et le seuil est dépassé pour d'autres fourmis. Celles-ci vont alors venir prêter main forte à la nourriture des larves.

Pour aller de la fourmilière à un lieu de nourriture, les fourmis utilisent toujours le chemin le plus court. En fait, elles avancent aléatoirement, jusqu'à rencontrer une trace laissée par une autre fourmi de sa fourmilière. Elle suivra alors cette trace (voir Fig 1.7).

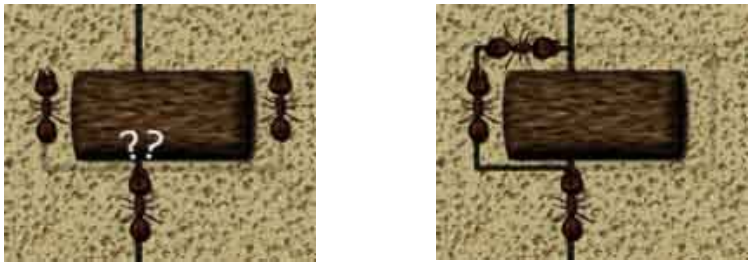


Fig 1.7 Dans cet exemple, la fourmi peut contourner l'objet par la gauche ou par la droite. La « reconnexion » à l'autre partie de la trace se fait aléatoirement. Certaines fourmis iront donc par la gauche d'autres par la droite. Les fourmis qui passeront par la gauche pourront faire plus d'aller-retour que celles passant par la droite. Au fur et à mesure, la piste de gauche va se renforcer, et les fourmis n'emprunteront plus celle de droite qui disparaîtra. De cette manière, on le voit, les fourmis empruntent toujours le chemin le plus court. Seule, aucune fourmi n'est capable de trouver le chemin le plus court. On a donc émergence d'un phénomène grâce à l'interaction des différentes fourmis entre elles. **C'est ce genre de phénomènes qu'il serait intéressant de produire chez une collectivité de robots**.

1.6. Les robots

Les inventions de robots, sont innombrables. Mais c'est essentiellement la seconde guerre mondiale qui a accéléré le phénomène de leur création. En premier lieu, la bombe atomique, pour la fabriquer, les ingénieurs américains ont due manipuler des matières radioactives. Il faut donc le faire à distance. D'où l'idée de bras mécaniques « esclaves » dotés de pinces articulées et commandés par un opérateur humain le «maître » abrité derrière une paroi de verre épaisse.

Dès 1948, le chercheur britannique Walter Grey met en œuvre les idées de Norbert Wiener sous forme de deux tortues à roulettes, « Elmer » et « Elsie ». Elles vivent dans son appartement, se promènent sans tout casser grâce à des capteurs, dont une cellule photoélectrique. Leur tâche essentielle : repérer une prise de courant signalée par une lampe et aller y recharger leurs batteries.

Les ordinateurs sont eux aussi nés de la seconde guerre mondiale, mais pendant 20 ans ils sont restés trop gros, trop chers, et pas assez souples pour être utiles à la robotique. Le premier robot se contentera d'une programmation simple. Il s'agit des robots de première génération. Depuis lors la robotique a beaucoup évolué. Actuellement les roboticiens s'inspirent du vivant. Toute une branche s'est développée s'appuyant sur la construction de robots simples, dotés de quelques capacités élémentaires inspirés du comportement animal. C'est le vaste domaine des animats [66] (qu'on verra un peut plus loin). Pour programmer ces robots on utilise fréquemment des algorithmes évolutionnaires (robotique évolutionnaire).

De même, on s'inspire des travaux sur les insectes sociaux pour construire des systèmes composés d'un ensemble de robots simples dont les interactions permettent de réaliser des tâches plus ou moins complexes. On parvient ainsi par exemple à faire déplacer des objets trop lourds pour un individu seul, ou à amener un groupe de robots à trier des objets. Il est important de comprendre ici, que ces résultats sont obtenus sans coordination centrale et avec des machines dotées de capacités extrêmement simples. On peut espérer ainsi obtenir des systèmes de faible coût et d'une très grande robustesse dans la mesure où la destruction de quelques individus n'empêche nullement la réalisation de la tâche assignée.

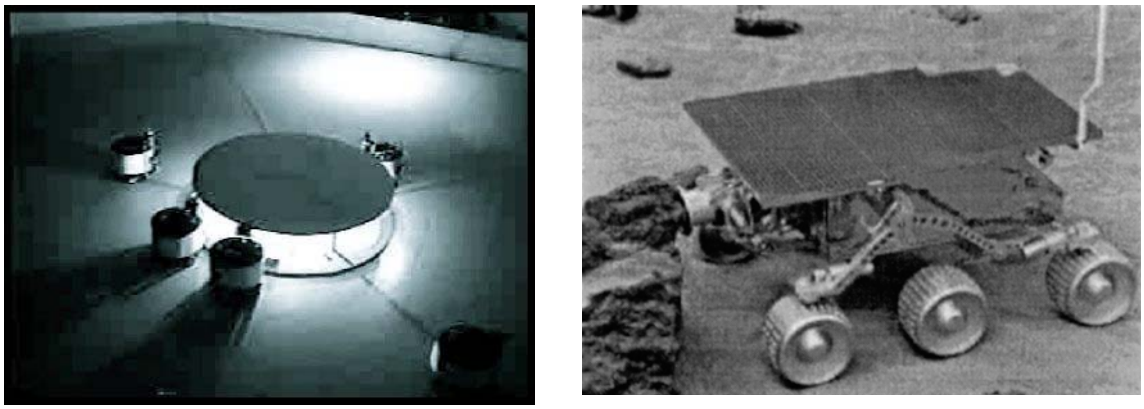


Fig1.8 A gauche les robots pousseurs de KUBE et ZHANG, à droite **«** Sojourner » un robot développé au JPL pour la NASA avec pour objectif l'exploration de la surface de la planète Mars. Ses tâches consistent à explorer les sites, à placer des instruments scientifiques, à rassembler des échantillons pour l'analyse après un éventuel retour sur terre. De telles actions demandent un haut niveau d'autonomie avec possibilité d'une navigation locale, d'identification de sites d'intérêt scientifique et de gestion des ressources propres et tout cela avec un minimum d'interventions télécommandées de la terre.

1.6.1. Définitions

Il existe diverses définitions du terme robot, mais elles tournent en général autour de ce qui suit :

- Un robot est une machine équipée de capacités de perception, de décision et d'action qui lui permettent d'agir de manière autonome dans son environnement en fonction de la perception qu'il en a.

En ce qui concerne la robotique on peut dire que c'est :

- le domaine scientifique et technologique qui étudie les mécanismes, les capteurs, les actionneurs, les méthodes de commande et le traitement de l'information nécessaires à la conception et l'utilisation des robots y compris leurs déplacements.
- C'est aussi le domaine d'application de l'informatique qui concerne l'automatisation de tâches mécaniques impliquant l'adaptation à l'environnement.
- C'est aussi un ensemble de techniques et études tendant à concevoir des systèmes mécaniques, informatiques ou mixtes, capables de se substituer à l'homme dans ses fonctions motrices, sensorielles et intellectuelles.

Il est important de mentionner que la vision de la robotique, donnée par l'Intelligence Artificielle, place au centre des préoccupations l'enchaînement du cycle **Perception / Décision / Action**.

Il est à noter aussi que la manière dont un robot gère ces différents éléments est définie par son architecture de contrôle, qui peut éventuellement faire appel à un modèle interne de l'environnement pour lui permettre alors de planifier ses actions à long terme.

1.6.2. Principe de fonctionnement d'un robot :

Le concept de ROBOT implique la faculté de perception de l'environnement, d'analyse de ses informations, la prise de décision et enfin la correction du comportement en fonction des informations captées sur l'environnement (voir Fig 1.9).

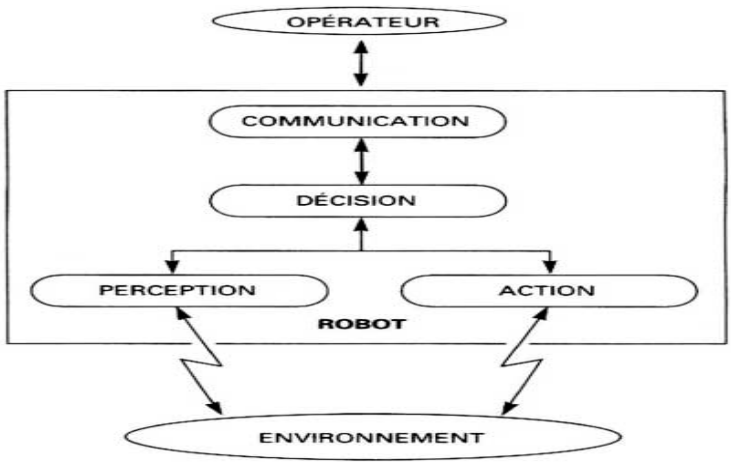


Fig 1.9 Fonctions de base dans le concept de robot

Ainsi, ce qui distingue le robot d'un automate, c'est précisément cette capacité d'appréhender son environnement, et d'en établir une certaine représentation pour en tenir compte dans son comportement. En cela, la perception de l'environnement constitue une fonction indispensable et caractéristique de tout système robotique. On notera que la perception de l'environnement (comme tout système de mesure) peut constituer une source d'interaction supplémentaire avec cet environnement (par exemple : éclairage spécial pour caméra, projection de faisceau par un télémètre laser, échos ultrasonores...).

L'interaction du robot avec l'environnement intervient donc sous deux formes distinctes : au niveau de la perception, puis au niveau de l'action physique exercée sur l'environnement.

La perception de l'environnement peut se définir par l'ensemble des fonctions d'acquisition de mesures et de traitement d'informations, permettant l'analyse et/ou la modélisation de l'environnement du robot, dans le but de supporter la prise de décision et la génération de commandes. Pour percevoir leur environnement, les robots utilisent des capteurs d'environnement encore appelés senseurs [77, 78].

De nos jours, l'utilisation des mécanismes du vivant au sein d'algorithmes ou de robots ouvre une voie originale aux développements technologiques du 21iéme siècle. Il est toutefois nécessaire de tempérer l'enthousiasme que pourraient susciter ces créations. La nature n'est pas avare, elle dispose de ressources quasiment illimitées et ne mesure pas son temps. Plus encore, il est très difficile de contrôler les résultats obtenus. Par exemple, en voulant programmer un robot devant se déplacer le plus loin possible en évitant les obstacles, on a pu obtenir une machine qui se contentait de tourner sur elle-même (ce qui n'était évidemment pas l'objectif), ses roues faisaient ainsi un très grand nombre de tours (ce qui était interprété comme un déplacement long) et le robot ne heurtait jamais d'obstacles.

Les systèmes biomimétiques* permettent de résoudre des problèmes difficiles, les résultats sont généralement d'une robustesse exceptionnelle et les mécanismes utilisés sont d'une souplesse que l'on ne rencontre que très rarement dans les réalisations plus traditionnelles. En ce sens, l'inspiration biomimétique est un outil nouveau, extrêmement puissant, qui vient compléter la palette de ceux dont disposent les chercheurs et les ingénieurs.

1.7. Le système de classifieur

Les systèmes de classifieurs (fondés par Holland) [63], mêlent dans le principe de leur méthode, apprentissage automatique et évolution. Ils sont constitués par un ensemble de règles nommées classifieurs, qui sont apprises ou oubliées suivant leurs performances lors de leur utilisation. Ainsi, une des différences par rapport à un système expert standard est l'importance des règles, celles-ci ne sont ni définies par le programmeur, ni par les experts du domaine. La seconde différence principale avec les systèmes experts est la possibilité de déclencher des règles en parallèle, et non plus obligatoirement de manière séquentielle, c'est à dire les unes après les autres.

Un système de classifieurs est composé de quatre parties :

- Une base de classifieurs représentant les connaissances du système.
- Une interface d'entrée composée de capteurs.
- Une interface de sortie composée d'effecteurs.
- Une liste de messages.

Deux algorithmes principaux gèrent alors les classifieurs :

- Un algorithme de répartition de crédits distribuant des récompenses aux classifieurs efficaces et pénalisant les inefficaces.
- Un algorithme génétique permettant l'évolution de la base de classifieurs.

A l'alphabet standard utilisé, A={0,1}, est ajouté un symbole supplémentaire, le wildcard #, celui-ci signifie soit 0 soit 1. Ainsi, le classifieur 11## : 0110 s'interprète comme « Si les deux premiers bits du classifieur sont à 1 et si les deux derniers bits sont indifférents, alors sélectionner le classifieur pour envoyer le message 0110 ». Le # permet donc de représenter plusieurs classifieurs par un seul classifieur. Si N est le nombre de # dans une chaîne, celle-ci est alors équivalente à 2^N classifieurs. Le classifieur de l'exemple précédent peut donc s'unifier avec quatre messages différents.

Un classifieur possède la forme générale suivante :

<classifieur> = <condition> : <action>
<condition> = <contribution>N
<contribution> = {0,1,#}L
<action> = {0,1}L

L'exemple précédent possède une contribution d'une longueur de 4 bits où L est la longueur d'une contribution et N est le nombre de contributions dans une action. Dans ce cas un système classifieur est défini d'une part par un ensemble de classifieurs, c-à-d un ensemble de règles de la forme : If condition(s) Then action(s), et d'autre part par un aspect de communication schématisé dans Fig 1.10

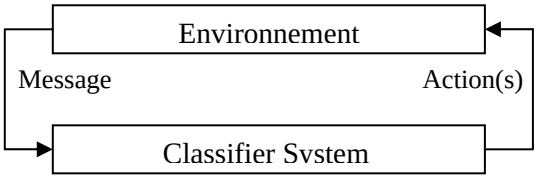
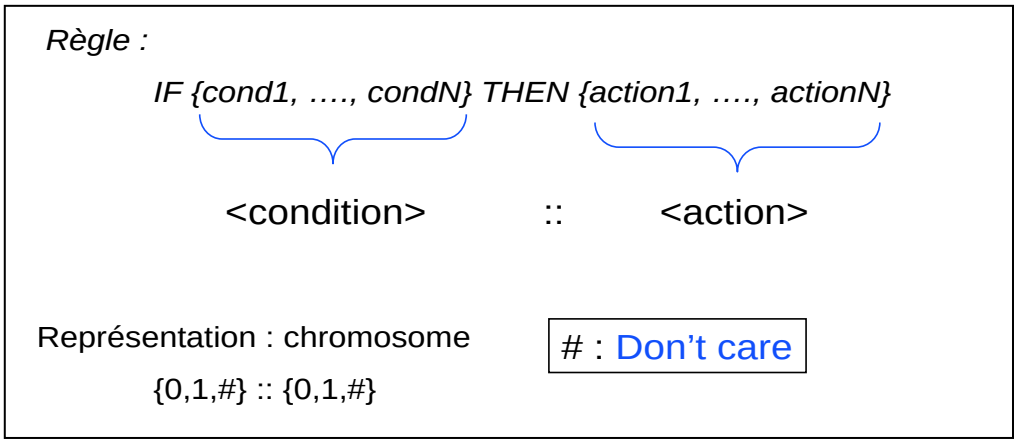


Fig 1.10 Système de classifieur (Aspect communication)

Exemple 1 : L'algorithme de répartition de crédit, bucket brigade algorithm définit par John Holland, utilise la métaphore de l'économie de marché. Le principe est de manipuler un système de vente aux enchères pour récompenser les classifieurs efficaces et écarter peu à peu ceux n'apportant guère de résultats. Cet algorithme est constitué de deux composantes principales : La vente aux enchères et la chambre de compensations.

Lorsqu'un classifieur peut être déclenché par un message, il se fait connaître du système et s'inscrit pour les enchères. Lorsque chaque classifieur a été consulté, la vente aux enchères commence, et tous les classifieurs inscrits proposent une enchère, et une seule, de valeur généralement proportionnelle à sa force. Un classifieur est donc choisi par cette méthode, et peut à son tour produire un nouveau message. Après avoir utilisé le message, le classifieur choisi doit passer à la chambre de compensations, y rétribuant de manière équitable le ou les classifieurs qui lui ont permis de s'activer, en offrant une partie de sa force. Cette récompense peut alors remonter le long d'une chaîne de classifieurs, d'où le nom de l'algorithme traduit par « algorithme des porteurs d'eau ».

Exemple 2 (Détection d'obstacles) **:** Dans le schéma ci-dessous (voir Fig1.11), un robot peut se déplacer selon les 8 directions usuelles. Ceci participe à la constitution de la partie action de nos règles « classifieurs », la partie condition étant constituée par la lecture de l'environnement où évolue le robot. Plusieurs cas se présentent, le robot peut avoir un obstacle à sa gauche, à sa droite, …, comme il peut être entouré complètement d'obstacles. La force des classifieurs réside dans le fait que plusieurs règles peuvent être regroupées sous l'écriture d'une seule règle en usant de « # ». On peut citer comme exemple, le cas où un robot détecte deux obstacles (en face, et à gauche). Plusieurs choix lui sont offerts dans une telle situation pour se déplacer dans les cinq directions restantes. Au lieu de plusieurs règles cette situation peut être représentée par une seule en utilisant le symbole « don't care ».



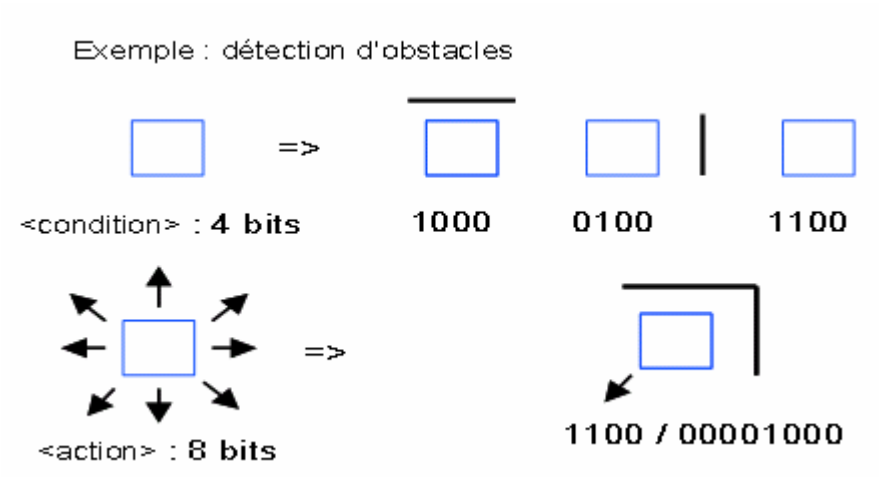


Fig 1.11 Règles permettant le déplacement d'un robot autonome en évitant des obstacles

Classifieur adaptatif : Le principe d'adaptation, sous-jacent aux systèmes de classifieurs, représente la faculté d'un tel système à modifier son comportement et ses connaissances en fonction de l'état de son environnement à un instant particulier (système de classifieurs apprenant : voir fig 1.12). Le problème n'est plus fixe, comme dans le cas d'un algorithme génétique utilisé pour l'optimisation, mais évolutif dans le temps. Cette adaptation s'effectue en fonction d'une situation dynamique dans le temps. Ainsi une action efficace à un instant « t », pourrait se révéler inefficace, voire néfaste, à un instant t + dt.

IF {cond1, cond2} THEN {action1} / Force1
IF {cond1, cond2} THEN {action2} / Force2
…

Rétribution :

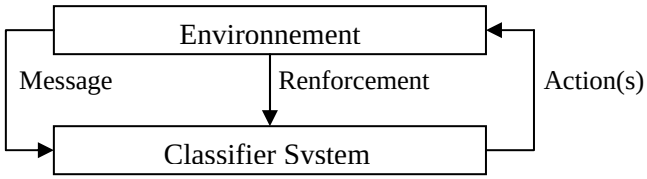


Fig 1.12 Système de classifieur apprenant

Notons que les systèmes de classifieurs sont souvent utilisés pour résoudre des problèmes d'apprentissage par renforcement et d'apprentissage latent (caché). En comparaison avec les algorithmes d'apprentissage par renforcement, les systèmes de classifieurs ont l'avantage de posséder un mécanisme de généralisation.

1.8. Animat

Bien que le principe d'un animal artificiel fut envisagé par L. Booker [66], le terme d'animat fut mentionné pour la première fois par S. Wilson lors de la première conférence internationale sur les algorithmes génétiques en 1985. Il y présenta son animat (animal artificiel), capable de découvrir de la nourriture au milieu d'arbres dans un monde discret de 18*58 cases, à l'image de l'environnement de la figure (Fig 1.13).

Pour définir son modèle, S. Wilson utilisa quatre caractéristiques des animaux :

- L'animal existe dans un océan de signaux sensoriels. A tout moment, seuls certains signaux sont significatifs, les autres sont non pertinents.
- L'animal est capable d'actions (comme le mouvement) qui tendent à modifier ses signaux.

- Certains signaux, comme ceux dépendants de la consommation de nourriture, ou l'absence de certains signaux, comme l'absence de douleur, ont un statut spécial.
- Il agit, aussi bien extérieurement qu'à travers des opérations internes, pour optimiser la quantité des signaux spéciaux.

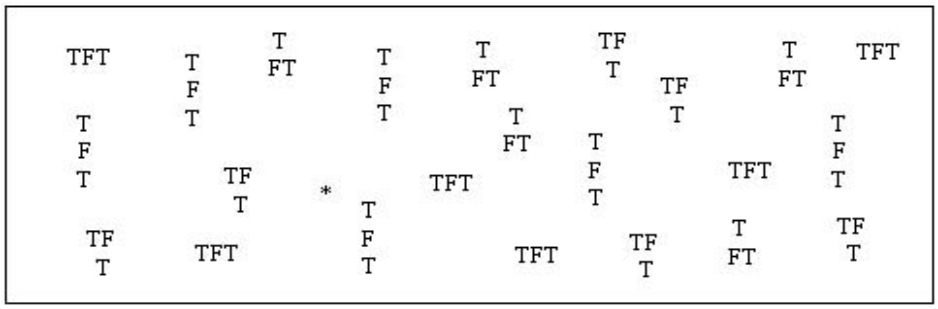


Fig 1.13. Environnement discret de 18x58 cases où **T** représente les arbres, ou obstacles infranchissables, **F** la nourriture et ***** l'animat.

L'animat parcourt son environnement (voir Fig1.13) dans le but de trouver la nourriture le plus rapidement possible en considérant uniquement son voisinage immédiat. Son apprentissage est dit par renforcement dans le sens où il renforce les règles utiles, et incrémental dans le sens où il n'apprend pas toutes ses connaissances en bloc, mais au fur et à mesure de son évolution dans le monde.

Outre l'utilisation des systèmes de classifieurs et des algorithmes génétiques, les animats possèdent un opérateur supplémentaire : L'opérateur d'intersection parallèle. Deux classifieurs disposant de la même partie action sont sélectionnés, deux points sont définis aléatoirement sur les classifieurs, une fonction ET est appliquée sur tous les gènes situés entre ces deux points : s'ils sont égaux alors ils sont conservés, sinon un # est positionné à la place. La chaîne résultante est formée avec les autres gènes du premier classifieur, comme le montre l'exemple de la figure (Fig 1.14) :

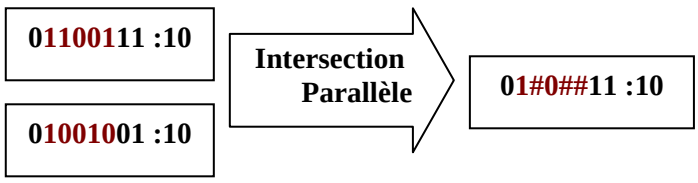


Fig 1.14 Opérateur d'intersection parallèle

Le nombre de déplacements moyens nécessaires à l'animat pour trouver de la nourriture au départ d'une simulation est de 8 à 10, après 2.000 cycles, ce nombre descend entre 4 et 5. Ces données sont à considérer en fonction du minimum mathématique qui est de 2.2 pas dans l'environnement décrit par Wilson.

Depuis les animats de S. Wilson, de nombreux travaux ont été réalisés dans le domaine, avec pour application pratique des projets tels que l'exploration d'environnements partiellement connus comme Mars ou les fonds sous-marins. Les animats doivent alors être capable d'agir et d'interagir avec l'environnement. Ils le modifient et sont à leur tour modifiés par son évolution.

Fig 1.15 Au laboratoire de neurocybernétique à l'E.N.S.E.A, l'équipe travaille avec des robots koala. Ils ont en particulier développé un algorithme de recherche de «nourriture» pour le robot. Dans un premier temps, le robot se déplace aléatoirement : Il cherche à trouver une feuille blanche collée au sol (grâce à un capteur placé sous le robot) symbolisant un lieu de nourriture (qui correspond donc à son but). A ce moment, le robot mémorise grâce à la caméra CDD, l'environnement. Pour cela il prend 5 panoramas successifs en reculant de quelques dizaines de centimètres du but, et à chaque fois en se décalant de 72°. Pour minimiser l'erreur, il recule, prends son panorama, puis retourne au but, pour se recaler, tourne de 70°, recule à nouveau, et réalise ainsi un nouveau cycle.

Le principe d'interaction ou de co-évolution représente l'interdépendance des constituants d'un monde et la capacité des uns à modifier le comportement des autres, et réciproquement. Ce principe met en valeur la théorie des niches écologiques, où l'environnement et les différents types d'agents le peuplant interagissent constamment, à tel point que la disparition d'une espèce peut entraîner la disparition de l'ensemble des populations. Ainsi, ce principe suit bien la notion d'adaptation définie par John Holland dans un de ces premiers articles : «L'étude de l'adaptation implique l'étude d'une part, du système adaptatif, et d'autre part de son environnement. En terme général, c'est une étude de comment les systèmes peuvent générer des procédures leur permettant de s'ajuster efficacement à leur environnement ».

Ce qui a permis à cette approche de prendre son essor est que ces principes adaptatifs ont pu être traduits en algorithmes efficaces grâce au développement du connexionnisme (Rumelhart et McClelland, 1986) et des méthodes évolutionnistes (Holland, 1975 ; Koza, 1992). En effet, les apprentissages que peuvent réaliser des réseaux de neurones à couches cachées (Rumelhart et McClelland, 1986) ou associatifs (Hopfield, 1982 ; Kohonen, 1984) sont plus performants que ceux de réseaux de neurones plus simples, comme le Perceptron à deux couches conçu par Rosenblatt en 1958.

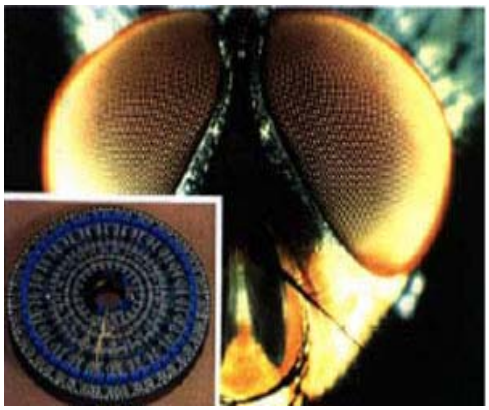


Fig 1.16 Exemple de réalisation de l'approche animat : Dans la conception des animats, les capteurs d'invention purement humaine, tels les télémètres lasers, ont peu à peu été remplacés par des dispositifs inspirés des organes sensoriels des animaux. Par exemple, au sein du Laboratoire mouvement et perception du CNRS, à Marseille, Nicolas Franceschini équipe des robots volants de systèmes visuels dérivés de l'œil à facettes de la mouche. Ces robots utilisent le flux visuel, c'est-à-dire la vitesse de défilement de l'image des obstacles sur la rétine, pour éviter les obstacles et atterrir.

En particulier, ils permettent au système artificiel de ne pas toujours avoir besoin de consignes humaines précises sur ce qu'il faut faire ou ne pas faire pour acquérir ou organiser des connaissances. L'apprentissage par renforcement, par exemple (voir chapitre II), retient les comportements qui ont 1. C'est pourquoi l'approche animat est parfois nommée « intelligence artificielle située » bien que, dans ce domaine, les chercheurs préfèrent parler de capacités adaptatives et non d'intelligence.

1.9. La Morphogenèse

La diversité des formes engendrées par la nature a toujours fasciné les hommes. Mais aussi importante soit-elle, cette diversité ne comprend qu'un nombre fini de formes, finalement assez limité. Toutes les formes possibles n'existent pas, les formes existantes sont le résultat de l'évolution et de la sélection naturelle.

Le processus par lequel un être vivant modèle se forme et acquiert ses caractéristiques, autrement dit développe son phénotype à partir de son génotype, s'appelle la morphogénèse. Elle constitue l'un des grands phénomènes du développement embryonnaire, qui s'effectue par des mouvements cellulaires, des multiplications cellulaires différentielles selon les régions et des morts cellulaires « programmés ».

Ces mécanismes résultent d'un jeu complexe d'interactions physico-chimiques qui ont pour médiateurs les protéines produites par les gènes. Ces derniers influent sur le développement de deux façons : soit directement à l'intérieur d'une cellule par régulation de l'activité d'autres gènes, soit à l'extérieur de la cellule par induction. Le processus de segmentation des insectes est l'un des exemples les plus connus de régulation du développement par les gènes [79, 80, 81].

Malgré son élégance, un modèle de développement limité à une suite d'activations et d'inhibitions qui est trop simple pour rendre compte des observations. Depuis la découverte de l'ADN, la tendance dominante au sein de la communauté scientifique fut de considérer le génome comme le seul élément déterminant la morphologie des individus.

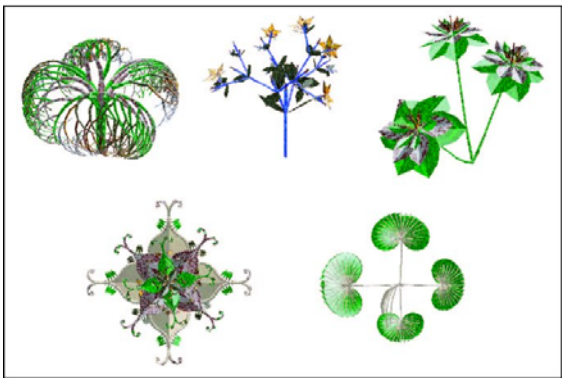
De nombreuses recherches ont été effectuées sur le thème de la reproduction des formes ou bien quelles procédures donnent des formes intéressantes ?

Le premier exemple de processus de développement artificiel est le célèbre programme « Blind Watchmmaker » de Richard Dawkins, chercheur au département de zoologie de l'université d'Oxford en Angleterre. Son programme crée un monde peuplé d'organismes à deux dimensions dont les formes sont surprenantes par leurs ressemblances avec certaines espèces vivantes. Ces êtres artificiels sont connus sous le nom de biomorphs. Dans son programme Dawkins a utilisé deux procédures, la première étant la reproduction qui, à partir d'un génotype parent produit un génotype* enfant, la seconde étant le développement qui, à partir d'un génotype, développe le phonotype*. La procédure reproduction est simple et bien connue, alors que la procédure développement requiert beaucoup plus d'attention. La procédure développement la plus évidente consisterait à associer un pixel à chaque gène. Autrement dit, chaque point image de l'écran correspondrait à un gène. Afin d'échapper aux problèmes de codage, Dawkins a remplacé les points par des lignes et il eut l'idée de coder la procédure de développement sous la forme d'une fonction récursive. Ainsi, la procédure de base conçue par Richard Dawkins est une simple fonction récursive, baptisée Tree.

Un autre exemple de recherche sont les L-systèmes [83] qui reposent sur une application récursive de règle de réécriture. Leur formalisme fut initialement proposé par Aristide Lindenmayer et Przemyslaw Prusinkiewicz pour résoudre les limitations des grammaires de Chomsky et des automates cellulaires, dans le cadre d'une approche algorithmique de description, d'analyse et de simulation des organismes multicellulaires. Ils ont montré en particulier, que certains processus dynamiques tels que la croissance des plantes pouvaient résulter du développement de quelques règles. Les développements des L-systèmes dessinent des courbes dont la croissance et la forme rappellent celles des plantes (voir Fig 1.17)

Les organismes vivants évoluent de génération en génération, les espèces s'adaptent aux modifications de l'environnement. Les premiers êtres vivants étaient des organismes très simples, constitués d'une cellule unique. Aujourd'hui, notre planète accueille un nombre considérable d'espèces végétales et animales, dont certaines présentent un très haut niveau d'organisation. En un certain sens, la terre peut, elle-même, être considérée comme un organisme vivant.

La théorie de l'évolution par sélection naturelle explique l'une des caractéristiques parmi les plus surprenantes du vivant : les organismes n'ont pas été programmés ou construits pour répondre à un problème spécifique. Ils évoluent et s'adaptent selon les nécessités que l'environnement leur impose. Ne pourrait-on concevoir des programmes d'ordinateurs conçus sur ce principe ? Les travaux récents sur les algorithmes génétiques (qu'on verra plus loin) nous permettent de répondre affirmativement à cette question.



Les processus de la morphogenèse restent mystérieux. Malgré les progrès de la génétique, de nombreux mécanismes nous restent inconnus. A. Lindenmayer (1925-1989) a proposé une méthode de description formelle de la structuration des plantes. Basée sur une forme récursive de grammaire générative, elle a été approfondie et mise en oeuvre graphiquement par P. Prunsinkiewicz dans les années quatre-vingt. Les procédures utilisées sont simples, mais nous ne les décrirons pas ici. Contentons nous de présenter quelques-unes des images obtenues

Fig 1.17 L'inspiration biomimétique des figures ci-dessus est indéniable. Elles ne sont pourtant que l'interprétation graphique de chaînes de caractères construites à partir d'un algorithme récursif.

Il est très difficile de cerner le lien entre la morphogenèse végétale et les L-systèmes. De même que les paysages, engendrés par les fractales, malgré leur ressemblance avec la réalité, n'ont, en aucune manière, suivi une évolution comparable aux paysages naturels, les similitudes entre plantes et systèmes ne signifient nullement que les génomes végétaux décrivent un mécanisme identique. Les L-systèmes n'en confortent pas moins l'hypothèse selon laquelle les processus morphogénétiques mettent en oeuvre des procédures répétitives, probablement souvent récursives.

1.10. Principe de l'évolution

Le principe de base de l'évolution est simple : il s'inspire de la théorie de Darwin sur l'évolution des espèces qui explique comment depuis l'apparition de la vie, les espèces ont su évoluer de façon innovante et souple, dans le sens d'une meilleure adaptation à l'environnement et qui éclaire ce chemin vertigineux qui, commencé par d'ambitieux organismes unicellulaires, a su nous mener à la vie telle qu'on la connaît aujourd'hui dans toute sa diversité exotique et spectaculaire. En permettant aux seuls individus bien adaptés à l'environnement de se reproduire, la nature assure la pérennité de leurs meilleures caractéristiques, qui se recombinent entre elles (chaque enfant reçoit de bonnes caractéristiques à la fois de son père et de sa mère) pour former à chaque génération de nouveaux individus toujours mieux adaptés à leur environnement.

1.10.1. L'évolution des espèces

Considérons un environnement quelconque dans lequel vit une population primitive : peu adaptée à cet environnement. Bien sûr, quoique globalement inadaptée, cette population n'est pas uniforme : certains individus sont mieux armés que d'autres pour profiter des ressources offertes par l'environnement (nourritures, abris, etc.) et pour faire face aux dangers qui y rôdent (prédateurs, intempéries, etc.).

Ces individus mieux équipés ont par conséquent une probabilité de survie plus grande que leurs congénères et auront de ce fait d'autant plus de chances de pouvoir se reproduire. En se reproduisant entre individus bien adaptés, ils vont transmettre à leurs enfants ces caractéristiques qui faisaient leur excellence.

La population qui résultera de cette reproduction sera donc globalement mieux adaptée à l'environnement que la précédente puisque la plupart des individus auront hérité de plusieurs (puisque chacun hérite à la fois de sa mère et de son père) des caractéristiques de l' « élite » de la génération précédente, et c'est ainsi, en recombinant à chaque génération les caractéristiques

élémentaires de bonne adaptation et en saupoudrant le tout d'un peu de hasard, que la population va évoluer vers une adéquation toujours meilleure avec l'environnement [43].

1.10.2. Le codage de l'information

Le principe est très convaincant mais une question s'impose : comment cette transmission d'informations est-elle possible ? L'explication viendra plus tard avec la découverte de la génétique (Mendel 1822-1884) et de l'ADN (Griffith 1928, Avery, McLeod et McCarthy 1944). On se rend compte que toutes les informations nécessaires à la genèse d'un individu (son patrimoine génétique, tout ce qui dicte sa construction et qui décrit toutes ses caractéristiques) sont contenues dans les molécules d'ADN dont chaque cellule de l'organisme possède une copie. Le processus de reproduction fait rencontrer deux cellules sexuelles (une du père, et une de la mère) qui voient leurs molécules d'ADN s'enchevêtrer, se croiser, ce qui donnera une cellule fille (futur individu) dont le patrimoine génétique sera un mélange des patrimoines des parents.

1.10.3. Applications aux algorithmes génétiques

Les algorithmes génétiques vont s'inspirer directement de ces principes pour résoudre une grande variété de problèmes d'optimisation. Face à un problème pour lequel il existe pour ainsi dire une infinité de solutions, plutôt que d'essayer naïvement toutes les solutions une à une pour trouver la meilleure, on va explorer l'espace des solutions en se laissant guider par les principes décrits plus haut. Plus concrètement :

Il faut d'abord définir un codage : établir une convention qui permet de décrire chaque solution possible sous la forme d'une chaîne de caractères (l'analogue de la molécule d'ADN). Ensuite, il faut définir précisément une fonction d'adaptation qui pour chaque solution possible donnera une valeur reflétant sa qualité pour résoudre le problème posé : plus la valeur de la fonction est élevée, meilleure est la solution.

Ceci fait, on se dote d'un ensemble de solutions choisies au hasard : c'est la population de départ. A l'aide de la fonction d'adaptation, on évalue chacune d'entre elles. On s'attelle maintenant à la construction de la génération suivante. Pour commencer à construire cette nouvelle génération, il nous faut deux parents : on les prend parmi la population de départ en les choisissant avec une probabilité d'autant plus grande que leur adaptation était bonne : c'est la phase de sélection. Partant de ces deux parents, on construit une descendance par croisement : on coupe les chaînes de caractères qui décrivent les parents en plusieurs morceaux dont la taille est tirée au hasard. On construit alors toutes les recombinaisons possibles de ces morceaux : l'enfant sera construit morceau par morceau : le 1er morceau du fils sera soit le 1er morceau de la mère, soit le 1er morceau du père, son 2ème morceau, soit le 2ème morceau de la mère, soit le 2ème morceau du père, etc. On construit ainsi autant d'enfants que de recombinaisons possibles [82].

Exemple : avec deux morceaux :
Père : **1111000**
Mère : **1001110**
Fils 1 : **1111110**
Fils 2 : **1001000**

On fait reproduire autant de couples de parents que nécessaire pour régénérer toute la population. Pour terminer, on procède au hasard à quelques mutations : chaque caractère de chaque chaîne a une faible probabilité de voir sa valeur changée complètement au hasard. Cette procédure a pour objet de dynamiser l'exploration de l'espace des solutions et surtout d'éviter aux nouvelles générations de rester bloquées dans un minimum local de la fonction d'adaptation dont on ne pourrait pas sortir en ne faisant que recombiner les caractéristiques des générations précédentes.

On a bien construit une nouvelle génération qui a hérité des meilleures caractéristiques de la génération précédente (puisque les individus possédant ces caractéristiques se sont plus probablement reproduits que les autres) et qui les a recombiné pour proposer des solutions originales et probablement plus efficaces au problème. Il ne reste plus qu'à recommencer jusqu'à ce qu'on obtienne des solutions satisfaisantes au problème.

1.11. Emergence

L'évolution rapide de l'informatique aussi bien matérielle que logicielle nécessite des approches nouvelles pour la conception des systèmes informatiques notamment distribués. Les futurs systèmes qui en résultent doivent permettre de prendre en considération de manière plus sérieuse la complexité, une robustesse plus grande et une autonomie plus accentuée. Les techniques informatiques actuelles sont incapables de répondre à de tels besoins. En prenant comme source d'inspiration les systèmes naturels, l'émergence s'offre comme nouvelle voie à explorer pour la conception de systèmes complexes. Cependant et malgré les ambitions alléchantes que l'émergence laisse entendre et les nombreuses disciplines qui s'y intéressent, elle reste difficile à appréhender. Donc l'un des objectifs actuels est de bien comprendre ce phénomène. Ce qui reste une tâche ardue.

1.11.1. Définition

- Sur le Web la définition de l'émergence est assez générique :
  - ✓ L'émergence est un phénomène qu'on trouve dans les domaines physiques, biologiques, écologiques et socio-économiques et tout autre système dynamique.
  - ✓ L'émergence désigne un phénomène particulier d'apparition de nouvelles propriétés au sein d'un système. Une propriété est dite « émergente » lorsqu'elle n'existe que dans la totalité du système fonctionnant et qu'elle ne peut être réduite aux éléments qui composent ce système, ni avoir d'existence en dehors d'eux.
- Marvin Minsky un des pionniers de l'Intelligence Artificielle Distribuée propose la définition suivante : Apparition inattendue, à partir d'un système complexe, d'un phénomène qui n'avait pas semblé inhérent aux différentes parties de ce système. Ces phénomènes émergents ou collectifs montrent qu'un tout peut être supérieur à la somme de ses parties.
- Selon l'association « Emergence » : L'émergence consiste dans le fait que d'une organisation, qu'elle quelle soit, à un moment donné apparaissent des propriétés et des qualités qui n'existent pas dans les éléments qui la constituent. Par exemple : l'organisation de l'homme est faite d'éléments uniquement physico-chimiques qui existent dans la nature, mais leur association et l'organisation même de l'homme produisent les capacités de se réparer, de se mouvoir en même temps que des qualités cognitives. Ainsi, Mind signifie l'émergence au cerveau humain de quelque chose qui est en connexion avec la culture. Quand une émergence apparaît, elle semble simple mais elle est produite par quelque chose de très complexe qui la compose.

Le concept de l'émergence dans ce cas est important pour décrire comment un phénomène non-compositionnel au niveau macro repose sur une structure d'interactions à un niveau micro. L'avantage est la possibilité pour le micro-niveau de s'auto-organiser fournissant ainsi à la fois flexibilité et robustesse malgré la possible simplicité des composants au niveau micro.

On peut dire alors qu'un phénomène est émergent si :

- il y a un ensemble d'entités en interaction dont la dynamique est exprimée dans un vocabulaire ou une théorie distincte du phénomène émergent à produire,
- la dynamique de ces entités interagissantes produit un phénomène global qui peut être un état structuré stable ou même la trace d'exécution,
- ce phénomène global peut être observé et décrit dans un vocabulaire ou une théorie distincte de la dynamique sous-jacente.

Il faut faire la distinction entre l'émergence comme explication (phénomène statique, par exemple : une force comme échange particulaire), et l'émergence comme surgissement (phénomène dynamique, par exemple : émergence de la vie). En effet l'émergence comme explication est un moyen de voir les choses. Selon l'utilisateur, l'interprétation et donc l'émergence peuvent avoir différentes formes.

Les principales propriétés de l'émergence sont [69] :

- la nouveauté : un phénomène émergent est en quelque sorte *unique* ;
- l'imprévisibilité : on ne peut prédire ni quand ni quel phénomène aura lieu ;
- la temporalité : le phénomène doit se construire dans le temps, il ne peut être instantané ;
- la stabilité : le phénomène doit laisser le système dans lequel il se trouve stable ;
- l'irréductibilité : on ne peut pas réduire le phénomène à ce que l'on observe, il a une cause, un but, etc.;
- l'absence d'intentionnalité : le système ne doit pas être définit pour observer ce phénomène.

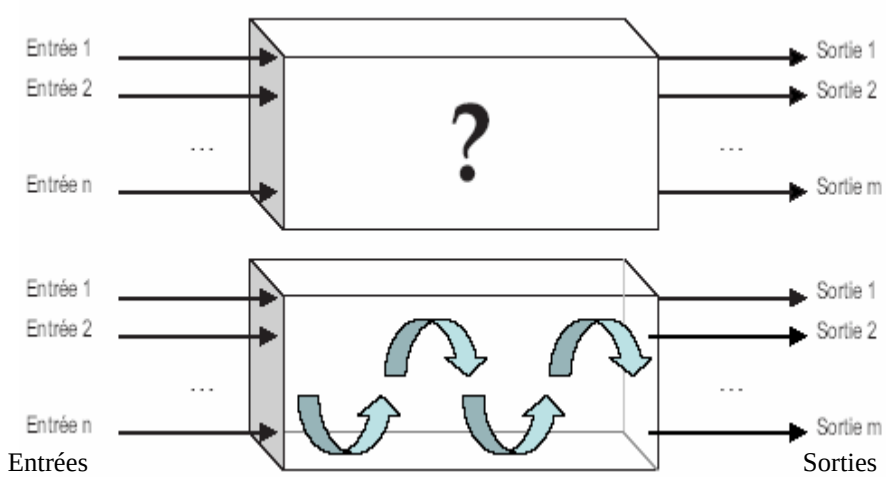
L'intérêt croissant pour comprendre et maîtriser les phénomènes émergents trouve sa justification dans le fait que nous construisons des systèmes artificiels de plus en plus complexes et leur réalisation demande de plus en plus de moyens. Souvent, le résultat final est seulement approximatif quant au fonctionnement souhaité et une attention humaine continue est nécessaire pour réparer les erreurs et développer le système. De plus, de nombreux domaines sont, de par leur nature même, difficiles à traiter : problèmes fortement dynamiques, répartis, nombreuses entités en interaction. Notons par la même occasion que IBM a récemment décidé de s'investir massivement dans un axe de recherche qu'ils appellent «autonomic computing» pour répondre à ces problèmes.

L'origine de l'émergence pourrait bien être le postulat datant de la Grèce antique : «le tout est plus que la somme de ses parties ». On retrouve des traces du concept d'émergence et d'auto-organisation dans des écrits de Thalès et Anaximandre. On se sert alors de cette notion pour expliquer certains phénomènes que l'on n'arrive pas à décomposer.

1.11.2. Le proto-émergentisme : la boîte noire

Cependant, il faut attendre le milieu du XIXe siècle pour voir apparaître un mouvement de pensée autour du concept de l'émergence. Ce mouvement sera plus tard appelé proto-émergentisme et se poursuivra jusqu'à dans les années 30. Ses débuts sont surtout caractérisés par la distinction que fait G. H. Lewes entre résultant et émergent. Ce philosophe anglais dont les travaux sont basés sur ceux de J. S. Mill, explique en 1875 que pour le résultant la séquence d'étapes qui produisent un phénomène est traçable, alors que pour l'émergent nous ne pouvons pas tracer les étapes du processus. Ainsi, nous ne pouvons pas voir dans le produit le mode d'opération de chaque facteur. L'émergence ne peut pas être réduite ni à la somme ni à la différence des forces co-opérantes. Les proto-émergentistes [42] cherchaient surtout à définir l'émergence afin de reconnaître un phénomène émergent et le différencier de phénomènes explicables grâce à d'autres théories ou modèles. Ainsi, le processus d'émergence est vu comme une boîte noire (voir Fig 1.18). On ne peut discerner que les entrées de plus bas niveaux et les sorties de plus hauts niveaux. On ne sait pas comment les entrées sont transformées et reliées aux sorties.

1.11.3. Le néo-émergentisme : l'exploration



Mais ce n'est que dans la deuxième moitié du XXe siècle que la science s'est dotée de moyens permettant d'explorer cette boîte noire. On peut appeler la recherche récente sur l'émergence le néo-émergentisme. Elle est liée à la théorie de la complexité actuelle et prend ses racines dans diverses approches comme la dynamique des systèmes en physique, en mathématiques et en informatique.

Fig 1.18 (En haut) Proto-émergentisme: l'émergence est considérée comme une boîte noire.
(En bas) Néo-émergentisme : nous essayons de comprendre et de manipuler l'émergence.

Nous pouvons maintenant ouvrir la boîte noire grâce aux découvertes de constructions mathématiques pertinentes et de nouvelles méthodes de recherche ainsi que grâce à la puissance des ordinateurs actuels (voir Fig1.18).

1.11.4. Identification d'un phénomène émergent

Mais qu'est-ce qui nous permet de juger du caractère émergent d'un phénomène ?
En creusant dans la littérature nous pouvons synthétiser les propriétés inter-reliées, communes qui permettent d'identifier le phénomène comme émergent :

- l'observation d'un phénomène ostensible (qui s'impose à l'observateur) au niveau global ou macro,
- la nouveauté radicale du phénomène (il n'est pas observé au niveau micro et n'est pas prévisible),
- la cohérence et la corrélation du phénomène (il a une identité propre mais liée fortement aux parties qui le produisent),
- l'observation d'une dynamique particulière (le phénomène n'est pas prédonné, il y a « auto-maintien » du phénomène).

1.11.5. Caractéristiques de l'émergence

Plutôt que de prétendre donner une définition exacte et exhaustive de l'émergence (chose qu'on est incapable de faire actuellement), nous allons plutôt donner exhaustivement les particularités de l'émergence (citées ci-avant) [71].

a. Nouveauté

- D'une part, l'émergence présuppose qu'il y a apparition de nouveauté : propriétés, structures, formes ou fonctions, et d'autre part, elle implique qu'il est impossible de décrire, d'expliquer ou de prédire ces nouveaux phénomènes en termes physiques à partir des conditions de base définies aux niveaux inférieurs.

- Théorie selon laquelle la combinaison d'unités d'un certain ordre réalise une entité d'ordre supérieur dont, les propriétés sont entièrement nouvelles.

b. Auto-organisation

L'émergence fait référence à l'apparition durant le processus d'auto-organisation dans un système complexe de structures ou de schémas (patterns) ou de propriétés nouvelles et cohérentes.

c. Irréductibilité

Churchland définit l'émergence en termes d'irréductibilité des propriétés associées à une théorie de haut niveau à des propriétés associées à des composants dans une théorie de plus bas niveau.

d. Interdépendance des niveaux

Des structures et/ou des fonctions apparaissent à un niveau macroscopique sans que la seule observation des propriétés des constituants permette de les prédire.

Langton (1989) définit l'émergence en terme de relation de feedback entre les niveaux dans un système dynamique. Les micro-dynamiques locales causent les macro-dynamiques et les macro-dynamiques globales contraignent les locales.

Comme exemples de phénomènes qui sont admis pour être émergents on peut citer: les constructions et comportements des insectes sociaux (fourmis, termites, ...), les phénomènes sociaux (embouteillages, applaudissements, ...), les phénomènes économiques, la circulation de l'information sur Internet, les automates cellulaires (comme le jeu de la Vie), ...

1.11.6. Exemple d'émergence

Systèmes physiques : le système de Bénard [41]

Dans les systèmes complexes physiques, des phénomènes émergents apparaissent typiquement dans des états de non-équilibre proche du chaos. En effet, l'équilibre pour un système empêche l'auto-organisation et le garde stérile alors que le non-équilibre permet l'apparition d'organisations nouvelles et de créativité. Ceci peut être observé dans des situations tout à fait courantes comme lorsque l'on fait chauffer de l'eau dans une casserole. La chaleur est transmise par un flux tout d'abord régulier du fond vers la surface par conduction. Lorsque l'on continue à chauffer, on quitte cet état d'équilibre pour un état loin de l'équilibre dû à la différence de température entre les deux régions qui fait que la gravité s'exerce plus intensément sur les couches supérieures, plus froides et donc plus denses. Ceci conduit à l'apparition progressive de circulations chaotiques du fluide (remous et tourbillons) qui s'intensifient jusqu'à l'état d'ébullition.

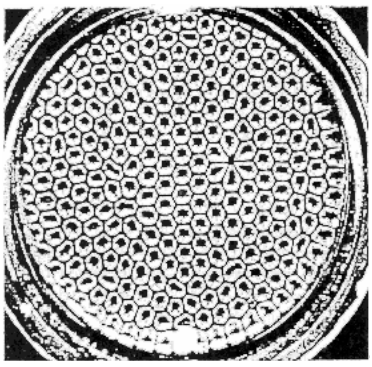


Fig 1.19 Réseau de cellules de Bénard (vu de dessus)

Mais avant l'ébullition proprement parlée, il y a un point critique où il n'y a pas encore de grands courants de circulation pour permettre la dispersion rapide de la chaleur au sein du fluide. On observe alors une auto-organisation du système qui quitte son état chaotique et produit un réseau de courants hexagonaux appelés cellules de Bénard (Voir Fig 1.19). Si la température augmente, ce phénomène disparaît. On peut parler d'une certaine forme d'émergence car des millions de molécules, subissant une contrainte de l'environnement, s'auto-organisent pour former une structure observable cohérente.

1.12. Relation entre Intelligence artificielle et Vie artificielle

Le paradigme cybernétique* a vu sa concrétisation technologique à travers deux disciplines dont les enjeux et les concepts de départ diffèrent sensiblement. Il s'agit de l'Intelligence Artificielle (IA) et de la Vie Artificielle (VA). L'IA, la plus ancienne de ces disciplines, est contemporaine des premiers développements de la cybernétique. Elle s'intéresse à la conception de machines pouvant simuler la cognition humaine. La VA, plus récente, diverge de l'IA sur certains points et la rejoint sur d'autres. Son domaine d'étude est plus vaste. Elle explore les caractéristiques du vivant en général. Afin de disposer de points de repère, il est question de poser, ici, les principales étapes de développement de ces concepts. Nous, nous intéresserons également à leurs applications technologiques.

L'intelligence Artificielle est directement issue des concepts cognitivistes. C'est-à-dire qu'elle envisage le fonctionnement cérébral sous un angle logico-déductif, et considère, de fait, que l'acte cognitif s'effectue à travers une manipulation de symboles élémentaires. A travers le cognitivisme, l'IA établit une analogie fondamentale entre le fonctionnement cérébral et celui de l'ordinateur. C'est pourquoi certains auteurs préfèrent le terme de computationnalisme pour désigner cette orientation théorique des sciences cognitives dont l'IA est le développement technologique le plus évident.

Initialement, les ambitions de l'IA étaient particulièrement élevées [40]. On espérait pouvoir créer des automates capables de traiter des problèmes très généraux comme la reconnaissance du langage, la traduction des langues naturelles ou la résolution universelle de problèmes. Il était donc question de modéliser le savoir-faire de l'expert. Mais, comme le souligne F. J. Varela : « au fur et à mesure que ces tentatives devenaient plus modestes, il devenait plus clair que l'intelligence la plus profonde et la plus fondamentale est celle du bébé qui acquiert le langage... » . Contrairement à ce qu'on pourrait croire, la VA ne relève pas d'ambitions aussi généralistes que l'IA à ses débuts et évite ainsi l'écueil de l'universalité. Il est vrai que la VA est une discipline plus jeune et qu'elle bénéficie, par conséquent, de l'expérience de l'IA. Le programme actuel de la VA semble assez bien défini et dispose de bases théoriques et conceptuelles plus diversifiées. La Vie Artificielle exploite les concepts issus des courants cognitivistes et connexionnistes qui partagent les sciences cognitives. Elle propose donc des solutions qui s'appuient tantôt sur l'approche symbolique, tantôt sur l'approche auto-organisationnelle, ou construit ses modèles à partir d'une combinaison des deux méthodologies.

1.13. Conclusion

Depuis l'antiquité, les hommes n'ont cessé de vouloir reproduire artificiellement la vie. Platon (427-347 avant J-C.) décrivait des statues vivantes qu'il fallait entraver pour qu'elles ne s'enfuient pas. Dédale, grand sculpteur crétois du VIIe siècle avant notre ère, créa des statues qui, selon la rumeur, pouvaient se déplacer seules comme des êtres vivants. Mais ce phénomène n'est pas spécifique à l'antiquité. De tout temps et en tout lieu, l'homme a eu le désir de créer des êtres artificiels.

L'avènement de l'ordinateur est un phénomène beaucoup plus récent dans notre histoire, mais qui marque une étape décisive par ses capacités universelles de simulation. Celles-ci, en effet, permettent d'envisager ce qui autrefois ne relevait que de la pure spéculation. Mais une question intrigante demeure toujours sans réponse : *Un ordinateur pourra-t-il un jour se programmer lui-même?* (Tous les informaticiens se sont posés au moins une fois cette question).

Dans ce premier chapitre, on a tenu à faire un tour d'horizon de la vie artificielle (des automates cellulaires en passant par la robotique, et la morphogenèse jusqu'à l'émergence), afin d'avoir une vue globale sur le domaine d'appartenance de l'intelligence collective (représentant dans ce projet notre axe d'intérêt). La vie artificielle en tant que science de l'avenir s'est avérée aussi vaste et aussi diversifiée que l'est son inspiratrice : la vie. Des problèmes comme ceux ayant une relation avec

l'émergence, gardent toujours aussi bien leurs secrets. N'empêche que les résultats très prometteurs obtenus dans certains cas, comme ceux concernant l'application des approches évolutionnistes pour des problèmes d'optimisation, laissent croire que l'avenir de la science est dans cette direction. L'école probabilistique semble toute prête à prendre sa revanche sur l'école déterministe.

Dans le chapitre 2, nous allons plutôt focaliser sur des concepts fondamentaux à savoir l'auto-organisation, l'adaptation et l'apprentissage, indispensables dans toute étude de la dynamique de systèmes complexes en relation avec la vie artificielle,

1.14. Références

Références bibliographiques

[1] J.C. Heudin, « La Vie Artificielle », Hermès, Paris, 1994.

[2] J.C. Heudin, , « L'évolution au bord du chaos », Editions Hermès Sciences, Paris, 1998.

[3] J.C. Heudin, , « Virtual Worlds - Synthetic Universes, Digital Life, and complexity, Perseus ». Books, Reading (USA), 1998.

[4] S. Forrest, « Emergent computation : self-organizing, collective, and cooperative phenomena in natural and artificial computing networks »,Proceedings of the ninth annual CLNS conference, 1990.

[5] J-M. Alliot, Nicolas Durand, « Algorithmes génétiques », 2005.

[6] G. Van de Vijyer, « Emergence et explication », Intellectica : Emergence and explanation, 1997/2 no25, ISSN no0984-0028 185-194, 1997.

[7] J.von Neumann, « Theory of Self-Reproducing Automata », (édité et complété par A.W.Burks), University of Illinois Press,1966.

[8] S. Wolfram, « Universality and Complexity in Cellular Automata », Physica D n°10, Janvier 1984.

[9] G.J. Chaitin, « To a mathematical theory of life », ACM SICACT News 4, jun-1970.

[10] G.J. Chaitin, « Toward a mathematical theory of life », in the volume, The Maximum Entropy Formalism , R.D.Levine and M. Tribus (eds), MIT Press, 1979.

[11] S. Wolfram, «New Kind of Science », Wolfram Media 2002.

[12] N. Bohr, R. Dubas, « The mysteries of life », The New Encyclopedia Britannica,15-th edition, vol.Propaedia, p.92, 1994.

[13] C. G. Langton, (Ed.) « Artificial Life ». Reading, MA: Addison-Wesley, 1989.

[14] C. G. Langton, , Taylor, C., Farmer, J. D., & Rasmussen, S. « Artificial Life II ». Reading, MA : Addison-Wesley,1992.

[15] M. Mitchell, , J.P. Crutchfield, , R. Das, « Evolving Cellular Automata with Genetic Algorithms », A Review of Recent Work In Proc. of the First International Conference on Evolutionary Computation and Its Applications (EvCA'96). Moscow, Russia: Russian Academy of Sciences, 1996.

[16] J.H. Holland, K.J. Holyoak, R.E. Nisbett, P. Thagard, «Induction: Processes of Inference, Learning, and Discovery ». Cambridge, Man : MIT Press, 1986.

[17] C. Adami, R.Seki, R. Yirdaw, « Critical Exponent of Species-Size Distribution », Evolution Artificial Life VI, MIT Press, 1998.

[18] R.K. Belew and Mitchell, « Adaptive Individuals in Evolving Populations: Models and Algorithms », Massachusetts, Addison-Wesley, 1996.

[19] P. Turney, D. Whitley, R. Anderson, « Evolution, Learning, and Instinct : 100 Years of the Baldwin Effect », Special Issue of Evolutionary Computation on the Baldwin Effect, V.4, N.3, 1996.

[20] C. Fournier, « Modélisation des interactions entre plantes au sein des peuplements. Application à la simulation des régulations de la morphogenèse aérienne du maïs (Zea mays L.)

par la compétition pour la lumière », Thèse de l'Institut National Agronomique Paris-Grignon, 2000.
[21] M. Gardner, « Mathematical Games : The fantastic combinations of John Conway's new solitaire game life », Scientific American n°223, p. 120-123, Octobre 1970.
[22] N. Ollinger, « Automates cellulaires: structures », thèse de doctorat de l'école normale supérieure de lyon; Laboratoire de l'informatique du parallélisme, 2002.
[23] J.D. Farmer et A. Belin, « to Artificial life », seconde conférence Artificial Life, 1990,
[24] Ch. Adami, « Introduction to Artificial Life», Springer-Verlag, New-York, 1998.
[26] H.A. Gutowitz, C.G. Langton, « Methods for designing Cellular Automata », with Interesting Behavior, 1994.
[27] A. Langlois, M. Phipps, « Automates cellulaires. Application à la simulation urbaine ». Hermès, Paris, 1997.
[28] J.-Ph. Rennard, « Vie artificielle. Où la biologie rencontre l'informatique », Vuibert, 2002.
[29] S. Wolfram, « Universality and complexity in cellular automata », Physica D, 10:1-35, 1984.
[30] S. Wolfram, «A New Kind of Science », Wolfram Media, 2002. Jean-Philippe Rennard 12/2000.
[31] V.Airault, S. Espié, C. Lattaud & J.M. Auberlet, « Interaction between Pedestrians and their Environment when Road-crossing: a Behavioural Approach », in Proceedings of the 24th Urban Data Management Symposium, Fendel E. & Rumor M. eds, Italie, 2004.
[32] C. Lattaud, « Co-Evolution in Artificial Ecosystems : Competition and Cooperation using Allellopathy », in Lecture Notes in Computer Science, vol. 2936, Liardet & al. eds., Springer-Verlag, 2004.
[33] M. Métivier & C. Lattaud, « Modélisation du comportement à base de systèmes de classifieurs durant un processus d'avatarisation », in Extraction des connaissances et apprentissage vol. 1:3, Hermès, pages 61-85, 2001.
[34] S. Sikora, « Evolution morphologique d'un animat dans un espace 3D », in Actes du colloque Jeunes Chercheurs en Sciences Cognitives 4, 2001.
[35] C. Lattaud, « Autonomous Adaptive Agents in a ResourceAcquisition Problem », Technical Report n°98/01, LIAP-5, Université Paris V, 2001.
[36] C. Lattaud & C. Cuenca, « A Model for the Evolution of Environments, in Lecture Notes in Artificial Intelligence », vol. 1434, Heudin J.C. ed., Springer-Verlag, pages 218-228, 1998.
[37] C. Lattaud & M. Irvine, « Adaptation of Animats Without Fitness Control », in the 3rd International Conference on Computational Intelligence and Neurosciences, Triangle Research Park, 1998.
[38] C. Lattaud, « A Classification for the Control of the Evolution of Adaptive Agents », in Proceedings of the 10th International Florida AI Research Symposium, Daytona, pages 449-454, 1997.
[39] P. Collet, E. Lutton, « Take it EASEA », Parallel Problem Solving from Nature VI, vol. 1917, Paris, Springer, p. 891-901, 09/2000.
[40] F. Varela, « Invitation aux sciences cognitives », Seuil, Paris, p. 56,1996.
[41] J. Gleick « La théorie du chaos, vers une nouvelle science » Champs Flammarion, 1987.
[42] Van de vigver, « Emergence, et explication », Intellectica : Emergence ans explanation, n°25 1997.
[43] R. Dawkins, « The evolution of evolvability, Artificial Life, Proceedings of an Interdisciplinary Workshop on the Synthesis and Simulation of Living Systems », Los Alamos, Addison-Wesley Publishing, pp. 201-220, September 1987.
[44] S. W. Wilson, « The animat path to ai ». In Meyer, J. and Wilson, S., editors, The First International Conference on Simulation of Adaptive Behavior, pages 15-21. The MIT Press, Cambridge, MA, 1990.
[45] Wikipédia, l'encyclopédie libre sur le web, 2006.
[46] C.G. Langton, « Studying Artificial Life with cellular automata », Physica D22, 1986.
[47] C. Langton, « Artificial Life in The philosophy of Artificial Life, » Boden M. A. dir, Oxford readings in Philosophy, Oxford University Press, pp. 64 et suivantes, 1996.
[48] C. Emmeche, « A semiotical reflection on biology, living signs and artificial life », Biology & Philosophy 6(3): 325-340,1991.

[49] C. Emmeche: « Life as an abstract phenomenon: is Artificial Life possible? » in Proceedings of the First European Conference on Artificial Life. The MIT Press, Cambridge, Mass. , 1992.
[50] C. Emmeche, « Modeling Life: A note on the semiotics of emergence and computation in artificial and natural systems », pp. 77-99 in:Biosemiotics. The Semiotic Web 1991, Thomas A. Sebeok & Jean Umiker-Sebeok, eds., Mouton de Gruyter Publishers, Berlin & New York. ,1992.
[51] C. Emmeche, « Is life as a multiverse phenomenon? », pp.553-568 in: Christopher G. Langton, ed.: Artificial Life III (Santa Fe Institute Studies in the Sciences of Complexity, Proceedings Volume XVII), Addison-Wesley Publishing Company, Reading, Massachusetts,1993.
[52] C. Emmeche, « The garden in the machine. », The emerging science of artificial life. Princeton University Press, Princeton. (translation of a Danish introduction to Artificial Life, transl. by Steven Sampson. (Also in other translations. More about this book here), 1994.
[53] C. Emmeche, « The computational notion of life »,Theoria - Segunda Epoca 9 (21): 1-30,1994.
[54] C. Emmeche, « A-life, organism and body: The semiotics of emergent levels », pp. 117-124 in Workshop and Tutorial Proceedings. Ninth International Conference on the Simulation and Synthesis of Living Systems, Boston Massachusetts, September 12th, 2004.
[55] M. Gardner, « Mathematical Games : The fantastic combinations of John Conway's new solitaire game », life, Scientific American n°223, p 120-123, Octobre1970.
[56] E. F. Codd, « Cellular Automata », New York Academic Press, 1968.
[57] C. Bays, « Patterns for Simple Cellular Automata in a Universe of Dense-Packed Spheres», Complex Systems, pages 853-875, 1987.
[58] C. Bays, « Classification of Semitotalistic Cellular Automata in Three Dimensions », Complex Systems, pages 235-254,1988.
[59] C. Bays, « A Note on the Discovery of a New Game of Three-dimensional Life », Complex Systems, pages 255-258,1988.
[60] C. Bays, « A New Game of Three-Dimensional Life », Complex Systems, pp 15-18. 5, 1991.
[61] C. W. Reynolds, « ACM SIGGRAPH Computer Graphics », Proceedings of the 9th annual conference on Computer graphics and interactive techniques SIGGRAPH ’82, 1982.
[62] C. W. Reynolds, « ACM SIGGRAPH Computer Graphics », Proceedings of the 14th annual conference on Computer graphics and interactive techniques SIGGRAPH '87, 1987.
[63] J. Holland, « Outline for a Logical Theory of Adaptive Systems », Journal of the Association for Computing Machinery, vol. 9 : 3, p 297-314, 1962.
[64] J. H.Holland, « Escaping brittleness: the possibilities of general-purpose learning algorithms applied to parallel rule-based systems », In R. S. Michalski, J. G. Carbonell & T. M. Mitchell (Eds.), Machine learning, an artificial intelligence approach, Volume II. Los Altos, California : Morgan Kaufmann, 1986.
[65] J. H. Holland, « Adaptation in Natural and Artificial Systems », MIT Press, 1992.
[66] L. B. Booker, «Intelligent Behavior as an Adaptation to the Task Environment », The University of Michigan, 1982.
[67] L. Booker, « Classifier Systems that Learn Internal World Models, », in Machine Learning Journal, Volume 1, Number 2,3, 1988.
[68] S. Rupert, « La memoire de l’univers » Edition du rocher 1988.
[69] A. Grumbach, « À propos d’émergence . Emergance ou explication », Intellecta emergence and explanation 1997/2 n°25, ISSN N° 0984-0028 185-194, 1997.
[70] J.H. Holland, « Emergence-from order to Chaos », Addison-Wesley, 1997.
[71] J.P. Gorgé, « Emergence », IRIT/2003-12-R, 2003.
[72] D. Ichbiah, « Robots, genèse d'un peuple artificiel ». Editions Minerva. 2005.
[73] P. Merlet, « Parallel manipulators: state of the art and perspective », Advanced Robotics, 1994.
[74] J-P. Merlet. « Parallel manipulators: state of the art and perspective», In T. Takamori and K. Tsuchiya, editors, Robotics,Mechatronics and Manufacturing Systems. Elsevier, 1993.
[75] J-P. Merlet. « Geometry and Kinematic Singularities of closed-loop manipulators », day in Laboratory Robotic and Automation, University of Michigan p 85-96, 1992.
[76] J-P. Merlet, « Parallel manipulators: state of the art and perspective », Journal of Robotics Society of Japan, 10(6):57-62, 1992.

[77] A. Guy, « Capteurs d'environnement en robotique: introduction », encyclopédie : technique de l'ingénieur, traité informatique industrielle.
[78] A. Priski, « Robots mobiles autonomes », encyclopédie : technique de l'ingénieur, traité informatique industrielle.
[79] J.Ferber, « Coopération réactive et émergence », Intellectica, 1994/2, 19, pp. 19-52.
[80] J-P. Georgé, « Résolution de problèmes par émergence, Étude d'un environnement de Programmation Émergente », Thèse de l'université de Toulouse III, 2004
[81] D. Phan, « Régimes et changements structurels. Essais d'économie historique », Thèse de l'université d'Orleans, 2002.

Références sur le Web

http://www.vieartificielle.com/article/?id=7T
http://fr.wikipedia.org/wiki/Robot
http://www.linux-france.org/prj/jargonf/V/vie_artificielle.html
http://www.admiroutes.asso.fr/larevue/2000/3/definition.htm
http://www.futura-sciences.com/comprendre/d/dossier19-3.php
http://www.comm.uqam.ca/~GRAM/C/term/mul/mult290.html
http://www.tout-savoir.net/lexique.php?rub=definition&code=7951
http://www.synomia.fr/img/fr/Vie_artificielle.html
http://membres.lycos.fr/viart/definitions.html
http://www-valoria.univ-ubs.fr/m2ri/Fiches/SSI.pdf
http://www.angelfires-fr.com/reflexion/robot.php
http://www.utc.fr/~webva/
http://www.bio.net/hypermail/Computational-Biology/2000-June/001992.html
http://www.pps.jussieu.fr/~fdls05/AtelierAutomatesCellulaires.html
http://www.cybersciences.com/cyber/4.0/sept96/ordi-1.asp
http://www.cti.ecp.fr/~laurent/MapModule1/PrMod1-04/
http://guillaume.bour.free.fr/index.php?lang=fr&page=bloats%2F%24%7Blang%7D%2Falife.blt
http://b0op.com/blog/index.php/2005/01/01/1582-simulation-de-vie-artificielle
http://www.nachez.fr/IAetSF.htm
http://www.infoscience.fr/web/siteweb.php3?pa=2&cat=9
http://www.mines.inpl-nancy.fr/~tisseran/cours/objet/jeu_vie.html
http://cormas.cirad.fr/fr/applica/fireautomata.htm
http://fr.wikipedia.org/wiki/Vaisseau_(automate_cellulaire)
http://www.limsi.fr/Individu/jps/enseignement/examsma/2003/mamlouk_benouirane/domaine.htm
http://www.univ-st-etienne.fr/criter/stats/http/web/2003/search_200308.html
http://www.lxxl.pt/aswarm/fr_aswarm2.html
http://www.infobiogen.fr/deambulum/liste.php?page=cbio_morphogenese
http://perso.wanadoo.fr/philippe.gascuel/morphogenese.htm
http://www.vieartificielle.com/article/index.php?action=article&id=26
http://www.salonagr.qc.ca/exposant.php?CID=32
http://ail.jinak.cz/
http://www.rennard.org
http://users.pandora.be/martin.melchior/fr/lexique.html
http://users.lug.com/gbarraud/Enseirb/thirdyear/project/circus/glossary_fr.html
http://www.world-of-dawkins.com
http://www.lifl.fr/~wegrzyno/deugMIAS1/PRJ-Scheme/projet2000/doc/ principal.htm
http://www.biologie.uni-hamburg.de/b-online/virtuallaboratory/Animations.html
http://www.genetic-programming.com/johnkoza.html
http://www.ping.be/chaoflight/pagefr/ss11.htm
http://www.poleia.lip6.fr/ANIMATLAB/
http://www.inria.fr/Themes/Theme3-fra.html

http://www.inria.fr/epidaure/GT-RV/gt-rv.html
www.math-info.univ-paris5.fr/~latc/va/va.html
http://www.math-info.univ-paris5.fr/alife/
http://minimum.inria.fr/evo-lab/EVO-easea.html.
http://www.geocities.com/tperz/L4Home.htm.
http://www.cpsc.ucalgary.ca/Research/bmv/vmm-deluxe/TitlePage.html.
http://www.red3d.com/cwr/boids/.
stephenwolfram.com (site officiel de stephen Wolfram)
http://www.vieartificielle.com (serveur de l'institut de SantaFe)
http://www.cpsc.ucalgary.ca/Research/bmv/vmm-deluxe/TitlePage.html.

Autres références

- CD Universalis 2005
- CD Encarta 2005
- Techniques de l'ingénieur version électronique et papier
- Bertalanffy (von) L., « Théorie générale des systèmes », Dunod, Paris, 1973.
- Morin E., « La Méthode. 1-La Nature de la Nature », Seuil, Paris 1977.
- Morin E., « La Méthode. 2-La vie de la vie », Seuil, Paris 1980.
- Gleick james, « la théorie du chaos, vers une nouvelle science », Albin Michel, 1989.
- J. Darnell, H. Lodish, D. Baltimor, « la cellule : biologie moléculaire », Edition Vogot, 1989.
- Penrose R. « Les ombres de l'esprit : à la recherche d'une science de la conscience », Interedition 1995.

La théorie des systèmes complexes actuelle s'intéresse tout particulièrement à la dynamique des systèmes, en se basant sur des concepts clés inspirés essentiellement du vivant (biologie, éthologie, …). Ces concepts sont en quelque sorte le découplage qui permet de théoriser loin des détails des systèmes réels. On arrive ainsi à appréhender la complexité de tels systèmes en discrétisant leurs aspects continus et en réduisant par le biais de l'abstraction un monde qui pourra nous faire noyer dans un millier de détails dont on n'on na pas forcément besoin. On retrouve en tête de ces concepts l'auto-organisation, l'adaptation et l'apprentissage. Dans ce chapitre, nous allons expliciter ces trois notions, et essayer de les projeter dans le champ de l'intelligence collective représentant notre domaine cible.

2.1. Auto-organisation

Le terme auto-organisation fait référence à un processus dans lequel l'organisation interne d'un système (habituellement un système hors équilibre) augmente automatiquement sans être dirigée par une cause extérieure. Typiquement, les systèmes auto-organisés ont des propriétés émergentes (bien que cela ne soit pas toujours le cas).

Avant d'entamer le concept de l'auto-organisation, nous allons en premier expliciter la notion de système, nécessaire à la compréhension de ce qui va suivre.

2.1.1. Concepts de système

Un système est plongé dans un environnement duquel il ne peut être dissocié et avec qui il est amené à interagir (voir Fig 2.1). L'environnement représente tout ce qui est extérieur au système. On dit alors que le système est couplé à son environnement. Les rapports d'un système avec son environnement sont matérialisés par des entrées et des sorties. Ces rapports sont plus au moins intenses selon que le système est dit fermé ou ouvert. Les systèmes fermés vivent totalement repliés sur eux-mêmes, alors que les systèmes ouverts effectuent de nombreux échanges avec tout ce qui les entoure (leur environnement). Un système et son environnement entretiennent des échanges réciproques qui leur permettent de s'ajuster mutuellement. On remarque que l'interaction est d'un intérêt central dans ce cas.

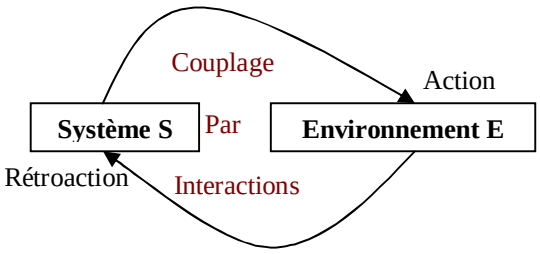


Fig 2.1 Dans le monde réel, un système **S** perçoit et agit localement dans l'environnement **E** dans lequel il est plongé (action). Son comportement modifie l'état de l'environnement qui exerce en retour, une pression sur le système (feedback ou rétroaction). D'une manière générale, les échanges système-environnement entraînent des influences réciproques conduisant à un ajustement mutuel. Ce processus a été observé depuis longtemps et modélisé avec des systèmes artificiels par exemple dans la boucle de rétroaction en cybernétique* (Von Foerster, 1962) [53].

Joël de Rosnay en 1975 et Daniel Durand en 1979 ont proposé tous deux une double description du système, l'une structurelle et l'autre fonctionnelle :

a) Aspect structurel : Au niveau structurel, ils dénombrèrent essentiellement quatre éléments entrant dans la constitution d'un système :

- La frontière qui le sépare de son environnement (Exemple : interface homme-machine, membrane d'une cellule, …).
- Les éléments qui peuvent être identifiés, dénombrés et classés. Ces éléments sont plus au moins hétérogènes en fonction du système étudié (Exemple : Dans le cadre d'un système multi-agents, il s'agit des agents).

- Le réseau de relation, de transport et de communication qui véhicule soit des matières solides, liquides ou gazeuses, soit de l'énergie, soit des informations sous toutes les formes possibles (Exemple : Dans un systèmes multi-agents, ce réseau de relation est représenté par les liens de communication inter-agents, sur lesquels se fondent peu à peu les connaissances sur autrui des agents).
- Les réservoirs dans lesquels sont stockés ces matières, cette énergie, ces produits, ou ces informations.

b) Aspect fonctionnel : Au niveau fonctionnel, ils distinguèrent également quatre composants pour un système :

- Des flux de natures diverses : des matières, des produits, de l'énergie,…, et de l'information. Ils circulent dans les divers réseaux et arrivent dans les réservoirs du système. Dans le domaine informatique, il s'agit des signaux échangés entre entités.
- Des centres de décision qui reçoivent les informations et les transforment en actions, en agissant sur les débits des différents flux. (Exemple : Dans un système multi-agents, ces centres de décisions sont une partie des compétences et des aptitudes des agents. Ces derniers jouissent d'une autonomie suffisante pour prendre par eux-mêmes les décisions opportunes).
- Des boucles de rétroaction qui ont pour objet d'informer les déclencheurs de ce qui se passe en aval, donc de leur permettre de prendre leurs décisions en connaissance de cause. Il s'agit là d'un comportement dit non linéaire. Le mécanisme de rétroaction est en particulier utilisé pour mettre en œuvre les apprentissages par renforcement (notion qu'on verra plus loin).
- Des délais de réponse qui permettent de procéder aux ajustements de temps nécessaires à la bonne marche du système.

Par le biais de l'auto-organisation, un système (ainsi décrit) doit conserver son adéquation fonctionnelle [46] malgré les variations de son environnement (voir Fig 2.2).

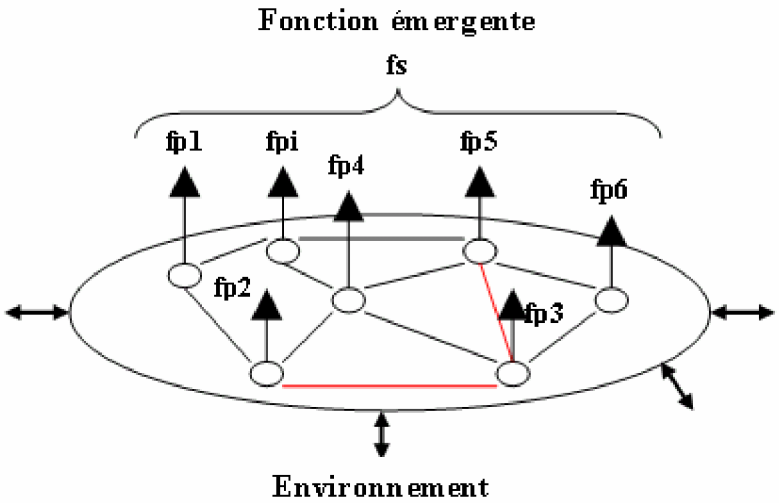


Fig 2.2 L'auto-organisation consiste en une modification des liens existants entre les parties réalisées de manière autonome et qui devrait permettre au système d'être plus performant dans le sens où la coordination et/ou la coopération s'améliorent au cours de sa vie. Ceci est réalisé en recherchant une meilleure organisation pour qu'une partie soit au bon endroit, au bon moment (f_{pi} représente la fonction accomplie par chaque partie i).

Notons que l'une des formes d'interaction particulières est la rétroaction (qu'on verra un peu plus loin).

2.1.2. Définition(s)

Plusieurs spécialistes issus de domaines différents ont proposé des définitions d'auto-organisation, qu'on peut regrouper succinctement dans ce qui suit :

- Le terme d'auto-organisation a initialement été défini par Farley et Clark du laboratoire Lincoln comme suit : un système auto-organisateur est un système qui change sa structure de base en fonction de son expérience et de son environnement [63].
- Toru Ishida et *al.* donnent une définition plus générale de l'auto-organisation ou OSD (organisation self-design) : Elle permet à une organisation d'agents de s'adapter elle-même aux situations dynamiquement changeantes [66].
- Selon Young-pa So et *al.* les agents coopérants travaillent typiquement dans des environnements changeants. Il est donc crucial que les mécanismes de coordination qu'ils utilisent soient capables de s'adapter à ces changements. En particulier, l'organisation des agents doit évoluer dans le temps lorsque les circonstances le nécessitent. Dans le domaine de l'intelligence artificielle distribuée (IAD), une telle réorganisation adaptative est appelée auto-organisation lorsqu'elle est réalisée par les membres de l'organisation [67].
- Pour Pierre Glize et son équipe, l'auto-organisation permet à un système de s'adapter de manière autonome aux conditions changeantes du milieu. Elle consiste plus précisément en une transformation de la topologie (c'est-à-dire des relations entre les agents) en tant que résultat du fonctionnement de ce même réseau dans le cadre de son couplage structurel avec l'environnement. Ils ajoutent que les systèmes auto-organisateurs appartiennent à la classe des systèmes autonomes (systèmes spécifiés par des mécanismes internes d'auto-organisation) et non pas hétéronomes (systèmes définis par des mécanismes extérieurs de contrôle). Cela implique que les règles d'organisation sont intérieures au système qui apparaît ainsi informationnellement clos [68, 69].
- Le Web propose la définition suivante : L'auto-organisation est un phénomène de mise en ordre croissant, et allant en sens inverse de l'augmentation de l'entropie*; au prix bien entendu d'une dissipation d'énergie qui servira à maintenir la structure du système auto-organisé.

Toutes ces définitions concourent pour définir l'auto-organisation, on peut dire que chacune représente un point de vue perçu selon le domaine de spécialisation de ses auteurs. Dans tous les cas elles ne peuvent être que complémentaires.

2.1.3. Caractéristiques des systèmes auto-organisés

Les systèmes auto-organisés sont particulièrement bien adaptés aux systèmes de nature distribuée, fortement dynamiques, où aucun élément entrant dans sa construction, ni même le concepteur, ne possède une vue globale du système. Contrairement à un système déterministe, un système auto-organisé résiste très bien aux pannes; il s'adapte de lui-même, de manière autonome aux perturbations en provenance de son environnement.

Les systèmes auto-organisateurs ont aussi la faculté de réagir de manière non prédonnée à plusieurs perturbations similaires (ils sont imprévisibles). Ils ont par conséquent la faculté d'apprendre au fur et à mesure qu'ils évoluent.

Une autre caractéristique essentielle, et particulièrement intéressante des systèmes auto-organisés est qu'ils peuvent accomplir des tâches complexes en utilisant des comportements individuels simples et concis. Donc, l'auto-organisation n'exige pas des mécanismes compliqués pour sa mise en œuvre (puisque c'est un phénomène spontané et imprévisible), cependant elle fait apparaître des structures qui, peuvent paraître complexes. On peut dire dans ce cas qu'ils sont des systèmes non linéaires, étant donné que la linéarité permet de tracer l'activité du niveau local pour interprétation au niveau global. Les interactions entre composantes du système sont obtenues de manière non linéaire. Ceci signifie que toute partie du système doit pouvoir être influencée de manière indirecte par d'autres composantes du système (voir fig 2.2). C'est le cas dans les systèmes à feedback négatif ou positif.

Les systèmes auto-organisateurs sont capables d'effectuer des tâches complexes qu'aucun élément les composant n'aurait pu effectuer seul. La notion d'émergence (voir chapitre 1.11) est donc très

fortement liée à la propriété d'auto-organisation, et en dépit de ses avantages, elle reste très peu utilisée.

Exemples : Les exemples les plus évidents des systèmes auto-organisés sont issus de la physique. C'est d'ailleurs dans ce domaine que le terme est apparu pour la première fois. L'auto-organisation est aussi présente en chimie où elle a souvent été synonyme d'auto-assemblage, comme la synchronisation d'oscillateurs couplés, les ondes observées dans certains systèmes chimiques alimentés en continu, ou l'apparition de motifs périodiques dans un liquide chauffé par le dessous (cellules de convection). Le concept d'auto-organisation est aussi central dans les systèmes biologiques, que ce soit au niveau cellulaire ou social, des exemples comme le développement de colonies de bactéries, variation périodique des populations dans un système prédateur-proie, déplacement cohérent d'un banc de poissons, et fourmilières abondent. On trouve aussi de nombreux exemples de phénomènes auto-organisés dans d'autres disciplines comme les sciences sociales, l'économie ou encore l'anthropologie. Le principe des systèmes auto-organisés s'étend même à la formation des dunes, des rivages, et des galaxies. Notons en dernier que l'un des premiers mécanismes mathématiques proposés pour étudier l'auto-organisation de manière formelle sont les automates cellulaires.

2.1.4. La rétroaction

Lorsqu'un système interagit avec son environnement, les entrées sont le résultat de l'influence de l'environnement sur le système et les sorties sont l'influence du système sur l'environnement. Donc, Lors d'une rétroaction, l'information sur le résultat de la transformation induite par les entrées est transmise au système sous forme d'une donnée d'entrée.

D'autres définissent la rétroaction (ou feed-back), comme l'action qui agit en retour sur le phénomène qui se trouve à sa source. Dans l'ordre du vivant, elle assure la régulation des mécanismes biologiques. En éducation, les situations rétroactives sont multiples. Ainsi, par exemple, l'apport du groupe agit sur l'activité d'apprentissage de l'individu, et ce que l'individu retire de cet apport agit en retour sur le groupe lui-même.

On peut aussi envisager la rétroaction dans l'apprentissage par tâtonnement individuel. Le résultat de ceci est une connaissance nouvelle qui rétroagit sur le savoir qui lui a donné lieu, en le transformant en savoir nouvellement organisé.

La rétroaction se fait dans la durée (il faut du temps pour l'élaboration du processus) et dans l'incertitude (on ne sait jamais quel effet produira un phénomène donné). Elle permet ainsi au processus d'évoluer tout en se transformant, garantit l'efficacité créatrice des actions, et facilite les émergences, en renforçant la dynamique du processus [62].

La récursion signale cet aspect d'un processus qui fait que les effets sont nécessaires à sa propre production. Donc le processus dépend des effets qu'il engendre pour se maintenir.

« Pratiquer permet de comprendre et de bien analyser. En retour, mieux comprendre et analyser permet de mieux pratiquer.»

2.1.3.1. Rétroaction positive : Lorsque les nouvelles données agissent dans le même sens que l'action principale, c'est-à-dire amplifie la transformation, la rétroaction est dite positive. Dans ce cas, ces effets sont cumulatifs (voir Fig 2.3 à gauche).

2.1.3.2. Rétroaction négative : En revanche, si les nouvelles données produisent un résultat dans le sens contraire à celui fourni par les précédents résultats, la rétroaction est dite négative; ses effets stabilisent le système (voir Fig 2.3 à droite).

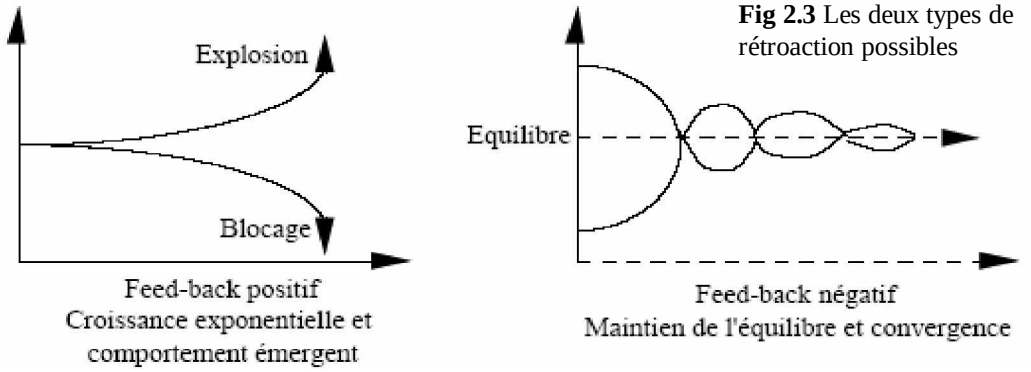


Fig 2.3 Les deux types de rétroaction possibles

2.1.5. Les mécanismes de l'auto-organisation

Le préfixe «auto» souligne qu'il peut apparaître des phénomènes collectifs robustes dans un ensemble d'éléments en interaction, sans qu'il y ait besoin ni d'un chef d'orchestre, ni d'une préparation initiale homogène, ni de conditions extérieures biaisant les interactions ou la dynamique individuelle. Le terme central d'*organisation* suggère une apparition d'ordre et renvoie aux notions d'entropie* et d'information. Par exemple, une structure spatiale va émerger d'un mélange homogène de composants. En termes techniques, on parle d'une diminution (locale) de l'entropie. Il n'y a en cela nulle violation du second principe de la thermodynamique, puisque le système est ouvert : la diminution d'entropie se fait aux dépens d'une consommation d'énergie [67].

Notons que l'émergence est restée longtemps qu'un concept philosophique manipulable en tant que tel, le domaine de l'auto-organisation a tout de suite essayé de chercher des règles générales de fonctionnement expliquant la croissance et l'évolution des structures systémiques observées, de trouver les formes que les systèmes peuvent prendre, et finalement de produire des méthodes pour prédire les futures organisations apparaissant à la suite de changements effectués sur les composants des systèmes. Les résultats éventuels sont censés être applicables à tout autre système exhibant les mêmes caractéristiques (recherche de fonctionnements génériques).

Un des meilleurs exemples est certainement le célèbre modèle réacto-diffusif (ou de réaction-diffusion) de Alan Turing, découvert alors qu'il explorait les mécanismes de la genèse du cerveau. Ce modèle explique comment des structures régulières peuvent apparaître grâce à la seule combinaison de mécanismes d'activation et d'inhibition engendrés par les mécanismes de production eux-mêmes : lors de la genèse du système, il y a production d'un activateur, pour produire certains types de structures, qui s'autocatalyse (grande production de ce type de structures) mais qui active également un inhibiteur qui empêche la formation de ces structures. Comme les taux et les vitesses de diffusion diffèrent (l'inhibiteur est plus rapide), la production des structures est perturbée régulièrement par l'inhibiteur dès que la quantité d'activateurs augmente, c-à-d il y a mise en place d'une régulation. Ce modèle pourtant simple et répondant entièrement à la définition de l'auto-organisation modélise particulièrement bien un vaste panel de phénomènes naturels, de la morphogenèse de certaines structures embryonnaires à la production des « dessins » sur les animaux, jusqu'à certains comportements des insectes sociaux. Le fait d'aboutir effectivement sur des modèles ou des mécanismes concrets permet alors une approche d'expérimentation et de simulation : il suffit d'intégrer le comportement à tester au sein d'un programme pour observer et analyser ce qu'il produit. On peut alors tester efficacement des hypothèses sur le fonctionnement du monde qui nous entoure. De nombreux travaux se sont ainsi focalisés sur l'expérimentation de systèmes fonctionnant sur des mécanismes d'auto-organisation, notamment en biologie et plus particulièrement en éthologie* où les phénomènes observés incitent naturellement à se tourner vers l'auto-organisation.

On peut ainsi citer les travaux de Jean-Louis Deneubourg, Scott Camazine, Eric Bonabeau et Guy Théraulaz, ensembles ou séparément, sur les insectes sociaux comme les fourmis et les termites [66].

Pour résumer, nous dirons que l'auto-organisation est l'ensemble des processus au sein d'un système, issus de mécanismes basés sur des règles locales, qui conduisent ce système à produire des structures ou des comportements spécifiques non dictés par l'extérieur du système.

L'auto-organisation a ainsi apporté l'idée que des formes stables et statistiquement reproductibles peuvent découler d'un équilibre dynamique, mettant en jeu des règles locales et stochastiques. Il s'agit d'un phénomène essentiellement collectif : les propriétés globales ne peuvent se réduire à celles d'un ou de plusieurs éléments isolés; on rejoint là les notions (encore inachevées) d'émergence et de complexité.

2.1.6. Règles comportementales locales

Dans la plupart des définitions d'émergence et d'auto-organisation, il est question sous une forme ou une autre de règles locales et de comportements en résultant, avec comme contrainte forte pour ces comportements de n'être pas imposés, dictés, explicitement informés ou contraints par l'environnement du système. Le caractère local d'une règle pose un cadre strict et clair. Mais pour ce qui est de l'influence de l'environnement sur le système, directement ou au travers des règles, la caractérisation exacte de cette influence est particulièrement difficile et floue. Prenons l'exemple des cellules de convection de Bénard qui est un exemple classique d'auto-organisation. Le phénomène produit par auto-organisation est ici la forme des déplacements des molécules d'eau qui crée cette structure de courants observable et particulière. Les règles locales sont ici les déplacements et les chocs entre molécules. Le fait que les molécules se déplacent plus aisément lorsqu'elles vont dans le même sens (car subissant moins de chocs) crée des sens de circulation. Mais les parois du conteneur ainsi que l'apport de chaleur qui oblige les molécules à se déplacer font bien partie de l'environnement et influent sur le comportement du système. Il faut alors décider de l'impact et de la nature de cette influence sur les comportements des molécules et du système. On peut alors argumenter que c'est bien cet apport de chaleur qui impose aux molécules de se déplacer et que les parois contraignent ce déplacement. Mais cela ne suffit pas pour expliquer comment les molécules doivent se déplacer, et ce sont bien les règles locales de collision qui vont faire apparaître les cellules hexagonales visibles de dessus. Le cadre d'auto-organisation semble ici relativement clair. Mais plus le système sera complexe et riche, plus les interactions avec l'environnement auront une influence et plus il sera dur de juger de la nature de cette influence [71].

2.1.7. L'auto-organisation en intelligence collective

a) Robotique collective : La robotique collective est l'un des domaines de l'intelligence collective où un bon nombre de chercheurs étudient l'auto-organisation. Citons par exemple Cem Ünsal [63] qui a choisi la collecte d'objets comme finalité de ses expériences et désire établir un comportement coopératif entre agents. Les tâches de recherche et de collecte d'échantillons de roche sont réalisées par un groupe de robots autonomes qui doivent les porter ensuite à un endroit déterminé. Les agents n'ont qu'une vue partielle; ils ne perçoivent qu'un nombre limité d'agents et d'objets. La difficulté est accrue chez Steels en raison d'un terrain parsemé d'obstacles, de collines, de vallées, etc.Les agents sont individuellement incapables de transporter la charge. Chez Ünsal, les agents sont obligés de se regrouper pour pouvoir faire office de transporteur. Chez Steel, la charge est fractionnable, ce qui permet aux agents de transporter des fragments de charge. Ils doivent cependant, dans ce dernier cas, faire en sorte d'optimiser leur déplacement. Les échantillons étant groupés en certains points, les robots à force de coopération, vont pouvoir accomplir leur tâche.

Leur problématique peut se résumer de la manière suivante : Comment des robots autonomes et distribués géographiquement peuvent se coordonner pour trouver des échantillon et les transporter jusqu'à un endroit déterminé.

b) Insectes sociales : Voyons un exemple plus complexe qui est celui de la construction de nids des termites autour d'une reine. Ces constructions ont des formes bien particulières avec une chambre pour la reine au centre de constructions concentriques élevées composées de nombreuses arches et passages. Les termites étant des insectes avec des cerveaux simples et des capacités de communication limitées, toute forme de planification complexe permettant d'atteindre une forme connue de termitière est bien sûr exclue, et les éthologues ont très tôt envisagé la piste de l'auto-organisation. Les modèles actuels de comportements des termites expliquent relativement bien le résultat final [71].

Il est à noter que l'auto-organisation est étudiée aussi dans les systèmes naturels, le domaine de la systémique. On la retrouve dans les réseaux connexionnistes, les algorithmes génétiques, etc.

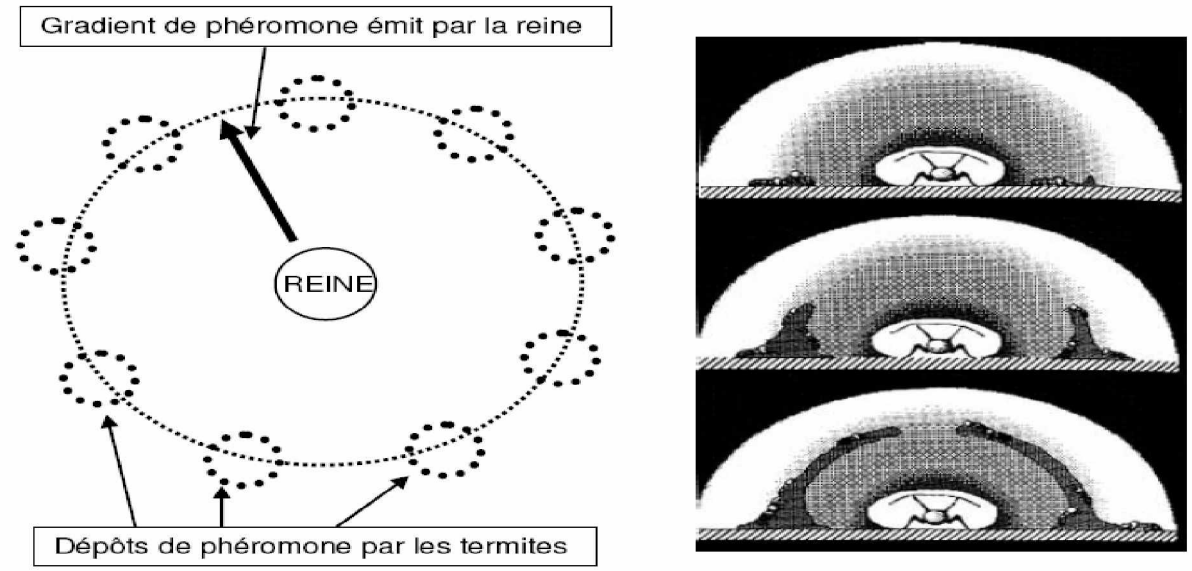


Fig 2.4 Le modèle d'auto-organisation expliquant la construction des termitières autour d'une reine.

Ces modèles (voir Fig 2.4) sont basés sur la diffusion de phéromones* par les termites et par la reine, ainsi que par la prise en compte de ces deux types de phéromones par les termites lors du dépôt des composants de la termitière. En simplifiant, les termites préfèrent déposer les matériaux près de fortes concentrations de phéromones et lorsqu'ils les déposent, ils émettent également des phéromones. Les phéromones émises par la reine incitent donc les termites à déposer les matériaux en cercles concentriques autour d'elle et les phéromones émises par les termites eux-mêmes conduisent à créer les arches, colonnes et murs formant effectivement la structure. Si l'on se concentre sur la termite individuelle, elle est effectivement autonome (elle n'est pas contrôlée) et se base sur des règles locales pour son comportement : le dépôt de matériaux de préférence sur de fortes concentrations en phéromones et l'émission de phéromones lors des dépôts. On peut alors considérer que l'on a développé un modèle basé sur l'auto-organisation. Pourtant, si les apparitions d'arches sont bien un phénomène émergent (on retrouve les mêmes phénomènes chez les fourmis quant à la création de tas), le fait d'avoir des cercles concentriques pour les murs ne l'est certainement pas : il est évident que l'émission de la phéromone de la reine sous cette forme conduit effectivement à la forme concentrique. La reine indique bien comment déposer les matériaux pour obtenir la circularité. Pourtant, nous avons bien des règles locales pour les termites. Mais il semble que ces règles, en ce qui concerne la forme circulaire, sont trop explicitement influencées par l'environnement (si on considère l'ensemble des termites excepté la reine comme étant le système étudié) ce qui exclut la caractéristique d'émergence pour cette forme. Peut-on cependant parler

d'auto-organisation? Le flou autour de la notion d'influence vient perturber inlassablement les réflexions. En fait, dans la communauté scientifique, on retrouve abondamment un mélange entre de l'auto-organisation qu'on peut qualifier de stricte et la notion de template (plan ou schéma). La phéromone de la reine est clairement un template pour les termites, mais la température du sol, l'humidité et tout simplement la forme, en sont aussi [71].

Ces templates ont une influence sur les comportements des entités composant le système. Plus ces templates sont précis et imposés, moins on parle d'auto-organisation, même si les entités elles-mêmes ont des comportements autonomes et suivent des règles absolument locales. Lorsqu'on utilise l'auto-organisation comme fonctionnement interne du système, il faut donc porter un regard attentif et critique sur l'influence de l'environnement sur le système.

On peut dire en fin de compte que l'auto-organisation correspond (dans le cadre de ce qui nous intéresse) à la capacité d'une société d'agents à apprendre de manière autonome et à s'adapter à tous signaux perçus par l'environnement ou échangés entre entités qui constituent cette société. Les entités qui composent le système ne poursuivent qu'un objectif individuel et ne connaissent rien de la fonction globale devant être produite par le système. Elles profitent du signal perçu de l'environnement pour enrichir les liens d'interaction qui les unissent les unes avec les autres.

Donc, l'auto-organisation explique l'émergence d'un comportement collectif « intelligent » et macroscopique par des interactions simples au niveau microscopique. Il est clair que l'auto-organisation n'exclut pas la complexité au niveau individuel (microscopique) mais elle suppose qu'à un niveau plus bas les entités sont plus simples.

2.2. Adaptation

La notion intuitive d'adaptativité précise qu'un système qui a cette propriété se comporte de façon adéquate face aux sollicitations de son environnement, et l'on entend par système un objet construit rationnellement, dans le but de servir les objectifs de ses utilisateurs, en usant de méthodes classiques et éprouvées .

Les systèmes adaptatifs se caractérisent par l'existence de tendances fondamentales conduisant à leur réorganisation ; ce sont des systèmes complexes qui modifient sans cesse les couplages entre leurs composants et qui modifient ces composants, mais qui restent soumis à des tendances les poussant à s'adapter. Avant de parler de système adaptatif, nous allons tout d'abord donner les différentes définitions proposées pour la notion d'adaptation. Nous montrons par la suite que l'adaptativité est un caractère organisationnel distribué dans tous les composants du système [73, 75].

2.2.1. Définition(s)

- L'adaptation est la modification à court ou long terme du comportement, de la morphologie et/ou de la physiologie d'un être vivant, afin d'optimiser sa survie et/ou celle de son espèce dans son environnement.
- L'adaptation est une modification (de son organisation, de son code génétique...) afin d'être plus en adéquation avec quelque chose (son milieu naturel, une situation politique nouvelle, une technologie qui bouleverse son organisation...).
- C'est la capacité des êtres vivants à se développer dans des conditions particulières (milieu humide, sec, salé, désertique, etc.).
- C'est une caractéristique qui aide un animal à survivre dans son habitat.
- L'adaptabilité est la capacité d'un système à ajuster ses mécanismes, ses processus et sa structure à des changements climatiques hypothétiques ou réels. L'adaptation peut être spontanée ou planifiée ; elle peut se produire en réponse à, ou en prévision d'une évolution des conditions.

- C'est le processus et le résultat de l'ajustement d'un organisme vivant ou d'une société aux conditions d'un environnement donné, ce qui lui permet de survivre, de se reproduire et de se développer.
- L'adaptation est le fait, pour le sujet, d'enregistrer des modifications à partir de l'interaction qu'il développe avec la réalité. Elle résulte de « l'équilibre entre l'assimilation et l'accommodation ».

De ces différentes définitions tirées essentiellement d'encyclopédies et de dictionnaires, on comprend que l'adaptation est une notion intuitive qui fait référence au couple (système, environnement), et que le système qui à cette caractéristique se comporte de façon adéquate face aux sollicitations de son environnement.

2.2.2. Système adaptatif

Un système adaptatif est un système dont la raison à fonctionner est conduite par la satisfaction de ses tendances fondamentales. Placé dans son environnement et pour satisfaire ses tendances, le système doit pouvoir adapter sa structure, agir en modifiant ainsi son comportement. Il sera composé de trois parties : une organisation d'entités permettant son fonctionnement rationnel, une organisation réifiant ses tendances et une composante les couplant fortement.

Ce type d'adaptativité est entendu au sens organisationnel, en précisant que la structure du système change, et évolue sans cesse, pour que celui-ci puisse se placer en situation de concordance avec son environnement et avec ses tendances fondamentales, c'est-à-dire pour qu'il s'adapte [75].

2.2.3. Architecture d'un système adaptatif

La façon de représenter l'architecture du système adaptatif, c'est-à-dire sa composition au niveau conceptuel, revient à définir trois sous-systèmes distincts et couplés [76, 77](voir Fig2.5) :

1. une organisation* de ses constituants d'expression, qui contient une partie interfaçant le système avec son environnement et une autre chargée de définir une représentation conforme de la situation courante et également de construire la conception distribuée du plan d'action, par composition des plans locaux générés par les constituants, en utilisant les communications entre eux et la formation de groupes,
2. une composante connexionniste, appelée réseau membranaire, qui représentera explicitement les tendances fondamentales? Les caractères de l'information à portée globale, en réifiant les caractères typiques de cette information,
3. une composante de couplage, qui représente la liaison entre l'ensemble des constituants et la composante connexionniste, pour faire en sorte que la réorganisation opérée par les constituants d'expression soit concordante avec les tendances fondamentales exprimées par la composante connexionniste. Cette composante est formée d'une organisation de constituants de couplage.

Il est à noter que certains systèmes, naturels ou artificiels, sont conçus pour satisfaire à des besoins très généraux qui orientent leurs comportements de manière décisive. C'est ces besoins généraux qui sont nommés tendances fondamentales du système.

Les tendances fondamentales, pour les organismes naturels, sont de survivre, de se nourrir, de se reproduire, de se reposer, de maintenir leur situation d'existence dans leur environnement. Ce sont ces raisons qui conduisent leurs comportements, en réorganisant leur structure et en faisant action dans leur environnement. Par la nécessité à se comporter et à agir que leur imposent ces tendances fondamentales, ils sont amenés à résoudre des problèmes variés, et à apprendre à les bien résoudre. La résolution de problèmes est donc, pour ces systèmes, un moyen et non un but. Nous appellerons ces systèmes, dont le comportement est conduit par des tendances fondamentales, des systèmes adaptatifs.

On considère que les tendances fondamentales présentes dans le système sont nombreuses, et peuvent être contradictoires.

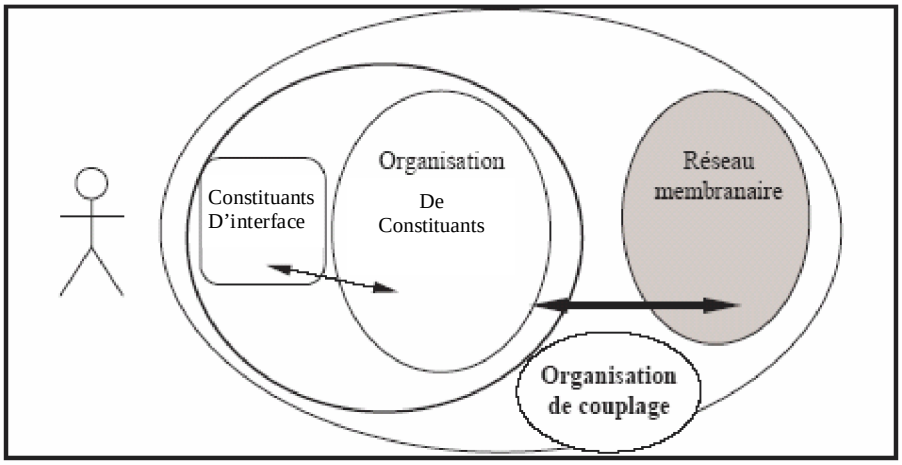


Fig 2.5 Architecture d'un système adaptative

De nos jours, l'un des exemples d'application des systèmes adaptatifs qui peut retenir le plus d'attention est celui concernant l'utilisation des appareils mobiles dans les systèmes multimédias, où le développement de systèmes adaptatifs devient une nécessité.

2.3. Apprentissage

Puisque le sujet central traité dans ce mémoire s'inscrit dans le domaine de la robotique collective nous allons plutôt focaliser sur les méthodes d'apprentissage les plus importantes, adoptées dans le domaine de la robotique.

Depuis les débuts de cette discipline, l'approche traditionnelle de programmation des robots la plus utilisée est basée sur l'analyse et la géométrie. Cette approche a permis de résoudre un grand nombre de problèmes, et d'obtenir des preuves de convergence et de stabilité.

Malgré ceci de nombreux problèmes persistent. Le principal étant celui d'envisager l'ensemble des cas possibles pendant la programmation. En effet, la réaction d'un système pré-programmé face à une situation inconnue est incertaine. Il a donc fallu trouver des solutions alternatives à l'approche mathématique (qui est de nature déterministe) pour permettre aux machines d'être plus autonomes. Ces solutions sont inspirées du vivant. Les techniques qui ont le plus de succès dans ce sens, seront présentées un peu plus loin. Mais avant d'entamer ceci, et pour avoir une idée générale sur le concept de l'apprentissage nous allons commencer par définir le concept de l'apprentissage du point de vue de différents spécialistes issus de différents domaines.

2.3.1. Définition(s)

- L'apprentissage est l'acquisition de nouveaux savoirs ou savoir-faire, c'est-à-dire la mise en relation entre un événement provoqué par l'extérieur (stimulus) et une réaction adéquate du sujet.
- C'est l'acquisition d'un ensemble de savoir faire nouveaux, en vue de faciliter l'adaptation à des acquisitions ultérieures.
- C'est un processus qui implique des étapes, des phases et commence dès la naissance; il est marqué par un temps, un lieu, et un moment donné. On affirme aujourd'hui que même la

perception (visuelle) est un acte de représentation, qui s'apprend, et se développe, elle est éminemment culturelle.
- C'est un acte de perception, d'interaction et d'intégration d'un objet par un sujet.
- C'est une acquisition de connaissances ou de développement d'habiletés ou d'attitudes.

Toutes ces définitions expliquent l'apprentissage, mais la définition qui nous intéresse elle est plus formelle. Dans cette perspective, on propose celle donnée par Mitchell [24] :

« Un programme apprend par :
- L'expérience E par rapport à une classe de tâche T, et une mesure de performance P,
- Si sa performance mesurée par P pour les tâches de classe T s'améliore avec E. »

2.3.2 Méthodes d'apprentissage

La nature a développé des stratégies alternatives afin de pouvoir analyser rapidement une situation et en déduire les meilleures réactions. L'étude de l'homme, et du milieu animal restent probablement les sources d'inspiration les plus riches au monde. Que ce soit pour l'homme ou pour l'animal, il n'y a que trois origines possibles à ces comportements [74] :

- Ils sont soit le résultat d'une imitation. Certains singes disposent de cette faculté à transmettre par l'exemple certaines aptitudes à leur progéniture. Ils leurs montrent notamment comment utiliser à bon escient un bâton ou une pierre pour ouvrir une noix de coco. Le jeune singe, écarté de ses parents après la naissance, n'est pas capable d'acquérir par lui-même cette faculté. On parle dans ce cas de mimétisme*.
- Ils peuvent être le fruit d'un apprentissage individuel. Prenons l'exemple d'un jongleur : si vous n'avez jamais appris à jongler et que vous essayez pour la première fois, c'est l'échec assuré. Donc, la capacité de jongler n'est pas inhérente à notre patrimoine génétique. Ce n'est pas non plus en regardant quelqu'un jongler pendant un certain temps qu'on va pouvoir acquérir cette faculté. Le seul moyen d'apprendre à jongler est de s'entraîner. C'est donc bien le fruit d'un auto-apprentissage.
- Enfin, ils peuvent être le fruit de l'évolution : un poulain peut se lever et tenir debout quelques heures seulement après sa naissance. Il est évident qu'il n'a pas appris en quelques heures à se tenir debout, il transportait donc dans son patrimoine génétique cette aptitude à se maintenir sur ses pattes.

Les principales techniques d'apprentissage en robotique peuvent être classées (en s'inspirant des exemples cités ci-avant) comme suit :
- l'apprentissage par l'exemple,
- l'apprentissage par renforcement,
- et les méthodes évolutionnistes.

Dans ce qui suit, nous allons nous intéresser en premier lieu aux techniques d'apprentissage par l'exemple, à savoir les réseaux de neurones (inspirées de l'étude du cerveau). L'objectif consiste à montrer une série d'exemples, chaque exemple étant associé à un stimulus d'entrée. Après cette phase d'apprentissage, si le réseau est stimulé avec une configuration d'entrées, alors il saura automatiquement de quel exemple il s'agit. Les réseaux de neurones trouvent de nombreuses applications dans différents domaines, notamment en classification de données ou en reconnaissance vocale.

Par la suite, on présente les techniques d'auto-apprentissage. Ce sont des méthodes, basées sur les processus markoviens, et permettent notamment de construire un modèle d'apprentissage. Une fois ce modèle construit, la technique consiste à déterminer quelles sont les meilleures actions à réaliser. Lorsque ces actions permettent au système d'obtenir une récompense importante, leurs liens d'activation sont renforcés, on parle alors d'apprentissage par renforcement.

En dernier lieu, nous présenterons les algorithmes évolutionnistes (approche qu'on a adopté pour la résolution de notre problème). Inspirés de l'évolution darwinienne, le principe général consiste à attribuer à chaque individu une séquence génétique correspondant à un comportement. Régulièrement, les individus sont rassemblés pour pouvoir effectuer des croisements et des mutations sur leurs chaînes chromosomiques. L'apprentissage est donc découpé en générations successives et les performances de l'ensemble de la population sont ainsi accrues au fil de l'évolution.

Mentionnant tout de même que ces techniques d'apprentissage nécessitent un critère à optimiser, comment estimer la performance d'un individu au fil de l'évolution ? Ou encore comment récompenser un agent dans le cadre de l'auto-apprentissage ? On parlera de ceci en dernier lieu.

2.3.2.1 Réseaux de neurones

En 1943, deux bio-physiciens de l'université de Chicago, Mac Cullogh et Pitts proposent le premier modèle de neurone biologique. Ce neurone formel, aussi appelé neurone à seuil, est inspiré des récentes découvertes en biologie. C'est un neurone logique (0 ou 1). En 1949, le psychologue Donald Hebb introduit le terme connexionnisme pour parler de modèles massivement parallèles et connectés. Il propose de nombreuses règles de mise à jour des poids.

En 1958, le psychologue Frank Rosenblatt, combinant les idées de ses prédécesseurs, propose le premier perceptron*. Ce réseau, capable d'apprendre à différencier des formes simples et à calculer certaines fonctions logiques, est inspiré du système visuel. Au début des années 60, les travaux de Rosenblatt suscitent un vif enthousiasme dans le milieu scientifique. Mais en 1969, deux scientifiques américains de renom, Minsky et Papert, publient un livre [16] qui démontre les limites du perceptron proposé par Rosenblatt. En particulier, son incapacité à résoudre les problèmes non linéairement séparables, dont la fonction logique XOR est un célèbre exemple. Les travaux ralentissent considérablement jusqu'aux années 80. En 1982, Hopfield démontre l'intérêt des réseaux entièrement connectés [17]. Parallèlement, Werbos conçoit un mécanisme d'apprentissage pour les réseaux multicouches de type perceptron : la rétropropagation (Back-Propagation). Cet algorithme, qui permet de propager l'erreur vers les couches cachées, sera popularisé en 1986 dans un livre « Parallel Distributed Processing » par Rumelhart et al [15]. Depuis ces travaux, les applications des réseaux de neurones n'ont cessé de croître. Il a d'ailleurs été démontré qu'un réseau MLP (Multi Layer perceptron) avec seulement deux couches peut approximer n'importe quelle fonction de Rn dans Rm avec une précision arbitraire.

2.3.2.1.1. Notation

Nous rappelons brièvement le modèle général du neurone artificiel, qui est l'élément de base de beaucoup de réseaux. Il est composé des éléments suivants :

- Une ou plusieurs entrées pondérées,
- Un sommateur,
- Une fonction de transfert,
- Une sortie.

La figure Fig 2.6 montre le schéma général du neurone artificiel, avec :

- xi le stimulus d'entrée,
- wij la valeur du poids synaptique reliant le stimulus i au neurone j,
- σ() la fonction de sortie du neurone,
- s la sortie du neurone.

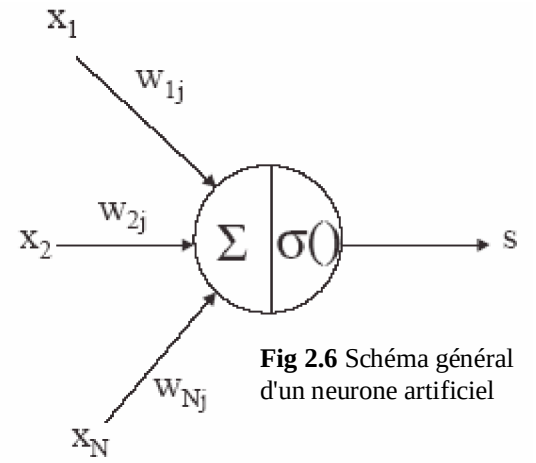


Fig 2.6 Schéma général d'un neurone artificiel

La fonction d'un neurone artificiel est donnée par l'équation suivante :

$$s=\sigma\left(\sum_{j=1}^{n} x_j.w_{ji}\right)$$

Équation 1 : Equation de la fonction de transfert d'un neurone

Un réseau de neurones n'est finalement qu'une représentation conviviale de fonctions mathématiques. En effet, chaque réseau peut s'écrire sous la forme d'une équation. La fonction de transfert de base des réseaux est donnée par l'Équation 1. La fonction de sortie des neurones est principalement utilisée pour mettre en forme les signaux de sortie des neurones. Si par exemple le système est binaire, une fonction de type seuil pourra être utilisée (voir Fig 2.7). Si le système est borné (par exemple un réseau dont la sortie est appliquée directement sur un actionneur) alors la fonction de sortie peut aussi être bornée (voit Fig 2.8). Nous avons présenté ici deux fonctions fréquemment rencontrées, mais d'autres fonctions orientées vers d'autres applications plus spécifiques existent aussi. Nous verrons notamment la particularité des réseaux RBF (Radial Basis Functions)

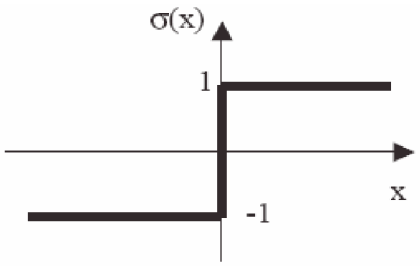


Fig 2.7 Fonction de sortie de type seuil

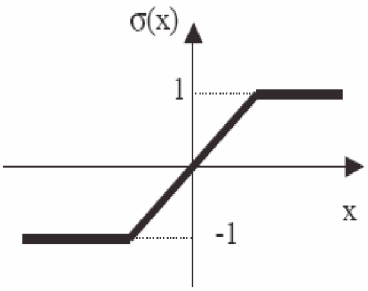


Fig 2.8 Fonction de sortie bornée

2.3.2.1.2. La retropropagation du gradient

La structure d'un réseau de neurones est constituée de neurones reliés entre eux par des liaisons synaptiques. A chacune de ces liaisons est associé un poids synaptique et ce sont précisément les modifications apportées à ces poids qui vont nous intéresser. De manière générale, l'apprentissage neuronal consiste à ajuster les valeurs des poids synaptiques du réseau pour que la réponse du réseau soit celle désirée. Dans un premier temps, nous allons présenter l'apprentissage des poids pour un réseau sans couche cachée.

a) **Réseaux sans couche cachée :** Dans ce type de réseaux, il y a autant de neurones que de sorties. Les entrées sont directement reliées aux neurones par l'intermédiaire des liaisons synaptiques. Le principe général consiste à calculer l'erreur de chaque sortie, en différenciant la sortie désirée et la sortie obtenue (Équation 2), puis à ajuster les poids afin de diminuer cette erreur (Équation 3). Le coefficient η représente le coefficient d'apprentissage, plus sa valeur sera faible, plus l'apprentissage sera long et stable.

Algorithme 1 (apprentissage d'un réseau sans couche cachée) [74] :

1. Initialisation des poids à des valeurs aléatoires de faible grandeur,
2. Sélection d'un exemple dans la base d'apprentissage <x,d>,
3. Propager l'entrée à travers le réseau et calculer les sorties oi
4. Pour chaque sortie du réseau calculer le terme d'erreur :

$$\Delta w_{ij} = \eta (d_i - o_i) x_i$$ **Equation 2**

5. Mettre à jour chaque poids synaptique du réseau :

$$w_{ij} = w_{ij} + \Delta w_{ij}$$ **Equation 3**

6. Retourner en 2. Tant que l'erreur est trop grande.
7. Fin

Les réseaux de neurones sans couche cachée ne peuvent modéliser que des fonctions linéaires. Pour illustrer ces propos, prenons l'exemple classique d'un réseau devant modéliser une fonction logique : il possède deux entrées x1 et x2, un perceptron unique et une sortie s. La fonction de transfert du réseau est donnée par l'Équation 4. Il s'agit là d'une fonction linéaire et nous allons optimiser les paramètres de la droite w11 et w21.

$$s = \sigma(x_1 . w_{11} + x_2 . w_{21})$$

Équation 4 : Fonction de transfert d'un réseau modélisant une fonction logique

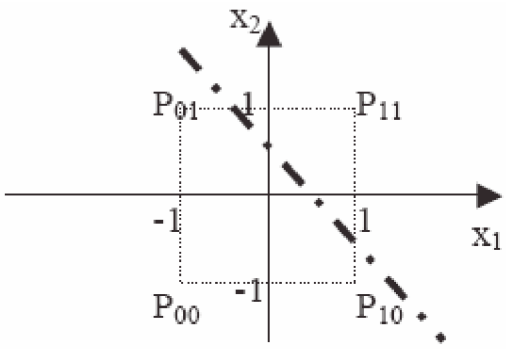


Fig 2.9 Représentation du réseau dans l'espace des entrées.

Par exemple, pour modéliser une fonction logique ET, il existe une infinité de droite permettant d'isoler le point P11 des points P00, P01 et P10 (voir Fig 2.9). En revanche, si l'objectif est de modéliser une fonction « OU EXCLUSIF », il n'existe aucune droite permettant d'isoler les points P11 et P00 des points P01 et P10. Les réseaux sans couche cachée ne permettent pas de modéliser toutes les fonctions. L'ajout d'une couche intermédiaire permet donc de découper l'espace en zones. Si une fonction de sortie linéaire est conservée, alors les zones sont séparées par des droites, mais il y a plusieurs droites, ce qui permet de dissocier des zones non linéairement séparables. Les réseaux de neurones sont d'ailleurs fréquemment utilisés pour résoudre des problèmes de classification.

b) Réseaux avec couche cachée : La méthode la plus connue est basée sur la rétropropagation du gradient. Cette méthode, dite supervisée, consiste à présenter des exemples aux réseaux multicouches, puis à en propager l'erreur entre la sortie désirée et la sortie obtenue à travers le réseau afin de corriger tous les poids, même ceux des couches cachées. Un exemple est composé d'un vecteur d'entrées et de sorties désirées <x,d>. L'algorithme de rétropropagation du gradient est décrit sur Algorithme 2.

Algorithme 2 (rétropropagation du gradient) [74] **:**

1. Initialisation des poids à des valeurs aléatoires de faible grandeur,
2. Sélection aléatoire d'un exemple dans la base d'apprentissage <x,d>,
3. Propager l'entrée à travers le réseau,
4. Appliquer l'entrée x sur le réseau et calculer la sortie o de chaque couche,

5. Pour chaque sortie du réseau calculer le terme d'erreur:

$$\delta_k = (d_k - o_k)\sigma'(x_i) \quad \textbf{Equation 5}$$

6. Pour chaque couche cachée, calculer le terme d'erreur :

$$\delta_k = \sum_{k=1}^{N} w_{ki}.\sigma'(x_i) \quad \textbf{Equation 6}$$

7. Mettre à jour chaque poids synaptique du réseau :

$$w_{ij} = w_{ij} + \Delta w_{ij} \quad \textbf{Equation 7}$$

$$\Delta w_{ij} = \eta.\delta_j.x_{ij} \quad \textbf{Equation 8}$$

8. Retourner en 2. Tant que l'erreur est trop grande.
9. Fin

La méthode présentée sur Algorithme 1 est en fait une version simplifiée de la rétropropagation du gradient puisque cela est équivalent à appliquer l'Algorithme 2 sur un système sans couche cachée. L'efficacité et la rapidité de convergence de cette méthode ne sont plus à prouver, seulement deux problèmes persistent : lors de la mise à jour des poids de la couche cachée, l'ensemble du réseau est modifié, c'est-à-dire que le comportement du réseau sera modifié pour tous les exemples entrés, même s'il n'a été mis à jour que pour un seul exemple. Dans l'espace des entrées, la modification n'est pas locale. Ensuite, il n'est pas toujours aisé d'estimer le nombre de neurones dans la couche cachée. Ce deuxième problème est principalement dû à la difficulté de donner une signification physique à cette couche cachée. Nous avons choisi de présenter ci-après une autre famille de réseaux de neurones qui permet justement de s'affranchir des deux problèmes précédents.

2.3.2.1.3. Les réseaux à fonctions radiales

Les réseaux à fonctions radiales (RBF : Radial Basis Functions) permettent l'apprentissage de fonctions complexes; ils sont basés sur une structure différente de perceptrons. Ces réseaux sont généralement composés de deux couches : une couche cachée de neurones RBF (dont la fonction de transfert est donnée par l'Équation 9) et une couche de sortie constituée de perceptrons.

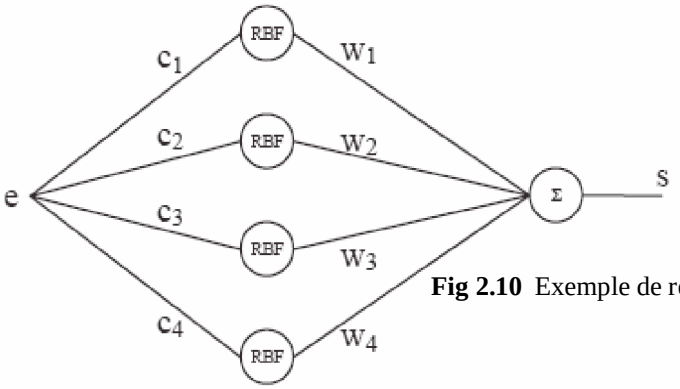


Fig 2.10 Exemple de réseaux RBF

L'exemple de la Fig 2.10 montre un réseau permettant l'approximation d'une fonction de *R* dans *R* (s=f(e)). Chaque fonction radiale possède un centre ci, la sortie globale est la somme des sorties des neurones de la couche cachée. La Fig 2.11 montre la courbe à estimer en continu, ainsi que la sortie de chaque neurone de la couche cachée en pointillé.

Chaque neurone a une influence locale sur la courbe estimée. Ce modèle peut être étendu à des fonctions de *Rn* dans *Rm*. La fonction de base, la plus couramment utilisée est donnée par l'Équation 9.

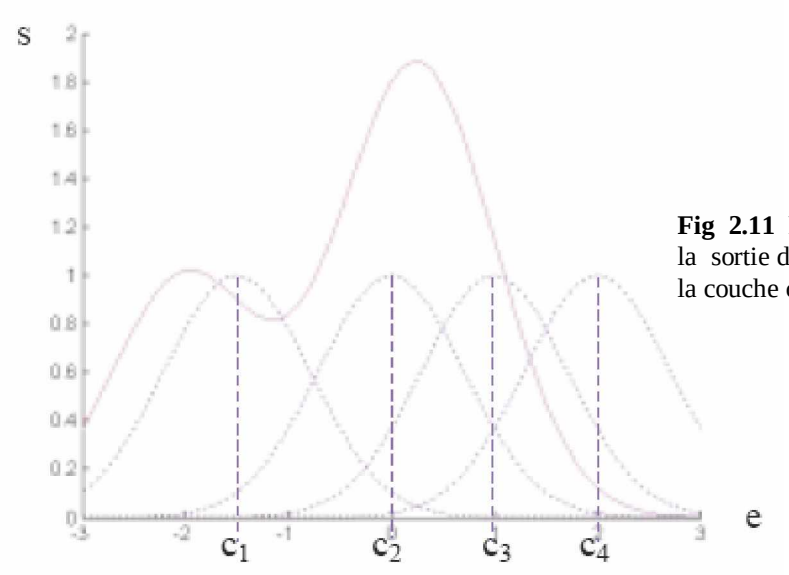


Fig 2.11 Fonction à estimer et la sortie de chaque neurone de la couche cachée

$$f(x)=e^{-(x-c_i)^2}$$

Equation 9 : Fonction de transfert d'un neurone de type RBF

Généralement, les centres sont fixés et seuls les wi sont modifiés. La règle de rétropropagation du gradient est utilisée sur la seule couche de sortie. Comme le réseau ne possède pas de couche cachée devant être mise à jour, la méthode présentée sur Algorithme 1 est parfaitement adaptée pour l'apprentissage. C'est pour ces mêmes raisons que la méthode est simple et rapide à implémenter. Toutefois, le choix de la position des centres et le nombre de neurones restent généralement arbitraires. Ils sont généralement répartis uniformément sur l'intervalle de définition de la fonction f = s(e). Il existe des méthodes de positionnement automatique des centres : ces méthodes consistent à déplacer les centres vers les points où l'erreur de modélisation est grande. L'accumulation de fonctions autour des zones complexes à modéliser améliore l'approximation, car les fonctions radiales ont une influence limitée autour de leurs centres.

Pour conclure cette section, on peut dire qu'il existe un grand nombre de méthodes permettant de réaliser l'apprentissage neuronal. De manière générale, ces méthodes sont supervisées, elles nécessitent des exemples pour réaliser l'apprentissage, et elles sont difficilement applicables dans les cas d'auto-apprentissage et d'adaptation où aucun exemple de comportement n'est connu a priori (avant toute expérience).

2.3.2.2. Apprentissage par renforcement

L'apprentissage par renforcement a pour objectif de maximiser la performance du système en renforçant les meilleures actions. L'agent obtient des informations de l'état de l'environnement (perceptions) et agit également sur l'environnement (actions) puis reçoit une estimation de sa performance : la récompense. Dans les algorithmes d'apprentissage par renforcement, cette récompense peut être immédiate ou retardée. Le schéma général est représenté sur Fig 2.12. L'agent connaît son état « S_t » dans l'environnement, il a la possibilité d'agir sur ce dernier « a_t », et il reçoit une récompense « r_t ». L'objectif de l'apprentissage par renforcement est d'associer à chaque état du système une action qui permet de maximiser la récompense.

On distingue deux types d'algorithmes; les stratégies « off policy » et les stratégies « on policy ». Dans le premier cas, l'apprentissage est divisé en deux phases :

- Une phase d'apprentissage, qui consiste à essayer des stratégies aléatoires afin de modéliser le système,
- Une phase d'application où l'agent n'exécute que les meilleures actions apprises dans la phase précédente.

Dans le cas des stratégies « on policy », l'agent applique la stratégie apprise au fur et à mesure de sa progression et de son apprentissage.

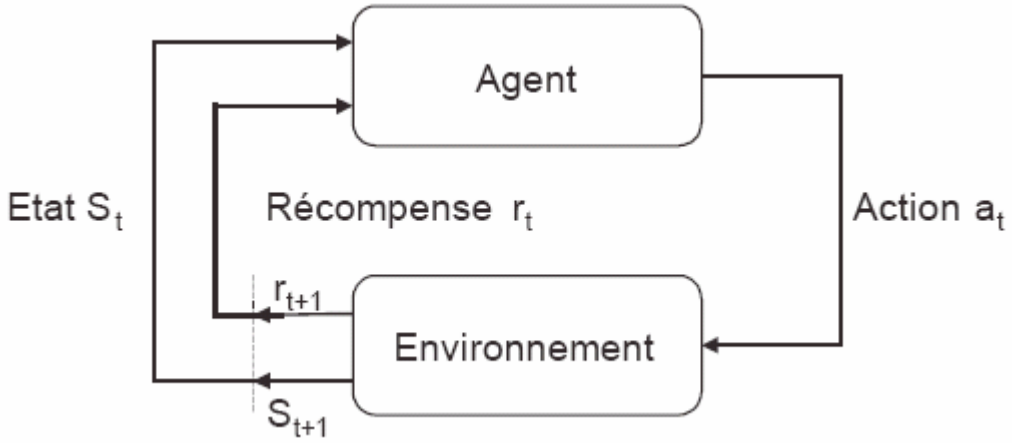


Fig 2.12 Système d'apprentissage par renforcement

2.3.2.2.1. Les processus markoviens

L'apprentissage par renforcement est basé sur l'apprentissage à temps discret des paramètres d'une chaîne de Markov. Les chaînes de Markov sont composées d'états et de transitions entre ces états. Prenons l'exemple d'un agent qui peut se déplacer dans un environnement discrétisé (voir Fig 2.13). L'environnement est décomposé en 16 cases représentant chacune un état du processus markovien. La position de départ est représentée par la case S et l'objectif est d'atteindre la case E. L'agent dispose de 4 actions possibles : se déplacer vers le nord, l'est, le sud ou l'ouest.

Dans ce cas, le système est déterministe, lorsque l'agent se trouve dans un état donné (S_n) et qu'il réalise une action donnée, l'état (S_{n+1}) suivant sera toujours le même. Dans un système non déterministe, la même action réalisée depuis le même état (S_n) peut aboutir à des états (S_{n+1}) différents. Dans ce cas, les transitions du processus markovien représentent la probabilité pour chaque état d'être atteint en fonction de l'action réalisée. Le schéma général de l'apprentissage

par renforcement consiste à associer à chaque action une fonction de cette probabilité et de la récompense associée à l'action. La récompense moyenne espérée pour chaque action peut ainsi être déterminée. De cette façon, il ne reste plus qu'à exécuter l'action possédant la meilleure récompense espérée dans la phase d'application de la stratégie.

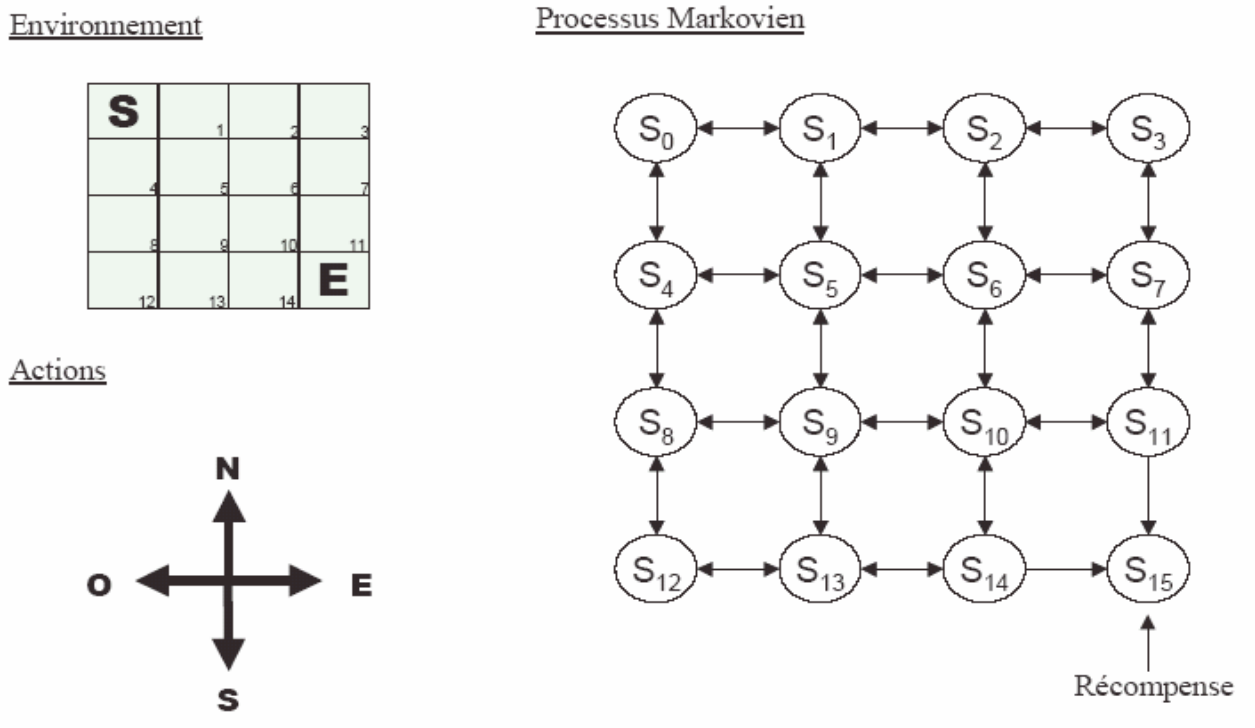


Fig 2.13 Exemple de processus markovien

2.3.2.2.2. Le Q-learning

Proposé en 1989 par Watkins [14], le Q-learning est probablement la méthode d'apprentissage par renforcement la plus étudiée actuellement. Les seules hypothèses de départ sont :

• Le système est modélisable par une chaîne de Markov à états finis,
• L'agent dispose d'un jeu d'actions discret.

Il n'est pas nécessaire de connaître les transitions entre les états, c'est le grand intérêt du Q-learning. Contrairement à la programmation dynamique, l'objectif n'est pas d'estimer la fonction d'évaluation de chaque état, mais celle de chaque action, en fonction de l'état courant.

Algorithme 4 (Q-learning)

1. Initialiser Q (s,a) de façon arbitraire, pour chaque état « s » et action « a »
2. Choisir une action « a »
3. récompense instantanée obtenue
4. s' nouvel état courant
5. Mettre Q(s,a) à jour selon l'équation suivante :

$$Q(s,a)=Q(s,a)+\alpha(r+\gamma\max_{a'}(Q(s',a'))-Q(s,a)) \qquad \textbf{Equation 10}$$

Avec: $\alpha=\frac{1}{1+visit(s,a)}$

γ coefficient d'amortissement

6. Incrémenter le nombre de visites de l'état (s,a) :

$$visit(s,a)=visit(s,a)+1$$ **Equation 11**

7. Retourner en 2.

8. Fin

La description du Q-learning est montrée sur l'Algorithme 4. Le Q-learning est basé sur l'estimation de la matrice Q(s,a), qui représente la récompense espérée en exécutant l'action « a » depuis l'état « s ». Le point 2 est volontairement resté imprécis. En effet, le Q-learning peut être utilisé comme une méthode « off-policy » ou « on-policy ». Dans le premier cas, il faudra alors choisir une action aléatoirement. Dans le deuxième cas, il faudra choisir une action en accord avec la stratégie apprise, c'est-à-dire l'action a qui maximise Q(s,a).

Le coefficient γ représente le coefficient d'amortissement. A la fin de l'apprentissage, la matrice Q peut être assimilée à un gradient qu'il suffit de remonter pour maximiser la récompense. La Figure (Fig 2.14) montre l'état de la matrice Q à la fin de l'apprentissage. On distingue clairement ce gradient pour chaque action. Le coefficient d'amortissement γ va modifier la pente de ce gradient. Un coefficient de 1 attribue la récompense espérée maximale à tous les états, alors qu'un coefficient de 0 attribue une récompense espérée de 0 à tous les états sauf aux états finaux. Sur la Figure (Fig 2.14), le coefficient γ a été arbitrairement fixé à 0,9.

Il a été démontré que pour des systèmes déterministes le Q-Learning permet d'atteindre la solution optimale après un temps d'apprentissage infini [80]. Il a d'ailleurs été démontré que la matrice Q converge vers l'espérance mathématique de la somme de la récompense instantanée et de la récompense espérée du meilleur état suivant possible (Équation 12). La même équation pouvant bien sûr s'écrire pour l'état suivant (Q(St+1,a')), on en déduit que remonter le gradient guidera l'agent vers la solution optimale.

$$Q(s_t,a)\approx E[r_{instantannée}+\gamma.\max_{a'}(Q(s_{t+1},a'))]$$

Equation 12 : espérance mathématique de la récompense espérée pour l'action à exécuter depuis l'état St

Pour cette section on peut dire que l'apprentissage par renforcement est un outil performant pour réaliser différents types d'apprentissage. Il existe de nombreuses autres méthodes dérivées de la programmation dynamique* et du Q-Learning, comme la différence temporelle (TD), Sarsa et Monte-Carlo. Toutes ces méthodes sont basées sur les processus markoviens et demandent de discrétiser l'environnement en états et les consignes sur les actionneurs en commandes échantillonnées. Pour parfaitement modéliser un problème réel appliqué à la robotique, la taille de la matrice Q serait proportionnelle au nombre de combinaisons états/actions possibles. Il est évident qu'il est impossible de stocker autant de données sur un système embarqué. De plus l'apprentissage serait extrêmement long, puisqu'il demanderait à l'agent de visiter au moins une fois chaque combinaison état/action possible.

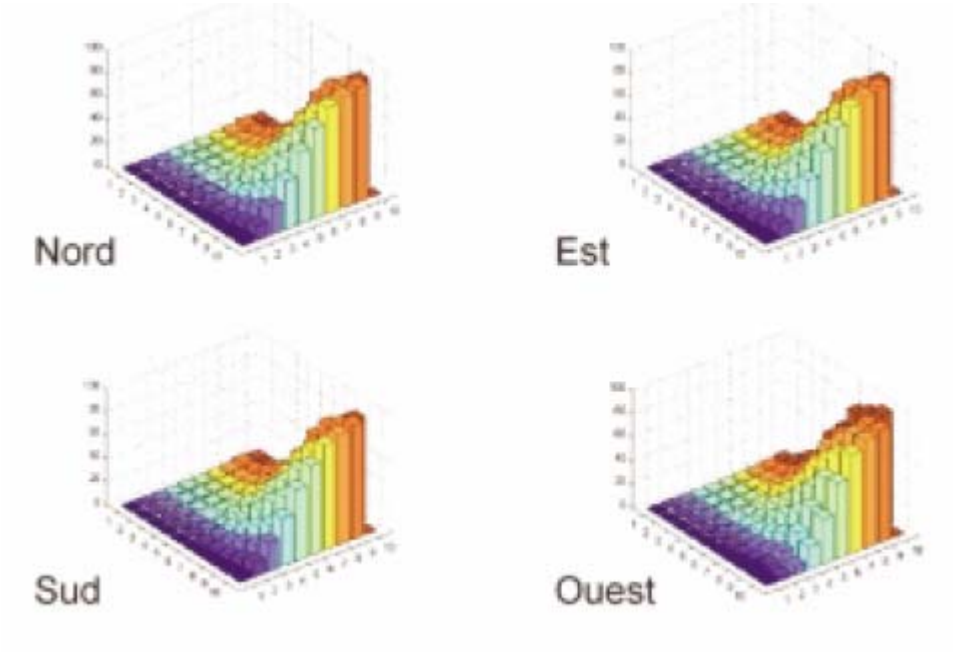


Fig 2.14 Matrice Q pour chacune des actions

Même si ces méthodes semblent mal indiquées pour l'apprentissage de comportements réactifs continus, elles peuvent être utilisées pour gérer les couches supérieures de l'architecture. En 2000, Maja Mataric et Dany Goldberg proposent une architecture multi-agents basée sur le principe des AMMs (Augmented Markovians Models) [13]. Il s'agit là d'une structure basée sur les processus markoviens, qui consiste à construire sa propre bibliothèque d'états au fur et à mesure de l'apprentissage. La technique permet de combiner deux états identiques et d'éviter de stocker des valeurs pour des états qui ne seront jamais rencontrés.

2.3.2.3. Algorithmes génétiques (AG)

Les algorithmes évolutionnistes sont inspirés de l'étude du vivant et plus précisément de l'évolution darwinienne. Le principe général [11] consiste à disposer d'une population d'individus possédant chacun une chaîne chromosomique dans laquelle est codé son comportement. La méthode nécessite de connaître une estimation de la performance de chaque agent (fitness). Dans la première génération, les chaînes chromosomiques sont choisies aléatoirement. Les expériences sont divisées en générations, lorsque les performances de tous les individus de la première génération sont connues des croisements sont réalisés entre les individus afin de créer la génération suivante. L'Algorithme 5 décrit le principe général de la méthode.

Algorithme 5 (Principe évolutionniste général d'après Mitchell [24])

1. Initialisation de la population P1: remplir aléatoirement la chaîne chromosomique de chaque individu hi.
2. Evaluer chaque individu de la population courante F(hi).
3. Créer la génération suivante en sélectionnant statistiquement les géniteurs. La probabilité qu'un individu soit utilisé pour générer la population suivante est donnée par :

$$\Pr(hi) = \frac{F(h_i)}{\sum_{j=1}^{Nbindividus} F(h_j)} \qquad \textbf{Equation 13}$$

4. Réaliser les croisements sur les chaînes chromosomiques.

5. Réaliser les mutations.
6. Retourner en 2.
7. Fin.

Prenons pour exemple le problème du juste chiffre : nous disposons de quatre nombres et le but est de s'approcher le plus près possible d'un cinquième nombre en réalisant des opérations arithmétiques avec les 4 premiers. Nous supposerons que les opérateurs sont : l'addition, la multiplication, la soustraction et la division. Nous émettrons également comme hypothèse que l'opération arithmétique doit être constituée de trois opérateurs sans priorité (il n'y a pas de parenthèse dans l'opération). Par exemple, si les chiffres sont : 24 25 30 et 7 et que le résultat doit être 142, une bonne solution sera : (((24 * 25 = 600) / 30) = 20)* 7 = 140. Si un algorithme génétique est utilisé pour résoudre ce problème, la chaîne chromosomique pourra être constituée de 14 bits : 8 bits pour les 4 nombres (voir Tab 1) et 6 bits pour les 3 opérateurs (voir Tab 2). Un exemple de chaîne chromosomique est donné sur le Tableau (Tab3).

Codage	Nombres
00	24
01	25
10	30
11	7

Tab 1 : Codage des chiffres dans la chaîne chromosomique

Codage	Opérateurs
00	+
01	-
10	*
11	/

Tab 2 : Codage des opérateurs dans la chaîne chromosomique

0	1	0	1	0	0	0	0	1	0	1	0	1	1
25		-		24		+		30		*		7	

Tab 3 : Exemple de chaîne chromosomique pour le problème du juste chiffre

2.3.2.3.1. Les opérateurs génétiques

Les opérateurs génétiques sont utilisés pour générer la population suivante. On distingue deux types d'opérateurs : les opérateurs de croisement, qui réalisent une opération sur deux individus et les opérateurs de mutation, qui n'opèrent que sur un seul individu.

a) Les croisements : L'opérateur de croisement produit deux nouvelles chaînes à partir de deux chaînes initiales. Le masque de croisement est un mot binaire de même longueur que les chaînes génétiques. Ce masque contribue à déterminer lequel des deux parents sera géniteur de chaque bit de la génération suivante. Trois grandes familles de croisements sont distinguées dans la littérature (voir Fig 2.15) :

- ✓ **Les croisements simple point :** Les deux chaînes initiales vont être divisées en deux. La première partie de la première chaîne sera associée à la seconde partie de la seconde

chaîne et inversement, deux nouveaux individus sont ainsi obtenus résultant d'un croisement entre les deux chaînes initiales.

- ✓ **Les croisements doubles points :** Le principe est assez proche des croisements simples points, à cette différence qu'il y a deux points de séparation des chaînes, la chaîne initiale est divisée en 3 parties et la combinaison de ces 3 parties permet d'obtenir deux nouvelles chaînes.
- ✓ **Les croisements uniformes :** Le masque de croisement est choisi aléatoirement, chaque bit peut être choisi indépendamment du reste de la chaîne.

Illustration 1 : Croisement simple point

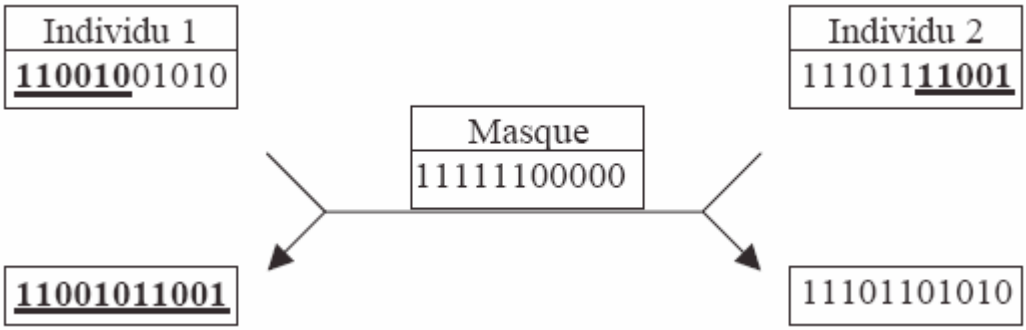


Illustration 2 : Croisement double points

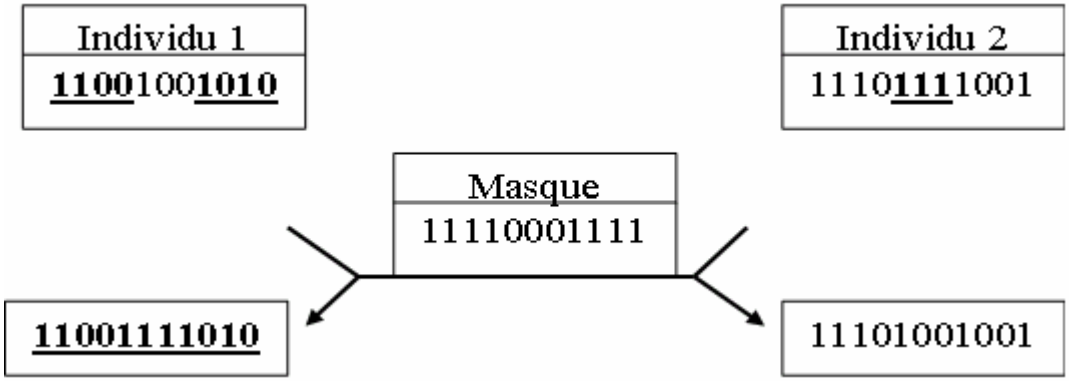


Illustration 3 : Croisement uniforme

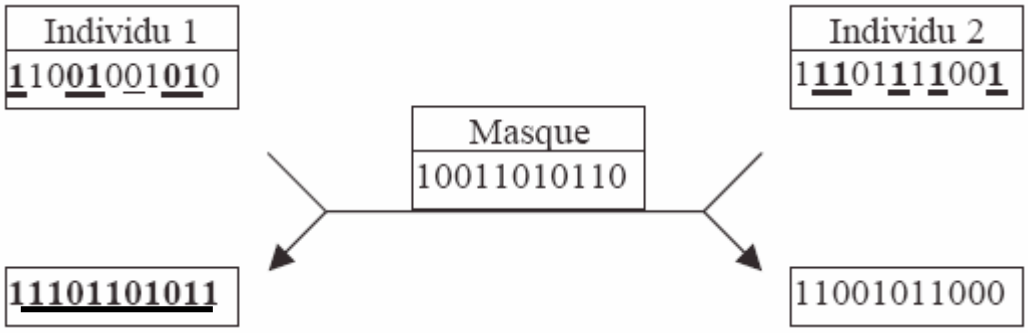


Fig 2.15 Opérateurs de Croisement sur des chaînes binaires

Il est assez courant d'utiliser la chaîne chromosomique pour y coder des valeurs binaires, entières ou réelles. De ce fait, les bits de la chaîne n'ont pas tous la même importance ; un changement sur un bit de poids fort n'a pas la même influence qu'une modification sur un bit de poids faible. Pour que le

sens des croisements soit justifié, des opérateurs numériques sont parfois utilisés : on réalise par exemple une moyenne pondérée des valeurs numériques des deux parents.

Individu géniteur 1

0	1	**0**	**1**	0	0	0	**0**	1	0	**1**	**0**	**1**	1
25		-		24		+		30		*		7	

Individu géniteur 2

1	**0**	1	1	**0**	**1**	**1**	0	**1**	**1**	1	0	1	**1**
30		/		25		*		7		*		7	

Individu de la génération suivante

0	0	0	1	0	1	1	0	1	1	1	0	1	1
24		-		25		*		7		*		7	

Fig 2.16 Exemple de croisement uniforme pour le problème du juste chiffre

b) Les mutations : Les mutations effectuent une modification à partir d'une seule chaîne initiale. Les mutations consistent à complémenter l'un des bits de la chaîne initiale (voir Fig 2.15). Les mutations permettent notamment d'obtenir des individus avec de nouvelles propriétés qui n'auraient pas été accessibles en ne réalisant que des croisements à partir de la population initiale.

Fig 2.17 Exemple de mutation

Comme pour l'opérateur de croisement, lorsque la chaîne représente des valeurs numériques, une mutation sur un bit de poids fort n'a pas la même influence que la même opération sur un bit de poids faible. Des opérateurs de mutation numériques sont parfois utilisés, le principe est de réaliser un tirage aléatoire centré sur la valeur numérique codée dans la chaîne initiale.

Si l'on reprend l'exemple du juste chiffre présenté précédemment, un exemple de croisement est présenté sur la Figure (Fig 2.17). L'apprentissage a été réalisé sur une population de 20 individus. L'évolution du meilleur individu est montrée sur la Figure (Fig 2.18). On distingue que la solution optimale et la stabilité du système est atteinte après une quarantaine de générations. En réalisant l'apprentissage plusieurs fois, on trouve 4 résultats équivalents (L'objectif était d'atteindre le chiffre 142) :

- ((7 * 25) – 7) – 25 = 143
- ((25 * 7) – 7) – 25 = 143
- ((7 * 25) – 25) – 7 = 143
- ((25 * 7) – 25) – 7 = 143

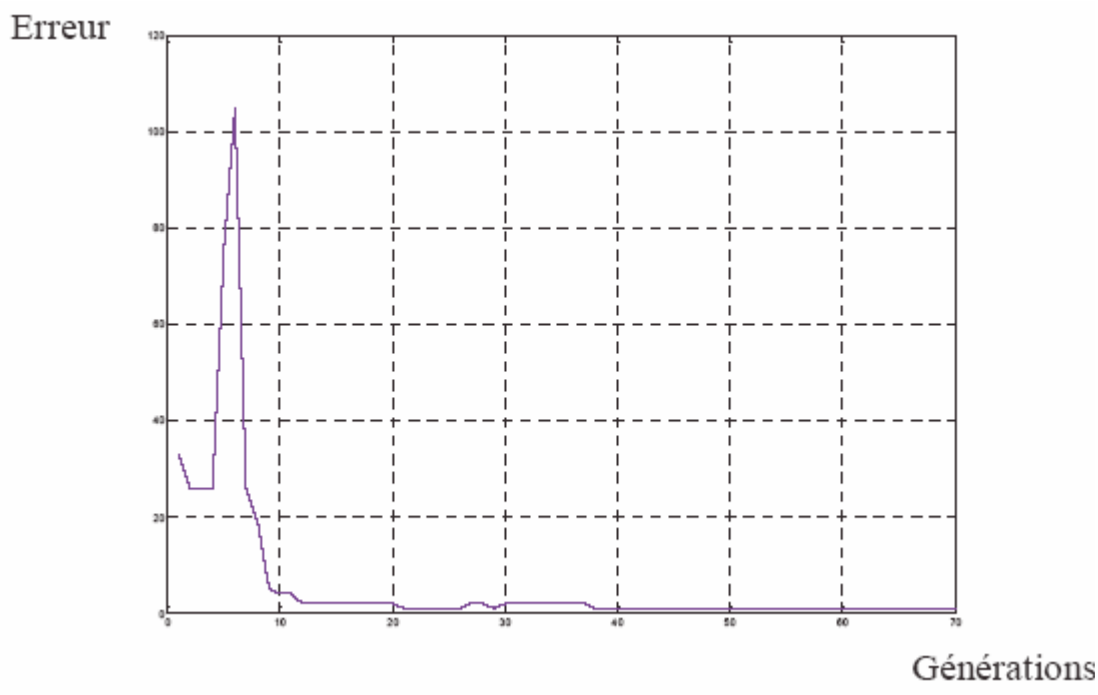


Fig 2.17 Evolution du meilleur individu au fil des générations

Les algorithmes évolutionnistes semblent particulièrement bien indiqués pour l'apprentissage des systèmes multi-agents. En premier lieu, il faut disposer d'une population d'individus, ce qui est le cas dans tout système multi-robots. Ensuite, la possibilité de coder toutes sortes de comportements dans la chaîne chromosomique ouvre un grand nombre de possibilités. Toutefois, la méthode est supervisée : après chaque expérience, l'ensemble des chaînes chromosomiques doit être rassemblé pour construire la génération suivante. Ce type d'apprentissage demande de centraliser les données. De plus, les agents ne peuvent pas toujours interrompre leur tâche courante pour réaliser les croisements.

2.3.3. Estimation de la performance

L'estimation de la performance est un paramètre essentiel de l'apprentissage. Le principal intérêt de l'apprentissage est de laisser l'agent trouver par lui-même la meilleure stratégie pour maximiser la récompense. Si la récompense est trop contrainte, l'apprentissage ne présente aucun avantage par rapport aux méthodes traditionnelles. Prenons l'exemple extrême du Morpion (Tic-Tac-Toe en anglais) où deux adversaires s'affrontent en disposant des croix et des cercles sur une grille carrée de 3 cases de côté. Le premier qui aligne trois symboles remporte la partie. Si le joueur est récompensé de la façon suivante :

- +1 si c'est un bon coup,
- 0 sinon.

La récompense suppose de connaître à priori les bons coups. De ce point de vue, l'apprentissage n'a aucun lieu d'être utilisé, autant programmer tous les bons coups. En revanche, si l'agent est récompensé de la manière suivante :

- +1 en cas de victoire,
- -1 en cas de défaite,
- 0 en cas de match nul.

Dans ce cas, l'agent devra trouver les meilleurs coups à jouer pour maximiser sa récompense qui ne lui sera accordée qu'à la fin de la partie. Cet exemple peut paraître évident, mais des récompenses

trop contraintes sont parfois utilisées et elles verrouillent l'apprentissage dans une stratégie qui n'est pas nécessairement optimale. Dans [10] Madani explique « Pour notre application la fonction de renforcement choisie permet de délivrer un signal +0.5 lorsque le robot choisit une bonne action (récompense) et -0.5 pour une mauvaise (punition) ». Si les bonnes et les mauvaises actions sont connues par avance, l'apprentissage n'a aucun intérêt. Il s'agit ici d'un ralliement de cible avec évitement d'obstacles; une récompense moins contraignante aurait pu être :

- +1 lorsque la cible est atteinte,
- -1 en cas de collision avec un obstacle,
- 0 sinon.

Un autre exemple intéressant est l'apprentissage d'un comportement d'évitement d'obstacles [9]. Il est important de préciser que l'expérience se déroulait dans un environnement très contraint. La récompense est donnée par l'Équation 14.

$$R_i = V.(1-\sqrt{\Delta v}).(1-i)$$

Equation 14 : Récompense instantanée attribuée à un robot mobile pour une tâche de navigation sans Collision (D'après Mondala [9])

Où :

- V est la vitesse moyenne de rotation des deux roues,
- v est la différence signée de la vitesse des deux roues,
- i est une valeur proportionnelle à la quantité d'obstacles proches du robot.

Cette estimation de la performance se divise en trois termes: le premier tend à maximiser la vitesse du robot, le second, à récompenser les trajectoires rectilignes et le dernier à minimiser les collisions. Francesco Mondada [9] a expliqué que l'environnement était tellement contraint qu'une des meilleures stratégies trouvées par le robot était de tourner en rond sur place, maximisant ainsi la vitesse moyenne des deux roues et minimisant le nombre de collisions. Il a donc choisi d'adapter la récompense à l'environnement et par conséquent l'apprentissage ne sera valide que dans des cas où l'environnement sera très contraint. D'après Lucidarme [25] les différents types d'estimations de la performance sont classés en 4 catégories :

- **L'erreur :** Utilisée dans les réseaux de neurones, l'erreur donne une information proportionnelle à la différence entre la sortie désirée et la sortie obtenue. De plus lorsque la sortie du système est un vecteur, l'erreur est également un vecteur de même dimension. Cette estimation de la performance est la plus contraignante car elle exige un modèle de comportement connu et fait tendre l'apprentissage à copier ce modèle.
- **Les récompenses instantanées :** Les récompenses instantanées donnent une estimation de la performance correspondant à la stratégie courante. Contrairement à l'erreur, même si la sortie du système est un vecteur, la récompense instantanée est un paramètre unidimensionnel. La récompense présentée par l'Équation 14 est un exemple de récompense instantanée.
- **Le coût :** La plupart des méthodes d'apprentissage sont présentées comme cherchant à maximiser la récompense. L'objectif de la tâche peut aussi être de minimiser un critère, par exemple une consommation d'énergie ou un temps de déplacement. Dans ces cas, on parle de coût et non de récompense. Mathématiquement, minimiser une fonction de coût peut toujours se ramener à maximiser une récompense.
- **Les récompenses retardées :** Une récompense retardée est attribuée à la fin d'une séquence d'actions. Elle estime la performance de l'ensemble de ces actions. Un exemple de récompense retardée est le Tic-Tac-Toe présenté ci-dessus. Le joueur est récompensé à la fin de la partie pour l'ensemble de ses coups. Ce type de récompense est parmi les moins contraignants.

2.4. Conclusion

Relation entre Adaptation, apprentissage et auto-organisation :

Par la modification de son organisation interne, un système auto-organisateur, va produire une fonction différente. Ce changement est dû à la pression environnementale, mais souvent ne se fait pas de façon immédiate. Le système s'adapte progressivement à son environnement de manière non supervisée. Le système et son environnement sont sous influence mutuelle [79]. Cette adaptation progressive correspond (entre autres) à un mode d'apprentissage comme on peut l'observer dans les réseaux de neurones formels. Mais ici, contrairement aux méthodes classiques d'apprentissage, l'adaptation du système n'est pas fondée sur une évaluation globale, comme par la sélection des meilleurs individus lors de la phase de sélection d'un algorithme génétique ou bien par la connaissance de l'erreur entre la sortie effective et la sortie théorique dans un réseau de neurones. Concernant la conception de systèmes adaptatifs, et donc apprenants. Ceci implique une possible décomposition récursive du système si une ou plusieurs de ses parties doivent être dotées de capacités d'apprentissage. En effet, une partie apprenante pourra produire la fonction souhaitée grâce à un sous-système auto-organisateur.

Avantages et inconvénients des techniques d'apprentissage :

On a présenté trois techniques d'apprentissage possédant chacune sa propre spécification :

- Les réseaux de neurones peuvent apprendre à mimer des fonctions mathématiques à partir d'une base d'exemples,
- L'apprentissage par renforcement permet d'apprendre des stratégies sans intervention extérieure, ni même d'exemple,
- enfin les algorithmes évolutionnistes peuvent faire évoluer des comportements afin d'en améliorer leurs performances.

Mais, Les réseaux de neurones demandent des exemples pour pouvoir fonctionner, ils sont incapables de générer d'eux-mêmes une stratégie permettant de résoudre un problème ou un conflit. Si l'apprentissage par renforcement permet d'apprendre sans exemple une stratégie optimale, cette technique demande de discrétiser l'univers en états et actions échantillonnés. Or, une quantité importante de données doit pouvoir être stockée, ce qui peut alors être disproportionné au regard des capacités de stockage actuelles. Les algorithmes évolutionnistes semblent bien appropriés à notre problématique (qu'on verra dans le chapitre 4), n'empêche que c'est une technique supervisée. Leur point faible réside dans le rassemblement des séquences génétiques pour pouvoir réaliser les croisements et les mutations.

Dans le prochain chapitre, nous allons introduire les systèmes multi-agents (en focalisant sur les agents réactifs), et la robotique collective ainsi que la relation qui peut lier ces deux disciplines.

2.5. Références

Références bibliographiques

[1] H. Abdi, « Les Réseaux de neurones, Sciences et technologies de la connaissance », PUG, Grenoble, 1994.

[2] P. Almquist, « Type of Service in the Internet Protocol Suite », RFC 1349 July 1992.

[3] G.Apostolopoulos, S. Kamat, R. Guérin, A. Orda, T. Przygienda, and D. Williams, « QoS Routing Mechanisms and OSPF extensions », RFC 2676 August 1999.

[4] T. Artieres, P. Gallinari, « Neural models for extracting speaker characteristics in speech modelization systems », In EuroSpeech, pp 2263-2266, 1993.

[5] F. Baker, S. Brim, T. Li, F. Kastenholz, S.Jagannath, J. K. Renwick, « IP Precedence in Differentiated Services Using the Assured Service », IETF Internet Draft, 1998.

[6] Y. Bennani, P. Gallinari, « Connectionist approaches for automatic speaker recognition », ESCA workshop on speaker recognition, Martini, Suisse 1998

[7] Y. Bernet, « Requirements of Diff-serv Boundary Routers », IETF Internet Draft, 1998.

[8] J. A. Boyan and M. L. Littman, « Packet Routing in Dynamically Changing Networks: A Reinforcement Learning Approach. », In Cowan, Tesauro and Alspector (eds), Advances in Neural Information Processing Systems 6, 1994.

[9] D. Floreano et F. Mondada, « Autonomic creation of an autonomous agent : Genetic evolution of a neural-network driven robot », Simulation of Adaptive Behavior, from animals to animats 3, 1994.

[10] T. Madani, A. Benallegue and N.K. M'SIRDI, « Apprentissage par renforcement pour la Navigation d'un Robot Mobile dans des Environnements Inconnus », Journée des Jeunes Chercheurs en Robotique, 2002.

[11] D. E. Goldberg, « Genetic Algorithms in Search, Optimization and Machine Learning », Addison-Wesley, 1989.

[12] D. Goldberg et M.J. Mataric, « Coordinating mobile robot group behaviour using a model of interaction dynamics », Proceedings, Autonomous agents Seattle, WA, p. 100-107, 1999.

[13] D. Goldberg and M.J. Mataric, « Robust Behavior-Based Control for Distributed Multi-Robot Collection Tasks », USC Institute for Robotics and Intelligent Systems Technical Report IRIS-00-387, 2000.

[14] C. Watkins, « Learning from delayed rewards », thèse de doctorat, Cambridge University, 1989.

[15] D.E. Rumelhart and J.L. Mc Clelland, « Parallel Distributed Processing », The MIT Press, vol. 1 et 2, Cambridge, 1986.

[16] M. Minsky and S. Papert, « Perceptron », The MIT Press, Cambridge, 1969.

[17] J.J. Hopfield, « Neural Networks and Physical Systems with Emergent Collective Computational Abilities », Proceedings of the National Academy of Sciences, p. 460-464, 1982.

[18] D.O. Hebb, « The Organisation of Behaviour », Wiley, New York, 1949

[19] F. Rosenblatt, « The Perceptron : a Probabilistic Model for Information Storage and Organisation in the Brain », Psychological Review, p. 386- 408, 1958.

[20] W.S. Mc Cullogh and W. Pitts , « A logical calculus of the ideas immanent in nervous activity », Bulletin of Mathematical Biophysics 5, p. 115-133, 1943.

[21] C. Fillol, « Apprentissage et systémique : une perspective intégrée », Laboratoire CREPA, Université Paris IX Dauphine, Conférence 8 de l'AIMS. Normandie. Vallée de Seine, 2004.

[22] J. Quinqueton, « Apprentissages » , LIRMM et INRIA, 2005.

[23] P. Gallinari, « Apprentissage Numerique et Statistique », master d'informatique specialité IAD, Université Paris 6, 2004/2005.

[24] T. Mitchell, « Machine learning », McGraw Hill, 1997

[25] C. Bishop «Neural Networks for pattern recognition », Clarendon Press. 1995
[26] R.Axelrod « The Evolution of Cooperation », New York : Basic Books, 1984.
[27] P. Bieber, V. Camps, B. Carpuat, P. Cros, M.P. Gleizes, P. Glize, A. Machonin, Jo Pezet, « Accès intelligent à l'information répartie », Rapport final ARCADIA - Contrat CTI CNET Nº94CND46, 1998.
[28] J.Bollen , F.Heylighen, « Algorithms for the self-organisation of distributed, multi-user networks. Possible application to the future World Wide Web », Cybernetics and Systems'96, 1996.
[29] V. Camps, M.P. Gleizes., « Une technique multi-agent pour rechercher des informations réparties », 5èmes JFIAD&SMA, La Colle sur Loup, Éditions Hermès, pp. 29 – 46, 1997.
[30] V. Camps, M.P. Gleizes, « Cooperative and mobile agents to find relevant information in a distributed resources network, Workshop on Artificial Intelligence-based tools to help W3 users », Fifth international conference on WW*W*, Paris, 1996.
[31] V.Camps, M.P. Gleizes, « Principes et évaluation d'une méthode d'auto-organisation », 3iéme journées francophones sur l'Intelligence Artificielle Distribuée & Systèmes Multi-Agents, 1995.
[32] J.*P.* Dupuy, « Ordres et désordres : Enquête sur un nouveau paradigme », Paris : Edition Seuil, 1982.
[33] J.P. Dupuy, P. Dumonchel, « l'a*uto*-organisation. De la physique au politique », Colloque de Cerisy, Paris : Edition Seuil, 1983.
[34] J.P. Dupuy, « Aux origines des sciences cognitives », Paris : La découverte, 1994.
[35] J. Ferber , « Les systèmes multi-agents :Vers une intelligence collective ». InterEditions, 1995.
[36] S. Forrest, « Emergent computation : Self-organizing, Collective, and cooperative phenomena in Natural and Artificial Computing networks », Special issue of Physica D, MIT Press, North-Holland, 1991.
[37] I.Gasser, T.A.Ishida, « Dynamic Organisational Architecture for Adaptive Problem Solving ». *AAAI91*, p. 185-190, 1991.
[38] L. Gasser, C. Braganza, N. Herman., « Implementing Distributed Artificial Intelligence Systems Using MACE », *In* Proceedings of the Third IEEE Conference on Artificial Intelligence Application*s*, 1987.
[39] C.V.Goldman, J.S.Rosenschein, « Emergent Coordination through the Use of Cooperative State-Changing Rule », *AAAI*, 1994.
[40] F. Heylighen, « Evolution, Selfishness and Cooperation », 1992.
[41] T.Hogg, B.A.Huberman, « Better than the best: The power of cooperation », Lectures notes in complex systems, Addison Wesley, 1992.
[42] N.R. Jennings, « Cooperation in industrial mutli-agent systems », World Scientific Series in Computer Science *Vol 43 - World Scientific Computer Publishin*g, CO. Pte. Ltd, 1994.
[43] S. Kalinka, N. Jennings., « Socially Responsible Decision Making by Autonomous Agents », Fifth Int. Colloq. On Cognitive Science, Donostia, Spain, 1997.
[44] J. Marcia, « Auto-organisation : émergence de structures », *Journées du PRC GDR IA : les systèmes multi-agents*, 1996.
[45] J. Mataric Maja, « Interaction and Intelligent Behavior », PHD of Philosophy Massachussetts Institute of Technology May 1994.
[46] H.R. Maturana , F.J. Varela, « Autopoiesis and cognition, The realization of the living *» D.* Reidel Publishing Company, North Holland, 1980.
[47] C. Piquemal-Baluard, V. Camps, M.P. Gleizes, P. Glize, « Cooperative agents to improve adaptivity of multi-agent systems », Intelligent Agent Workshop of the British Computer Society, *In* Specialist Interest Group on Expert Systems & Representation and Reasoning, Oxford, 1995.
[48] M. Sekaran, S. Sen, «To help or not to help », Seventeenth Annual Cognitive Sciences Conference, Pitsburg Pennsylvannia, 1995.
[49] S. Sen, M. Sekaran, «Using reciprocity to adapt to others », IJCAI, 1995.
[50] M. J Shaw. and A. B. Whinston, « Learning And Adaptation in DAI », in Gasser and Huhns. 1989.

[51] S. Sian « Adaptation Based On Cooperative Learning In Mas », in Proceedings of the second workshop on Modelling Autonomous Agents in Multi-Agent World, *Edito*rs Y. Demazeau & J-P. Müller, Editions North Holland, 1991.
[52] L. Steels, « The spontaneous Self-organization of an Adaptive Language, Machine Intelligence 15 », Oxford University Press , Oxford, Muggleton S. (Ed.) March 29, 1996.
[53] H. Von Foerster, « Principles of self-organization », Pergamon Press, 1962.
[54] G. Weiß, « Learning To Coordinate Actions In Multi-Agent Systems », *in* Proceedings of the International Joint Conference on Artificial Intelligen*ce*, 1993.
[55] G. Weiß, « Adaptation and Learning in Multi-Agent Systems », In Lecture Notes in Artificial Intelligence 1042. Springer Verlag, 1996.
[56] M. Wooldridge, N.R. Jennings, « A theory of cooperative problem solving », Intelligent Agents Workshop of the British Computer Society Specialist Interest Group on expert Systems & the Representation and reasoning Special Interest Group, Edited by N.S. Taylor and J.L. Lealon, 1995.
[57] P. De Buyl, N. Tabti, A. Wagner, « Auto-organisation chez les insectes sociaux », Printemps des sciences, 2003.
[58] J. Millor, M. Pharm-Delegue, J. L. Deneubourg, S. Camazine, « Self-organized defensive behavior » in honeybees,1999
[59] J. Deneubourg. et al., « Plan d'organisation et population dans les sociétés d'insectes », p. 141-155, dans Prigogine I. (dir.), L'homme devant l'incertain, Paris, Odile Jacob, 2001.
[60] M. Dorigo, G. DI Caro, « Ant Algorithms for Discrete Optimization », Artificial Life, vol. 5, n°3, p. 137-172, 1999.
[61] Valerie CAMPS, « Vers une théorie de l'auto-organisation dans les systèmes multi-agents basée sur la coopération : application à la recherche d'information dans un système d'information répartie ». Thèse de l'université de Paul Sabatier, 1998.
[62] C. Fillol, « Apprentissage et systémique : une perspective intégrée », Laboratoire CREPA - Université Paris IX Dauphine, 2004.
[63] Û. Cem « self-organisation in large population of mobile robots », Master of sciences in electrical engineering, Blacksburg, Virginia, 1993.
[64] J. F. Varela, E. Thompson, E. Rosch, « L'inspiration corporelle de l'esprit » Edition du seuil, 1993.
[65] F. J. Varela, « Autonomie et connaissance : essai sur le vivant » , Edition du seuil, 1988.
[66] I. Toru, L. Gasser, M. Yokoo. « Organization Self-Design of Distributed Production systems ». IEEE Transactions on Knowledge and Data Engineering, vol. 4, No. 2, April 1992.
[67] Y. So, H. Edmund Durfee. « An Organizational Self-Design Model for organizational Change », In Working Notes of the AAAI-93 Workshop on AI and Theories of Groups And Organizations, July 1993.
[68] C. Piquemal-Baluard, P. Glize, « Des aptitudes non cognitivistes d'agents pour l'auto-organisation », Actes de la Journée PRC-GDR Intelligence Artificielle sur le thème des Systèmes Multi-Agents, Toulouse, 2 février 1996.
[69] V. Camps et M-P. Gleizes. « Principes et évaluation d'une méthode d'auto-organisation », Journées francophones IAD&SMA, Saint Baldoph, pp. 337-348. Mars 1995.
[70] V. Camps, M-P. Gleizes, « Cooperative and mobile agents to find relevant information in a distributed ressources network », In WWW5 AI Workshop Fifth International World Web Conference, Paris, France, May 6-10, 1996.
[71] J-P. Georgé, « Résolution de problèmes par émergence : Étude d'un environnement de Programmation Émergente ». Thèse de l'université de Toulouse III Paul Sabatier, 2004.
[72] G. picard, « Systémes Multi-Agents Coopératifs Méthodologie de développement de systèmes multi-agents adaptatifs et conception de logiciels à fonctionnalité émergente ». Thèse de Paul Sabatier, Toulouse III, 2004.
[73] J-P. Vacher, « Un système adaptatif par agents avec utilisation des algorithmes génétiques multi-objectifs : Application a l'ordonnancement d'atelier de type job-shop N _M » Thèse l'université du Havre, 2000.
[74] P. Lucidarme, « Apprentissage et adaptation pour des ensembles de robots réactifs

coopérants », de l'université de Montpellier II, 2003.
[75] A. Cardon, Z. Guessoum, « Systèmes multi-agents adaptatifs », Laboratoire d'informatique de Paris VI (LIP6), 2000.
[76] Z. Guessoum and J-P.Briot, « From Active Objects to Autonomous Agents, IEEE Concurrency ». 7(3): 68-76, November 1999.
[77] Z. Guessoum, A. Cardon, A. Ramdani, « Toward self-adaptive multi-agent systems », Maamaw'97, Valencia, 1999.
[78] L. Hugues, « Apprentissage de comportements pour un robot autonome ». Thèse Pierre et Marie Curie – Paris 6, 2002.
[79] V. H. Foerster, « On self-organising systems and their environments, in: Self-Organising Systems », M.C. Yovits and S. Cameron (eds.), Pergamon Press, London, pp. 30-50, 1960.
[80] C. Szepsvari, « The assymptotic convergence-Rate of Q-learning », in Proc. Neural Information Processing Systems, 1997.

Références sur le Web

http://pespmc1.vub.ac.be/papers/SelfOrganWWW.html
http://arti.vub.ac.be/steels/publications.html
http://www.info.unicaen.fr/~serge/3wia/workshop/papers/paper30.html
http ://www.rennard.org/
http ://tecfa.unige.ch/tecfa
www.alliance21.org/fr/proposals/summaries/global.htm
http://www.thetransitioner.org/wiki/tiki-download_file.php?fileId=6
http://www.austms.org.au/Jobs/Library4.html
http://dieoff.org/page95.htm
http://pespmc1.vub.ac.be/SUBOPTIM.html
www.thetransitioner.org/wikifr/tiki-read_article.php?articleId=4
www.vision-nest.com/cbw/Quest.html
http://www.kcl.ac.uk/neuronet/
www.agro-montpellier.fr/cnam-lr/statnet
fr.wikipedia.org/wiki/Apprentissage
www.vie-publique.fr/politiques-publiques/apprentissage-enseignement-professionnel/rub714/
psychobiologie.ouvaton.org/glossaire/txt-p06.20-01-glossaire.htm
www.knowup.com/pages/ressources/lexique.cfm/
www.legissimo.com/lexique.htm
membres.lycos.fr/bfsavate/lexique.htm
www.er.uqam.ca/nobel/r33554/lexique.html
www.erudium.polymtl.ca/html-fra/glossaire.php
http://www.ncrg.aston.ac.uk/netlab/
http://www.cs.waikato.ac.nz/~ml/weka/
http://www.brint.com/Systems.htm
http://www.informatik.umu.se/~rwhit/ReadingRoom.html
http://www.lyber-eclat.net/lyber/friedman/utopies.html

Autres références

- CD UNIVERSALIS 2005
- CD ENCARTA 2005
- Techniques de l'ingénieur 2005 version papier et CD
- Ervin Laszlo, « La cohérence du réel : Evolution cœur du savoir », Bordas, 1989.
- Murray Guell-Mann, prix nobel en physique, « Le quark et le Jaguar, voyage au cœur du

simple et du complexe », Albin Mihel, 1995.
- Humberto Maturana, Francisco Varela, « L'arbre de la connaissance », Addison-Wesley, 1992.
- Mark Ludwig, « Mutation d'un virus », Addison-Wsley, 1994.
- David E. Goldberg « Algorithmes Génétiques », Addison-Wesley, 1991.
- Eric Davalo, Patrick Naïm, « Des Réseaux de Neurones », Eyrolles, 1993.
- J-F. Jodouin, « Les réseaux neuromimétiques : modèles et applications », Hermes, 1993.
- J.L Austin, « Quand dire c'est faire ». Editions du Seuil, Paris, 1991.
- J.R. Searle, « L'intentionnalité », essai de philosophie des états mentaux. Les Editions de Minuit.

Les Systèmes Multi-agents (SMA) et la Robotique Collective (RC) ont bon nombre de points similaires. Cela dit, la robotique collective puise dans le monde de la réalité alors que les SMA puisent dans des mondes virtuelles crées de toute pièce pour des objectifs bien définis dans des ordinateurs. Le transfert des problèmes rencontrés dans la robotique collective vers le monde virtuelle des ordinateurs via la simulation multi-agents, constitue un passage (qu'on ose dire) obligé pour la plupart des chercheurs en RC. Ceci est motivé en partie par la grande similarité des deux domaines, et qu'il est plus facile et plus économique d'expérimenter ses idées par le biais d'un programme de simulation.

Dans ce chapitre nous allons tout d'abord présenter les systèmes multi-agents qui penchent plutôt vers l'intelligence artificielle distribuée. Par la suite on pénètre dans le domaine de la robotique collective, qui penche plutôt vers la vie artificielle, et on parachève ce chapitre en présentant les points en communs qui lient ces deux disciplines.

3.1. Système Multi-Agents (SMA)

Les SMA se situent à l'intersection entre Intelligence Artificielle Distribuée (IAD) et la Vie Artificielle (VA). Les premières applications de la vie artificielle sont apparues avec l'avènement de l'informatique (voir chapitre 1). L'IAD est apparue vers la fin des années 1970. Hewitt, confronté à un problème de résolution de théorèmes, proposa une solution en 1977 avec des entités actives appelées « acteurs » et considéra la résolution comme la confrontation de leurs « points de vue ». En 1980, Erman et bien d'autres chercheurs ont enrichit le concept d'échange d'informations en élaborant l'idée du tableau noir (blackboard) avec le projet « HERSAY II ». Notons que le tableau noir est une zone de travail commune, destinée à la transition d'informations entre les différents agents. Chacun peut venir le consulter à sa guise, y prélever et y déposer des objets qu'il peut également modifier. Le tableau structure la modélisation du domaine d'application comme l'espace des hypothèses et des solutions.

C'est à partir de ces premières réalisations que nous avons vu apparaître l'idée des SMA. Leur utilisation est aujourd'hui distribuée selon les deux axes fondateurs de l'IAD et de la VA : d'une part dans la résolution des problèmes dits « distribués », tant physiquement que fonctionnellement et nécessitant de fait, la participation cohérente de plusieurs systèmes (spécialisé chacun dans un aspect du problème). D'autre part, dans l'étude des dynamiques « globales » résultant des interactions entre agents (travaux amenant des éléments de réponse à l'émergence « spontanée » de structures organisées comparables à celles observées dans le règne animal).

Il est important de mentionner à ce stade que les systèmes multi-agents mettent en oeuvre un ensemble de concepts et de techniques permettant à des logiciels hétérogènes, ou à des parties de logiciels (nommés agents) de coopérer suivant des modes complexes d'interaction. D'après Ferber [1] le développement de la technologie des SMA s'est fait motivé par quatre raisons principales :

- La première raison est la limite de l'intelligence artificielle classique sur le plan de la structuration et de l'organisation des connaissances. La difficulté qu'il y a à traduire un ensemble d'expertises sous une forme unifiée a amené les chercheurs à développer ce que l'on a d'abord appelé des systèmes multi-experts, c'est-à-dire des systèmes mettant en jeu plusieurs bases de connaissances plus ou moins coordonnées. Ce faisant, on a pu constater que le problème de la coopération entre plusieurs bases de connaissances se révélait un enjeu crucial qui dépassait de loin le problème de la multi-expertise.
- La deuxième raison trouve son origine dans la nécessité de trouver des techniques de modélisation et de simulation performantes dans le domaine des sciences du vivant au sens large du terme. L'évolution des écosystèmes habités, notamment, montre qu'il est difficile de rendre compte de leur évolution par un ensemble d'équations différentielles. Une approche dans laquelle les individus sont directement représentés sous forme d'entités informatiques semble une voie prometteuse et a contribué à l'essor du domaine.

- La troisième raison provient de la robotique. Le développement de la miniaturisation en électronique permet de concevoir des robots qui disposent d'une certaine autonomie quant à la gestion de leur énergie et à leur capacité de décision. On a pu alors montrer qu'un ensemble de petits robots ne disposant que de capacités élémentaires de décision et d'intelligence pouvait facilement rivaliser avec les performances d'un seul robot « intelligent », nécessairement plus lourd et plus difficile à gérer. Le problème alors revient, ici encore, à faire coopérer ces entités de manière à ce qu'elles assurent les fonctions désirées.
- la quatrième et dernière raison est issue du développement de l'informatique et des systèmes distribués en particulier. Avec la généralisation des réseaux et des ordinateurs parallèles, il devient de plus en plus important de pouvoir faire coopérer plusieurs composants logiciels au sein d'environnements hétérogènes et distribués. Le problème ne réside plus dans le contenu des programmes, qui peuvent avoir des fonctionnalités diverses, mais dans leur capacité à collaborer avec d'autres programmes pour la réalisation d'un objectif commun.

Notons que dans le cadre de ce projet, c'est essentiellement la deuxième et la troisième raison qui nous intéressent tout particulièrement.

3.1.1. Définitions

Comme l'élément fondamental dans l'approche multi-agents est l'agent, il est évident qu'on doit commencer tout d'abord par le définir. La plupart des concepts fondamentaux, sont généralement vagues (relativement). La notion d'agent n'échappe pas à cette règle : on peut distinguer plusieurs manières de concevoir et de comprendre la notion d'agent. Chacune de ces notions renvoie à un courant de recherche particulier dans le domaine de ce qui touche au paradigme « agent ». Cependant, tous les SMA peuvent être ramenés à un ensemble d'entités physiques ou virtuelles possédant un ensemble de caractéristiques communes (voir 3.1.1.2.)

3.1.1.1. Définition d'un Agent

Si on se réfère à la définition du dictionnaire : du latin « agens » : celui qui agit. « Un agent est une personne chargée des affaires et des intérêts d'un individu, d'un groupe ou d'un pays, pour le compte desquels elle agit ». Le terme « agir » est défini par le petit Larousse comme : faire quelque chose, s'occuper, produire un effet. De ceci on peut retirer deux aspects fondamentaux :

- Un agent accomplit quelque chose,
- Un agent agit à la demande d'un autre agent ou d'un utilisateur.

Dans le même sens d'idées Caglayan et Harrison [2] définissent un agent comme suit : « Un Agent logiciel est une entité informatique qui réalise de manière autonome des tâches pour un utilisateur ». On peut conclure de ces définitions qu'un agent informatique (plus exactement une application agent) devra faire quelque chose pour le compte d'une personne ou d'une application.

3.1.1.2. Caractéristiques d'un Agent

D'après Giot et all. [48] les caractéristiques d'un agent peuvent êtres classés comme suit :

a) Caractéristiques fondamentales

L'autonomie : L'agent travaille sans intervention directe, jusqu'à un point défini par l'utilisateur. L'autonomie d'un agent peut aller du simple lancement d'une sauvegarde la nuit, à la négociation du prix d'un produit choisi par son mandataire.

L'interactivité : L'agent doit pouvoir exercer des actions sur son environnement et réciproquement. L'interactivité d'un agent de sauvegarde sera sa possibilité de sauvegarder un

certain nombre de fichiers par une action exercée au travers du système d'exploitation. Ce dernier pourra informer l'agent d'une permission non accordée (exemple : un fichier dont l'utilisateur n'a pas de droit de lecture).

La réactivité : L'agent percevra son environnement (qui pourra être l'utilisateur à travers une interface graphique, un ensemble d'agents, etc.) en répondant dans les temps impartis aux changements qui surviennent sur cet environnement (exemple : un agent de sauvegarde pourra faire sa tâche à une heure donnée).

b) Caractéristiques d'intelligence

La capacité d'apprendre : Le dictionnaire définit « apprendre » (du latin appréhendée, saisir) comme le fait « d'acquérir la connaissance, l'information, l'habitude ». Un agent aura la capacité d'apprendre s'il sait acquérir de la connaissance, de l'information ou des habitudes.

> **Exemple :** Un agent grâce à sa capacité de réactivité, doit se déclencher à une certaine heure. Mais l'utilisateur l'arrête dans sa tâche (qui pourrait être une sauvegarde) car il ralentit le travail de l'utilisateur. L'agent va apprendre à différer son exécution pour éviter de gêner l'utilisateur.

La capacité sociale : les agents interagissent avec les autre agents (et éventuellement avec des êtres humains) grâce à des langages de communication entre agents. Cette capacité sera la base pour la coopération entre les agents.

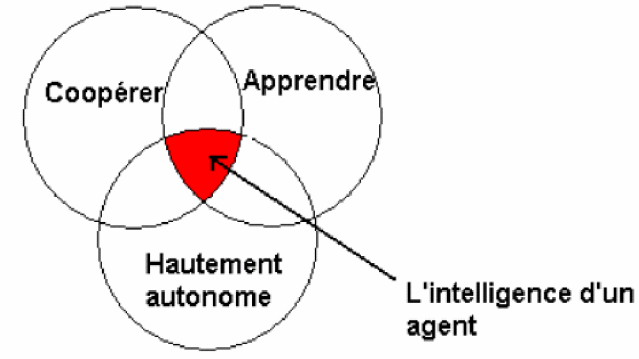
> **Exemple 1 :** Notre agent de sauvegarde rencontre un autre agent de sauvegarde sur un réseau. Ces deux agents peuvent se mettre d'accord pour se partager le travail afin que la tâche soit achevée plus vite.

> **Exemple 2 :** L'agent ne peut exécuter sa tâche pour des raisons techniques diverses (problèmes de droits d'accès, par exemple). L'agent va communiquer ce fait à l'utilisateur, en lui apportant la démarche à suivre dans un langage naturel compréhensible. L'utilisateur pourra ensuite, en utilisant le même langage, indiquer à l'agent comment résoudre le problème.

c) Haut degré d'autonomie

L'agent fonctionne sans intervention directe humaine ou autre (autonomie), en plus il a une forme de contrôle sur ses actions et sur leur état interne (haut degré).

Fig 3.1 Schéma de la représentation de l'intelligence d'un agent



Une intelligence plus poussée : La pro-activité est une caractéristique qui est intégrée dans les agents hautement intelligents (très rares actuellement). Les agents n'agissent pas seulement en réponse à leur environnement, mais ils sont capables d'avoir un comportement guidé par un but, en ayant la possibilité de prendre l'initiative. **Exemple :** Un agent réseau peut décider, de lui-même, pendant un temps où il est non actif, de faire des statistiques sur les routeurs pour améliorer son activité future. Cette notion de pro-activité peut être décrite comme l'ensemble des qualités suivantes : croyances, désirs, buts, intentions, choix, décisions, engagements, conventions et obligations

Exemple : Explication de la pro-activité avec la « vie d'un humain », ici Ali :

• **Connaissances** : Ali connaît le fait que les humains sont mortels
• **Croyances** : Ali a pris son parapluie parce qu'il croit qu'il va pleuvoir
• **Désirs**, **buts** : Ali désire avoir son doctorat
• **Intentions** : Ali a l'intention de travailler dur pour avoir sa thèse
• **Choix, décisions** : Ali a décidé de faire une thèse
• **Engagements** : Ali ne va pas s'arrêter de travailler avant d'avoir fini sa thèse
• **Conventions** : si, par hasard, Ali décide d'abandonner sa thèse, il va le dire à son professeur
• **Obligations** : Ali doit travailler pour entretenir sa famille

d) Mobilité

Nous devons prendre en compte deux types de mobilités :

- **Mobilité relative** (ou par requêtes) : Dans ce cas il n'y a pas un réel déplacement de l'agent. Celui-ci lance une succession de requêtes à destination de différents serveurs. C'est le cas des agents de recherche, tel que « Copernic », qui interroge différents moteurs de recherche afin de fournir à son utilisateur une synthèse des résultats.

- **Mobilité réelle** (de l'agent) : Le processus agent se déplace d'un serveur à un autre, sur le réseau. Le code de l'objet ainsi que ses données, sont transportés sur une nouvelle machine où il continu son exécution.

  Exemple : Un agent de sauvegarde peut se déplacer sur plusieurs serveurs dans le but de faire des restaurations de fichiers.

e) Coopération

Un système multi-agents (SMA) se distingue d'une collection d'agents indépendants par le fait que les agents interagissent en vue de réaliser conjointement une tâche ou d'atteindre conjointement un but particulier. Les agents peuvent interagir en communiquant directement entre eux ou par l'intermédiaire d'un autre agent ou même en agissant sur leur environnement. Chaque agent peut être caractérisé par trois dimensions: ses buts, ses capacités à réaliser certaines tâches et les ressources dont il dispose. Les interactions des agents d'un SMA sont motivées par l'interdépendance des agents selon ces trois dimensions: leurs buts peuvent être compatibles ou non; un agent peut désirer des ressources que les autres possèdent; un agent X peut disposer d'une capacité nécessaire à un agent Y pour l'accomplissement d'un des plans d'action de Y.

Ferber [1] remarque que les chercheurs ont développé différents points de vue sur la coopération. Ainsi, on peut considérer la coopération comme une attitude adoptée par les agents qui décident de travailler ensemble, ou bien on peut adopter le point de vue d'un observateur extérieur au système multi-agents, qui interprète à posteriori les comportements des agents pour les qualifier de coopératifs ou non suivant des critères préétablis, tels que l'interdépendance des actions ou le nombre de communications effectuées. Dans le cas de la coopération vue comme une attitude intentionnelle, les agents s'engagent dans une action après avoir identifié et adopté un but commun. C'est par exemple le cas d'un groupe de personnes acceptant de travailler dans une entreprise (le but étant par exemple d'augmenter les bénéfices de l'entreprise). Des chercheurs comme Galliers [7] et Conte [11] ont souligné l'importance de l'adoption d'un but commun pour ce type de coopération qu'ils considèrent comme un élément essentiel de l'activité sociale.

f) Agent situé

Le domaine des agents situés est né vers la fin des années 80, à partir de travaux issus de la robotique mobile. Dans ces travaux, la prise en compte d'un environnement complexe et dynamique, rend le problème de sélection d'action difficile à résoudre avec des approches

déterminantes classiques [2]. La mise en situation de l'agent a donc été proposée, pour permettre des comportements plus réactifs, face à la dynamique de l'environnement. C'est dans ce contexte que R. Brooks et son équipe [20, 27], puis d'autres [18, 24, 28, 19, 26] ont proposé l'approche comportementale inspirée de la biologie pour la robotique mobile, comme moyen de mise en oeuvre d'une intelligence « sans représentation » [21].

Cette approche s'est ensuite étendue à d'autres problématiques, comme la résolution distribuée de problèmes [19] ou la simulation multi-agents, qui a émergé depuis 1998 comme domaine de recherche à part entière.

L'approche comportementale tient son inspiration de la biologie. L'éthologie dans ce domaine a inspiré beaucoup de travaux, et a donné naissance au domaine de l'intelligence en essaim (swarm intelligence) [29]. Cette approche couvre des travaux de différentes natures comme le développement de métaheuristiques pour les problèmes d'optimisation [24], la robotique collective [44, 45, 46] ou encore des applications dans le domaine des réseaux Internet et réseaux des télécommunications [19], etc.

Si l'on reprend les idées de Vera, et Clancy et autres [42, 43], un agent est situé s'il utilise des représentations externes plutôt qu'internes, rien ne vaux l'expérimentation réelle à travers la boucle perception-action. Néanmoins il n'y a pas, sous cet angle, d'interdiction stricte de la notion de symbole ou de toute autre représentation interne, mais rien n'est fixé à l'avance. Nous allons définir un agent situé comme n'ayant pas à priori de système de représentation interne spécifié par le concepteur (puisqu'on souhaiterai se passer de l'intervention de ce dernier). Par contre, on n'exclu pas que de telles représentations apparaissent d'elles-mêmes au sein de l'agent (imaginez un agent qui découvre par exemple la notion de pomme ou de nombre).

On retrouve dans la littérature d'autres descriptions des caractéristiques d'un agent, dont les propriétés en commun sont généralement les suivants :

1) Capacité de perception d'un environnement.
2) Capacité d'agir dans cet environnement.
3) Capacité de se représenter une partie de cet environnement (voir pas du tout).
4) Capacité de communiquer directement avec d'autres agents.
5) Possession de ressources propres.
6) Possession d'une dynamique orientée par un ensemble de tendances (ces tendances peuvent être des objectifs individuels ou collectifs, ou bien l'optimisation d'une fonction de satisfaction ou de survie).
7) Possession de compétences qu'il peut éventuellement offrir.
8) Capacité de se reproduire (facultative).

En somme, on peut dire qu'un agent est doté d'un comportement tendant à satisfaire au mieux ses buts, au vu de la perception et de la représentation de l'environnement dont il dispose, des ressources et des compétences qu'il possède et de la teneur des informations échangées. Etant mus par des objectifs, les agents sont dits autonomes dans le sens où le créateur du système ne pilote pas leur comportement. C'est l'agent qui décide de ses actions par rapport à un éventail de possibilités données. L'agent peut-être ainsi vu comme une sorte « d'organisme vivant » dont le comportement vise à la satisfaction de ses besoins propres en fonction de tout ce dont il peut disposer.

3.1.1.3. Système multi-agents

D'après Jack Ferber [1], Un système multi-agents est :

1) Un environnement **E**, soit un espace disposant généralement d'une métrique.

2) Un ensemble d'objets **O**. Ils sont dits « situés », c'est à dire qu'à tout objet **O** on peut associer une position dans **E**. Ils sont dits passifs, à savoir qu'ils peuvent être perçus, manipulés, créés ou détruits par les agents, qui sont les entités actives du système.
3) Un ensemble **A** d'agents. Les agents sont des objets particuliers aux propriétés spécifiées plus haut.
4) Un ensemble de relations **R** qui unissent les objets entre eux (et donc également les objets aux agents et les agents entre eux)
5) Un ensemble d'opérations **Op** que les agents peuvent appliquer aux objets.
6) Des opérateurs chargés de répercuter l'application de ses opérations sur l'environnement. Ses opérateurs sont appelés « lois de l'univers » car ils modifient l'environnement consécutivement aux actions portées sur lui.

Il existe un cas particulier de SMA dans lequel A = O (il n'y a pas d'objet passif) et où E=Ø. Cette variété de système est appelée « SMA purement communiquant » car les relations R définissent un réseau entre agents dont les actions sur E se réduisent à la communication. Cette structure particulière est utilisée par l'intelligence artificielle distribuée, principalement pour la collaboration de modules amenés à résoudre un problème ou à élaborer une expertise. Cette organisation mime le mode de travail d'un organisme social, comme par exemple : une administration.

Lorsque E est non vide, on parlera d'agents situés. Rappelons que, les SMA sont étudiés en raison de leurs remarquables capacités d'auto-organisation.

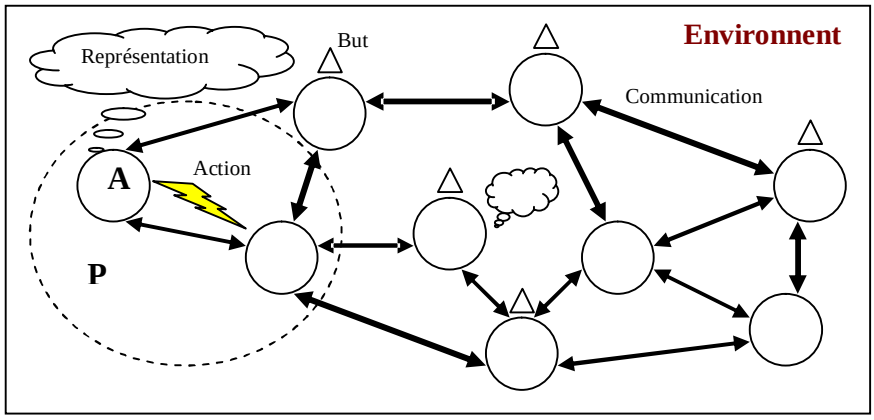


Fig 3. 2 Un système multi-agents est un ensemble d'agents qui agissent et interagissent dans le même environnement. Chaque agent à sa propre perception **P** de l'environnement, une représentation de ce qu'il connaît de cet environnement, et un but à atteindre.

Concept de l'interaction : Une des principales propriétés de l'agent dans un SMA est celle d'interagir avec les autres agents. Ces interactions sont généralement définies comme toute forme d'action exécutée au sein du système d'agents et qui a pour effet de modifier le comportement d'un autre agent.

Elles permettent aux agents de participer à atteindre un but global (satisfaire un besoin). Cette participation permet au système d'évoluer vers un de ses objectifs et d'avoir un comportement intelligent indépendamment du degré de complexité des agents qui le composent. En général, les interactions sont mises en oeuvre par un transfert d'informations entre agents ou entre l'environnement et les agents, soit par perception, soit par communication :

- par **la perception**, les agents ont connaissance d'un changement de comportement d'un tiers au travers du milieu.
- par **la communication**, un agent fait un acte délibéré de transfert d'informations vers un ou plusieurs autres agents.

L'interaction peut être décomposée en trois phases non nécessairement séquentielles :

- la réception d'informations ou la perception d'un changement,

- le raisonnement sur le comportement des autres agents à partir des informations acquises,
- une émission de message(s) ou plusieurs actions (plan d'actions) modifiant l'environnement. Cette phase est le résultat d'un raisonnement de l'agent sur son propre savoir-faire et celui des autres agents.

Le degré de complexité des connaissances nécessaires pour traiter les interactions dépend des capacités cognitives (de raisonnement) de l'agent et du fait que l'agent ait connaissance ou non de l'objectif du système global.

En effet, un agent qui poursuit un objectif individuel au sein du système, comme c'est le cas pour les agents dits réactifs, ne focalise pas son énergie pour interagir avec les autres, même s'il y est amené. Par contre, un agent qui participe à la satisfaction du but global du système tout en poursuivant un objectif individuel, va passer une partie de son temps à coopérer ou à se coordonner avec les autres agents. Pour cela, il doit posséder des connaissances sociales qui modélisent ses croyances sur les autres agents.

3.1.2. Dualité Agent / Organisation

D'après Ferber [49], La kénétique* se veut à la fois la science et la technique des organisations artificielles. Possédant comme outil la conception de SMA, cette discipline détient l'avantage expérimental du contrôle de la totalité des paramètres définissant le système.

Or, si le concepteur peut intervenir à sa guise sur l'ensemble des variables introduites, il n'a toutefois pas accès à un déterminisme absolu concernant l'évolution du système. Bien au contraire, les configurations évolutives des SMA sont soumises au principe des phénomènes chaotiques. Cela implique, que la moindre modification dans les conditions initiales, et l'introduction de la plus insignifiante variable aléatoire peut conduire le même système à des états finaux diamétralement opposés. En effet, ces infimes variations sont très rapidement amplifiées par les interactions entre les agents, interdisant ainsi la prévision précise des états ultérieurs du système [51].

Les SMA ne sont pas pour autant soumis à aucune loi. Le travail portant sur les SMA se situe au centre d'une irréductible dualité [51] : Agent / Organisation. Toute organisation est le fruit d'une interaction entre agents, et leurs comportements individuels se trouvent en retours modifiés par les contraintes qu'imposent les structures organisatrices créées. Le rapport existant entre agent et organisation est donc une réalité mouvante. Et mis à part les cas où l'organisation est intégralement décrite par le concepteur du système, la répartition du travail, la coordination des tâches, et l'assignation de rôles spécifiques résultent de la seule activité de ses membres. On assiste alors à l'apparition de propriétés émergentes, absentes de la programmation de départ. Il nous faut ici préciser les différences architecturales distinguant les deux courants de pensée relatifs à l'application des SMA. Ces différences de conception concernent la nature algorithmique des agents utilisés. Ainsi, on trouve d'un côté l'école cognitive (pour l'intelligence artificielle distribuée), et de l'autre côté, celle qui nous intéresse le plus dans le travail présenté ici : l'école réactive (pour la vie artificielle).

3.1.2.1. Agent cognitif

La conception de SMA cognitifs trouve son origine dans la volonté de faire communiquer et coopérer des systèmes experts classiques. Dans ce cas, le SMA sera composé d'un petit nombre d'agents « intelligents ». Chacun possédant une base de connaissance comprenant l'ensemble des informations et des savoir-faire nécessaires à la réalisation de sa tâche ainsi qu'à la gestion des interactions avec son environnement et les autres agents. Les agents sont parfois dits « intentionnels », c'est-à-dire qu'ils possèdent des buts et des plans explicites leur permettant de les atteindre. On parlera alors de conduites téléonomiques, par opposition aux conduites réflexives (lesquelles caractérisent plus fréquemment les agents réactifs) dont les buts sont définis en

réaction aux stimuli du milieu. Dans ce cadre cognitif, planifié au maximum, le SMA se conduira à la façon d'un petit groupe d'individus, régit par des règles sociales prédéfinies (par exemple en cas de situations conflictuelles, les agents seront amenés à négocier). Les chercheurs dans ce domaine s'appuient fréquemment sur des travaux sociologiques des organisations et des groupes restreints. Nous n'évoquerons pas davantage ce type de SMA, dont l'extrême planification stratégique laisse peu de place à l'émergence de nouveaux comportements.

Il faut noter que la structure présentée dans la figure (Fig 3.3) caractérise les agents cognitifs en général [53], et que la composante « interaction » avec d'autres agents n'est pas souvent présente dans le cas des agents réactifs. Ce qu'il faut retenir est que cette structure définit aussi d'autres types d'agents, notamment ceux avec états et avec buts que nous décrivons un peu plus loin.

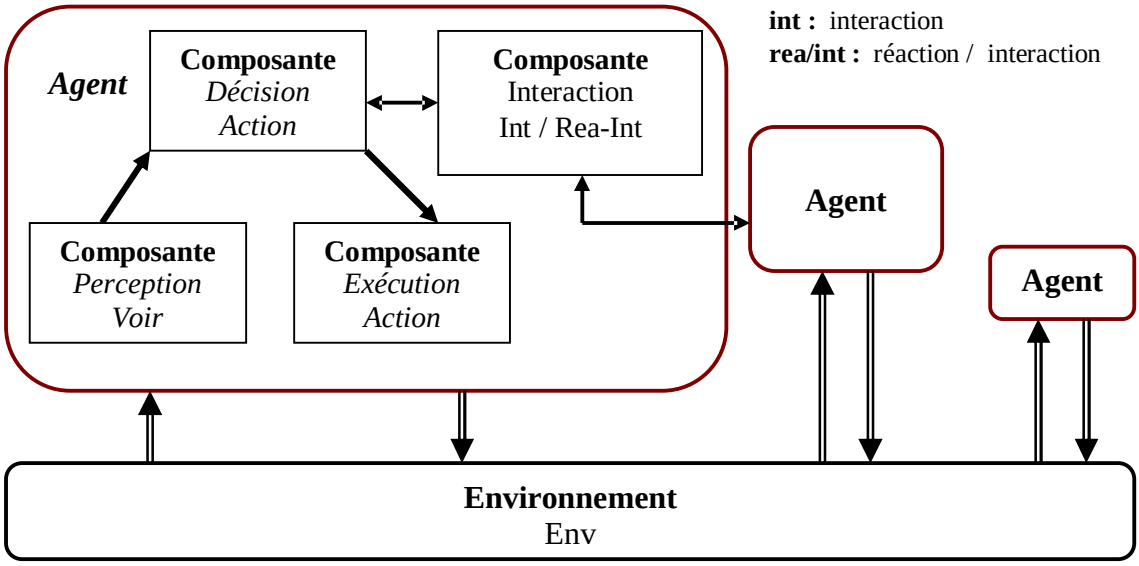


Fig 3.3 Structure d'un agent cognitif qui interagit dans un environnement Multi-agents

Dans le schéma de la figure (Fig 3.3) ***int*** est la fonction qui représente la décision de l'agent concernant son interaction avec un autre agent (par exemple un message) en fonction d'une perception qu'il a de l'environnement, et ***rea-int*** est la fonction qui décrit la réaction de l'agent, notamment l'action décidée par l'agent, à une interaction transmise par un autre agent.

Agent avec Etats : Les agents avec états, comme leur nom l'indique, maintiennent un état interne qui mémorise la séquence des perceptions de l'agent et, dans certains cas, les actions effectuées par l'agent. L'agent a besoin de maintenir cet état interne pour faire la distinction entre des états de l'environnement qui lui semblent identiques selon la perception qu'il en a, mais qui sont néanmoins différents.

Agent avec Buts : Connaître l'état de l'environnement et la séquence des perceptions mémorisées dans l'état interne n'est pas toujours suffisant pour décider quelle est la meilleure action à effectuer à un moment donné. Pour fonctionner d'une manière efficace, l'agent doit essayer d'obtenir le maximum de performance et doit choisir en conséquence ses actions. Il y a plusieurs façons de définir la mesure de performance. Une première solution est d'indiquer à l'agent ce qu'il doit faire en toute circonstance pour avoir du succès. C'est le cas des agents réactifs, où le concepteur définit les règles condition-action pour chaque cas de perception sur l'environnement. Toutefois, puisque les agents cognitifs sont autonomes et proactifs, on aimerait n'avoir à dire à notre agent que ce qu'il faut faire, sans avoir à lui dire exactement comment le faire dans chaque situation, dans ce cas, une deuxième solution pour définir la mesure de

performance de l'agent, est de fixer des **états-but** ou états désirables : but **: E** à **{0, 1}** est une fonction qui a la valeur 1 pour les états-but et 0 pour les autres.

Notons que les interactions dans un système d'agents cognitifs se traduisent par des schémas de conversations construits à partir des actes de communication et des prises de décision locales des agents (issus du caractère autonome et du comportement interne des agents) ou sous la forme de protocoles d'interaction pré-définis qui concernent le potentiel de tous les agents du système, et que l'une des architectures les plus utilisées pour ce genre d'agent est l'architecture BDI pour Belief (croyance), Desire (Désir), Intention.

3.1.2.2. Agent réactif

L'école réactive prétend qu'un système globalement intelligent ne nécessite nullement l'intelligence individuelle de ses agents. De simples mécanismes de réaction aux évènements, ne prenant en compte ni explicitation des buts (conduites le plus souvent « réflexives », par opposition aux conduites générales des agents cognitifs), ni mécanismes de planification, peuvent en effet résoudre des problèmes qualifiés de complexes (voir Fig 3.4) [53].

C'est dans ce deuxième type de SMA que l'on assiste à l'émergence de processus d'auto-organisation parfois spectaculaires, l'un des exemples les plus frappants étant celui d'une fourmilière artificielle : Alors que toutes les fourmis se situent sur un même pied d'égalité, et en l'absence d'une quelconque autorité stricte, les actions des agents se coordonnent de telle manière que la colonie survie et se développe. L'entité collective est en mesure de résoudre des problèmes complexes tels que la recherche de nourriture, les soins à donner aux oeufs, et aux larves, la construction de nid, la reproduction, etc. Ainsi, les organisations émergentes concernent quasi-exclusivement les SMA réactifs, car elles sont caractérisées par l'absence de structures organisationnelles prédéfinies (ce qui est rarement le cas des agents cognitifs). Dans une société d'agents réactifs, un processus d'auto-organisation permet à celle-ci d'évoluer et de s'adapter efficacement aux modifications de son environnement et aux besoins du groupe. Bien qu'il existe une certaine forme d'émergence dans les organisations prédéfinies, celle-ci est toujours contrôlée par des schémas de communication bien établis, et les organisations qui en résultent sont toujours décrites par des structures très précises.

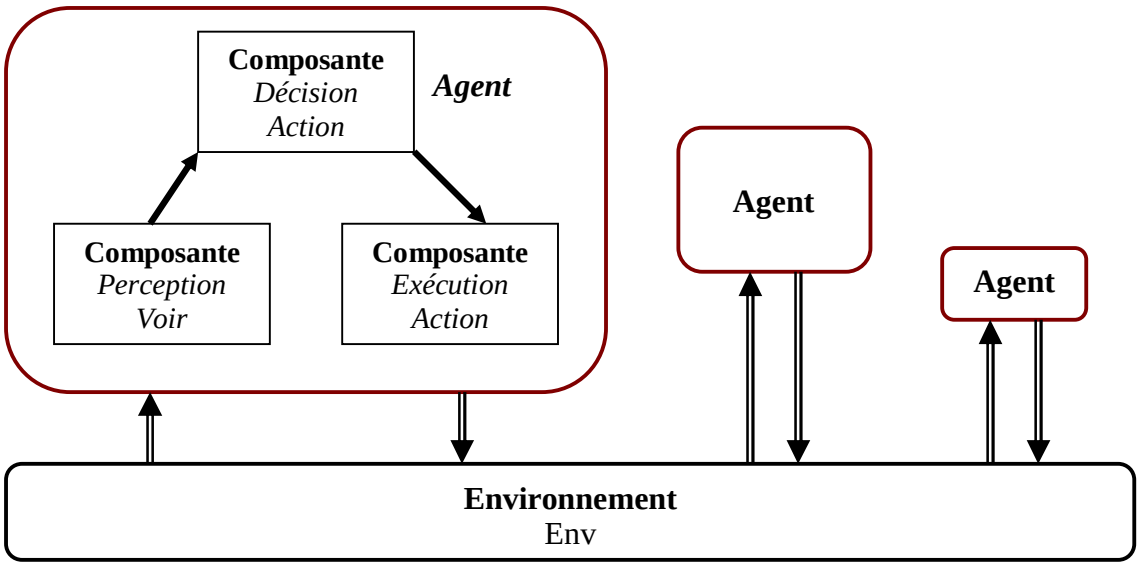


Fig 3.4 Structure d'un agent réactif dans un environnement multi-agents

La coordination réactive considère qu'il est possible de coordonner simplement les actions d'une population d'agents sans aucune planification initiale, en agissant naturellement au moment où l'action est nécessaire. Ces agents sont très simples et ne possèdent pas de représentation de leur environnement. Ils ne disposent pas non plus de mémoire ce qui les prive d'apprentissage et de toute anticipation aux évènements. Or, ces agents, aussi frustres soient-ils, montrent des comportements de groupe particulièrement impressionnants lorsqu'il s'agit de coordonner certaines actions, comme par exemple, leur déplacement.

3.1.2.2.1. Exemples

Grâce à très peu de paramètres élémentaires, les agents réactifs se montrent capables d'évoluer parfaitement en groupe, tout en s'évitant mutuellement, constituant par là une meute aux comportements très souples.

a) Boïds de Reynolds :

Le premier à s'être intéressé à la définition de comportements collectifs de meutes est Craig Reynolds. En 1987, il créé des créatures appelées « Boïds » (voir chapitre 1.5.1), des agents réactifs capables d'interactions sommaires dont émerge un comportement agrégatif sans faille, chacun des Boïds se contentant d'appliquer les trois règles comportementales suivantes :

- Maintenir une distance minimale par rapport aux objets de l'environnement y compris les autres Boïds.
- Adapter sa vitesse à la moyenne de celle de ses voisins.
- Aller vers le centre de graviter de l'ensemble des Boïds voisins.

Ces règles s'avèrent suffisantes pour générer un comportement global des Boïds, semblable à celui d'un vol d'oiseaux migrateurs. Contrairement aux oiseaux, qui possèdent un chef initiant le mouvement, les Boïds évoluent sans leader et sans contrôle global. La trajectoire du groupe reste cependant d'une fluidité parfaite : ils contournent tous ensemble un obstacle tout en restant naturellement groupés. Parfois la meute peut se diviser en deux pour se reformer sans problème par la suite.

Les travaux de Reynolds ont étés repris par Mataric [28] et appliqués à des agents réactifs robotiques. Chaque robot possède une batterie de comportements de haut niveau que ce dernier sélectionne en fonction de la situation dans laquelle il se trouve : évitement de collision, filature d'un autre robot, dispersion et répartition sur un territoire, agrégation, guidage sur un but. Ces comportements font appel à des primitives de bas niveau telles que la perception par infrarouge d'un obstacle ou le calcul d'une distance par sonar. Un comportement de meute peut s'obtenir par pondération des comportements d'évitement, de filature, d'agrégation et de dispersion. Les résultats sont ici moins brillants : les robots sont souvent trop éloignés les uns des autres, ce qui conduit à terme à la désagrégation du groupe, ou au contraire trop compactés, ce qui gène leur évolution. Mais Mataric attribue ces problèmes à la dynamique imparfaite de la mécanique des robots ainsi qu'à la limitation de leurs capteurs.

b) Système MANTA (simulation d'une fourmilière)

Ce système illustre parfaitement l'intérêt des modélisations multi-agents de type réactif (absence de planification). Ces dernières accordent aux sociétés qu'elles génèrent une grande capacité d'adaptation à des environnements très changeants, au prix, néanmoins, d'une certaine redondance de moyens.

Ce projet, issu de la collaboration entre éthologie et informatique, modélise notamment la sociogenèse d'une colonie de fourmis, c'est-à-dire la constitution d'une fourmilière mature à partir d'une ou de plusieurs reines. Il étudie également la capacité d'adaptation d'une telle colonie, les mécanismes de polyéthisme (division du travail) et la spécialisation des ouvrières.

Il s'agit en particulier de tester une hypothèse éthologique concernant l'aspect distribué de la prise de décision dans une colonie.

Le comportement des individus explique t-il à lui seul la stabilité des formes sociales observées? Une société d'agents peut-elle survivre et s'organiser en se passant de tout système de contrôle centralisé et d'une quelconque organisation hiérarchique? Le système MANTA [53] se propose de répondre à ces questionnements en ne paramétrant ni spécialisation, ni organisation préétablie, ni instance supérieure chargée de superviser l'ensemble.

b1) Architecture du système

Les agents utilisés peuvent être regroupés en deux catégories : les « assistés »: les oeufs, les larves et les cocons, et les « assistants », à savoir la reine, les mâles et les ouvrières. L'environnement de MANTA fait apparaître des points de nourriture, des zones d'humidité, et des sources de lumière, stimuli que les agents seront amenés à exploiter ou à éviter.

Les agents possèdent la capacité de choisir parmi plusieurs tâches en compétition, chacune d'elles relative à un seul type de stimulus de l'environnement. L'importance d'une tâche à accomplir (par rapport à une autre), est pondérée par l'intensité relative du stimulus qui la suscite, comparée à celles d'autres stimuli plus faibles perçus par l'agent. Cette intensité concerne la force d'un signal émis par le dit stimulus. Ce signal étant inversement proportionnel à la distance séparant stimulus et agent. En conséquence, parmi une population de stimuli, le signal le plus puissant entraînera la focalisation sur le stimulus associé de tout agent qui le percevra, ainsi que l'activation de la tâche unique concernant ce stimulus.

L'importance d'une tâche est donc dépendante, d'une part de l'intensité du stimulus qui la concerne, et d'autre part de son « poids ». Le poids d'une tâche est lui-même en relation avec le « niveau d'activité » de la tâche.

Ainsi, une tâche sera sélectionnée parmi d'autres si et seulement si :

- Le stimulus déclencheur dépasse un certain seuil de déclenchement (en dessous duquel le stimulus sera ignoré par l'agent)
- Le niveau d'activité de la tâche concernée est plus élevé que celui d'une autre tâche, suscitée en même temps.

Les tâches sont donc en compétition : si plusieurs tâches sont sélectionnables en même temps (l'agent percevant différents stimuli de même ampleur), la fourmi virtuelle choisira celle dont le niveau d'activité atteint chez elle la plus grande valeur.

b2) Rétroaction positive

Chaque agent voit son comportement renforcé. Le poids « w » de chaque tâche sera en effet augmenté d'une valeur chaque fois qu'une tâche « i » sera sélectionnée, ce qui peut s'écrire :

$w_i (t+1) = w_i (t) + x$ si i est sélectionnée. **Equation 14**

$w_i (t+1) = w_i (t)$ sinon. **Equation 15**

Ce mécanisme de renforcement (rétroaction positive) rend donc un agent plus sensible aux stimuli correspondant aux tâches qu'il a déjà exécuté. De ce fait, plus un agent accomplit une tâche, plus il aura tendance à l'exécuter de nouveau.

b3) Rétroaction négative

Les actions des ouvrières visent à maintenir l'homéostasie du nid. En effet, la plupart d'entre elles tendent à satisfaire les besoins exprimés par les agents assistés, à savoir les oeufs, les

cocons et les larves. Par exemple, une larve peut émettre un signal « affamée » si elle manque de nourriture. Si ce stimulus parvient à déclencher la tâche « nourrir larve » chez une ouvrière, la demande de la larve sera certainement comblée, et en conséquence, le signal de demande se verra en retour diminué. C'est cette diminution de signaux de demande qu'on appelle rétroaction négative. Par elle, les actions des ouvrières tendent à réguler le niveau des demandes.

Certains définissent un type d'agent supplémentaire communément appelé agent hybride [61]. Il reprend les caractéristiques des agents réactifs et des agents cognitifs pour former une entité capable de se comporter encore différemment.

Pour conclure cette section on peut dire que les SMA, représentent une notion « d'interaction de parties formant un tout émergent » qui s'étend à l'organisation sociale, à l'intelligence collective d'individus autonomes. Ainsi, l'une des caractéristiques majeures de la vie, «l'organisation» d'éléments primitifs, se dote d'une dimension fractale car elle semble se retrouver à tous les niveaux observés : Dans la brique initiale, dans les organismes qu'elle compose et dans les systèmes macroscopiques créés par ces organismes

3.2. Robotique collective

La robotique collective est née de l'observation des sociétés animales telles que les fourmilières ou les termitières. Elle est aussi le fruit d'une constatation : ***il vaut mieux une armée de robots très simples qu'un seul robot complexe***. En effet, les robots plus simples sont plus faciles à réaliser. Surtout, la perte d'un individu n'entraîne pas nécessairement l'échec de la mission, ce qui est le cas avec un seul robot très complexe. Ainsi, l'échec de la mission « Mars Polar Lander » aurait pu être évité par l'envoi d'un plus grand nombre de robots beaucoup plus simples.

La robotique modulaire [63] qui est un autre terme couvant la même idée, recouvre l'ensemble des robots constitués de parties plus ou moins autonome et se distingue de la robotique traditionnelle par le fait qu'il n'y a pas de contrôleur centralisé. Le choix d'une décentralisation à l'extrême est motivé par plusieurs points. Tout d'abord, du point de vue du contrôle, il est parfois plus facile de coordonner des mouvements complexes par une interaction locale entre éléments très réactifs (exemple : mouvement de chaque « morceau » de la colonne vertébrale d'un serpent pour la reptation). Ensuite, la présence de plusieurs éléments quasi-autonomes permet de mieux résister aux pannes et de faire varier sous certaines conditions la taille du robot (exemple : la longueur de la colonne vertébrale du serpent n'influent pas à priori sur la complexité du contrôle). Les propriétés inhérentes à la robotique modulaire (contrôle distribué, résistance aux pannes, reconfiguration) rendent les perspectives d'applications nombreuses en particulier dans les environnements hostiles à l'homme (résistance aux pannes utiles dans l'espace, reconfiguration utiles en terrain accidenté, etc.) [62, 60].

Du point de vue de la recherche, la robotique modulaire présente un domaine relativement nouveau, qui hérite certaines problématiques de la robotique collective et en pose de nouvelles. Les travaux actuels se regroupent dans deux catégories : tout d'abord les travaux sur les problématiques matériels de construction d'un prototype, ensuite, les travaux ayant pour sujet la conception automatique ou non de contrôleurs adéquats. Les recherches actuelles s'organisent autour de deux types de robots modulaires : le type « chaîne » où les modules sont fixés les uns aux autres, et le type « treillis » (lattice) où les blocs peuvent se déplacer les uns par rapport aux autres, voire se séparer de l'ensemble. Bien sur, des architectures hybrides sont aussi considérées.

3.2.1. Motivations

Les motivations qui ont poussé à la prolifération de la recherche dans le domaine de la robotique collective peuvent êtres résumées dans ce qui suit : 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 inférieure, peuvent êtres d'un **coût 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.

3.2.2. Contrôle d'un robot

Matarić [54] part de la classique planification pour arriver aux architectures purement réactives; 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 difficiles 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 qui 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. Noter que c'est à cette version de robot qu'on s'intéresse.

- 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 : un contrôle de bas niveau, gérant les réflexes de sauvegarde, purement réactifs comme l'évitement d'obstacles par exemple, et un contrôle de haut niveau gérant 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 [55]. 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 il est souvent à fonctionnalité émergente.

3.2.3. Contrôle d'un collectif de robots

Dans cette section on va présenter trois politiques différentes de gestion d'un collectif de robots à savoir : une approche supervisée, une approche partiellement distribuée avec « coach » et enfin une approche totalement distribuée.

- **Collectif supervisé** (approche 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 des 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.
- **Collectif « coaché »** (approche plus robuste) : La métaphore du coach (ou entraîneur) que l'on peut découvrir dans l'article de Drogoul et all.[56], 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.
- **Totalement distribuée** (approche plus générique) : où tous les robots sont indépendants. Cette vision trouve son origine dans l'éthologie et l'observation des communautés d'insectes. 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 voir Fig 3.5 et Fig 3.5 bis) ou bien différents. Dans le cas où tous les robots sont identiques, 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 de nouvelles notions ont étés introduites avec des métriques associées afin d'étudier l'hétérogénéité ou l'homogénéité d'une société.

Fig 3.5 Groupement de frisbees par un collectif de robots homogènes selon leurs couleurs.

Fig 3.5 bis Ces robots, développés au MIT et baptisés « R1 », ont pour tâches de rechercher, de récolter et de rassembler les palets disséminés sur la table. Chacun est équipé d'une pince, d'un système de positionnement et d'un système de communication radio. Ils accomplissent cette mission collectivement : chacun profite des informations récoltées par les autres robots pour trouver les palets. MAJA MATARIC

Il est à noter 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; cela même si les robots sont identiques du point de vue logiciel (même programme de contrôle) et physiquement (même corps/châssis). Dans notre cas, nos robots-agents sont réactifs et ne possèdent aucune faculté d'apprentissage, et initialement homogènes, ce qui nous écarte fondamentalement de ces propos.

3.2.4. Les insectes sociaux

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 c-à-d purement réactive, « caractérisée par un couplage direct de la perception à l'action » [58], 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 [58] 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) :

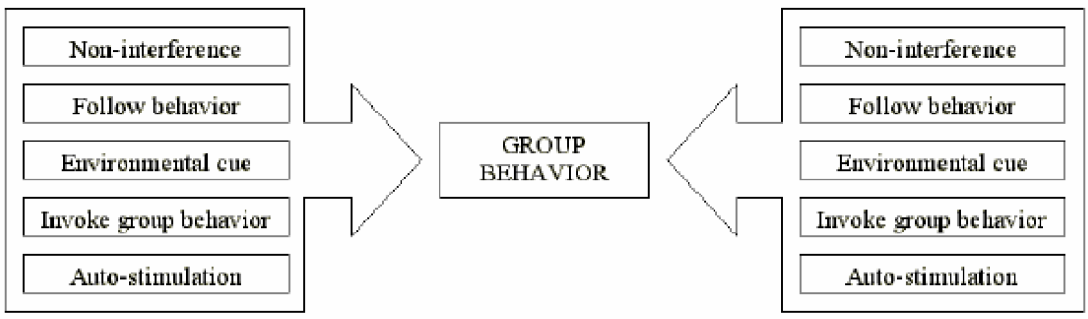


Fig 3.6 Cinq mécanismes pour un comportement de groupe

- 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 autostimulation. 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.

Kube et Zhang ont utilisé ces mécanismes pour étudier le problème du « box-pushing » (voir Fig 3.6). Ce jeu de mécanismes forme un exemple simple et frappant du fait de coller en 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. Pourtant, on peut soulever quelques questions au sujet de 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 certains 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 ciblées dans ce mémoire, cet ensemble de mécanismes 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. Dans le cadre de l'étude du comportement émergent, il 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.

Citons comme exemple témoignant de l'intérêt et du lyrisme qui animent les chercheurs dans cette perspective, L'équipe de Rodney Brooks du « MIT Artificial Intelligence Lab » qui travaille essentiellement sur la micro-robotique et les systèmes multi-agents. Elle travaille notamment sur un projet, appelé « The Ants », consistant à créer une fourmilière artificielle, inspirée du modèle biologique. Ceci afin d'obtenir une forme de communauté de micro-robots simples, structurée par leurs interactions.

3.2.5. 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. » [1] 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 dans le cas de robots réactifs.
- La **coordination d'actions** consiste en la bonne mise en oeuvre 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 le plus nous intéresser dans le sens où elle est l'un des principes de base de la théorie des AMAS [64], avec la notion de Situations Non-Coopératives (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.

3.2.6 Un exemple de résolution de conflits : la compétition agressive

Le problème soulevé ici est la résolution des interférences spatiales par un comportement dit « **agressif** » [65] : Soit 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 gagne 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 des sociétés de robots hétérogènes.

Les auteurs de cet exemple [65] 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 manoeuvrer 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. Ils 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 peut être non adapté.

3.2.7. Un exemple de méthodologie en Robotique Collective : *Cirta*

Pour parer à la complexité des phénomènes émergents, J.P. Müller et all. ont proposé une méthodologie, appelée *Cirta* [41], ayant pour but de produire des systèmes de micro-robots réactifs à comportement collectif émergent. Rappelons dans ce qui suit les points les plus adoptés dans la définition de l'émergence :

- un ensemble d'entités ou agents interagissant qui peuvent être exprimés dans une théorie ou vocabulaire de micro niveau;
- la dynamique d'interaction produit un phénomène global qui peut être une règle, un état stable ou un processus;
- ce phénomène est distingué 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 :

1) 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 (spatio-temporels).

2) 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. L'exemple donné par les auteurs dans ce sens est celui de fourmis fourrageuses. 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.

3) 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.

C'est par cette technique que müller et all. [41] ont étudié la dynamique des mini-robots ou bien des formations des pistes. D'après ces auteurs, « si un tel processus se stabilise, il conduit à une émergence structurelle et dynamique ». Ainsi, pour valider le caractère émergent d'une application, il suffit de vérifier la stabilisation des processus de Markov. Ils 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 leur définition. 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.

3.3. Conclusion

La technologie « Agent » est porteuse de beaucoup d'espérances, et elle est en plein essor. Beaucoup d'encre coule et coulera encore sur ce sujet, sans parler des meetings et des conférences organisés chaque année autour des SMA et tout ce qui peut en dériver. De nombreux acteurs sont impliqués dans cette technologie. Pourtant, malgré les efforts déployés, il n'y a pas encore eu l'émergence d'un ou de plusieurs chefs de files. Parmi les acteurs, on peut citer les deux géants de l'informatique : IBM, et Microsoft, ainsi que le géant des télécommunications : British Telecom.

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 aussi bien pour l'un comme pour l'autre a toujours trouvé naturellement sa place.

La Robotique Collective est en perpétuel changement. Elle s'avère être aussi un formidable outil d'étude des phénomènes de masse, et de l'apparition de comportements collectifs. Soulignons que l'étude des systèmes multi-agents nous permet de nous intéresser au domaine, très proche, de la Robotique Collective; les problèmes sont à quelques différences prés les mêmes dans les deux disciplines : Les agents des SMA représentent au niveau de l'ordinateur les robots qui doivent accomplir une tâche spécifique comme le transport de ressources. Ils sont autonomes et semblables à ces derniers surtout du point de vue communication; Ils ne disposent d'aucun moyen direct pour communiquer entre eux (radio, lumière, …), par contre, ils possèdent des capteurs et des effecteurs (simulés) afin d'interagir avec leur environnement, et ainsi communiquer indirectement avec leurs pairs.

Les limitations de la robotique collective peuvent être résumées en ce qui suit :

- Les expérimentations actuelles sont effectuées en environnement contrôlé et adapté aux robots. Ceci a pour conséquence de contraindre la dynamicité, et le réalisme, et donc de compliquer la tâche pour étendre cette approche à de vrais environnements.
- Ce type de systèmes est conçu pour la réalisation de tâches précises, approche très fonctionnelle (RoboCup). Ceci implique peu de recherches sur des systèmes génériques, capables de s'adapter à d'autres tâches et/ou à d'autres environnements.

L'un des problèmes types traités en robotique collective est celui de la classification. Ceci représente le sujet du prochain chapitre, où l'idée de base et de se servir de l'émergence pour accomplir la tâche de classification dans un environnement de robotique collective. Les Agents-robots de la collectivité simulée son dotés de règles comportementales simples trouvées via un processus évolutionniste élaboré dans ce sens. Les solutions proposées dans cette perspective utilisent l'idée de la généralisation de la formation de Tas à la formation multi-Tas. Pour faire simple, la classification est étudiée au début selon un seul critère, par la suite l'étude est généralisée à plusieurs critères.

3.4. Références

Références bibliographiques

[1] J. Ferber, « Les Systèmes multiagents. Vers une intelligence collective », InterEditions, 1995.

[2] A. Caglayan et C. Harrisson « Les Agents ». InterEditions, 1997
[3] A. H. Vera et H. A. Simon « Situated action : A symbolic interpretation ». Cognitive Science, 17 : 7-48, 1993.
[4] W. J. Clancey, « Situated Cognition ». Cambridge University Press, Cambridge, 1997.
[5] N. Avouris et L. Gasser, « Distributed Artificial Intelligence: Theory and Praxis », Kluwer Academic Publishers, 1992.
[6] Bond A. et L. Gasser , « Readings in Distributed Artificial Intelligence ». San Mateo, California, Morgan Kaufman, 1988.
[7] J. R. Galliers. « A Theoretical Framework for Computer Models of Cooperative Dialogue, Acknowledging Multi-Agent Conflic »,PhD thesis, Open University, UK, 1988.
[8] J. R. Galliers. « Cooperative interaction as strategic belief revision », In S. M. Deen, editor, CKBS-90 - Proceedings of the International Working Conference on Cooperating Knowledge Based Systems, pages 148-163. Springer-Verlag: Heidelberg, Germany, 1991.
[9] L. Gasser. « Social conceptions of knowledge and action: DAI foundations and open systems semantics ». Artificial Intelligence, 47:107-138, 1991.
[10] M. P. Georgeff, « Communication and interaction in multi-agent planning », In Proceedings of the Third National Conference on Artificial Intelligence (AAAI-83), Washington, D.C., 1983.
[11] R. Conte, M. Miceli, and C. Castelfranchi, « Limits and levels of cooperation », In Y. Demazeau & J.-P. Müller, editors, Decentralized AI 2 - Proceedings of the Second European Workshop on Modelling Autonomous Agents and Multi-Agent Worlds (MAAMAW- 90), pages 147-160. Elsevier Science Publishers B.V.: Amsterdam, The Netherlands, 1991.
[12] D. D. Corkill and V. R. Lesser, « The use of meta-level control for coordination in a distributed problem solving network », In Proceedings of the Eighth International Joint onference on Artificial Intelligence, pages 748-756, Karlsruhe, Federal Republic of Germany, August 1983.
[13] M. Côté and N. Troudi, « Une architecture multi-agent pour la recherche », L'expertise Informatique, 3(3), 1998.
[14] A. Drogoul, M. Tambe, T. Fukuda, « Collective Robotics », Lecture Notes in Artificial Intelligence 1456, Proceedings, 1st International Workshop, CRW'98, Paris, France, July 1998.
[16] R.A. Brooks, « A robust layered control system for a mobile robot », IEEE Journal of Robotics and Automation, 286(1) :14–23, 1986.
[17] P. Agre and D. Chapman, « Pengi : An implementation of a theory of activity », In Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87), pages 268–272, 1987.
[18] J. L. Deneubourg, S. Goss, N. Franks, A. Sendova-Franks, C. Detrain, and L Chretien. « The dynamics of collective sorting », In From Animals to Animats : International Conference on Simulation of Adaptive Behavior, pages 356–363. MIT Press, 1990.
[19] A. Drogoul and J. Ferber. « from tom-thumb to the dockers : Some experiments with foraging robots ». In From Animals to Animats II, pages 451–459, Cambridge, 1993. MIT Press.
[20] R.A. Brooks, « A robot that walks : Emergent behaviors from carefully evolved networks », Neural Computation, 1(2) :253–262, 1989.
[21] R.A. Brooks, « Intelligence without representation », Artificial Intelligence Journal, 47 :139–159, 1991.
[22] W. Ross Ashby, « An Introduction to Cybernetics », Chapman & Hall, London, 1956.
[23] R.Schoonderwoerd, O. Holland, and J.Bruten, « Ant-like agents for load balancing in telecommunications networks », In Proceedings of the 1st International Conference on Autonomous Agents, pages 209–216, February 5-8 1997.
[24] S. Goss and J.L. Deneubourg, « Harvesting by a group of robots », In F. Varela and P. Bourgine, editors, Proceedings of the First European Conference on Artificial Life Conference (ECAL91), pages 195–204, Paris, France, MIT-Press., 1991.
[25] M. Sebring, E. Shellhouse, M. E. Hanna, and R. A. Whithurst, « Expert systems in intrusion detection : A case study », In Proc. 11th NIST-NCSC National Computer

Security Conference, pages 74–81, 1988.

[26] L. Steels, « Building agents with autonomous behavior systems », In L. Steels and R. Brooks, editors, The artificial life route to artificial intelligence. Building situated embodied agents. Lawrence Erlbaum Associates, 1994.

[27] P. Maes and R.A. Brooks, « Learning to coordinate behaviors », In Proceeding of the 8th National Conference on Artificial Intelligence, p 796–802, Menlo Park, CA, AAAI, 1990.

[28] M. Mataric, « Designing emergent behaviors : From local interactions to collective intelligence », In J-A. Meyer, H. Roitblat, and S. Wilson, editors, roceedings, From Animals to Animats 2, Second International Conference on Simulation of Adaptive Behavior (SAB-92). MIT Press, 1992.

[29] E. Bonabeau, M. Dorigo, and Theraulaz, « From Natural to Artificial Swarm Intelligence », Oxford University Press, 1998.

[30] E. Bonabeau, F. Henaux, S. Guerin, D. Snyers, P. Kuntz, and G. Theraul, « Routing in telecommunication networks with smart ant-like agents », In Proceedings of IATA'98, Second Int. Workshop on Intelligent Agents for Telecommunication Applications. Lectures Notes in AI vol. 1437, Springer Verlag, 1998.

[31] E. Bonabeau, G. Theraulaz, J.-L. Deneubourg, and S. Camazine, « Self-organisation in social insects », Trends in Ecology and Evolution, 12(5) :188–193, 1997.

[32] E. Tzafestas, « Vers une systémique des agents autonomes : des cellules, des motivations et des perturbations » , Thése de l'Université PARIS 6, 1995.

[33] S. Nolfi, « Power and the Limits of Reactive Agents », Institute of Psychology, National Research Council, Roma, Italy, 2000.

[34] B. Drieu, « L'intelligence artificielle distribuée appliquée aux jeux d'équipe situés dans un milieu dynamique : l'exemple de la RoboCup », Mémoire de Maîtrise de l'Université Paris 8, 2001.

[35] J-P. Briot et Y. Demazeau, « Introduction aux agents : Principes et architecture des systèmes multi-agents », collection IC2, Hermès, 2001.

[36] J-P. Briot, « Introduction aux Agents, Thème OASIS », Laboratoire d'Informatique de Paris 6, Université Paris 6 – CNRS, Paris, France, 2000.

[37] A. Dutech, O. Buffet, F. Charpillet, « Développement autonome des comportements de base d'un agent », Loria - INRIA-Lorraine, 2004.

[38] P. Maes, « Modeling Adaptive Autonomous Agents », MIT Media-Laboratory, 1995.

[39] J-P. Sansonnet, « Modeling Adaptive Autonomous Agents », LIMSI-CNRS. 1995.

[40] D. Lestel, B. Grison, A. Drougoul, Les agents réactifs et le vivant dans une perspective d'évolution cooperative », Intellectica, pp. 73-90. 2/1994.

[41] J-P. Müller, « Des systèmes autonomes aux systèmes multi-agents : Interaction, émergence et systèmes complexes », Rapport présenté pour l'obtention de l'Habilitation à Diriger les Recherches en Informatique, novembre 2002.

[42] A. H. Vera et H. A. Simon, « Situated action : A symbolic interpretation », Cognitive Science, 17 : 7–48, 1993.

[43] W. J. Clancey, « Situated Cognition », Cambridge University Press, Cambridge, 1997.

[44] G.PICARD, « Etude de l'émergence comportementale d'un collectif de robots par auto-organisation cooperative », IRIT - équipe SMAC, 2001.

[45] A. Munoz Melendez sous la direction de A. Drogoul et de D. Duhaut, « Robotique Collective : implantation de deux comportements strategiques inspires de comportements animaux », Stage de DEA IARFA, 1999.

[46] N. Monmarché, S. Mohamed, G. Venturini, « L'algorithme AntClass : classification non supervisée par une colonie de fourmis artificielles », Laboratoire d'Informatique de l'Université de Tours, 2001.

[47] O. Buffet, « Apprentissage par renforcement dans un système multi-agents », Mémoire de DEA, de l'université de Henri Poincaré, 2000.

[48] O. Bibot, V. Brakeva, « Intelligence artificielle : Les agents intelligents », Travail réalisé dans le cadre du cours d'intelligence artificielle de Mr. Giot, ISIB, Bruxelles, 2005.

[49] J. Ferber, « La kénétique : Des systèmes multi-agents à une science de l'interaction »,

Revue Internationale de Systémique, 8(1):13-27,1994.
[50] J. Ferber, « Coopération réactive et émergence », Intellectica, 19(2):19-52, 1994.
[51] Sbastien F. « Intégration de la dimension spatiale au sein d'un modèle multi-agents à base de rôles pour la simulation : Application à la navigation maritime », Thése de l'université de Rennes 1, 2005.
[52] F. Van Aeken, « Les systèmes multi-agents minimaux**,** Un Modèle Adapté à l'Etude de la Dynamique Organisationnelle dans les Systèmes Multi-Agents Ouverts », Thése de l'Institut National Polytechnique de Grenoble, 1999.
[53] M. Le Bars, « Un Simulateur Multi-Agent pour l'Aide à la Décision d'un Collectif : Application à la Gestion d'une Ressource Limitée Agro-environnementale », Thèse de l'Université de PARIS IX-DAUPHINE, 2003.
[54] J. M. Mataric, M. Nilsson, K. T. Simsarian, « Cooperative Multi-Robot Box-Pushing », IEEE/RSJ International Conference on Intelligent Robots and Systems, Pittsburg, PA, 1995.
[55] R. A. Brooks, « A Robust Layered Control System for a Mobile Robot », IEEE Journal of Robotics and Automation, RA-2, 14-23, 1986.
[56] A. Drogoul, M. Tambe, T. Fukuda, « Collective Robotics »,Lecture Notes in Artificial Intelligence 1456, Proceedings, 1st International Workshop, CRW'98, Paris France, July 1998.
[57] T. BALCH, « Social Entropy : a New Metric for Learning Multi-Robot », Teams Proceedings, 10th International FLAIRS Conference (FLAIRS-97), 1997.
[58] C. Ronald KUBE, H. ZHANG, « Collective Robotics Intelligence : From Social Insects to Robots », Proceedings, 3rd International Conference on Simulation of Adaptive Behavior, 1993.
[59] R. C. Kube and H. Zhang, «The use of perceptual cues in multi-robot box-pushing. », In Proceedings IEEE International Conference on robotics and Automation, 1996.
[60] P. Gauthier, « Etude de l'émergence comportementale d'un collectif de robots par auto organisation cooperative », Rapport de DEA de l'IRIT 2000-2001.
[61] A. Cardon, « Etude de la conception et contrôle comportemental d'une organisation massive d'agents », LIP6, Laboratoire d'informatique de Paris VI, Thème OASIS, 2002.
[62] A. Martinoli, « Swarm intelligence in autonomous collective robotics : from tools to the analysis and synthesis of distributed control strategies » , thèse de l'école polytechnique fédérale de Lausanne, 1999.
[63] N. Bredeche, Stage Master au sein de l'équipe Inférence&Apprentissage-TAO du LRI, 2005.
[64] V. CAMPS , « Vers une théorie de l'organisation dans les systèmes multi-agents basés sur la coopération », Thèse IRIT, n° 2890, 1998.
[65] T. Richard Vaughan, K. Stoy, S.Gaurav Sukhatme, J. M. Matric, « Go ahead, make my day : Robot Conflict Resolution by Aggressive Competition », Proceedings, 6th International Conference on the Simulation Adaptive Behavior (SAB-2000), Paris, France, 2000

Références sur le Web

www.veille.com
www.agentland.com
http://www.strategic-road.com/recherch/agents/agentsfr.htm
http://www.agentintelligent.com/agent_intelligent/agents_intelligents.html
http://ressources.abondance.com/agents.html
http://euro.ecom.cmu.edu/resources/elibrary/eclbots.shtml
http://agents.umbc.edu/
http://www-2.cs.cmu.edu/~softagents/
http://www.botspot.com/
http://www.agentbase.com/survey.html
http://www.compinfo-center.com/ai/intelligent_agents.htm
http://agents.media.mit.edu/projects/

http://www-cia.mty.itesm.mx/~lgarrido/Repositories/IA/
http://www.cs.tcd.ie/research_groups/aig/iag/
http://raynaud.userworld.com/detailVeille.htm
http://www.stratenet.com/veille-strategique.html
http://concurrentielle.veille.co.uk/
http://www.australisintelligence.com/
http://www.journaldunet.com/rubrique/veille/veille020426.shtml
http://www.planete-commerce.com/agents-intelligent.html
http://www.agentintelligent.com/veille/livre_veille_agent.html
http://www.journaldunet.com/rubrique/veille/veille020927.shtml
http://www.damas.ift.ulaval.ca/publications/index.html
http://www.anacip.com
http://www.agentland.fr
http://www.sam-mag.com/archives
humain.htm
http://turing.cs.pub.ro/auf2
http://www.damas.ift.ulaval.ca
http://www.decisionnel.net/agentintelligent
http://www.enpc.fr/enseignements/Legait
projet/victor/chercher/Agent.Int.html
http://big.chez.tiscali.fr/gbonnet/laii.html
http://www.dess-taii.univ-avignon.fr/pages
PagesEnseignements/Cours/AgentsIntelligents.pdf
http://c.asselin.free.fr/french/agentintelli.htm
http://www.geoscopie.com/sources
internet/g051motint.html
http://dmoz.org/World/Fran%C3%A7ais
Informatique/Intelligence_artificielle/Agents
http://www.limsi.fr/Individu/jps/enseignement
examsma/examsma.htm
http://www.irit.fr/ACTIVITES/EQ_SMI/SMAC
http://agents.umbc.edu
http://www.agentintelligent.com/ Veille technologique, chatterbot et agents intelligents
http://www2.parc.com/spl/projects/modrobots/chain/polybot/
http://www.newscientist.com/news/news.jsp?id=ns99996683
http://tao.lri.fr/index.php/Robotique_modulaire_chaine_(2004-2005)

Autres références

- **CD** Universalis 2005
- **CD** Encarta 2005
- Techniques de l'ingénieur version électronique et papier
- Françoise et Jean paul Bertrandias, « Mathématiques pour les sciences de la nature et de la vie », Collection Grenoble Science, 1993.
- Eric Bonabeau, Guy Théraulaz, « L'intelligence en essaim », Pour la science, N° 27, mai 2000.
- Stephen Wolfram, « Les logiciels scientifiques », Pour la science, Novembre 1984.
- Colin Harrison, Alper Caglayan, « Les agents : Applications bureautiques, Internet et intranet », Intereditions, 1998.

4.1. Introduction

La popularité, la complexité et toutes les variantes du problème de classification dans divers domaines ont donné naissance à une multitude de méthodes de résolution en informatique. Ces méthodes peuvent à la fois faire appel à des principes heuristiques* [34] ou encore mathématiques. Parmi celles-ci, il existe une branche qui s'inspire plus spécialement de la biologie. Les motivations des chercheurs dans ce cas sont de tester de nouveaux algorithmes sur le problème de classification et de connaître leurs apports. Mais aussi de proposer de nouvelles sources d'inspiration, étant donné que le phénomène de classification se rencontre souvent dans les systèmes biologiques de tous genres.

Dans ce contexte, l'un des domaines qui s'inspire le plus du vivant et sur lequel nous focalisons tout particulièrement est la robotique collective. Des chercheurs dans ce domaine essayent déjà depuis quelques temps de doter une famille de robots autonomes, mobiles et homogènes d'un ensemble de réactions simplifiées (au niveau micro), qui vont les mener à réaliser ou à faire émerger une tâche spécifique (au niveau macro), comme celle de la « classification ». La difficulté réside alors dans le fait de rechercher et d'interpréter (une fois trouvées) ces réactions dites sensori-motrices via : un environnement de simulation, un modèle à base d'agents réactifs et une méthode (entre autres) de recherche incrémentale à base d'algorithmes génétiques. Pour ce faire, nous allons au début définir les concepts de regroupement, de classification et de tri, et essayer de les comparer essentiellement dans le contexte du travail mené dans ce projet. On évoquera par la suite le problème de classification chez les fourmis à travers quelques exemples. On reviendra ensuite au domaine de la robotique collective pour analyser le problème étudié ici, à savoir la formation de Tas (heap-formation) et l'idée de sa généralisation à une formation multi-Tas dans le but de l'exploiter pour la classification d'objets. Au début, selon un seul critère, et ensuite selon plusieurs.

4.2. Définition de regroupement, classification et tri

4.2.1. Définition du regroupement

Parmi les définitions proposées pour le regroupement on retrouve ce qui suit :

- Un bloc d'informations et de services connexes rassemblés à partir de diverses organisations, en groupes qui sont significatifs pour les citoyens et les clients.
- Regroupement autour d'un terme de base de mots liés les uns aux autres par des rapports à la fois morphologiques et sémantiques. Les possibilités qu'offrent les systèmes de suffixation et de préfixation pour passer d'une construction de phrase à une autre construction, d'un verbe à un substantif, d'un substantif à un adjectif, etc.
- ActionT de grouper : Réunion de personnes, d'objets groupés. Groupement d'intérêt économique : entité juridique dotée de la personnalité morale et de la capacité juridique, constituée par contrat entre deux ou plusieurs personnes physiques ou morales en vue de mettre en oeuvre tous les moyens propres à faciliter ou à développer l'activité économique de ses membres, à améliorer ou à accroître les résultats de cette activité.

4.2.2. Définition de la classification

D'après quelques encyclopédies (Encarta, Universalis, 36 dictionnaires,…), la classification est l'action de ranger par classe, c'est aussi le résultat de cette action. C'est aussi le classement hiérarchique en catégories selon une logique et des critères précis.

Une autre définition est proposée selon le point de vue des biologistes :

La classification sert à 2 fins :

1. Elle soulage la mémoire; Le nombre des phénomènes naturels est infini, et la mémoire la mieux douée ne pourrait les retenir. La classification, en y mettant de l'ordre, en facilite l'étude. Par elle, on se souvient plus facilement des faits, on les reconnaît plus aisément quand ils se présentent dans la nature.
2. Elle nous permet de reconstituer l'oeuvre de la nature; Celle-ci a un certain plan. C'est en tentant de le retrouver que les sciences naturelles apportent une satisfaction à l'esprit.

La classification dans ce cas est l'opération par laquelle nous rangeons les êtres ou les faits observés en *groupes* distincts et subordonnés les uns aux autres. Il y a deux sortes de classifications :

- **Classifications *artificielles*** : qui rangent les faits ou les êtres d'après *certains caractères extérieurs*. Elles réalisent la première fin de la classification en aidant la mémoire, mais ne servent qu'à cela. Il ne faut pourtant pas croire que la classification artificielle soit purement arbitraire. Elle a un fondement naturel, et repose toujours sur un caractère réel, mais qu'elle choisit, pour bien atteindre sa fin, plutôt bien visible que réellement important.

- **Classifications *naturelles*** : Les classifications naturelles au contraire répartissent les êtres d'après *leurs véritables rapports*. Elles ne sont plus fondées sur un caractère extérieur, mais sur la nature intime des objets. Elles réalisent surtout le second but de la classification, satisfaire l'esprit en lui reconstituant en tout ou en partie le plan de la nature.

Moyens de classifications :

1. Au moyen de la comparaison des êtres ou des faits,
2. Au moyen de la subordination des faits ou des caractères,
3. Comparaison. La comparaison est une opération qui rapproche tous les caractères des êtres (ou des objets) pour déterminer ceux qui sont semblables et ceux qui sont différents. Mais ce premier travail ne suffit pas: après avoir établi un rapprochement entre les caractères semblables, il faut pour terminer la classification avoir recours à la subordination des caractères,
4. Subordination des caractères. Certains caractères semblent dominer les autres, en ce que les seconds ne peuvent exister sans les premiers. On ne peut être mammifère si l'on n'est vertébré. C'est par cette subordination que s'établit la classification.

Bien que ce travail soit fait dans le cadre du vivant, Il est clair qu'on peut étendre les moyens de classifications présentés ci-dessus aux objets, sans aucune perte ou déformation du sens de la classification.

On peut dire donc que, le principe de classification propose de retrouver le plan de la nature; en supposant qu'il y a un certain ordre dans les choses. Cet ordre vient du principe de finalité. La classification cherche donc à retrouver l'ordre des êtres ou des objets par les fins qui leur sont assignées. Son principe est donc *le principe de finalité*.

4.2.3. Définition du tri

Plusieurs définitions sont aussi proposées pour le tri :

- En informatique ou en mathématiques, un algorithme de tri est un algorithme qui permet d'organiser une collection d'objets selon un ordre déterminé. Les objets à trier font donc partie d'un ensemble muni d'une relation d'ordre (de manière générale un ordre total). Les ordres les plus utilisés sont l'ordre numérique et lexicographique (dictionnaire).
- Le tri en mathématique est l'opération qui consiste à classer des informations ou des données selon un ou plusieurs critères ou selon un ordre donné.

- Retrait des unités défectueuses d'un lot de poisson ou de produits du poisson.

Selon les définitions fournies ci-avant on peut dire ce qui suit : On remarque que le regroupement peut se faire sans critère de regroupement : on ne fait qu'amasser dans un voisinage donné (qu'il soit réel ou virtuel) des choses sans distinction. Le regroupement réalise dans ce cas une fonction topologique comme la formation de Tas. La seule distinction qu'on doit prendre en compte est celle de ceux qui amassent et de ce qu'ils amassent.

Si le regroupement est effectué selon un ou plusieurs critères donnés, le regroupement se comprend dans le sens de la classification. Donc, la classification est une tâche qui peut s'accomplir à base de regroupement. Il peut s'agir aussi de sous-regroupement, si une relation de hiérarchie est présente, l'ordre dans ce dernier cas est important.

L'opération de tri est une opération où l'ordre est important. Ceci nous amène à dire que le tri est une opération construite à base de classification et/ou d'ordonnancement. Notons que le tri peut se faire sans classification. Dans ce cas il devient synonyme d'ordonnancement. Les éléments qui vérifient un critère donné se trouveront d'eux même les uns avec les autres, sans qu'il y'est préméditation ou volonté de le faire.

Donc, L'opération de formation de Tas est une opération de regroupement, et la formation multi-tas est une opération constituée de plusieurs regroupements, vérifiant chacun une ou plusieurs qualification(s) donnée(s), ce qui nous ramène vers la classification. Le tri dans son sens exclusif doit réaliser en plus de la classification une relation d'ordre entre les différentes classes ainsi constituées. On peut dire dans ce cas, que pour le problème étudié dans ce mémoire, il est plus judicieux de dire qu'il s'agit de rechercher des règles sensori-motrices qui réalisent plutôt une classification qu'un tri, et que la classification et réalisée à base de regroupement.

4.3. Les algorithmes de classification biomimétique*

Le monde du vivant aujourd'hui permet de proposer des méthodes de résolution de problèmes qui s'inspirent de ses mécanismes. Les algorithmes génétiques, qui puisent dans la théorie de la sélection naturelle [29], en sont des exemples de plus en plus rencontrés, tant pour des problèmes d'apprentissage automatique que d'optimisation. Plus récemment les fourmis présentent un intérêt grandissant pour la résolution de problèmes complexes. On parle alors de fourmis artificielles qui s'inscrivent dans une démarche visant à exploiter l'intelligence collective qu'ont leur a attribué [30]**.** Notons que la classification fait partie de problèmes pour lesquels les fourmis suggèrent des heuristiques très intéressantes pour les informaticiens. Une des premières études relatives à ce domaine a été menée par Deneubourg [31] où une population d'agents-fourmis bouge aléatoirement sur une grille en 2D déplaçant ainsi des objets dans le but de les rassembler. Cette méthode a été étendue par Lumer [32] sur des objets simples puis par Kuntz [33] où un cas concret est abordé dans le but de résoudre efficacement un problème d'optimisation.

Dans ce qui suit, nous allons survoler les méthodes biomimétiques permettant de résoudre le problème de la classification : Allant des approches évolutionnaires, en passant par l'approche à base de fourmis artificielles, et d'intelligence en essaim (nuages d'agents) jusqu'aux systèmes immunitaires.

4.3.1. Approche évolutionniste

Les premiers travaux sur l'évolution artificielle ont concerné les algorithmes génétiques (AG), les stratégies d'évolution (SE) et la programmation évolutive (PE). Ces trois types d'algorithmes ont utilisé des principes globalement communs car ils se sont tous inspirés des mêmes principes du neo-darwinisme* : utilisation d'une population d'individus, évaluation des individus par une fonction de sélection des meilleurs et génération d'une nouvelle population avec des opérateurs de croisement et

de mutation. Cependant, des choix méthodologiques ont initialement opposé ces méthodes. Ainsi, les premiers AG utilisaient plutôt un codage binaire des individus alors que les SE utilisaient un codage en nombre réel. Ensuite, dans les années 90 est apparue la programmation génétique (PG) qui introduit notamment des représentations arborescentes.

Pour toutes ces approches, la représentation va également imposer des opérateurs particuliers pour engendrer de nouvelles solutions. Par exemple, l'un des principes fondamentaux des AG étant d'utiliser un opérateur de croisement combinant utilement les gènes de deux individus, le problème posé est alors de définir des opérateurs de croisement permettant l'échange de caractéristiques entre deux classifications. Les SE utilisent plutôt des mutations à base de lois gaussiennes qui vont modifier les paramètres réels d'un individu.

4.3.1.1. Algorithmes génétiques

Les premiers travaux, proposant un AG (et plus généralement un algorithme évolutionnaire) pour le problème de la classification, sont dus à Raghavan et Birchard [27]. Le nombre de classes est fixé à l'avance et la représentation de longueur n associe une classe à chaque donnée.

Les opérateurs génétiques sont une adaptation directe des opérateurs génétiques binaires pour le cas d'un individu représenté par une chaîne n-aire (croisement avec un point de coupure). Par exemple, le croisement à un point d'échange des étiquettes de classe entre deux individus. Cet opérateur peut donc faire disparaître des classes. Seule la mutation peut faire apparaître de nouvelles classes. La fonction d'évaluation consiste à minimiser une erreur quadratique. Notons que ce codage est utilisé également par Hansohm, [15] dans le contexte du bipartitionnement : le premier vecteur d'entiers code la partition sur les données, le deuxième sur les variables. De nouveaux codages, utilisant des permutations, ont été introduits par plusieurs auteurs. Dans [16], un premier codage consiste à utiliser une permutation des n données représentées par leur indice en ajoutant en plus des symboles servant de séparateurs : par exemple pour n = 6, l'individu (2,4,-,5,1,3,-,6) représente un partitionnement en 3 classes. Pour obtenir k classes, le nombre de séparateurs introduits doit être égal à k − 1. Un autre codage à base de permutation proposé dans le même travail consiste à utiliser deux parties dans un même individu. Les prototypes sont codés sur la première partie de la partition, puis la suite de la partition représente un ordre sur la manière d'affecter les données restantes à ces prototypes. Dans [7], ce type de permutation est utilisé uniquement pour fixer un ordre sur les données : un algorithme heuristique décide alors comment construire la partition en « coupant » la permutation en des endroits judicieux. Pour ces codages sont utilisés des opérateurs génétiques de croisement définis dans le cadre du problème du voyageur de commerce (OX et PMX, voir [13]). Un autre codage alternatif consiste non pas à représenter une classe pour chacun des objets dans un individu de longueur n, mais plutôt k prototypes de ces classes [21]. Ces k prototypes sont choisis parmi l'ensemble des n donnés. Ainsi, un individu devient alors un vecteur d'indices de prototype :

Ensuite, pour calculer la partition résultante, les données sont affectées à chaque prototype de classe sur la base d'un algorithme de type plus proche voisin. Dans cette représentation, les opérateurs génétiques classiques peuvent poser des problèmes : à la suite d'un croisement, un même prototype peut se retrouver deux fois dans un individu.

Le codage introduit dans [6] consiste à représenter le partitionnement à l'aide d'une matrice M booléenne de type classe × données. M(i,j) prend la valeur 1 ($i \in [1,k]$, $j \in [1,n]$) si la donnée dj appartient à la classe i, 0 sinon.

Dans cette représentation, l'opérateur de croisement est défini cette fois en 2D. Un point important à noter dans cette représentation est la possibilité de la généraliser à des classifications recouvrantes ainsi que floues : dans le premier cas plusieurs 1 peuvent apparaître sur une même ligne, dans le deuxième les valeurs ne sont plus booléennes mais représentent des degrés d'appartenance.

Un codage permettant de manipuler directement des groupes a été proposé dans [11]. Une classification est constituée de n gènes représentant la classe de chaque donnée (comme dans le premier codage introduit dans cette section), suivis de la liste des groupes apparaissant dans l'individu (par exemple si les données appartiennent à trois classes, l'individu finit par (3, 2, 1)). Cette représentation utilise donc un opérateur de croisement permettant d'échanger directement des groupes.

L'opérateur de mutation agit également au niveau des groupes (éclatement, regroupement, etc.) avec des heuristiques locales (réaffectation des données isolées). C'est Greene (voir [14]) qui a développé à notre connaissance le seul AG apprenant une classification hiérarchique présentée sous la forme d'un arbre de centroïdes. Cet algorithme est restreint aux données numériques mais ne fait pas d'hypothèses sur le nombre de classes.

4.3.1.2. Autres approches évolutionnistes

Les trois autres catégories d'algorithmes évolutionnaires ont été nettement moins développées que les AG. Par exemple, dans [5], les SE ont été utilisés avec un codage matriciel : chaque colonne du tableau représente un centre de classe de la même dimension que l'espace numérique de description des données. La PE a également été utilisée mais dans un seul travail à notre connaissance [12]. Également, nous n'avons pas trouvé d'articles traitant du problème général de la classification avec la PG.

Plusieurs approches hybrides ont cependant été proposées en utilisant conjointement les AG avec des approches plus classiques comme K-Means ou encore Fuzzy-CMeans. Ces heuristiques sont utilisées par exemple juste après l'AG qui sert donc à trouver une bonne partition initiale [4]. Elles peuvent également servir au même titre que les opérateurs génétiques dans la boucle de l'AG [18] : elles sont appliquées sur chaque individu. Cela permet d'accélérer la convergence des AG tout en conservant les avantages d'une méthode globale. Ces hybridations restent cependant liées aux données numériques.

4.3.2. Fourmi artificielle

Les fourmis réelles ont inspiré les chercheurs en informatique dans de nombreux domaines. Cela se justifie particulièrement quand on connaît la richesse comportementale de ces animaux. L'un des modèles les plus connus (ACO pour Ant Colony Optimization) a été introduit par Clorni et all.[8] initialement dans le cadre du problème du voyageur de commerce. Les fourmis utilisent des phéromones pour marquer des arcs entre les villes. Ces phéromones représentent en fait une distribution de probabilités qui est mise à jour en fonction des résultats observés (longueur totale du chemin par exemple). Cette approche a été depuis, largement développée et appliquée à de nombreux problèmes d'optimisation combinatoire et numérique. Un survol de ces articles ne rentre cependant pas dans le cadre de notre étude puisque ce modèle n'a pas été utilisé à notre connaissance pour résoudre le problème de la classification (voir cependant [2] mais apparemment, aucune suite n'a été donnée à ces travaux). Pourtant, les sections suivantes vont montrer que le modèle des fourmis artificielles est très riche dans ce domaine. La raison vient du fait que d'autres comportements observés chez les fourmis peuvent être directement mis en relation avec le problème de la classification, à commencer par la classification du couvain.

4.3.2.1. La classification du couvain*

On constate expérimentalement, en renversant sur une planche le contenu du couvain d'une fourmilière, que les fourmis classent spontanément les oeufs par stades d'évolution : elles font de petits tas d'oeufs, de larves et de nymphes. Elles sont même capables de distinguer plusieurs stades d'évolution pour les oeufs. Comment peut-on expliquer ce comportement complexe alors qu'une

fourmi est un être très simple ? Nous allons voir dans ce qui suit que l'on peut modéliser par ordinateur ce comportement observé par les entomologistes* [53].

a) Données dont dispose une fourmi sur son environnement

Les fourmis sont capables d'émettre et de capter des molécules appelées phéromones. Un oeuf de fourmi émet des phéromones différentes suivant son stade d'évolution. On peut donc légitimement supposer qu'une fourmi est capable d'estimer la concentration en oeufs du même stade au point où elle se trouve en « reniflant » les phéromones (qu'elles captent avec leurs antennes). Par contre, la myopie des fourmis leur interdit d'avoir une vue d'ensemble du couvain. On peut aussi supposer qu'une fourmi dispose d'une mémoire à court terme des oeufs qu'elle a rencontré durant les dernières secondes de son déplacement.

On peut donc réaliser deux modèles de comportement : le premier suppose que la fourmi a une perception spatiale de la concentration en oeufs d'un type donné, le deuxième fait intervenir la mémoire à court terme de la fourmi.

b) Modélisation du comportement des fourmis

Pour modéliser le comportement de la fourmi, il faut simplifier l'environnement des fourmis virtuelles. On suppose dans ce cas que les fourmis se déplacent sur une grille. Sur cette grille seront disposés au début de la simulation des oeufs de 2 types : des Rouges et des Bleus.

La simulation est découpée en tours. A chaque tour, la fourmi se déplace aléatoirement sur une case adjacente (au nord, au sud, à l'est ou à l'ouest mais pas en diagonale). Lorsqu'une fourmi arrive sur une case contenant un oeuf, elle peut décider de l'emporter ou de le laisser.

Quelque soit le modèle de la perception de la concentration des oeufs, on peut établir un ensemble de règles régissant les fourmis lors de la classification du couvain :

1. Si une fourmi ne portant rien, rencontre un oeuf et que la concentration locale de ce type d'oeufs est faible, la fourmi aura tendance à l'emporter. Au contraire, si la concentration est forte, elle aura tendance à le laisser sur place et à continuer sa route.
2. Si une fourmi porte un oeuf et qu'elle arrive sur une case vide, elle aura tendance à déposer cet oeuf si la concentration locale de ce type est forte et à le garder si la concentration est faible.

c) Résultats de la simulation informatique

D'après Michel Casabianca (joignable à l'adresse casa@sweetohm.net) les résultats de la simulation selon les deux modèles de perception (spatiale et temporelle) sont assez convaincants (Voir Fig 4.1. et Fig 4.2.) :

On constate bien que les oeufs sont regroupés par tas d'oeufs semblables. La méthode semble plus efficace avec une perception spatiale qu'avec la temporelle.

Cette simulation permet de valider les modèles du comportement de la fourmi artificielle. Même si cela ne prouve pas que ces modèles soient justes, cela permet de penser qu'ils sont efficaces et conduisent à des résultats semblables à ceux que l'on observe chez les fourmis réelles.

Cette étude est intéressante pour les entomologistes, mais intéresse aussi les informaticiens. En effet, cette simulation met en oeuvre une intelligence collective : à partir de comportements simples, l'ensemble des fourmis classent les oeufs, alors qu'une fourmi en est incapable seule. Cette situation de travail collectif se présente en informatique lorsqu'on fait travailler ensembles plusieurs processeurs.

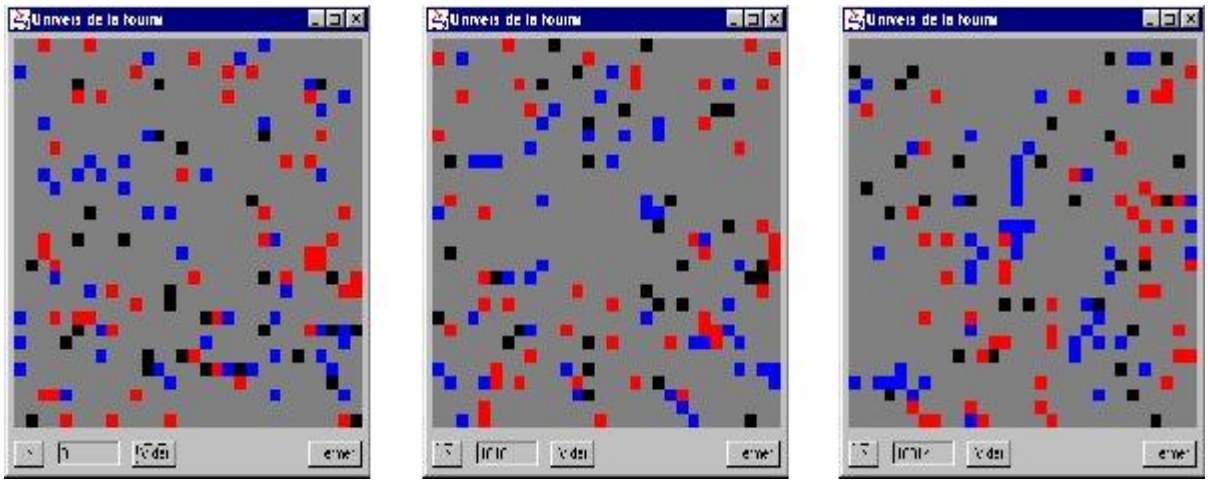


Fig 4.1 Simulation avec perception spatiale au temps initial 0 et 1000 et 10 000

Chaque processeur est simple, mais doit travailler avec ses voisins de manière à ce que la tâche qu'on leur a confié avance.

Les avantages d'une architecture parallèle sont multiples : tout d'abord, à puissance égale, plusieurs petits processeurs coûtent moins cher qu'un unique processeur plus perfectionné (c'est ce qui est mis en oeuvre dans la BeBox), d'autre part, si un processeur vient à tomber en panne, les autres peuvent continuer à fonctionner, ce qui augmente fortement la tolérance à la panne.

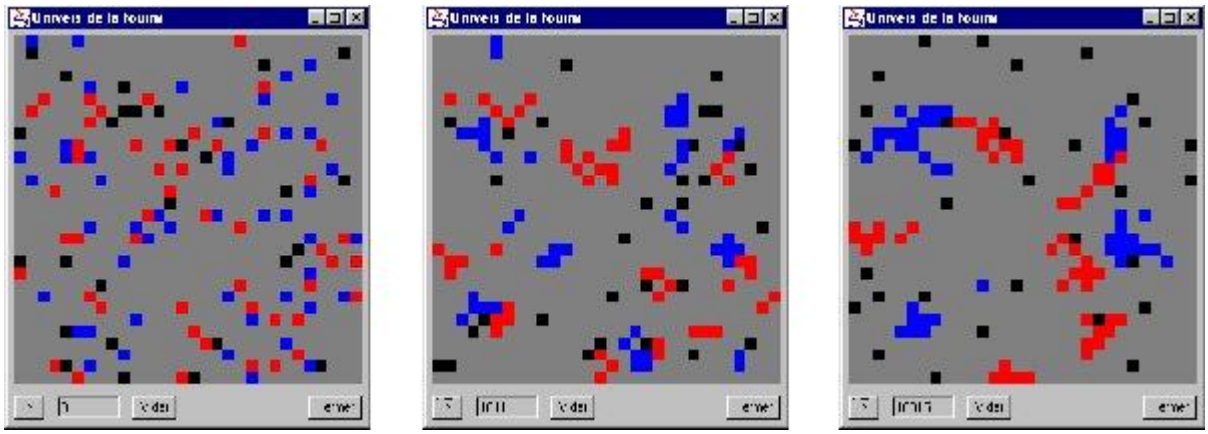


Fig 4.2 Simulation avec perception temporelle au temps initial 0 et 1000 et 10 000

4.3.2.2. Autres travaux

Ces travaux ont débuté en 1990. Il s'agit d'abord des travaux de biologistes s'intéressant à la modélisation des fourmis en termes mathématiques et informatiques, et à l'utilisation concrète de ces modèles. Deneubourg apparaît comme un pionnier dans le domaine de classification d'objets par des fourmis artificielles. Dans [10], il propose avec ses collègues les principes suivants : des fourmis artificielles se déplacent sur un plan. Les objets à rassembler sont répartis sur ce plan. Une fourmi ne dispose que d'une perception locale de ces objets et ne communique pas avec les autres. Au lieu de cela, la configuration des objets sur le sol va influencer leurs actions. Lorsqu'une fourmi rencontre un objet, elle le ramasse avec une probabilité c1 / (c1 + f), où « f » représente la fréquence de rencontre d'objets dans un passé récent. Autrement dit, plus une fourmi rencontre d'objets, moins elle a de chance d'en prendre un (elle se trouve dans une zone avec beaucoup d'objets). Ensuite, une fois un objet ramassé, la fourmi se déplace au hasard dans le plan, et elle

dépose l'objet avec une probabilité f / (c2 + f). Cette probabilité est d'autant plus grande que la fourmi a rencontré récemment des objets. Ces principes relativement simples font qu'il apparaît des regroupements d'objets. L'approche peut être généralisée à plusieurs types d'objets (les fréquences « f » sont spécifiques à chaque type d'objets) : cet algorithme permet alors le regroupement des objets.

Le pas qui sépare le regroupement d'objets de la classification a ensuite été franchi dans [22]. Ils ont adapté l'algorithme présenté précédemment (voir Fig.4.3) : les données sont initialement réparties aléatoirement sur une grille 2D. Chaque fourmi est située dans une case de cette grille et ne perçoit que les données situées dans son voisinage (8 voisins par exemple). Ensuite la fréquence « f » utilisée dans l'algorithme de classification vu précédemment peut être remplacée par une moyenne des similarités entre une donnée « di » portée par une fourmi et les

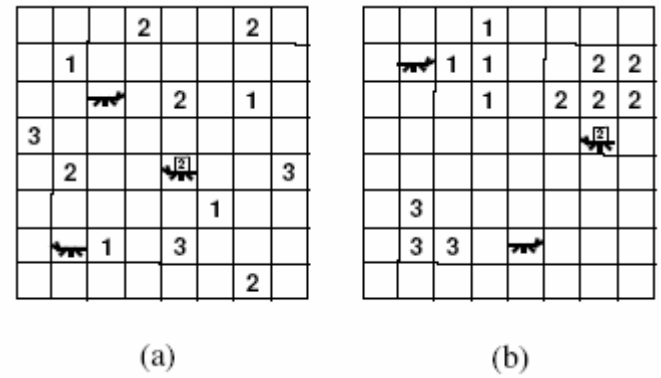


Fig. 4.3 Principe de la classification de données par des fourmis artificielles selon l'algorithme présenté par Lumer et Faieta [22]. En (a) les objets sont répartis aléatoirement sur la grille. Les fourmis peuvent s'en saisir et les déposer dans des cases où la densité d'objets similaires est élevée. Il en résulte la formation de groupes comme en (b).

données « dj » situées dans son voisinage. Une donnée « di » sur la grille est ramassée avec une probabilité d'autant plus grande qu'elle est peu similaire aux données voisines. De la même manière, une donnée « di » portée par une fourmi est plus facilement déposée dans une région comportant des données qui lui sont similaires. Cet algorithme a été depuis étendu à d'autres applications comme le partitionnement de graphes [19] ou la classification de sessions sur des sites Web [1].

4.3.2.3 Approches récentes

Une extension de l'algorithme de Lumer et Faieta [22] a été présentée par Monmarché dans [23]. D'une part, les fourmis peuvent empiler les objets les uns sur les autres dans une même case de la grille. Lorsqu'elles rencontrent un tas d'objets, elles peuvent ainsi se saisir de l'objet le plus dissimilaire. D'autre part, une hybridation a été effectuée avec l'algorithme des K-Means. Cette hybridation consiste à utiliser la séquence d'algorithmes suivante : AntClass, K-Means, AntClass, K-Means. AntClass fournit une partition initiale, les K-Means corrigent des erreurs d'AntClass qui mettraient beaucoup plus de temps à être corrigées avec AntClass seul. Labroche et all. Dans [20] ont introduit un nouveau modèle à base de fourmis pour la classification utilisant le système d'identification chimique des fourmis. Celui-ci est fondé sur la construction d'une odeur coloniale qui est le fruit des apports génétiques, environnementaux et comportementaux. Cette odeur est construite par les individus pour identifier celui qui fait partie du groupe et celui qui doit être rejeté. A partir de ce modèle, un nouvel algorithme de classification a été proposé dans lequel chaque donnée est une fourmi dont l'odeur est déterminée par les valeurs prises par les attributs décrivant cette donnée. Les fourmis effectuent des rencontres aléatoires et décident d'appartenir au même groupe ou non. Il en résulte l'établissement d'une classification. Enfin, dans [3], un nouveau modèle a été introduit, permettant d'effectuer rapidement une classification hiérarchique. Il s'agit de copier la manière dont les fourmis construisent des structures vivantes en s'accrochant les unes aux autres en fonction de critères locaux (la forme de la structure influençant le comportement d'accrochage ou de décrochage). Dans ce modèle, chaque fourmi artificielle

représente une donnée. Les fourmis sont placées initialement à la racine de l'arbre et vont pouvoir se déplacer dans cet arbre et s'accrocher afin de construire une structure hiérarchique dont chaque noeud représente une donnée. L'objectif est de construire automatiquement un site portail (données textuelles) et d'obtenir la propriété suivante : chaque noeud o de l'arbre est une catégorie composée de toutes les données portées par les sous-arbres de o.

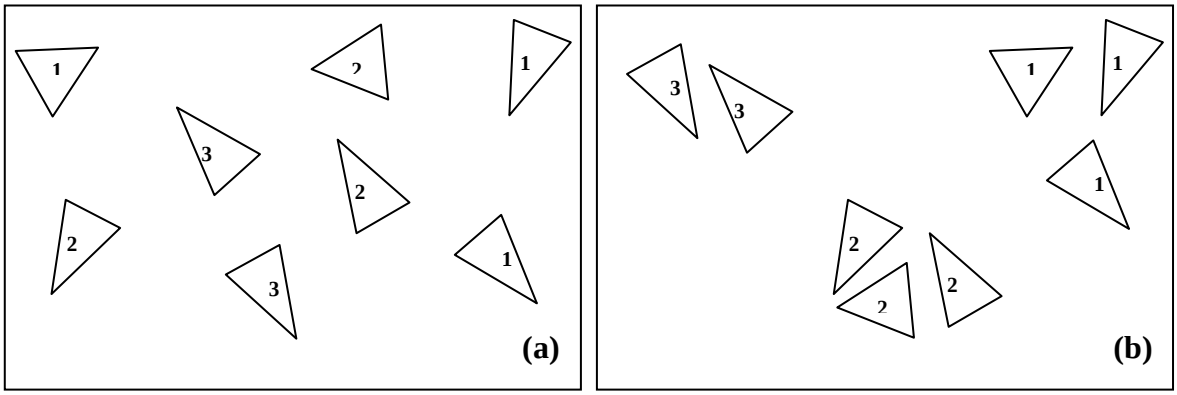


Fig. 4.4 Principes utilisés pour la classification par nuages d'agents. Les agents sont placés initialement avec des coordonnées et des vecteurs vitesse aléatoires (voir (a)). Les mouvements d'un agent dépendent des autres agents perçus dans son voisinage et des similarités entre les données qu'ils représentent. Le comportement local de chaque agent tend à former globalement des groupes d'agents similaires se déplaçant de manière cohérente (voir (b)).

Les sous catégories (représentées par les noeuds connectés à o) doivent être très similaires à leur mère dans l'arbre, mais également les plus dissimilaires entre elles. Les résultats obtenus sont très compétitifs par rapport à la classification ascendante hiérarchique notamment.

4.3.2.4. Intelligence en essaim

L'intelligence en essaim (swarm intelligence) regroupe de nombreux algorithmes à base de population d'agents. Les fourmis artificielles en font partie mais nous nous intéressons, dans cette section, à des algorithmes plus spécifiques qui utilisent les déplacements d'un essaim d'agents pour résoudre un problème. A titre d'exemple, les algorithmes PSO (particle swarm optimization) utilisent un ensemble de particules caractérisées par leur position et leur vitesse pour maximiser une fonction dans un espace de recherche. Des interactions ont lieu entre les particules afin d'obtenir des comportements globaux efficaces. Dans la biologie, de nombreux chercheurs se sont intéressés à la manière dont les animaux se déplacent en groupe. Aucun individu ne contrôle les autres mais pourtant des formes et des comportements complexes peuvent apparaître lors de ces déplacements.

Reynolds dans [28] a été probablement le premier à proposer une utilisation informatique de tels modèles, simulations qui sont utilisés notamment dans l'industrie du cinéma pour donner des mouvements réalistes à des groupes d'individus. Dans ces travaux, chaque individu évolue dans un espace 3D. Il est donc caractérisé par sa position et sa vitesse. Un individu perçoit les autres dans un voisinage donné. Des règles comportementales généralement simples permettent aux individus de se déplacer en groupe, d'éviter des obstacles, etc.

En 1998, ces principes ont été appliqués pour la première fois à un problème de classification [26] (voir figure Fig.4.4). Les agents représentent chacun une donnée. Un agent réagit aux autres agents présents dans son voisinage en tenant compte de la similarité des données. Un agent se déplacera plutôt vers des données qui lui sont similaires. Cette règle comportementale permet donc de former des groupes de données similaires. Dans [24], cet algorithme a été amélioré et

évalué d'une manière plus systématique. Une distance idéale entre individus est définie, distance qui dépend de la similarité entre les données. Un critère d'arrêt est utilisé également en mesurant l'entropie spatiale du nuage d'agents. Cet algorithme a été intégré dans un système de fouille visuelle de données utilisant la réalité virtuelle.

4.3.2.5. Systèmes immunitaires

Les systèmes immunitaires (SI) sont un ensemble de modélisations du système immunitaire humain et animal appliquées à différents problèmes en informatique. Ils utilisent les principes suivants : des agents (lymphocytes) qui génèrent des anticorps vont apprendre à reconnaître le soi du non-soi (les antigènes). Pour cela, ces agents doivent d'abord être engendrés en utilisant un principe de composition de briques élémentaires. Ensuite, ils subissent un test de sélection (dit de sélection négative) : les agents rejetant le soi sont éliminés, et les autres, qui vont rejeter le non-soi, sont gardés. Chaque fois qu'il y a reconnaissance d'un antigène par un anticorps, la présence des lymphocytes générant ces anticorps est favorisée par un processus de sélection par clonage et par la disparition des lymphocytes non stimulés par les antigènes. Ce clonage donne donc lieu à des interactions entre les lymphocytes et peut mettre en oeuvre des mutations. Certains lymphocytes, lorsqu'ils sont souvent utilisés, prennent un rôle d'élément de mémorisation à long terme. Ces systèmes disposent de propriétés complexes car ils sont capables de générer des solutions et de les sélectionner en fonction de leur efficacité selon des heuristiques originales. En ce qui concerne la classification, les principes des systèmes immunitaires sont, à un niveau général, les suivants (voir par exemple le système aiNet [9]) : les données d1,...,dn représentent les antigènes. Ces antigènes sont présentés itérativement au système jusqu'à l'obtention d'une condition d'arrêt. On suppose que les données sont numériques, et donc qu'un antigène est un vecteur de dimension n. A chaque itération, l'antigène présenté va activer des anticorps (assimilés dans cette modélisation à des lymphocytes-B). Un anticorps est également représenté par un vecteur de dimension n. Les anticorps suffisamment proches de l'antigène (au sens de la distance euclidienne) vont subir des clonages avec mutation (interaction anticorps/antigènes) afin d'amplifier et d'affiner la réponse du système. Egalement, ces anticorps vont subir une sélection (interaction anticorps/anticorps) : ceux qui sont trop proches les uns des autres seront diminués en nombre. Après ces itérations, le système converge en plaçant des anticorps (qui agissent comme des détecteurs) de manière judicieuse et en nombre adapté aux données. D'autres modèles plus complexes existent. Ainsi dans [17] le système utilise plusieurs niveaux de cellules et d'interaction (anticorps, lymphocytes, cellules de mémorisation). Dans [25], ce même système est généralisé et amélioré en utilisant des fonctions d'appartenance floue plutôt qu'une distance euclidienne et un seuil.

4.4. Problème de formation de Tas en robotique collective

Le problème de formation de Tas (comme d'ailleurs le problème des écoulements), revient assez souvent en certains domaines de la littérature informatique. Il s'agit de problèmes types qui paressent simples mais qui sont difficiles à modéliser selon les approches classiques connues (que se soit mathématique ou algorithmique). Les automates cellulaires basés sur le principe de l'émergence ont par contre montré leur force d'expression, ainsi que leur efficacité dans cette perspective.

D'après quelques définitions génériques de dictionnaires et encyclopédies, le tas exprimerait l'accumulation de choses de la même espèce. Le problème de formation de tas dans ce cas consiste en un regroupement désigné ou non (s'ils n'ont pas de relation les uns avec les autres, comme des choses quelconques) d'éléments, …, ou de particules (s'ils participent par exemple à une même composition), éparpillés sur une surface donnée. L'un des problèmes clé dans ce cas (du moins à notre avis) est la localisation ou la désignation du point de formation du tas. Ce point de rencontre ou de rendez vous (s'il s'agit de robots mobiles par exemple) n'est pas évident à trouver pour des entités stupides ou de

moindre intelligence, sauf s'il est préétabli par défaut au niveau de chaque entité. On n'a pas encore de preuve expérimentale suffisante, mais on estime que ceci peut se faire de deux manières :

- En convenant les coordonnées de rencontre pour toutes les entités mobiles. Le problème devient alors : revenir à l'endroit initial quelque soit l'endroit où l'on se trouve en évitant les obstacles et en ramenant un élément du tas à former, qu'il faudra trouver sur le chemin, ou en le cherchant avant de revenir (mémorisation des coordonnées d'un emplacement).
- En convenant la direction à suivre pour toutes les entités mobiles. Le problème devient alors : suivre toujours la même direction en évitant les obstacles et en ayant un élément du tas à former, qu'il faudra trouver sur le chemin, ou en le cherchant avant d'emprunter la direction en question, et déposer l'élément transporté si on rencontre une quantité suffisante des éléments de la même espèce (mémorisation d'une direction).

Il est à noter que aussi bien le point de rencontre que la direction a à suivre n'est pas fixé au préalable par une entité externe au système, mais il émerge plutôt d'une sorte de travail coopératif qui se déroule entre les concernés.

Les expériences menées jusqu'à lors (en adoptant d'autres idées) montrent que ces entités mobiles conçues à base de modèle d'agent réactifs, imitent plutôt la nature (comme les fourmis entre autres) en construisant tout d'abord des tas de petites tailles qui grandissent au fur et à mesure que le temps passe pour fusionner en tas d'une plus grande ampleur (voir Fig 4.5).

L'approche que nous avons suivie ne puise pas dans cet état d'esprit. On peut dire même quelle est diamétralement opposée. Étant donné que nous n'allons pas doter nos entités mobiles de comportement basique que nous allons vérifier expérimentalement par la suite, mais plutôt fixer le but à atteindre via une formulation prédonnée, et chercher parmi toutes les combinaisons de comportements possibles chez nos entités mobiles, celles qui nous permettent d'aboutir à notre fin (la formation de Tas).

Notons que l'idée de chercher parmi toutes les combinaisons possibles de comportements, celle qui convient le mieux n'est pas toujours possible (problème d'explosion combinatoire). Ceci est rendu possible en utilisant les algorithmes génétiques.

Ajouter à ceci, la dimension du réalisme qu'on doit prendre en considération, étant donné qu'on travaille dans le domaine de la robotique collective, et le résultat auquel on va aboutir, via la simulation, doit être vérifié sur le terrain par des robots mobiles réels. En résumé, le problème peut être posé comme suit :

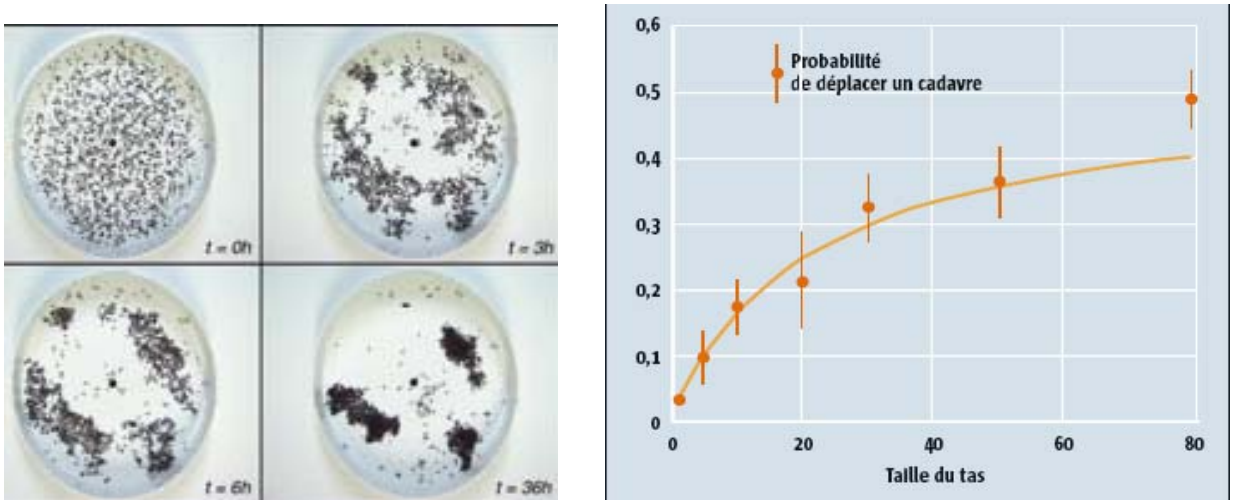


Fig 4.5 AGRÉGATION DES CADAVRES CHEZ LA FOURMI MESSOR SANCTA. Au début de l'expérience, 1 500 cadavres de fourmis sont éparpillés au hasard (en haut à gauche). Après trente-six heures, les ouvrières ont formé trois tas (en bas à droite). La probabilité de déposer un cadavre sur un tas déjà formé augmente rapidement avec la taille des tas (graphique). PHOTO G. TERAULAZ

On veut faire émerger la tâche de formation de tas engendrée par une coopération de robots mobiles, autonomes et de moindre intelligence, simulés via un collectif d'agents réactifs dans un environnement plat et fermé dans les deux directions.

Pour ce faire nous allons utiliser :

- les automates cellulaires pour réaliser le système de contrôle des robots mobiles qui est le même chez tous les robots,
- une approche évolutionniste, basée sur le principe d'un algorithme génétique pour converger vers le système de contrôle le plus adéquat à atteindre notre objectif.

4.4.1. Principe du fonctionnement réactif d'un robot mobile

Chaque robot possède un certain nombre de senseurs, et de déclencheurs d'actions, lui permettant de réagir au changement de son milieu externe. La donnée captée par un senseur est à l'état brut sous forme continue. Afin de pouvoir la traiter, une solution proposée par Kube et Zhang [42] consiste à réaliser sur le senseur réel un ensemble de senseurs virtuels, et indépendants, produisant une valeur discrète (qualifiée parfois d'orthogonale). Dans leur proposition Kube et Zhang [42] distinguent entre l'orthogonalité spatiale et celle modale (voir Tab 4.1) :

- Dans l'espace, les senseurs orthogonaux ne regardent pas la même région de l'espace (étant donné qu'ils sont situés à des endroits différents. Voir Fig 4.7).
- Les senseurs modaux orthogonaux ne regardent pas le même type de données. Dans notre exemple ci-dessous, l'un détecte les objets de couleur bleue, alors que l'autre détecte les objets de couleur rouge (Voir Fig 4.7).

Ceci veut dire que la donnée introduite par les senseurs (qui était à l'état brut continu) est transformée par les senseurs virtuels en une donnée discrète plus facile à utiliser (voir Fig 4.8).

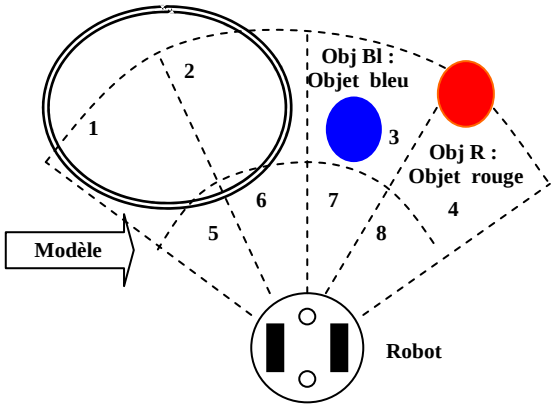


Fig. 4.6 Représentation d'un robot avec son espace orthogonal

	Obst	Obj Bl	Obj R
1	1	0	0
2	1	0	0
3	0	1	0
4	0	0	1
5	0	0	0
6	0	0	0
7	0	0	0
8	0	0	0

Orthogonalité spatiale

Orthogonalité modale

TAB 4.1 Numérisation de l'espace Orthogonal sensoriel.

La sortie du système de commande ordonnant au robot de réagir d'une certaine manière est discrète dans sa première phase, dans le sens ou un robot doit choisir entre une panoplie de comportements basiques prédéfinis et finis [43], une fois ceci fait, le résultat discret est converti en une commande réelle destinée aux actionneurs physiques.

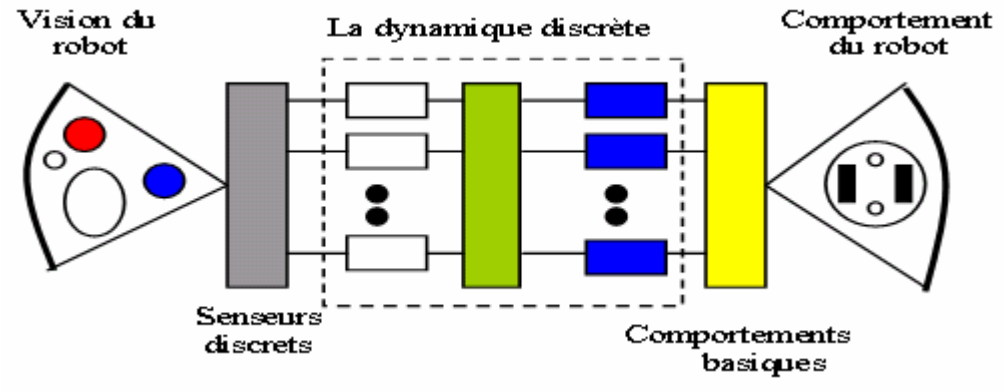


Fig. 4.7 Schéma du model de contrôle de robot autonome ; La dynamique interne est discrète. Le rectangle gris représente les senseurs physiques, le rectangle jaune les actionneurs physiques, le rectangle vert le schéma d'arbitrage, les petits rectangles blancs les n senseurs discrets SN1, .., SNn, et les petits rectangles bleus les m

On peut considérer la transformation des données continues en données discrètes comme une phase de prétraitement, et la conversion d'un comportement basique en une commande réelle destinée aux actionneurs physiques comme une phase de post-traitement. Une fois que le pré et le post traitement sont installés, la difficulté demeure au niveau de l'établissement d'un schéma approprié d'arbitrage, accueillant en entrée une séquence discrète (de taille N), et fournissant en sortie une séquence discrète appropriée (un des comportements de base CBi / i = 1, m).

Pour ce faire, on opte pour une table de correspondance discrète, mémorisant les relations sensori-motrices, c-à-d associant chaque entrée de la table (représentant une vision locale du robot mobile) à un comportement basique choisi entre les m pré-proposés, et qu'on nommera par la suite : table de consultation. Noter qu'on peut tout à fait l'assimiler à un automate cellulaire.

En faite, ce travail est inspiré de celui effectué par T. D. Barfoot et G. M. T. Deleuterio de l'université de Toronto [35], et que lui-même est la suite d'un autre effectué par un groupe de l'institut de Santa Fe, au Mexique concernant l'évolution des automates cellulaires. Le travail essentiel de ce dernier est qu'il a pu montrer que les algorithmes génétiques étaient capables de faire évoluer des automates cellulaires réalisant des tâches qui exigeaient une coordination globale. En ce qui concerne la robotique collective, atteindre un tel objectif est contraint par une difficulté supplémentaire se rapportant au respect du réalisme de l'environnement où évoluent les robots mobiles (il est hors de question d'utiliser par exemple de la phéromone pour guider le comportement des robots étant donné que les robots actuels sont incapables aussi bien de la produire que de la détecter). Ce type de contrôle au niveau des robots est qualifié souvent de réactif, c-à-d pour chaque séquence d'entrée possible, le schéma de l'automate cellulaire stocke une valeur discrète de sortie. Autrement dit pour chaque vue locale perçue par le robot, correspond un des comportements de base préconçus. Donc, à chaque changement local dans sa vue limitée, le robot recherche le comportement adéquat, prescrit dans sa table de correspondance, et l'exécute. Ce changement peut être dû à l'apparition ou la disparition d'éléments dans son voisinage, comme il peut être dû au déplacement du robot lui-même.

Pour les senseurs binaires (1 : détection d'un type d'objet, 0 : non détection), la table de consultation est de 2^n (n étant le nombre de senseurs). On remarque que la taille de la table est proportionnelle exponentiellement au nombre des senseurs; si le nombre de senseurs est important, la taille de la table de consultation devient très grande. Ceci implique que l'espace de recherche peut devenir immense, entraînant un temps de recherche au-delà d'un délai raisonnable. Cette approche est rendue faisable dans ce cas, en réduisant le nombre de senseurs. Un autre point important est celui du nombre m des comportements de base pré-établis : ce nombre n'affecte pas la taille d'une table de consultation donnée; par contre il affecte le nombre total des tables envisageables, qui est de m^k

avec $k = 2^n$ pour les senseurs binaires. Afin de limiter le nombre de ces tables, on réduit aussi le nombre des comportements à une valeur donnant la solution en un temps acceptable.

On peut dire dans ce cas, que aussi bien le nombre de senseurs discrets que celui des comportements de base influe directement sur la taille de l'espace de recherche.

4.4.2. Principe de l'approche évolutionniste utilisée

Toute l'approche proposée, repose sur la découverte des tables de consultation, participant au succès d'une collectivité de robots identiques à réaliser une tâche exigeant une coopération globale.

En somme, la discipline de la robotique collective toute entière s'appuie sur la découverte d'un ensemble de règles locales produisant un comportement global désiré. Plusieurs questions intriguent les chercheurs à ce stade :

- Existe-il réellement ces règles locales ou ces comportements de base ?
- Dans le cas affirmatif, on se demande pourquoi (c'est bien) ces règles et non pas d'autres ?

On peut ajouter à ceci une autre question, aussi intéressante que les deux premières :

- si on peut établir une relation entre tâches à l'échelle macro, existe-il une relation équivalente à l'échelle micro ?

Et, qui nous permettrait de déduire les règles faisant émerger l'une de ces tâches en relation, sans passer par le processus qui les découvre.

La méthode la plus évidente pour résoudre le problème évoqué dans la première question est de proposer des règles locales manuellement, et de les tester à chaque fois, jusqu'à obtention du but voulu. Ceci s'avère difficile, voire irréalisable dans certains cas.

Généralement, deux méthodes (entre autres : recuit simulé, …) sont employées pour trouver les règles locales, qui produisent des comportements globaux intéressants, à savoir :

- L'optimisation,
- L'apprentissage par renforcement.

Dans le travail présenté ici, on s'intéresse plutôt à la première méthode en ciblant en particulier les techniques d'optimisation évolutionniste globale, plus connue sous le nom d'algorithmes génétiques.

Les algorithmes génétiques (largement utilisés actuellement, pour résoudre des problèmes d'optimisation) sont déjà présentés dans le chapitre 2. Ce que nous allons décrire ici, c'est plutôt comment les utilise t-on pour trouver l'ensemble des réactions des robots conduisant à l'accomplissement d'une tâche spécifique. Dans notre cas, on considère aléatoirement une population initiale de P tables de consultation représentant chacune un automate cellulaire jouant le rôle d'arbitre dans un entraînement donné, d'une collectivité de robots identiques (tous intégrant la même table de consultation). Ces P tables de consultation représentent en termes de génétique P chromosomes, générés tous aléatoirement. A chaque génération une évaluation dite de performance ou de fitness est calculée pour chaque table de consultation, permettant de déterminer si elle peut faire partie de la prochaine génération. Il est à noter que cette évaluation n'est attribuée qu'après un certain délai d'entraînement suffisant des robots simulés sur le terrain ; Si elle est classée dans les k premiers de sa génération elle fera partie directement de la population de la prochaine génération, sinon on s'intéressera à sa progéniture, en effectuant des croisements d'emplacement simple à un endroit aléatoire avec une probabilité Pc et des mutations aléatoires d'emplacement avec une probabilité Pm par emplacement sur les P – k individus qui restent après l'opération de sélection. Le choix des valeurs des paramètres Pc et Pm prises parmi des distributions aléatoires uniformes joue un rôle décisif dans le succès de tout le processus de recherche. Pm est une valeur généralement très

faible par rapport à Pc. Ce qui s'explique par le fait que dans la réalité (source de notre inspiration), la mutation ne se produit pas aussi souvent que le croisement.

4.5. Problème de classification monocritère

En adoptant la même approche, déjà présentée dans la section précédente, nous allons décrire ici la macro tâche de classification monocritère (selon le principe de formation multi-tas) qu'on veut faire émerger, à partir d'un ensemble de comportements de base, dont on dote chaque robot simulé de notre collectivité [56]. Il s'agit de regrouper les objets manipulés par ces robots selon l'une de leurs propriétés. On peut penser à la forme de ces objets, à leur couleur, à leur poids, à leur volume, etc. S'il s'agit de caractéristiques physiques. On peut aussi songer à toute autre caractéristique logique, comme leur désignation, leur famille (si on s'intéresse à des produits industriels par exemple). Toute cette panoplie de propriétés pour nous est ramenée à des valeurs numériques ou à une forme de codification qu'on exploite pour distinguer entre les objets en question. Notre objectif étant de trouver les réactions des robots {perception locale, comportement de base} qui vont les pousser à regrouper ces objets dans un environnement plat (à deux dimensions), de préférence à des endroits différents, selon la propriété désignée. Si les robots simulés ont bien fait leur travail, on retrouve en fin du parcours, chaque groupe d'objets ayant la même valeur de la propriété choisie entassé dans un endroit à part (voir Fig.4.8). Donc, idéalement les constituants de chaque tas formé doivent vérifier deux propriétés essentielles à la formulation de l'équation de fitness qu'on verra un peu plus loin dans la première solution proposée :

- Appartenir au même voisinage (qu'on désignera plus tard par cellule).
- Avoir la même valeur du critère de classification considéré.

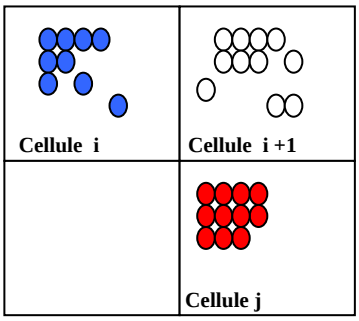


Fig. 4.8 Classification selon la propriété « couleur »; Formation idéale de trois TAS, chaque tas vérifie que tous ses éléments appartiennent à la même cellule, et que tous les objets appartenant au même tas, ont la même valeur de couleur, rouge blanche ou bleue (La variation de couleur au niveau de chaque cellule est réduite à zéro). Noter que les trois cellules sont différentes

4.4.3.1. L'exemple de couleur

Si on prend comme propriété, celle des couleurs, les objets appartenant au même voisinage doivent avoir la même couleur (voir Fig. 4.8.). Il est clair que chaque couleur est représentée par une valeur donnée.

Pour cet exemple l'opérateur de l'évaluation de fitness doit donner la valeur maximale. Généralement, on normalise les valeurs de l'opérateur d'évaluation de telle façon à les avoir tous dans la fourchette des réels appartenant à [0, 1]

4.6. Solutions proposées pour l'approche monocritère

Pour pouvoir évaluer le taux de performance des robots mobiles, on doit en premier lieu s'intéresser à modéliser l'opérateur d'évaluation, qui a pour rôle d'attribuer une note à chaque individu de la population, en fonction de son adaptabilité aux critères de l'ensemble des solutions. Autrement dit, il s'agit comme dans le monde réel, d'une comparaison entre le phénotype de chaque individu et ses

critères, le génotype étant exclu de cette étape. Cet opérateur est plus connu sous le nom de *fonction de fitness* qu'on peut formuler mathématiquement comme suit : ***Note = F(phénotype)***. Notre but à ce stade est de formuler l'équation F. Pour ce faire, nous allons suivre trois approches; dans la première approche nous allons proposer une nouvelle formulation de la fitness, et dans les deux autres, nous allons plutôt adapter l'équation de fitness utilisée par Barfoot et all dans [35, 36] à notre problème de classification.

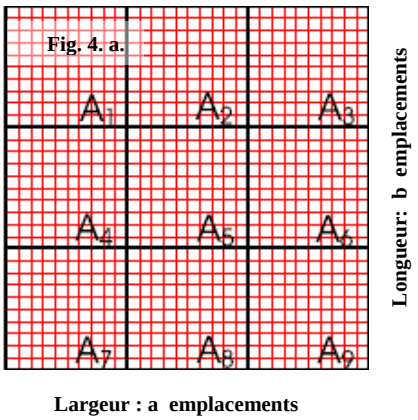
4.6.1. Première solution

Nous allons commencer par avancer un certain nombre d'hypothèses, permettant de décrire l'environnement de simulation :

- le monde de simulation est constitué de a x b emplacements (physiquement chaque emplacement ne peut contenir qu'un seul objet ou un seul robot à la fois ou bien un robot portant un objet). Le monde ainsi constitué est fermé, et donne l'impression de se déplacer sur une sphère.
- les a x b emplacements sont structurés en J cellules,
- sur ces a x b emplacements se déplacent w robots simulés,
- sur ces a x b emplacements on a disposé aléatoirement t * n objets appartenant à l'une des t couleurs différentes considérées, (voir Fig .4.a. et Fig . 4.b.)

Dans notre exemple, on suppose qu'on a seulement trois couleurs (bleue, blanche, rouge), et qu'on a n objets de chaque couleur. Deux critères d'évaluation sont alors à considérer :
- Le taux de regroupement des objets, dans la même cellule,
- Le taux de variation des couleurs dans la même cellule.

La performance dans ce cas est inversement proportionnelle au taux de variation de couleurs, c-à-d que plus le taux de variation de couleur diminue dans une cellule donnée, et plus notre performance pour réaliser la tâche de classification augmente au sein de cette même cellule. Contrairement, le taux de regroupement est proportionnel à la performance, dans le sens où plus le taux de regroupement augmente dans une cellule, et plus notre performance a tendance à augmenter dans cette même cellule.
Notre opérateur de performance doit comporter, les deux évaluations, tout en sachant que la diminution ou l'augmentation de l'un des deux taux influence l'autre, et l'annulation de l'un des deux annule l'autre.



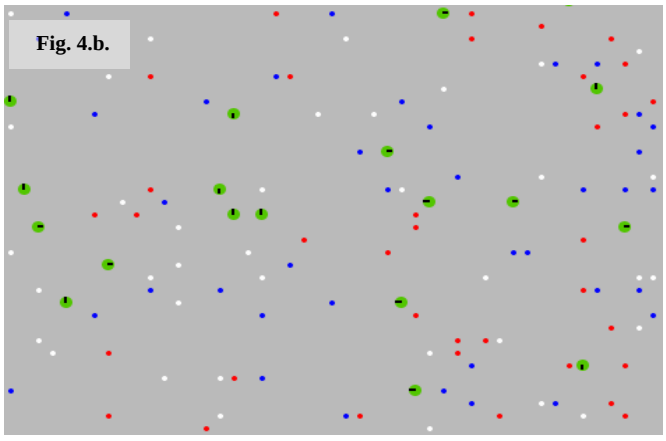


Fig. 4.a. l'environnement où évolue la collectivité des robots simulés est structuré en cellules Aj, dans l'exemple ci-contre j vari de 1 à J = 9.

Fig. 4.b. partie de l'environnement où évolue un certain nombre de robots simulés, symbolisés par des disques verts, dont l'orientation est repérée par un trait noir, et un certain nombre d'objets, de trois couleurs différentes, symbolisées par des disques simples de couleurs rouge, bleue, et blanche.

Posons ce qui suit :

- Nc : le nombre de couleurs,
- Val(Ci) : la valeur de la iéme couleur Ci,
- Cm = $\sum_{i=1}^{Nc} Val(Ci)$ représente la couleur moyenne par rapport au monde simulé.
- Cm(Aj) = $(\sum_{i=1}^{Nc} Tci(Aj) \times Val(Ci)) / Nc$ représente la couleur moyenne au niveau de la cellule Aj.

Sachant que Tci(Aj) est le taux de regroupement d'objets de couleur ci dans la cellule Aj, calculé par rapport au nombre total des objets de couleur ci. Dans ce cas :

- Tci(Aj) = Nci (Aj) / Nci, où Nci représente le nombre d'objets de couleur ci,
- Tv(Aj) = $\sum_{i=1}^{Nc} \frac{|k + Cm(Aj) - Val(Ci)|}{|k + Cm - Val(Ci)|}$ représente le taux de variation de couleur au niveau de chaque cellule. La variable K permet de translater le problème vers R+, tous en conservant l'échelle d'évaluation. Ceci nous permet de conserver la valeur de Tv(Aj) dans l'intervalle [0 1]. Donc K = 5000 * |Cm(Aj) – Val(Ci)|
- Pj = $(\sum_{i=1}^{Nc} Tv(Aj) Tci(Aj)) / Nc$ représente l'évaluation du taux de variation de couleur et de la concentration d'objets au niveau de chaque cellule j, concernant les Nc couleurs,

Dans ce cas la fonction de fitness peut être exprimée comme suit :

Ft = 1 + $(\sum_{j=1}^{J} Pj \ln Pj) / (J / e)$ **Equation 16**

e : constante exponentielle

t : exprime le tième entraînement concernant la même table de consultation. J/e nous permet de normaliser la valeur de la fonction Ft dans l'intervalle [0, 1].

Pour un nombre Nb d'entraînements (représentant le nombre de conditions initiales), la fonction Ft se transforme en une moyenne statistique plus représentative de la valeur théorique de fitness de la table de consultation testée.

Fm = $(\sum_{t=1}^{Nb} Ft)$ / Nb **Equation 17**

En ce qui concerne la formation de Tas, on a :

- Le taux de mélange ou de variation de couleurs le plus grand est atteint quand touts les objets de toutes les couleurs sont ensembles dans la même cellule.
- Le taux de regroupement le plus élevé est obtenu dans ce même cas. A ce stade on doit distinguer entre deux cas : - le premier cas représente celui où la cellule est considérée comme une composante constituée d'autres emplacements, et le mélange est pris en compte par rapport à ces emplacements, - le deuxième cas est celui où l'on considère que la cellule est un tout, dans ce cas le regroupement de touts les objets dans cette cellule représente une classification, et la fonction de fitness vaut 1.
- Notre situation idéale par rapport à une couleur donnée est obtenue, quand on arrive à avoir dans une cellule donnée, le plus grand taux de regroupement concernant cette même couleur, et le taux de variation le plus faible (pratiquement zéro). La fitness dans ce cas est localement maximale,
- La situation idéale globale consiste à avoir une formation d'un nombre de tas égal au nombre de couleurs différentes donné. Cette formation peut se faire dans la même cellule ou dans des cellules séparées. Dans ce cas l'évaluation de la fitness donne 1.

- La situation la plus équilibrée (si on veut faire un parallèle avec l'entropie) concerne la répartition équitable de touts les objets de chaque couleur sur les J cellules. Dans ce cas l'évaluation de la fitness donne une valeur très faible qui tend vers zéro.

Tous ces cas, qu'on peut qualifier de cas à l'extrême, ont été vérifiés formellement, et expérimentalement par l'équation proposée (voir Equation 16).

4.6.2. Deuxième solution

On peut assimiler la classification dans le cas du problème étudié ici, à une formation multi-tas. La formation de tas devient alors un cas particulier dans notre processus de classification. Dans l'exemple présenté dans la section précédente, la classification des objets selon la propriété « couleur », sachant qu'on a trois couleurs (rouge, bleue, et blanche) nous conduit vers la formation de trois tas, chacun concernant une des trois couleurs en question (voir Fig. 4.8). A première vue, on peut résumer le procédé de classification proposé comme suit :

- Regrouper les objets de couleur rouge, en suite,
- Regrouper les objets de couleur bleue, en suite,
- Regrouper les objets de couleur blanche.

La même table de consultation (trouvée par le processus évolutionniste expliqué ci-avant) est utilisée pour les trois regroupements, en ne faisant que changer la couleur des objets à amasser à chaque fois. Les autres objets des autres couleurs seront considérés pour les robots simulés comme des obstacles à éviter (Attitude que le robot simulé adopte envers un autre robot simulé). Il est important de considérer les autres objets (des autres couleurs), sinon la contrainte du réalisme de notre environnement sera faussée. Par exemple, un robot n'a pas le droit de déplacer un objet vers un endroit s'il est réellement occupé (voir Fig 4.9). Ajouter à ceci, le fait que les tas formés peuvent cerner quelques objets d'une autre couleur que la leur dans un endroit donné où il serait impossible aux robots d'arriver (voir fig 4.10), il serait dans ce cas plus intéressant de former les trois tas simultanément (et non pas en cascade). Ceci implique qu'au niveau du processus évolutionniste, à la recherche de la meilleure table de consultation, donnant une fitness proche de 1, on doit, pour la même table de consultation, entraîner les robots à regrouper simultanément les objets rouges, les objets bleus, et les objets blancs, puis évaluer la situation après un certain temps t, pour chaque couleur à part, et faire ensuite la moyenne ou prendre le minimum, de ces trois évaluations. La meilleure table de consultation trouvée à la fin du processus exprime que les robots sont capables de former les trois tas, et même si on rencontre un état bloqué (voir Fig 5.b.), ce ne sera que momentanément.

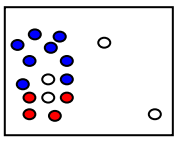


Fig. 4.9. La formation finale des tas rouge, et bleu, empêche la formation du tas blanc, puisque une partie des objets blancs est coincée à l'intérieur, entre les deux autres tas.

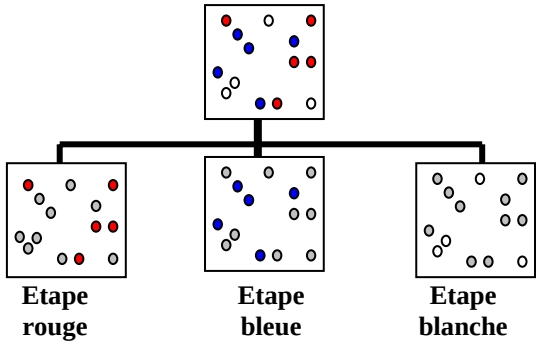


Fig. 4.10. À chaque étape couleur, les robots distinguent entre les objets de la couleur qu'ils doivent regrouper, et les autres objets qui représentent pour eux des obstacles (cercle en gris). Notons que chaque étape couleur, influence les autres étapes (étape rouge, étape bleue, et étape blanche).

Les comportements associés aux agents, diffèrent aussi, par rapport à la première méthode. Au niveau du déplacement, l'agent porteur d'un objet doit le déposer s'il est de la couleur-étape autorisée, et s'il trouve une place, sinon il tente de se déplacer. Celui qui ne porte pas d'objet, doit prendre un objet de la couleur de son étape couleur, s'il ne trouve pas d'objet à prendre dans son environnement local, il tente de se déplacer. Ainsi, si un objet est coincé quelque part, à la prochaine étape-couleur ou à la suivante, un chemin pourra se dégager pour qu'il devienne accessible. La fitness calculée par rapport à chaque couleur à la fin de chaque entraînement suit la formule de schanon présentée par Barfoot et all. dans [35, 36] (voir équations 18) :

$$Fik = 1 + \sum_{j=1}^{J} Pjk \ln Pjk \Big/ \ln J \qquad \textbf{Equation 18}$$

$$Pjk = n(Ajk) / \sum_{j=1}^{J} n(Ajk)$$

Après un temps d'entraînement *t* on calcul les Fik par (18), puis

$$Fi' = \text{Min}_{k \text{ allant de 1 à K}} \{Fik\} \qquad \textbf{Equation19}$$

On peut aussi utiliser la moyenne :

$$Fi' = \left(\sum_{k=1}^{K} Fik\right) / K \qquad \textbf{Equation 20}$$

J représente le nombre de cellules, n (Ajk) le nombre d'objets de couleur k, dans la cellule Aj, et K le nombre de couleurs. Il est clair que pour m entraînements selon m conditions initiales différentes, Fi' se transforme en une moyenne Fm' en utilisant l'équation (2).

Après l'évaluation, on pourra appliquer les opérateurs de sélection, de croisement et de mutation, à la recherche de nouveaux espaces de solutions, éventuellement plus intéressants. Ce procédé est répété pour plusieurs générations. A la fin, on doit pouvoir aboutir à la solution, qui consiste à trouver la table de consultation qui permet de faire émerger la classification d'objets.

4.6.2. Troisième solution

L'idée de la troisième solution dérive de celle de la deuxième solution, dans le sens où l'on généralise la formation de Tas à la formation de multi-Tas, en utilisant toujours la fonction de fitness mentionnée par Barfoot et all. dans [35, 36]. Cependant on change la phase d'expérimentation et d'évaluation, en entraînant les agents-robots pendant un certain temps suffisant sans distinguer entre des étapes-couleurs, c'est-à-dire sans prendre en compte le critère de classification, à savoir : « La couleur ». C'est au moment de l'évaluation qu'on distingue entre les trois couleurs utilisées (rouge, bleue, et blanche) par trois fonctions de fitness distinctes (fr, fb, fbl), chacune ressemblant exactement à celle utilisés par Barfoot et all. dans [35, 36]. La fonction de fitness globale (fi) permettant d'évaluer l'état de l'environnement, après un entraînement suffisant des agents-robots sur le terrain, étant donnée par la moyenne des trois fonctions précédentes :

fi = (fr + fb + fbl) / 3.

En la généralisant, on retrouve la même forme que celle de l'équation (Equation 20). Le reste du procédé évolutionniste se déroule exactement comme pour les deux solutions précédentes.

4.7. Problème de classification multicritères

Nous allons décrire ici la macro tâche de classification multicritères (selon le principe de formation multi-tas) qu'on veut faire émerger, à partir d'un ensemble de comportements de base, dont on dote chaque robot simulé de notre collectivité. Il s'agit de regrouper les objets manipulés par ces robots selon deux voir plus de leurs propriétés.

On peut combiner entre plusieurs caractéristiques comme la forme de ces objets, la couleur, le poids, le volume, etc. Il s'agit de caractéristiques physiques. On peut aussi penser à combiner tout autre caractéristique logique, comme la désignation, la famille, etc. Notre objectif dans ce cas (d'ailleurs comme dans le premier problème) étant de trouver les réactions des robots {perception locale, comportement de base} qui vont les pousser à regrouper ces objets dans un environnement à deux dimensions, selon les propriétés désignées. Si les robots simulés ont bien fait leur travail, on retrouve à la fin du temps d'entraînement chaque groupe d'objets ayant les mêmes valeurs des propriétés choisies entassé dans un endroit à part (voir Fig.4.11). Donc, idéalement les constituants de chaque tas formé doivent vérifier deux critères :

- Appartenir au même voisinage (cellule).
- Avoir les mêmes valeurs des critères de classification considérés.

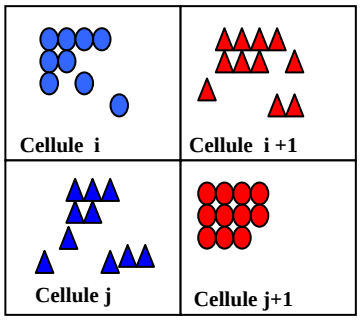


Fig. 4.11 Classification selon la propriété « couleur » et la propriété « forme »; Formation idéale de 4_TAS, chaque tas vérifie que tous ses éléments appartiennent à la même cellule, et que tous les objets appartenant au même tas, ont la même valeur de couleur : rouge ou bleue et la même valeur de la forme : cercle ou triangle.

4.8. Solution proposée pour l'approche multicritères

Comme dans les solutions deux et trois du problème de classification monocritère, on peut assimiler la classification dans le cas du problème étudié ici, à une formation multi-tas. La formation de tas devient alors un cas particulier dans notre processus de classification. Dans l'exemple présenté dans la section 4.7, on a classifié les objets selon les propriétés de « couleur », et de « forme ».

L'idée donc est de généraliser la formation de Tas à la formation multi-Tas, en utilisant toujours la formulation de fitness mentionnée par Barfoot et all dans [35, 36]. On entraîne les agents-robots pendant un temps suffisant, et au moment de l'évaluation on distingue entre les couleurs utilisées : rouge et bleue, et les formes utilisées : cercle et triangle (voir fig 4.11) par des fonctions de fitness distinctes. Pour l'exemple de la section précédente on utilise quatre fonctions de fitness (ftr, ftb, fcr, fcb). La fonction de fitness globale (fi) permettant d'évaluer l'état de l'environnement, après un temps d'entraînement suffisant des agents-robots sur le terrain, étant donnée par la moyenne des quatre fonctions précédentes :

fi = (ftr + ftb + fcr + fcb) / 4. Sachant que :

ftr veut dire fitness des objets vérifiant les critères : triangle et rouge
ftb veut dire fitness des objets vérifiant les critères : triangle et bleu
fcr veut dire fitness des objets vérifiant les critères : cercle et rouge
fcb veut dire fitness des objets vérifiant les critères : cercle et bleu

En la généralisant on retrouve la même forme que celle de l'équation (Equation 20), avec K représentant le nombre de combinaison entre couleur et forme. Le reste du procédé évolutionniste se déroule exactement comme pour les solutions précédentes.

4.9. Regroupement sélectif exclusif au mieux

Le regroupement sélectif exclusif au mieux consiste en une formation de tas selon un ou plusieurs critères en dispersant le reste. Pour nos agents-robots le reste sera compris dans le sens d'objets à écarter au mieux de la place où le tas se forme. Donc, l'agent-robot doit distinguer entre ce qu'il doit regrouper (déplacer et déposer les uns à côté des autre dans un même voisinage) et ce qu'il doit écarter (mettre hors du périmètre de la zone où se forme le Tas). Si on reprend l'exemple présenté dans la section précédente, le résultat virtuel de la requête de regroupement d'objets ayant la couleur rouge peut être imaginé comme suit :

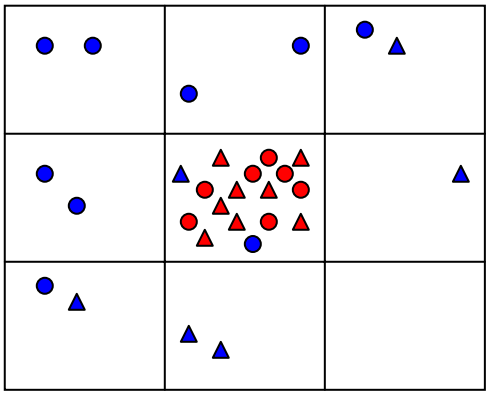


Fig 4.12 La cellule centrale contient le résultat type de la requête : regrouper exclusivement au mieux les objets de couleur rouge. Noter que les autres objets se trouvent éparpillés au maximum hors de la cellule centrale.

Pour qu'on puisse converger vers la tâche ainsi considérée, le processus évolutionniste utilisé peut être modélisé à travers une fonction qu'on peut qualifier de bi-objectifs, dans le sens où l'on doit maximiser la formation d'un Tas selon un ou plusieurs critères donnés dans un voisinage donné (cellule), et en même temps minimiser la formation en Tas au mieux du reste des objets (les écarter) dans le même voisinage. Cette fonction peut être désignée de MiniMax et formalisée comme suit :

$$(\text{Mini } F'i, \text{Max } Fi) = (\text{Mini } (1 + \sum_{j=1}^{J} P'_j \ln P'_j \Big/ \ln J), \text{Max } (1 + \sum_{j=1}^{J} P_j \ln P_j \Big/ \ln J)) \qquad \textbf{Equation 21}$$

$$P'j = n'(Aj) / \sum_{j=1}^{J} n'(Aj)$$

$$Pj = n(Aj) / \sum_{j=1}^{J} n(Aj)$$

- n'(Aj) représente le nombre des objets qui ne vérifient pas les critères considérés dans la cellule Aj,
- n(Aj) représente le nombre d'objets qui vérifient les critères considérés dans la cellule Aj,
- J représente le nombre de cellules.

4.9. Conclusion

La classification que se soit d'objets ou de données est un problème type qu'on rencontre souvent comme un testeur standard ou un benchmark d'idées nouvelles inspirées en particulier du vivant (comportement des essaims, etc.). L'idée dont on veut vérifier la validité à travers le travail présenté ici peut se poser comme suit : «étant donnée la tâche de classification qu'on veut faire émerger au niveau macro, et dont on peut décrire les caractéristiques externes via l'état final du système où elle est

appliquée, quelles sont les règles de base que les robots mobiles doivent respecter au niveau micro pour arriver à l'accomplir avec succès (atteindre cet état final)».

La recherche de ces règles n'est pas aussi évidente que ça peut le paraître, et peut même s'avérer impossible dans certains cas. Le fait de discrétiser l'espace en cellules et le temps en durée nécessaire à l'accomplissement d'un travail élémentaire significatif ou en événements significatifs rend plus facile la détection des interactions entre agents-robots mobiles, ceci d'une part, et d'autre part, facilite l'évaluation du travail accompli par ces agents-robots. Reste la réduction de l'espace de recherche qui peut prendre une taille démesurée ; plusieurs techniques existent actuellement (voir chapitre 2), nous avons opté dans ce travail pour une approche évolutionniste. Plusieurs solutions ont été présentées dans cette perspective, se basant essentiellement sur l'idée de la généralisation de formation de Tas à la formation multi-Tas [56].

Le regroupement sélectif exclusif au mieux reste une idée à l'état embryonnaire qu'on projette d'analyser, de mettre en œuvre et d'utiliser par la suite pour réaliser (entre autres) un opérateur de « regroupement » dans un environnement de base de données distribué, favorisant la consultation d'un type de données par rapport à un autres type en l'amassant dans une zone donnée caractérisée par la fréquence d'utilisation d'un certain nombre de critères de sélection. Sachant que la « restriction strict», tel qu'on le connaît dans la littérature bases de données, peut être réalisé par un regroupement sélectif strict. Dans le même sens d'idées, on projette d'utiliser la tâche de classification multi-critères pour réaliser une version distribué de l'opérateur « Groupe-by » toujours dans un environnement Bases de Données Disturbées.

Dans le cinquième et dernier chapitre nous présentons la mise en œuvre des solutions présentées dans ce chapitre, et toute la démarche de réalisation qui nous a mené vers la confection de plusieurs outils de simulation (programmes), et la validation du travail ainsi élaboré à travers en particulier la mise à l'échelle des règles détectées, et leurs interprétations en les comparants aux règles décelées chez les fourmis pour la classification de couvain.

4.10. Références

Références bibliographiques

[1] A. Abraham et V. Ramos. « Web usage mining using artificial ant colony clustering and linear genetic programming. », In The Congress on Evolutionary Computation, pages 1384–1391, Canberra, Australia, IEEE-Press, 08-12 December 2003.

[2] D. Alexandrov. « Randomized algorithms for the minmax diameter k-clustering problem. », In Proceedings of ECCO 13, pages 193–194, Capri, Italy, May 2000.

[3] N. Azzag, H. Monmarch´e, M. Slimane, G. Venturini, et C. Guinot. « Anttree : a new model for clustering with artificial ants. », In IEEE Congress on Evolutionary Computation, Canberra, Australia, 08-12 December 2003.

[4] G.P. Babu et M.N. Murty. « A near-optimal initial seed value selection in k-means algorithm using a genetic algorithm. », In Proceedings of the First European Conference on Artificial Life, pages 763–769, 1993.

[5] G.P. Babu et M.N. Murty. « Clustering with evolution strategies. », In Proceedings of the First European Conference on Artificial Life, pages 321–329, 1994.

[6] J.C. Bezdek, S. Boggavarapu, L. Hall, et A. Bensaid. « Genetic algorithm guided clustering ». In Proceedings of the First IEEE Conference on Evo-lutionary Computation, pages 34–39, 1994.

[7] J.N. Bhuyan, V.V. Raghavan, et V.K. Elayavalli. « Genetic algorithms for clustering with an ordered representation. », In R.K. Belew et L.B. Booker, RNTI - C – 1, 2001.

[8] A. Colorni, M. Dorigo, et V. Maniezzo. « Distributed optimization by ant colonies. », In Proceedings of the First European Conference on Artificial Life, pages 134–142, 1991.

[9] L.N. de Castro et F.J. Von Zuben. « An evolutionary immune network for data clustering. », In

In Proceedings of the IEEE SBRN'00 (Bra-zilian Simposium on Artificial Neural Networks), Pages 84–89, 2000.

[10] J.-L. Deneubourg, S. Goss, N.R. Franks, A. Sendova-Franks, C. Detrain, et L. Chretien. «The dynamics of collective sorting: robot-like ant and ant-like robots. », In Proceedings of the First International Conference on Simulation of Adaptive Behavior, pages 356–365, 1990.

[11] E. Falkenauer. « A new representation and operators for genetic algorithms applied to grouping problems. Evolutionary Computation », In Proceedings of the First European Conference on Artificial Life,pages 123–144, 1994.

[12] D.B. Fogel et P.K. Simpson. « Evolving fuzzy clusters. », In ICNN93, pages 1829–1834, San Francisco, 1993.

[13] D.E. Goldberg, «Genetic Algorithms in Search, Optimization and Ma-chine Learning», Addison-Wesley, 1989.

[14] W.A. Greene. «Unsupervised hierarchical clustering via a genetic algorithm.», In IEEE Press, editor, Proceedings of the 2003 Congress on Evolutionary Computation, pages 998–1005, Canberra, Australia, 2003.

[15] J. Hansohm. «Two-mode clustering with genetic algorithms.», In Classification, Automation, and New Media: Proceedings of the 24th Annual Conference of the Gesellschaft Fur Klassifikation E.V., pages 87–94, 2000.

[16] D.R. Jones et M.A. «Beltramo. Solving partitioning problems with genetic algorithms. », In R.K. Belew et L.B. Booker, editors, Proceedings of the Fourth International Conference on Genetic Algorithms, pages 442–449, San Diego, CA, 1991. Morgan Kaufmann.

[17] T. Knight et J. Timmis. «On data clustering with artificial ants.», In J. Garibaldi A. Lotfi et R. John, editors, Proceedings of the 4th International Conference on Recent Advances in Soft Computing, pages 266–271, Nottingham, UK., December 2002.

[18] K. Krishna et M. Murty. «Genetic k-means algorithm.», IEEE Transactions on Systems, Man and Cybernetics - Part B, 29(3):433–439, 1999.

[19] P. Kuntz, P. Layzell, et D. Snyers. «A colony of ant-like agents for partitioning in vlsi technology.», In P. Husbands et I. Harvey, editors, Proceedings of the Fourth European Conference on Artificial Life, pages 417–424, 1997.

[20] N. Labroche, N. Monmarché, et G. Venturini. «A new clustering algorithm based on the chemical recognition system of ants. », In F. van Harmelen, editor, Proceedings of the 15th European Conference on Artificial Intelligence, pages 345–349, Lyon, France, july 2002. IOS Press. RNTI – C – 1 Classification et méthodes biomimétiques

[21] C.B. Lucasius, A.D. Dane, et G. Kateman. «On k-medoid clustering of large data sets with the aid of a genetic algorithm: background, feasibility and comparison. », Analytica Chimica Acta, 282:647–669, 1993.

[22] E.D. Lumer et B. Faieta. «Diversity and adaptation in populations of clustering ants. », In Proceedings of the Third International Conference on Simulation of Adaptive Behaviour», pages 501–508, 1994.

[23] N. Monmarché, M. Slimane, et G. Venturini. «On improving clustering in numerical databases with artificial ants. », In D. Floreano, J.D. Nicoud, et F. Mondala, editors, 5th European Conference on Artificial Life (ECAL'99), Lecture Notes in Artificial Intelligence, volume 1674, pages 626–635, Swiss Federal Institute of Technology, Lausanne, Switzerland, Springer-Verlag. 13-17 September 1999.

[24] N. Monmarché, C. Guinot, et G. Venturini. « Fouille visuelle et classification de données par nuage d'insectes volants. », RSTI-RIA-ECA : Méthodes d'optimisation pour l'extraction de connaissances et l'apprentissage, (6):729–752, 2002.

[25] O. Nasaroui, D. Dasgupta, et F. Gonzalez. «The fuzzy artificial immune system : Motivations, basic concepts, and application to clustering and web profiling. », In Proceedings of the IEEE International Conference on Fuzzy Systems at WCCI, pages 711–716, May 12-17 2002.

[26] G. Proctor et C. Winter. «Information flocking : Data visualization in virtual worlds using emergent behaviours. », In J.-C. Heudin, editor, Proc. 1st Int. Conf. Virtual Worlds, VW, volume 1434, pages 168–176. Springer-Verlag, 1998.

[27] V.V. Raghavan et K. Birchard. «A clustering strategy based on a formalism of the reproductive

process in natural systems. », In Information Implications into the Eighties, Proceedings of the Second International Conference on Information Storage and Retrieval, pages 10–22. ACM, 1979.

[28] C. W. Reynolds, «Flocks, herds, and schools : A distributed behavioral model.», Computer Graphics (SIGGRAPH '87 Conference Proceedings), 21(4):25–34, 1987.

[29] J. Holland, «Adaptation in natural and artificial systems », University of Michigan Press, Ann Arbor, 1975.

[30] E. Bonabeau, M. Dorigo, G. Theraulaz, «Swarm Intelligence : From Natural to Artificial Systems », Oxford University Press, New York, 1999.

[31] J. Deneubourg, S. Goss, N. Franks, A. Sendova, C. Detrain, L. Chretien, « The dynamics of collective sorting : robot-like ant and ant-like robots », RSTI-RIA-ECA : Méthodes d'optimisation pour l'extraction de connaissances et l'apprentissage, 2002.

[32] E. Lumer, B. Faieta, « Diversity and Adaptation in Populations of Clustering Ants », University of Michigan Press, Ann Arbor, 1998.

[33] P. Kuntz, P. Layzell, D. Snyers, « A Colony of Ant-like Agents for Partitioning in VLSI Technology », Proceedings of the Fourth European Conference on Artificial Life, MIT Press, Boston, p. 417–424, 1997.

[34] J. Dréo, A. Pétrowski, P. Siarry et E. Taillard: « Métaheuristiques pour l'optimisation difficile», Eyrolles, 2003.

[35] T. D. Barfoot, G. M. T. D'Eleuterio, « An Evolutionary Approach to Multiagent Heap Formation ». Presented at the Congress on Evolutionary Computation, Washington, 1999.

[36] T. D. Barfoot, G. M. T. D'Eleuterio, « Learning Distributed Control for an Object-Clustering Task ». Technical Report, University of Toronto, Institute for Aerospace Studies, Canada, 2003.

[37] N. Stefano, « Power and the Limits of Reactive Agents. », Institute of Psychology, National Research Council (Italy), 1999.

[38] G. Prencipe, V. Gervasi, « On the Intelligent Behavior of Stupid Robots. », Département d'informatique, Université de Pisa (Italy), 2002.

[39] J. Thangavelautham, T. D. Barffot, G. M. T. D. D'Eleuterio, « Ccoevolving Communication and cooperation Lattice Formation Tasks », University of Toronto, Institute for Aerospace Studies, Canada, 2003.

[40] J. Wolff, « Modélisation du déplacement et de la formation de tas de cadavre chez les fourmis. », SCILAB à l'Ecole nationale des ponts et chaussées, 2005.

[41] E. Hoyt, «The Earth Dwellers : Adventures in the land of ants. », Simon ans Schuster, New Yorck, 1996.

[42] R. C. Kube and H. Zhang, «The use of perceptual cues in multi-robot box-pushing. », In Proceedings IEEE International Conference on robotics and Automation, 1996.

[43] M. J. Mataric, «Behaviour-based control : Examples from navigation, learning, and group behaviour. », Journal of Experimental AND Theoretical Artificial Intelligence, 9(2):232-336. Special Issue on Software Architectures for Physical Agents, Editors: H. Hexmoor,I. Horswuill, D. KortenKamp, 1997.

[44] P. Lucidarme , A Liégeois, « Aprentissage et adaptation pour des ensembles de robots réactifs cooperants. », LIRMM, Montpellier, 2001.

[45] A. Dutech, O. Buffer, F. Charpillet, « Développement autonome de comportements de base d'un agent. », Loria – INRIA-Lorraine, National ICT Australia, 2004.

[46] P. Lucidarme, « Apprentissage et adaptation pour des ensembles de robots réactifs coopérants. », Thèse de doctorat de l'université de Montpellier II, 2003.

[47] J-P. müller, « Des systèmes autonomes aux systèmes multi-agents Interaction, Emergence et Systèmes complexes. », Rapport présenté pour l'obtention de l'habilitation à Diriger les recherches en informatiques, 2002.

[48] J-P. Renard, « Auto-organisation chez les insectes sociaux. », Rapport de DEA, 2003.

[49] V. Thomas, C. Bourjot, V. « Chevrier un formalisme pour la construction automatique d'interactions dans les SMA réactifs - étendu. », Rapport interne, INRIA, 2004.

[50] V. Thomas, Y. Murat, « Présentation des algorithmes génétiques et de leurs applications en

économie. », Université de Nantes, Université Montesquieu Bordeaux IV, 2003.
[51] P. Dabrowski, « Introduction aux algorithmes génétiques. », EIVD, 2005.
[52] M. Schoenauer, «Les algorithmes évolutionnistes : état de l'art et enjeux. », Projet fractale, INRIA Rocquencourt, France, 2001.
[53] G. Theraulaz, E. Bonabeau, S. Goss et J-L. Denneubourg, « Algorithme génétiques »,Pour la science N°198 avril 1994.
[54] H. Azzag et al., «Genetic Algorithms », In Proceedings of the Fourth International Conference on Genetic Algorithms, pages 408–15, San Diego, CA, 1991.
[55] J.Meyer, S. Wilson, Eds., «Adaptive Behavior system», Proceedings of the First International Conference on Simulation of Adaptive Behavior, MIT Press, Cambridge, Massachusetts, 1990.
[56] M. R. Abdessemed , M. C. Batouche, « Proposition d'une méthode de tri monocritère : Généralisation d'une approche multi-agent évolutionniste pour la formation de tas à la formation multi-tas » CIP'05 Conférence international, Tlemcen, 2005.

Références sur le Web

www.communication.gc.ca/glossaire.html
www.dictionnaires.culture.fr/textsec3_3.html
www.ups.com/content/ca/fr/resources/select/receiving/customs/terms.html
www.inspection.gc.ca/francais/anima/fispoi/manman/fpimip/definf.shtml
www.ac-rouen.fr/pedagogie/equipes/eps/enseignement/didaaps/escalade/glossaire.htm
sunsite.queensu.ca/rmc/BUS204/Finance/pages/glossaire.html
http ://www.rennard.org/ alife
http ://tecfa.unige.ch/tecfa
http://www.santafe.edu. (Santa Fe Institute, Nouveau Mexique, USA)
fr.wikipedia.org/wiki/Tri
variance.free.fr/notes%20techniques/dechets%20de%20chantiers/dico/texteindex.htm

Autres références

- **CD** UNIVERSALIS 2005.
- **CD** ENCARTA 2005.
- 36 dictionnaires
- Techniques de l'ingénieur 2005 version papier et électronique.
- Brian S Everitt, « Cluster Analysis », Arnold, 1992.
- Louquet P., Vogt A, « Probabilités : combinatoire & statistiques», Armand colin, 1971.
- Seymour L., « Probabilités : cours et problèmes », 1982.

5.1. Introduction

La simulation consiste à effectuer des expériences sur un modèle d'un système réel (dans notre cas il s'agit d'une collectivité de robots autonomes et homogènes dans un environnement spécifique et fermé). Pour ce faire en décompose généralement cette activité en quatre étapes :

- On analyse en premier le système réel et les informations que l'on désire obtenir sur son comportement, pour élaborer ensuite un modèle dans lequel on ne retient que les caractéristiques qui semblent pertinentes par rapport aux résultats recherchés.
- L'étape qui suit consiste à représenter ce modèle sous forme d'un programme de simulation.
- On aborde ensuite la phase d'expérimentation dans laquelle on fait évoluer le modèle, c'est à dire que l'on interprète le programme de simulation à l'aide d'un schéma d'exécution adapté (le simulateur),
- La dernière étape consiste à analyser les résultats produits lors de la phase précédente, pour en déduire des informations sur le comportement du système réel.

Il est clair que la première étape a été traitée dans le chapitre 4, Il s'agit dans ce chapitre de prendre en charge les trois autres étapes, en utilisant comme outil de simulation l'environnement Netlogo (voir Annexe A), et comme approche de simulation celle dirigée par événements (utilisée généralement pour les systèmes discrets), sachant qu'il existe aussi la simulation dirigée par le temps (utilisée généralement pour les systèmes continus).

5.2. Mise en œuvre de la simulation

Dans cette perspective, la simulation va nous permettre de tester la validité des solutions proposées dans le chapitre 4 (voir aussi [13]), et de trouver les règles permettant de faire émerger la tâche de classification en question. L'approche évolutionniste suivie (voir chapitres 2 et 4) étant validée dans des travaux antérieurs, ne constitue pas un but en soit à vouloir atteindre. Nos robots autonomes, mobiles et homogènes (voir chapitres 3 et 4) sont modélisés via des agents réactifs (voir chapitre 3). L'espace dans lequel ces agents errent est sous forme d'une grille bidimensionnelle avec des cellules carrées et des frontières fermées dans les deux directions spatiales. On peut imaginer la forme de cet environnement comme celle d'un tore ou d'une bouée de sauvetage. Les mises en œuvre concernant chacune des solutions proposées (guidée essentiellement par le procédé d'évaluation utilisé) sont expliquées ci-après :

5.2.1. Mise en œuvre de la première solution monocritère

Chaque agent ne peut visionner que 6 places (voir Fig 5.1.) et ne peut prendre ou déposer qu'un objet à la fois. Dans les 5 emplacements qui l'entourent, un agent ne peut voir que ce qui suit :

- rien,
- un objet rouge,
- un objet bleu,
- un objet blanc,
- un autre agent.

Dans la sixième place, celle qu'il occupe, les états dans lesquels il peut se trouver sont :

- libre,
- portant un objet rouge,
- portant un objet blanc,
- portant un objet bleu.

Ceci signifie qu'il y a 5^5 x 4 entrées dans la table de consultation, et qui vaut à 12500 entrées, ce qui est un nombre très grand. Pour réduire ce nombre, nous allons considérer seulement trois cases dans l'environnement qui entoure l'agent; la case de gauche, de droite, et celle de devant, les autres n'étant pas très intéressantes puisque réellement le robot ne peut pas se déplacer vers elles. Notre espace de recherche dans chaque table de consultation se trouve réduit alors à 500 entrées (5^3 X 4).
Les deux comportements de base que l'agent peut adopter sont :

- **Se déplacer** : dans ce comportement, l'agent se déplace en avant, si l'emplacement qui est devant lui est libre, autrement il tourne à droite, si l'emplacement situé à sa droite est vide, autrement il tourne à gauche, si l'emplacement situé à sa gauche est vide, autrement il attend que son environnement local change.

- **Manipuler un objet** : Dans ce comportement, l'agent-robot prend/dépose un objet de/dans l'emplacement situé directement devant lui, sinon, de/dans l'emplacement situé à sa droite, sinon de/dans l'emplacement situé à sa gauche, sinon il adopte le comportement de déplacement. On fait remarquer que la décision de prendre ou de déposer un objet que l'agent doit prendre, s'il a à manipuler un objet, dépend de son état courant dans lequel il se trouve (s'il porte un objet, il le dépose, sinon il prend un objet s'il en trouve un dans son environnement local réduit).

Avec deux comportements de base et une table de consultation de 500 entrées, on se retrouve facilement avec 2^{500} tables de consultation. Ce nombre est énorme, et l'idée de les essayer toutes, l'une à la suite de l'autre pour trouver la meilleure des tables de consultation est tout de suite à écarter.

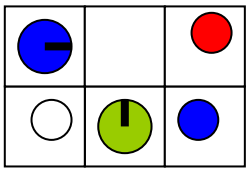


Fig 5.1 Exemple d'une vision locale d'un agent-robot libre (en vert); en face de lui l'emplacement est libre, à sa droite se trouve un objet bleu, à sa gauche un objet blanc, à sa diagonale droite un objet rouge, et à sa diagonale gauche un autre agent-robot portant un objet bleu.

Pour pallier cette difficulté, l'idée est d'utiliser les algorithmes génétiques, en choisissant aléatoirement un certain nombre de ces tables de consultation, pour qu'elles constituent notre population initiale, et de les faire évoluer en essayant de couvrir un espace de recherche, de plus en plus significatif, et se rapprocher ainsi de génération en génération de l'optimum global (la table de consultation la plus pertinente). Les agents considérés dans ce système fonctionnent d'une manière complètement déterministe; pour la même perception locale, l'agent réagit toujours de la même manière, ce qui revient à dire que notre système, à partir du même état initial se déroule toujours de la même manière. Comme nous nous sommes inspirés des travaux déjà effectués auparavant, on a utilisé les mêmes valeurs que Barfoot et all, ont proposé dans [34, 35] : une population fixe à chaque génération de 50 tables de consultation, en retenant à chaque génération les 5 tables de consultation dont la valeur de fitness est la plus grande pour qu'il intègre directement la prochaine génération, une probabilité de croisement Pc = 0.6 et une probabilité de mutation Pm = 0.005. Ainsi qu'un environnement (dans lequel évoluent les agents-robots) de 21 x 21 emplacements, structuré en 9 cellules. Avec 15 agents et 10 objets de chaque couleur (sachant que les couleurs considérées sont le rouge, le bleu et le blanc). Dans la table de consultation (voir Tableau 5.1) nous avons réservé 3 bits pour la case en face de l'agent-robot, 3 pour la case située à sa droite, et 3 pour la case à sa droite, étant donné que le robot réel ne peut percevoir que l'un des cas suivants :

- Rien, codé par 000
- Un objet rouge, codée par 011
- Un objet bleu, codée par 010
- Un objet blanc, codée par 100
- un autre robot, codé par 001

face			droite			gauche			état robot		c
1	0	0	1	0	1	0	0	0	0	1	1
0	0	1	0	0	1	0	1	1	1	1	0
0	0	0	0	0	0	0	1	1	0	0	0
.	.	.	.	.	.	.	.	.	.	.	.

Environnement local — **Chromosome**

TABLEAU 5.1. STRUCTURE DE LA TABLE DE CONSULTATION

Nous avons aussi réservé 2 bits pour l'état de l'agent-robot, étant donné que le robot réel ne peut être que dans l'un des états suivants :

- libre, codé par 00
- portant un objet, rouge codée par 01
- portant un objet bleu, codée par 10
- portant un objet blanc, codée par 11

Comme on n'a que deux comportements, un bit suffit pour désigner l'un des deux comportements de base, 0 : pour se déplacer, et 1 : pour manipuler un objet.

Le chromosome binaire est constitué alors, de toute la colonne « c » du Tableau 5.1 concernée par le comportement, et est de taille 500 bits, représentant le nombre des entrées. Ceci étant donné que les entrées dans chaque table de consultation sont exactement les mêmes (voir Tableau 5.1, partie environnement locale), et ont les mêmes positions.

L'algorithme génétique se présente dans ce cas comme suit :

Pour *NG générations de taille P chacune*
Pour *chaque table de consultation de la génération courante*
Répéter Nb-exp
Entraîner les robots simulés à classer les objets pendant un temps t
Evaluer la fitness dans ***Ft***
Som_Ft = Som_Ft + Ft
/ ********* *D'autres formules peuvent être utilisées à la place de la moyenne, comme celle du Min.*
Fin Répéter
Evaluer la fitness moyenne de la table dans Ftmoy = Som_Ft / Nb-exp
Fin pour
Sélectionner les K meilleures tables de consultation
/ Pour qu'ils intègrent la prochaine génération*
Eliminer les K dernières tables de consultation
/*** *représentant les chromosomes les plus mauvais*
Pour *les P-K tables de consultations restantes*
Appliquer le croisement et la mutation avec respectivement les probabilités Pc et Pm
Fin Pour
Fin pour

Il est clair que l'opération de croisement s'effectue au même emplacement choisi aléatoirement des deux chromosomes. Le fait de choisir aléatoirement l'endroit de coupure est important dans le sens où, ceci permet d'éviter de rester coincé dans un ensemble de groupes de chromosome dont le préfixe (partie gauche de la coupure) est toujours le même, et d'élargir ainsi la recherche à toutes les classes. Ceci est valable aussi pour l'opérateur de mutation.

Il ne faut pas confondre entre la fonction aléatoire qui permet de choisir l'emplacement où l'opération doit s'effectuer (que ce soit celle de croisement ou de mutation) et la probabilité qui permet de dire si l'opération doit s'effectuer effectivement ou non et qu'on a réalisé à travers les seuils Pc et Pm (si on dépasse le seuil l'opération est interdite).

5.2.2. Mise en œuvre de la deuxième solution monocritère

Dans cette solution [13], on garde la même structure de la table de consultation, en ne faisant que changer le comportement des agents, comme suit :

- S'il s'agit d'un déplacement, deux cas sont à considérer :

a) L'agent ne porte pas d'objet : dans ce cas l'agent peut se déplacer exactement comme dans la première solution.
b) L'agent porte un objet : dans ce cas, l'agent se déplace seulement si l'objet en sa possession est de la même couleur que celle de l'étape-couleur autorisée.

- S'il s'agit d'une manipulation d'objets : L'agent ne peut déposer ou prendre d'objet que si l'objet sujet à la manipulation est de la même couleur que celui de l'étape couleur autorisée.

On fait remarquer qu'on applique les mêmes opérateurs que ceux de la première solution, aux chromosomes des tables considérées à savoir : Evaluation, Sélection, Croisement, et Mutation, avec les mêmes paramètres de croisement et de mutation, Pc et Pm. En ce qui concerne l'évaluation, on utilise trois fonctions de fitness; une par couleur (exactement de la même forme que celle utilisée par Barfoot et all dans [1, 2] pour la formation de Tas). L'état des lieux étant évalué par la moyenne de ces trois fonctions.

L'algorithme générique dans ce cas est de la forme suivante :

***Pour NG** générations de taille **P** chacune*
***Pour** chaque table de consultation de la génération courante*
Répéter Nb-exp
TQ** le temps d'entraînement n'est pas épuisé **Faire
Pour** chaque couleur C**oul
Entraîner les robots simulés à classer les objets pendant un temps dt
Fin Pour
Fin TQ
Evaluer les fitness de chaque couleur respectivement dans F_couleur1, …, F_couleur_k
Ftmoy = Ftmoy + Moyenne (F_couleur1, …, F_couleur_k)
Fin Répéter
*/ * D'autres formules peuvent être utilisées à la place de la moyenne, comme celle du Min.*
Evaluer la fitness moyenne de la table : Ftmoy = Ftmoy / Nb-exp
Fin pour
Sélectionner les k meilleures tables de consultation selon la valeur de Ftmoy pour rejoindre directement la prochaine génération
Eliminer les k chromosomes (tables de consultation) les plus mauvais.
***Pour** les P-K tables restantes*
Appliquer le croisement et la mutation
Fin Pour
Fin pour

Les agents-robots doivent prendre en considération dans cette solution aussi bien leur perception de l'environnement local que la couleur autorisée à chaque étape, afin de pouvoir choisir le comportement de base à adopter. Pour que ceci soit réalisable il faut que les agents-robots synchronisent leur étape-couleur autorisée (la couleur autorisée doit être la même pour tous les agents-robots à un instant donné). On peut réaliser ceci de deux manières :

- Diffuser la couleur autorisée à chaque étape-couleur à tous les agents-robots,
- Utiliser une horloge au niveau de chaque agent-robot fonctionnant périodiquement en modulo « nombre de couleurs ». Il suffit dans ce cas d'initialiser toutes les horloges au début à zéro et les déclancher. Chaque numéro du compteur représente alors une couleur donnée.

5.2.3. Mise en œuvre de la troisième solution monocritère

La troisième solution est la plus simple à mettre en œuvre. On reprend la même table de consultation utilisée dans les deux premières solutions, et on utilise trois fonctions de fitness, chacune pour évaluer la formation en Tas concernant une couleur donnée. La fonction de fitness évaluant la tâche de classification étant la moyenne de ces trois fonctions. Les comportements de bases concernant

chaque robot ne prennent pas en considération le critère de classification couleur (comme dans la deuxième solution). En d'autres termes, il s'agit des mêmes comportements de base adoptés dans la première solution (se déplacer et manipuler un objet).

L'algorithme générique dans ce cas est de la forme suivante :

***Pour NG** générations de taille **P** chacune*
  ***Pour** chaque table de consultation de la génération courante*
  Répéter Nb-exp
    Entraîner les robots simulés à classer les objets pendant un temps t
     Pour** chaque couleur **coul
       *Evaluer la fitness dans **fi_coul***
     Fin pour
    *Faire leur moyenne dans **fm = moyenne (fi_coul)***
    Accumuler les valeurs d'évaluation des Nb-exp expériences dans Ft = Ft + fm
   */ * D'autres formules peuvent être utilisées à la place de la moyenne, comme celle du Min.*
   Fin Répéter
  Evaluer la fitness moyenne de la table dans Ft = Ft / Nb-exp
 Fin pour
  Sélectionner les k meilleures tables de consultation pour rejoindre la prochaine génération
 Eliminer les k plus mauvais chromosomes
  ***Pour** les P-K restants*
   Appliquer le croisement et la mutation
 Fin Pour
Fin pour

5.2.4. Mise en œuvre de la solution multicritères

Comme pour les solutions monocritère, chaque agent ne peut visionner que 6 places (voir Fig 5.2.) et ne peut prendre ou déposer qu'un objet à la fois. Dans les 5 emplacements qui l'entourent, un agent ne peut voir que ce qui suit :

- rien, codé par 000
- un cercle rouge, codé par 011
- un cercle bleu, codé par 010
- un triangle rouge, codé par 100
- un triangle bleu, codé par 111
- un autre agent, codé par 001

Dans la sixième place, celle que l'agent-robot occupe, les états dans lesquels il peut se trouver sont :

- libre, codé par 000
- portant un cercle rouge, codé par 010
- portant un cercle bleu, codé par 100
- portant un triangle rouge, codé par 001
- portant un triangle bleu, codé par 011

En ne considérant que trois positions (de devant, de gauche et de droite) Notre espace de recherche dans chaque table de consultation est de (6^3 X 5) = 1080 entrées.

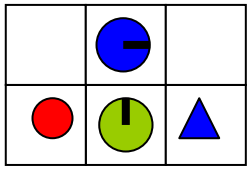


Fig 5.2 Exemple d'une vision locale d'un agent libre (en vert); en face de lui un autre agent portant un objet bleu, à sa droite se trouve un triangle bleu, à sa gauche un cercle rouge.

Les deux comportements de base que l'agent peut adopter étant les mêmes que ceux précités dans pour les autres solutions.

Il est à noter que la table de consultation utilisée dans cette solution a exactement la même structure que celle des autres solutions, excepté qu'on doit codifier six valeurs pour identifier les objets pouvant se trouver dans l'environnement local de l'agent en question (on a besoin dans ce cas de trois bits pour codifier les six valeurs), et cinq valeurs pour les états de l'agent-robot (on a besoin dans ce cas de trois bits pour codifier les cinq valeurs). La taille de chaque entrée dans ce cas est de 3x3 bits (pour l'environnement local) + 3 bits pour l'état de l'agent-robot = 12 bits, la sortie étant un bit pour choisir entre l'un des deux comportements (se déplacer ou manipuler un objet).

L'algorithme générique dans ce cas est de la forme suivante :

***Pour NG** générations de **P** chromosomes chacune*
  ***Pour** chaque chromosome (table de consultation) de la génération courante*
  Répéter Nb-exp
    Entraîner les agents-robots à classer les objets pendant un temps t
     ***Pour** chaque type d'objets vérifiant la même couleur et la même forme*
       *Evaluer la fitness dans **f_coul_form***
     Fin pour
    *Faire leur moyenne dans **fm = moyenne (f_coul_form)***
    Accumuler les valeurs d'évaluation des Nb-exp expériences dans Ft = Ft + fm
   Fin Répéter
  Evaluer la fitness moyenne du chromosome dans Ft = Ft / Nb-exp
 Fin pour
  Sélectionner les k meilleures tables de consultation pour rejoindre la prochaine génération
 Eliminer les k moins bons chromosomes
  ***Pour** les P-K restants*
   Appliquer le croisement et la mutation
 Fin Pour
Fin pour

5.3. Choix entre simulation synchrone et asynchrone

Pour le problème étudié ici la simulation asynchrone est plus intéressante que la simulation synchrone, étant donné que dans la première, le robot simulé via l'agent essaie d'exécuter ses actions dans le plus bref des délais (il fait au mieux), alors que dans la simulation synchrone il doit attendre le signal d'approbation ou son tour. Le problème avec la simulation asynchrone est que deux robots peuvent entrer en conflit, en essayant par exemple de prendre le même objet ou d'occuper le même emplacement, ce qui n'est pas cohérent avec la contrainte qu'on a imposé auparavant, et qui stipule qu'on ne peut pas trouver deux éléments en même temps, à la même place, excepté si c'est un robot simulé qui porte un objet. Pour résoudre ce problème, et se conformer à notre contrainte, on peut utiliser un mécanisme de synchronisation assurant l'exclusion mutuelle entre les différents robots simulés, mais ceci augmentera la complexité de notre algorithme et fera diminuer du coup les performances du système.

L'environnement de simulation qu'on a utilisé (Netlogo) pour concrétiser nos idées, et évaluer la performance de l'approche adoptée, ainsi que la consistance de notre opérateur d'évaluation de fitness offre les deux modes d'exécution; pour une exécution asynchrone il suffit d'utiliser la commande Ask nom-du-groupe, qui s'adresse à tous les robots simulés (constituant un groupe spécifique). Pour une exécution synchrone, il suffit de préciser le numéro d'identification de l'agent en question (Ask num-entité). En faisant varier ce numéro, on arrive à manipuler l'ensemble de tous les agents dans une boucle.

5.4. Résultats de la simulation

5.4.1. Règles décelées

a) pour la première et la troisième solution monocritères

Pour faciliter l'interprétation des règles sensori-motrices, nous avons mis en œuvre un programme permettant d'interpréter graphiquement la table de consultation trouvée en fin du parcours de la recherche. Plus encore, une fois que nous avons mis la main sur les règles en question, nous avons conçu un autre programme pour vérifier que ces règles réalisent effectivement cette macro-tâche. Autrement dit, une fois qu'on a trouvé les règles sensori-motrices par un processus apparenté au « reverse-emergence», et qu'on a pu les interpréter en déduisant des règles plus significatives (de haut niveau), nous avons appliqué un processus apparenté à l'émergence pour vérifier que ces règles sont réellement bonnes dans le contexte de notre macro-tâche.

Les règles qu'on a pu tirer en ce qui concerne la première et troisième solution, en relation avec les équations de fitness proposées, sont les suivantes :

- L'agent-robot dépose un objet, si son environnement contient au moins un objet de la même couleur que celui en sa possession, sinon il se déplace avec l'objet.
- L'agent-robot prend un objet, s'il est tout seul dans son environnement local, sinon il se déplace.

Il va de soit que l'agent-robot prend un objet dans le cas où il n'en porte pas déjà un, et réciproquement, dépose un objet dans le cas où il en porte déjà un. On a pu constater aussi, qu'il y'a deux approches possibles :

- **Approche optimiste :** qui stipule que l'agent-robot (libre) ne prend pas un objet s'il y a au moins un autre objet de la même couleur dans son environnement local (s'il y a deux objets de même couleur dans son environnement local, l'agent-robot espère qu'il y'a formation d'une pile ou d'un Tas. Il prend dans ce cas celui qui ne leur ressemble pas, s'il existe dans la position qui reste, sinon il se déplace). Pour déposer un objet, l'agent-robot suit la même logique : il suffit de trouver un objet qui ressemble à celui en sa possession pour qu'il le dépose.
- **Approche pessimiste :** qui stipule que l'agent-robot ne prend pas d'objets seulement dans le cas où tous les objets de son environnement local sont de même couleur (dans tous les autres cas il prend un objet, il déduit qu'il n'y a pas de formation de pile ou de Tas). Pour déposer un objet en sa possession l'agent-robot doit trouver deux objets qui ressemblent à celui qu'il porte.

b) Pour la solution multicritères

Pour réaliser la tâche multicritères, les règles décelées pour la tâche monocritère restent valables. Mais, au lieu de vérifier un seul critère pour prendre ou déposer un objet, on en vérifie plusieurs. Dans le cas de l'exemple avec lequel on a travaillé, il s'agit de la « couleur » et de la « forme ».

Notons que pour la deuxième solution monocritère, les règles n'ont pas encore été décelées, puisque le processus de convergence n'a pas encore atteint un seuil significatif.

5.4.2. Résultats des expériences effectuées dans le cas monocritère

La décision prise par un agent-robot de prendre un objet de couleur donnée ou de déposer un objet de couleur donnée dépend en premier lieu du fait qu'il a en sa possession un ou non, et en second lieu de son environnement local.

Le choix de ces types de comportement est jusqu'ici arbitrairement naturel. Avec 2 comportements de base et une table de consultation de la taille de 500 entrées, il y a 2^{500} tables de consultation possibles. Ce nombre est tout à fait grand. Mais, de bonnes solutions peuvent encore être trouvées.

On précise aussi que nos agents fonctionnent d'une façon complètement déterministe. À partir d'un état initial particulier, le système se déroule toujours de la même manière. Ce que nous présentons ici concerne les expériences menées pour la troisième solution monocritère (Pour les autres solutions le procédé est le même) : Nous avons utilisé une taille de population de P = 50, un nombre de générations G = 150, un nombre d'individus retenus lors de l'opération de sélection K = 5, une probabilité de croissement P_C = 0.6, et une probabilité de mutation Pm = 0.005. Pour trouver de bonnes règles nous avons utilisé un monde bidimensionnel de taille 21 x 21 de 15 agents-robots, et de 10 objets de chaque couleur, trois couleurs, et un temps d'entraînement allant de T = 150 s, à 240s, un nombre de cellules impliquées dans le calcul de l'évaluation de fitness J = 9 et un nombre de conditions initiales aléatoires par évaluation de fitness I = 3. La figure Fig.5.3 montre un historique de convergence typique de l'AG ainsi appliqué dans le cas de la troisième solution monocritère.

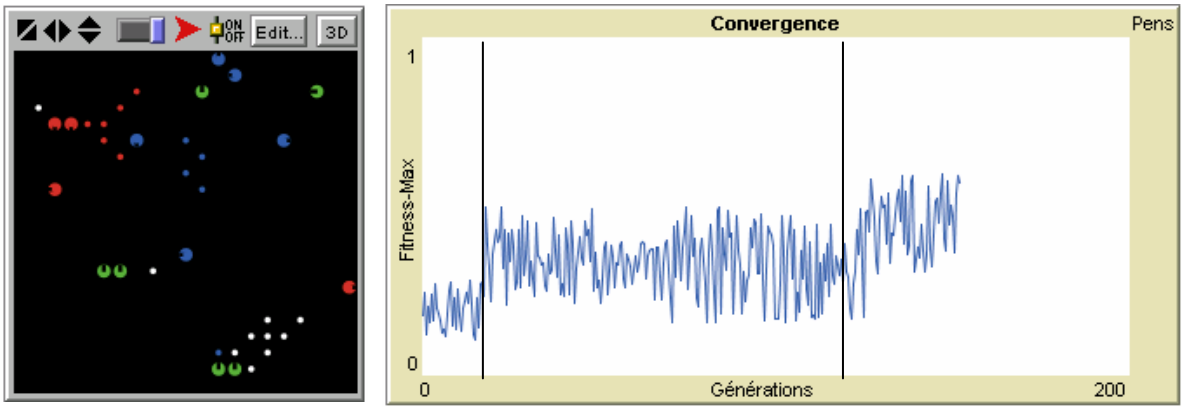


Fig.5.3 A gauche un instantané du système arrivé à une fitness de 0.53, A droite, l'historique de convergence du même système dans le cas de la troisième solution monocritère. Pour un environnement de taille 21x21, contenant 15 agents, et 10 objets à classer de chaque couleur (rouge, bleu, ou blanc). Notons que chaque population est entraînée trois fois avant son évaluation finale.

Il est à noter que le processus évolutionniste de convergence utilisé pour approcher l'une des solutions éventuelles est fractionné en phases : dans la première étape on a utilisé un temps d'entraînement de 120 s, après une quinzaine de générations on a augmenté ce temps pour atteindre 180 s, et pour les trente cinq dernières générations on a utilisé un temps de 240 s (voir fig 5.3). Ceci dans l'espoir d'accélérer la convergence. Partant du principe que plus on avance, et plus on se rapproche de générations prometteuses. Donc, il ne faut pas trop s'attarder au début (ceci reste à vérifier en comparant notre façon de faire avec celles proposées par d'autres auteurs tel que Barfoot et all.). Il est important de mentionner aussi que plus on avance vers l'optimum global, et plus on augmente le nombre d'entraînements pour chaque table de consultation. Ceci nous permet de stabiliser la solution trouvée dans le sens où on trouve une solution indépendante de l'état initiale de disposition des agents-robots et des objets (la raison précitée pour l'augmentation progressive du temps d'entraînement reste valable pour l'augmentation progressive du nombre d'entraînement : on a utiliser 3 entraînements par table de consultation dans les premières phase jusqu'à avoir atteint le seuil de fitness de 0.53, et 10 par la suite - on remarque que le seuil chut puis remonte timidement -. On espère utiliser 30 expériences / table après avoir dépasser le seuil de 0.70).

A la fin de 150 générations, nous prenons la meilleure table de consultation de cette dernière génération produite par l'algorithme génétique adopté pour être notre solution partielle. La figure

Fig.5.4 montre quelques instantanés d'une classification partielle. À partir d'une première distribution aléatoire des objets, les agents commencent par former de petites piles pouvant contenir plus d'une couleur, qu'ils fusionnent par la suite pour former des piles plus importantes en taille, et contenant de moins en moins de variation de couleurs (la structure de classification commence à émerger).

Pour comprendre le comportement d'une telle table, nous avons entraîné nos agents-robots, dotés de cette dernière, une certaine durée en prélevant à chaque pas de temps variable ou fixe (importe peu) la valeur de la fonction de fitness utilisée pour l'évaluation. Le résultat est présenté dans la figure (fig 5.4).

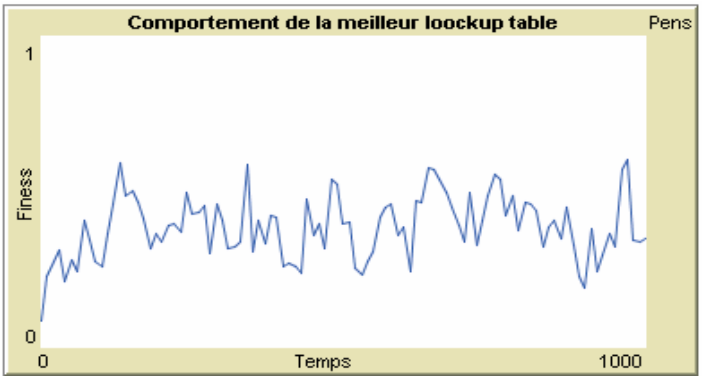


Fig 5.4 Comportement de la meilleure table de consultation (0.53 de fitness) à différents intervalles de temps (10 s pour chaque pas).

On remarque que la collectivité des agents-robots arrivent à atteindre plusieurs fois la fitness cible (0.53), le problème est que la table de correspondance utilisée (contenant les règles sensori-motrices) ne renferme pas encore les règles qui permettent de conserver cette structure. Donc, on doit encore faire entraîner nos agents-robots à la recherche d'un jeu de règles plus complet. Une solution serrai d'augmenter le nombre d'expériences effectuées par table de consultation progressivement (en passant d'une génération à une autre) jusqu'à atteindre un certain nombre (Barfoot et all. ont utilisés 30) par table de consultation effectuées pour tous les individus de la même génération.

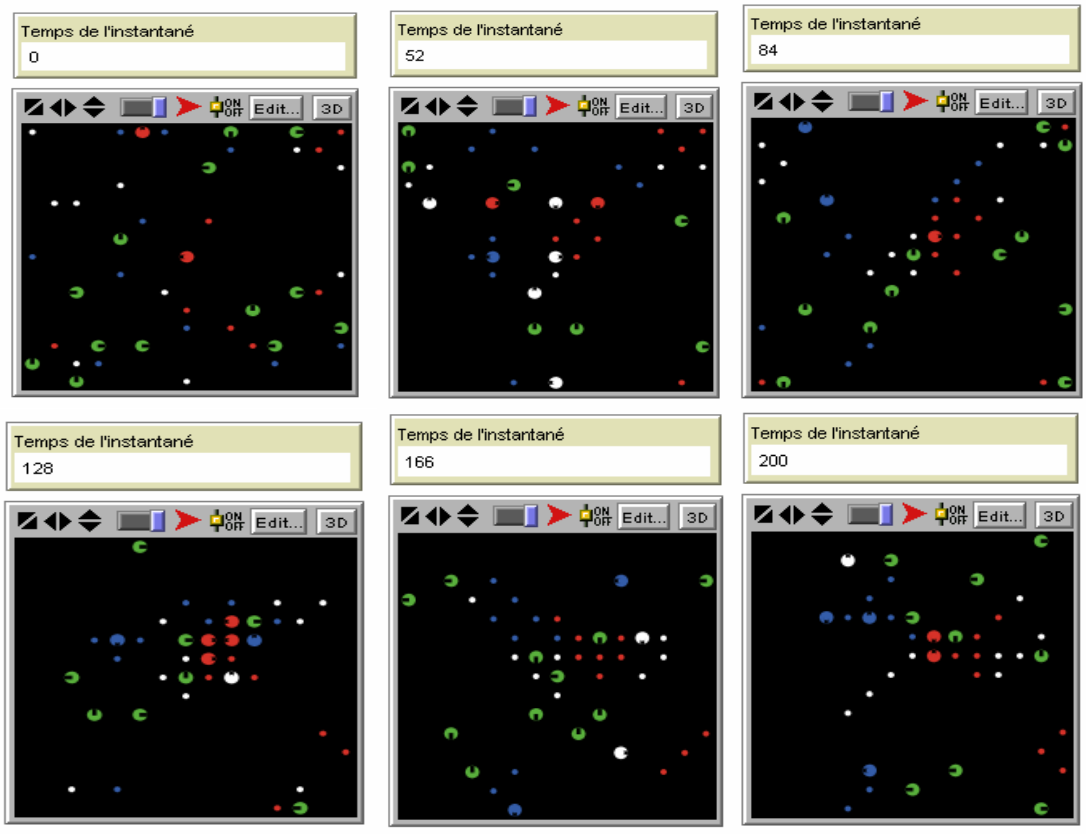


Fig.5. 5 états instantanés du système à des temps variés en secondes (0; 52 84; 128 ; 166 ; 200) La taille de l'environnement est de 21 x 21, contenant 15 agents (disques verts avec des lignes) et 10 objets de chaque couleur (disques rouges, bleus, et blancs).

5.4.3. Mise à l'échelle du nombre d'agents-robots

Pour ce faire, on fait varier le nombre d'agents-robots de 15 à 100 en fixant la taille de l'espace de déplacement des agents-robots à 21x21 et le nombre des objets à classer de chaque couleurs à 10. Ceci nous donne la courbe suivante.

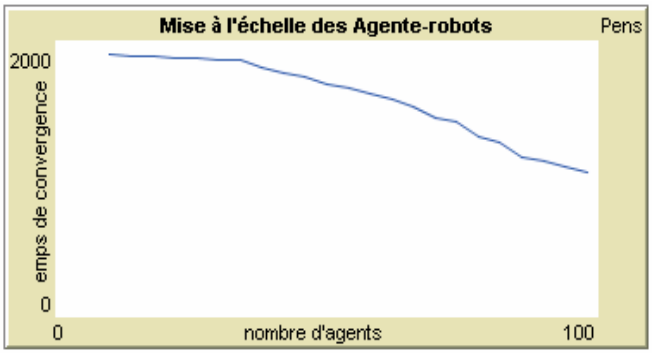


Fig 5.6 Courbe exprimant la convergence de la collectivité des agent-robots dotés de l'automate cellulaire adéquat, vers la tâche de classification (fitness de 0.53), et nombre d'agents variant de 15 à 100.

Plus le nombre d'agents-robots augmente et plus le système a tendance à accélérer le processus de classification, jusqu'à une certaine limite (l'accélération devient moins accentuée). On peut dire que le nombre d'agents est inversement proportionnel au temps de convergence vers l'émergence de la structure de classification en question.

5.4.4. Mise à l'échelle du nombre d'objets

Pour ce faire, on fait varier le nombre d'objets pour chaque couleur (rouge, bleue, blanc) de 10 à 100 en fixant la taille de l'espace de déplacement des agents-robots à 21x21 et le nombre d'agents-robots à 20. Ceci nous donne la courbe suivante.

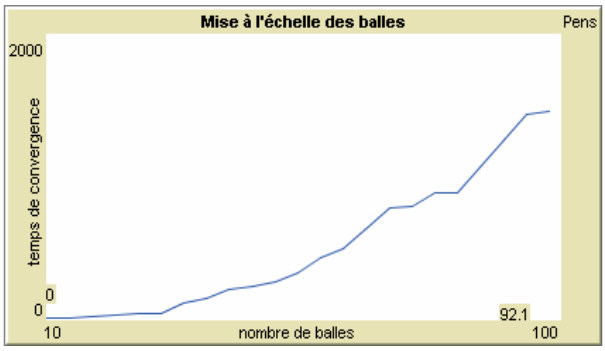


Fig 5.7 Courbe exprimant la convergence de la collectivité des agents-robots dotés de l'automate cellulaire optimal, vers la tâche de classification, avec une fitness de 0.53 en faisant varier le nombre d'objets à classer par couleur de 10 à 100. Notons que cette expérience est faite pour un même nombre d'objets par couleur.

Plus on augmente le nombre d'objets à classifier et plus la courbe tend à être linéaire : le temps de l'émergence de la structure de classification atteignant la fitness désirée (0.53) augmente presque linéairement à partir d'un certain nombre d'objets à classifier par couleur. Il est à noter que la convergence est toujours atteinte. On peut dire que le temps de convergence vers l'émergence de la structure de classification est proportionnel au nombre d'objets à classifier.

5.4.5. Mise à l'échelle de l'espace d'entraînement

Pour ce faire, on fait varier la taille de l'espace de 21x21 à 51x51 en fixant le nombre d'agents-robots à 20 et le nombre d'objets à classer pour chaque couleur à 15. Ceci nous donne la courbe suivante.

Sur la figure fig 5.8 on remarque que l'espace d'entraînement est proportionnel au temps de convergence : plus l'espace augmente et plus le système a tendance à prendre plus de temps pour converger (les agents-robots parcourent plus de terrain pour classer les objets).

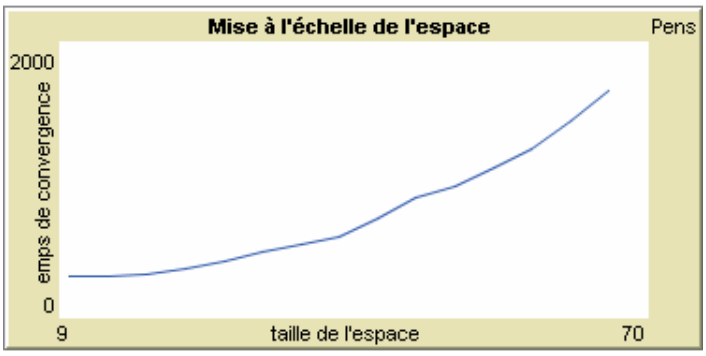


Fig 5.8 Courbe exprimant la convergence de la collectivité des agents-robots dotés de l'automate cellulaire optimal, vers la tâche de classification, avec une fitness de 0.53 quelque soit la taille de l'espace de recherche utilisé.

5.5. Comparaison des règles trouvées avec celles décelée chez les Fourmis

On sait maintenant (voir chapitre 4) que les fourmis classent spontanément les oeufs par stades d'évolution : elles font de petits tas d'oeufs, de larves et de nymphes. Elles sont même capables de distinguer plusieurs stades d'évolution pour les oeufs. L'ensemble de règles qui peuvent régir les fourmis lors de la classification du couvain peuvent êtres décrites comme suit :

1) Si une fourmi libre rencontre un oeuf et que la concentration locale spécifique au type de cet oeuf est faible, la fourmi aura tendance à l'emporter. Au contraire, si la concentration est forte, elle aura tendance à le laisser sur place et à continuer sa route.
2) Si une fourmi porte un oeuf et qu'elle arrive sur une case vide, elle aura tendance à déposer cet oeuf si la concentration locale spécifique à ce type d'œuf est forte et à le garder si la concentration est faible.

L'analogie avec les règles décelées pour les agents-robots (voir 4.5.1) est immédiate : Le type de chaque œuf dans le cas des fourmis représente une couleur différente dans le cas des agents-robots, le nombre d'objets similaires dans le cas des agents-robots représente la concentration spécifique de chaque type d'œuf dans le cas des fourmis. Plus ce nombre est grand (en parallèle de l'autre coté : taux de concentration du type d'œuf considéré élevé), et plus l'agent-robot a tendance à déposer l'objet emporté, et plus ce nombre est petit (concentration spécifique faible) et plus l'agent-robot a tendance à l'emporter (s'il n'est pas déjà en sa possession) ou le garder et se déplacer (s'il est déjà en sa possession).

On fait remarquer que l'expression de « nombre suffisant » d'objets de même couleur est une appréciation laissée aux agents-robots selon qu'il travaillent avec une approche pessimiste ou optimiste. Notons aussi que la table de consultation trouvée (renfermant les règles sensori-motrices en question) n'est pas unique. Ceci explique en partie les tendances (pessimiste ou optimiste) des agents-robots qui peuvent varier, d'une solution à une autre. Ceci dit, les directives principales restent inchangées.

Cette comparaison nous permet de confirmer la consistance et la véracité des règles ainsi proposées.

5.6. Problème de la boucle infinie

Le problème de la boucle infinie est un problème qu'on ne discerne qu'au moment de la mise en œuvre du système en question. On observe dans le cas du phénomène de la boucle infinie que les agents-robots n'accomplissent pas de travail utile, ils ne font que tourner en rond indéfiniment (étant donné que l'environnement d'entraînement est fermé dans les deux directions de l'espace). On peut dire que le système ne transite pas vers un nouvel état, il ne fait que boucler périodiquement sur un ensemble d'états. Ceci est du aux couloirs vides où se déplace un sous-ensemble des agents-robots constituant notre système, ainsi qu'à leurs voisinages vides. Notons que le reste des agents-robots se trouvent bloqués. Donc, aucun nouvel événement ne peut surgir pour faire changer aux agents-robots leurs directions, d'où la consommation d'étapes d'entraînement en vain, et l'augmentation en conséquence du temps de convergence. Ce phénomène est accentué lorsqu'on réduit le nombre d'agents-robots ou le nombre d'objets à classifier ou lorsqu'on étend l'espace d'entraînement. On a pu constater ce phénomène à la phase de recherche de l'une des solutions optimales, et on a confirmé son émergence une autre fois aux moments des mises à l'échelle.

Pour résoudre ce problème, on propose deux solutions :

1) L'agent-robot doit :

   - identifier la boucle infinie,
   - changer de direction en conséquence afin d'obliger le système à changer d'état.

   Pour identifier la boucle infinie, l'agent-robot peut compter dans le cas d'un déplacement tout droit (sont changement de direction) le nombre de cases traversées, si ce nombre dépasse la taille de l'arrête de l'espace d'entraînement, on conclu que l'agent-robot a effectué une rotation complète et qu'il peut rentrer dans une boucle infinie, on lui fait changer alors de direction (lorsque c'est possible).

2) Une autre solution consiste en le déplacement aléatoire de l'agent-robot selon l'une des directions : devant, à droite ou à gauche. Dans nos expériences on a utilisé une fonction aléatoire qui choisie entre les valeurs de l'ensemble suivant {devant, devant, devant, à droite, à gauche}. On remarque qu'ainsi on favorise la direction « devant » par rapport aux autres, sans pour autant lui donner une priorité absolue sur les autres directions.

5.7. Conclusion

L'adéquation de chaque robot mobile simulé à son environnement (espace de déplacement + objets à classer + les autres robots mobiles) est estimée généralement selon un formalisme mathématique donné. Dans ce chapitre on a proposé plusieurs approches d'évaluation. Il est à noter aussi qu'on a cherché à accélérer le processus de convergence :

- En cherchant via un programme confectionné dans ce sens à débuter avec une génération de chromosomes (tables de consultations) prometteurs : On a entraîné et évalué quelques dizaines de générations de chromosomes un certain temps, puis on a choisi les meilleurs des chromosomes (ayant la meilleure fitness) pour faire partie de la génération initiale.
- En réduisant l'espace de recherche de 25 fois.
- En impliquant dans la phase de croisement les meilleurs chromosomes trouvés.

Les courbes concernant la convergence vers la table de consultation optimale (illustrées pour la troisième solution monocritére) montrent la lenteur des algorithmes génétiques. Mais, ceci n'enlève rien quant à leur robustesse pour trouver l'optimum global.

Pour cette même solution, on a montré la consistance des règles sensori-motrices ainsi découvertes, via la mise à l'échelle des nombres de robots, des nombres d'objets à classifier concernant chaque couleur, et de l'espace d'entraînement.

Il est clair que la simulation est d'une aide incontestable pour les chercheurs dans beaucoup de domaines, néanmoins, et comme il a été motionné par beaucoup de chercheurs, dans le domaine de la robotique collective, la meilleure façon de tester le comportement des robots reste le milieu effectif; avec une plate forme réelle, et de vrais robots : étant donné que le milieu simulé est un milieu discrétisé, parfaitement contrôlé, et où tout événement est répertorié (connu d'avance). Notons que ceci fait partie de nos futurs projets dans ce domaine.

5.7. Références

Références bibliographiques

[1] T. D. Barfoot, G.M.T. D'Eleuterio, « An Evolutionary Approach to Multiagent Heap Formation. Presented at the Congress on Evolutionary Computation, Washington », 1999.

[2] T.D. Barfoot, G.M.T. D'Eleuterio, « Learning Distributed Control for an Object-Clustering Task ». Technical Report, University of Toronto, Institute for Aerospace Studies, Canada, 2003.

[3] Gauthier Quesnel, Raphael Duboz, Eric Ramat, « Comparaison d'approches de simulations distribuées à événements discrets d'entités spatiales », Laboratoire d'Informatique du Littoral Mission de la Recherche Blaise Pascal, 2003.

[4] Fabien Miche, « Formalisme, méthodologie et outils pour la modélisation et la simulation de systèmes multi-agents », Laboratoire d'Informatique de Robotique et de Micro-électronique de Montpellier, 2004.

[5] George Draghici, Nicolae Brinzel, Ioana Filipas, « La modélisation et la simulation en vue de la conduite des systèmes de production », Université polytechnique de Timisoara, 1998.

[6] Philipe Ingels, Michel Raynal, « Simulation repartie de système a événements discrets : Partie 1 : Modélisation et schémas d'exécution », Rapports de recherche n° 1057, INRIA, 1989.

[7] Claude Evéquoz, « Simulation et modélisation discrètes », Département d'électricité et informatique, Ecole d'ingénieurs du Canton de Vaud, 2001.

[8] Papert, S. Mindstorms, « NY: Basic Books », 1980.

[9] Pea, R. D. & Maldonado, H. « Wild for learning: Interacting through new computing devices anytime, anywhere ». In K. Sawyer (Ed.), The Cambridge Handbook of the Learning Sciences. New York: Cambridge University Press, 2005.

[10] Wilensky, U. Evanston, IL. « NetLogo. Center for Connected Learning and Computer-Based Modeling », Northwestern University, 1999.

[11] Wilensky, U. GasLab: « An extensible modeling toolkit for exploring micro- and macroviews of gases ». In N. Roberts, W. Feurzeig, & B. Hunter (Eds.), Computer modeling and simulation in science education. Berlin: Springer Verlag, 1999.

[12] Wilensky, U. & Stroup, W. Evanston, IL. «HubNet. Center for Connected Learning and Computer-Based Modeling », Northwestern University, 1999.

[13] Abdessemed M. R., Batouche M. C. « Proposition d'une méthode de tri monocritère : Généralisation d'une approche multi-agent évolutionniste pour la formation de tas à la formation multi-tas », CIP'05, Conférence international, Tlemcen, 2005.

Références sur le Web

http://ccl.northwestern.edu/netlogo/
http://nura.no-ip.com/~julien/lactose/

http://education.mit.edu/starlogo.
http://www.brook.edu/dybdocroot/es/dynamics/models/
http://www.swarm.org.
http://repast.sourceforge.net.
http://ccl.sesp.northwestern.edu/netlogo.
http://www.agentsheets.com.
http://www.integratedmodelling.org.
http://www.madkit.org.
http://www.santafe.edu/projects/echo/echo.html.
http://www.iwr.uni-heidelberg.de/groups/comopt
http://espacestemps.net/document1563.html

Conclusion générale

La diversité et la richesse que renferme la vie artificielle malgré son jeune age, témoignent du grand intérêt qui lui est accordé. Ceci, que ce soit par la communauté scientifique ou par celle des grandes firmes et compagnies à l'échelle mondiale, comme Microsoft, et IBM.

L'intelligence collective est l'un des axes de recherche les plus en vogue actuellement, qui s'étale à l'intersection entre la vie artificielle et l'intelligence artificielle distribuée. Le principe sur lequel se base l'intelligence collective, se retrouve dans le monde animal et plus particulièrement chez certaines espèces comme les fourmis et les abeilles. Les fourmis sont « stupides » quand elles sont prises individuellement. Mais, si l'on prend toutes les fourmis d'une colonie, on remarque que la somme de toutes les individualités aboutit à l'émergence d'une forme d'intelligence collective, qui leur permet (entre autres) de construire des fourmilières immenses ou de surmonter de nombreux obstacles, choses qu'elles seraient incapables de réaliser seules, et de trouver toujours le chemin le plus court qui mène à la fourmilière. L'intelligence collective émerge de la dynamique des interactions entre les éléments d'un groupe ou d'une société. Elle n'est pas complexe. Au contraire, elle est issue de la somme d'éléments simples mais qui, en se cumulant ou se superposant, aboutissent à un système complexe. La meilleure illustration de ces dires est à chercher dans les images fractales ou dans le langage binaire. C'est à partir de simples lignes de code composées de 1 et de 0 que fonctionnent des programmes informatiques fort complexes. Le but ultime dans Cette perspective est de pouvoir mieux comprendre pour mieux agir sur des systèmes complexes naturels comme le corps humain ou la société, ou bien pour mieux confectionner des systèmes complexes artificiels comme les ateliers flexibles assistés par ordinateur ou les systèmes issus de la robotique collective.

Dans ce contexte, les Systèmes Multi-Agents (SMA) constituent une branche de l'intelligence artificielle distribuée qui s'intéresse plus particulièrement à la conception d'entités artificielles (homogènes ou hétérogènes) capables de s'organiser efficacement pour accomplir collectivement les tâches qui leur sont demandées. Ils permettent, de manière idéale mais dynamique, de modéliser et de représenter des problèmes complexes qui se rapportent entre autres à l'intelligence collective (notre domaine cible dans ce travail), et dont le fonctionnement global émerge des actions des entités. Les principales qualités des modélisations multi-agents sont leur capacité d'intégration et leur flexibilité. Ces systèmes ont ainsi la possibilité de solutionner des difficultés complexes tout en ayant les avantages de la résolution distribuée (vitesse, fiabilité...) et de l'intelligence artificielle (facilité de maintenance, autonomie de décision). Ils permettent aussi de faire intervenir des types d'interaction assez complexes tels que la coopération et la coordination d'actions.

La simulation multi-agents est fondée sur l'idée qu'il est possible de représenter sous forme informatique le comportement des entités qui agissent dans le monde, et qu'il est ainsi possible de décrire un phénomène comme le fruit des interactions d'un ensemble d'entités disposant de leur propre autonomie opératoire. Les SMA apparaissent de ce fait comme des outils de modélisation et de simulation distribués ayant un très grand succès; ils sont totalement adaptés à la complexité des systèmes dans des domaines aussi diversifiés que la biologie, les écosystèmes, le monde des insectes, le transport, les télécommunications, la sociologie, l'économie, et la robotique. Il est à noter que les systèmes à base d'agents réactifs prennent à cet effet (en plus de leur simplicité d'utilisation) une part d'intérêt et d'adoption de plus en plus grandissante, par rapport à leurs concurrents directs : les systèmes à base d'agents cognitifs.

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 d'un système de robots (généralement) autonomes et homogènes ; encore faut-il disposer de méthodes capables de garantir la maîtrise du comportement collectif. C'est pourquoi, il semble naturel de s'inspirer des groupes d'insectes sociaux et d'appliquer la théorie des Systèmes Multi-Agents Adaptatifs en vue d'effectuer des simulations dans ce sens. Comme exemple de comportement de groupe on peut citer

celui de la mise en place d'une circulation dans un collectif de robots mobiles et autonomes transportant des objets en vue de formation d'un Tas.

En s'inspirant de la théorie de l'évolution, des chercheurs essayent dans cet état d'esprit de retrouver les comportements individuels les plus adéquats (problème d'optimisation) menant à accomplir de telles tâches (représentant le comportement collectif). Les algorithmes génétiques appliqués à cet effet se caractérisent par deux propriétés essentielles :

- La diversification : qui évite que la solution se concentre sur de mauvaises zones de l'espace de recherche.
- L'intensification : qui essaie de se concentrer sur les meilleures solutions dans la région de l'espace de recherche en cours d'analyse.

Une fonction d'évaluation note, dans ce cas, la qualité de la solution trouvée à une étape (génération) donnée. On peut dire qu'elle exprime en quelque sorte le phénotype des individus, alors que le génotype est exprimé à travers la chaîne de codage du même individu (chromosome). L'espace de recherche de la meilleure solution est exploré de la manière la plus large possible, sans risque d'être coincé dans un optimum local, ceci est assuré par l'opérateur de croisement qui crée de nouveaux individus à partir de ceux déjà existants (nouveauté qui hérite du passé), plus forts encore, par l'opérateur de mutation, qui crée un nouvel individu spontanément (nouveauté sans héritage).

Cette approche évolutionniste appliquée dans le contexte du travail mené dans ce mémoire (à savoir la formation de Multi-Tas dans un environnement de robotique collectif en vue de réaliser une tâche de classification basée sur un ou plusieurs critères), nous a permis de trouver l'un des automates cellulaires des plus adéquats (implémenté sous forme d'une table dite de consultation : lookup table), guidant les réactions de chaque robot mobile de la collectivité en question afin d'accomplir la macro-tâche prédéfinie de classification.

L'adéquation de chaque robot mobile simulé à son environnement est évaluée selon la fonction d'évaluation précitée, plus connue dans la littérature du genre sous le nom de fonction de fitness. Dans ce mémoire on a proposé trois approches d'évaluation pour la classification monocritère : deux partant du principe de réutiliser ce qui existe déjà (équation proposée par Barfoot et all. Voir chapitre 4 [35, 36]), et la troisième en utilisant notre propre formalisme (voir chapitre 4). Nous avons aussi proposé une approche d'évaluation pour la classification multicritères.

Le problème difficile qu'on a rencontré ici est l'ajustement des paramètres permettant de nous guider vers la solution optimale globale ; comme le temps d'entraînement (qui doit être suffisant pour que les robots simulés puissent arriver à leur fin), le nombre d'agents, la taille de l'espace d'entraînement, et le nombre d'objets. Ainsi que le choix de la version de l'algorithme génétique qui permet de converger le plus rapidement possible vers la solution, et les valeurs des seuils de croisement et de mutation Pc et Pm.

On confirme une autre fois les propos avancés dans la conclusion du chapitre cinq : la simulation multi-agents est d'une aide incontestable pour les chercheurs dans beaucoup de domaines traitant des problèmes distribués en générale, néanmoins, et comme il a été motionné par plusieurs chercheurs, dans le domaine de la robotique collective, la meilleure façon de tester le comportement et d'évaluer l'efficacité du collectif robotique reste le milieu effectif; avec une plate forme réelle, et de vrais robots ; et ce étant donné que le milieu simulé est un milieu discrétisé, parfaitement contrôlé, et où tout événement est répertorié.

La mise à l'échelle du nombre d'agents-robots impliqués dans la simulation, du nombre d'objets à classifier par couleur, ainsi que de la taille de l'espace d'entraînement a permis de prouver l'indépendance des règles sensori-motrices décelées des variations significatives que peut subir le système ainsi considéré; Puisqu'elles arrivent à chaque fois à guider les agents-robots pour aboutir à l'accomplissement de la tâche de classification. Ceci quelque soit l'état initial à partir duquel démarre la collectivité des agents-robots.

Dans la deuxième solution monocritère la classification est réalisée à travers un paramétrage de l'opération de regroupement selon la propriété couleur. Les règles sensori-motrices conduisant à ce genre de regroupement spécifiques sont différentes de celles conduisant à la classification : Les règles trouvées dans cette solution permettent plutôt de faire un regroupement paramétré selon la propriété couleur, la classification devient alors une opération composite, qui invoque le groupement paramétré, en introduisant à chaque appel une nouvelle valeur de couleur. Les comportements de bases doivent prendre dans ce cas le paramétrage de couleurs en considération, autrement dit la couleur à traiter à chaque étape. La troisième solution monocritère essaye de sélectionner les règles sensori-motrices les plus aptes à réaliser la classification sans distinction entre les couleurs à la phase d'entraînement. Ce n'est qu'à la phase d'évaluation qu'on fait intervenir k fonctions de fitness (une pour chaque couleur). L'évaluation globale de l'état du système est calculée en prenant la moyenne de ces k fonctions de fitness. Ceci implique que les comportements de base ne sont pas contraints de prendre en considération des étapes distinctes de couleurs. La solution multicritères a été illustrée à travers deux caractéristiques des objets à classer : la « couleur », et la « forme ». Pour chaque combinaison de valeurs de ces deux caractéristiques proposées on utilise une fonction de fitness, leur moyenne estime l'état global de l'environnement après un temps d'entraînement suffisant.

Les applications effectives de ce travail ne manquent pas, étant donné que le problème de classification est un problème très répondu. Allant de la classification de données dans un environnement purement bases de données distribuées, par exemple, en passant par le domaine de la récupération (tri d'ordures pour séparer le plastique, du verre et des produits organiques dans un environnement de robotique collective par exemple), jusqu'au tri de pièces de fabrication dans des usines destinées à cette fin.

Ceci va nous permettre aussi de promouvoir et d'enrichir du coup notre connaissance de la façon dont les collectivités biologiques réalisent la coordination et la coopération. Les répercutions de ceci dans le domaine des systèmes artificiels sont immédiates. Beaucoup de chercheurs croient que les méthodes massivement parallèles et décentralisées sont la clé pour doter les systèmes artificiels d'intelligence, et l'on présume à cet effet que les systèmes répartis joueront un rôle prépondérant dans le développement de pareil processus dans le proche avenir.

Cela dit, avec tout ce que nous apprenons chaque jour, Il est bien clair que notre compréhension de l'intelligence est entrain de subir à cet effet des changements profonds.

Nos contributions dans ce travail peuvent êtres résumés en ce qui suit :

- Réduction de l'espace de recherche au niveau de chaque table de consultation de 25 fois (12500/500) en considérant trois positions dans l'environnement local de l'agent-robot (à gauche, à droite, et tout droit) au lieu de cinq positions, tout en aboutissant au même résultat. Il est clair que la réduction de l'espace de recherche implique une accélération dans le processus de convergence évolutionniste. Notons que le nombre total de tables de consultation est du coup réduit de 2^{12500} à 2^{500}.
- Proposition d'une fonction d'évaluation, permettant de noter la tâche de classification accomplie par chaque population de robots. Cette fonction a été validée expérimentalement.
- Généralisation du processus évolutionniste conduisant à la formation de tas, à la formation multi-tas, et exploitation de ceci pour réaliser la classification.
- Proposition d'une méthode de classification multicritères, utilisant la conjonction « et » : $critére_1$ **et** $critére_2$ **et** ……**et** $critére_m$.
- Les deux derniers points présentés juste au dessus nous amènent à conclure ce qui suit : La proposition d'une nouvelle méthode de classification, applicable en robotique collective, et éventuellement applicable dans le domaine de l'informatique distribuée : base de données distribuées, fouille de données*, réseaux (routage, etc.), etc.

Perspectives

Parmi nos objectifs futuristes, on peut citer ce qui suit :

- A court et moyen terme, on vise la réalisation d'autres macro-tâches en utilisant le même procédé évolutionniste, puis essayer de trouver des relations entre les comportements de base contribuant à la réalisation de chaque macro-tâche, et d'établir un processus permettant d'adapter automatiquement le comportement des agents-robots, selon la macro-tâche déduite de, inspirée de ou dictée par l'environnement où évoluent les agents-robots. On sait par exemple qu'une fourmi selon la lecture de son environnement local peut chercher de la nourriture ou trier des cadavres (changer de comportement). Une question reste tout de même posée : « est ce que les fourmis changent d'attitudes seulement en faisant une lecture de leur environnement local (comportement stigmergique*) ou bien alors d'autres paramètres rentrent en jeu pour influencer ce changement d'attitudes ? ». On projette aussi d'essayer d'autres techniques pour retrouver de pareils comportements de base, comme le renforcement et les réseaux de neurones (voir chapitre 2). On prévoit aussi (avec la collaboration du département d'électronique de l'université de Batna) de doter un groupe de robots autonomes et mobiles de règles sensori-motrices ainsi trouvées et de les faire évoluer dans un environnement fermé, et concret afin d'estimer le degré d'efficacité (réalisme) de ces règles. En dernier lieu, on projette aussi d'utiliser une nouvelle politique pour la formation de Tas (qu'on peut généraliser ensuite à la formation multi-Tas) à savoir : Faire émerger un point de rencontre ou de rendez-vous de tous les robots, ce qui représente la première étape dans le processus de formation de Tas. La deuxième étape consiste (après le consensus sur le point de rencontre et sa mémorisation) en le transfère optimal des objets manipulés vers ce point de rencontre. Les règles sensori-motrices décelées via le processus évolutionniste vont permettre aux agents-robots, dans un premier temps, de s'accorder sur un point de rendez-vous (Emergence du point de rendez-vous. Ceci peut intéresser les Entomologistes, puisqu'il va apporter du nouveau à leur façon de comprendre le comportement des insectes supposés utiliser de la mémoire dans leurs stratégies). La deuxième phase vise à retrouver les règles sensori-motrices permettant d'user de l'une des meilleures stratégies possibles pour transporter tous les objets vers ce point de rencontre (Emergence de stratégie. Il est clair dans ce cas qu'on doit prendre en considération dans notre fonction d'évaluation le paramètre « temps »). Il est à noter qu'un projet de fin d'étude pour l'obtention du diplôme d'ingénieur est proposé au département d'informatique de l'université de Batna pour cette année scolaire dans ce sens, et que les premiers résultats en ce qui concerne la première phase sont concluants.

- A long terme on projette de développer l'idée de la correspondance (verticale) entre macro-relation et micro-relation. Dans ce qui suit nous expliquons ceci brièvement : Considérons deux tâches distinctes T1 et T2. Dans ce cas, l'idée qu'on veut développer est la suivante : Si on peut établir une relation R entre T1 et T2 qu'on note R(T1, T2), et si T1 émerge à partir des relations sensori-motrices {SM11, SM12, …., SM1n}, on se demande alors, s'il est possible de déduire les relations sensori-motrices {SM21, SM22, …., SM2m} de T2 sans passer par le processus expérimental, précité ou tout autre processus donnant un résultat équivalent.

Pour expliquer ceci nous considérons deux exemples, en essayant d'illustrer pour chacun : une relation R entre deux tâches au niveau macro, et en établissant une relation R' entre les règles sensori-motrices faisant émerger chacune de ces deux macro-tâches. Par la suite on établira une correspondance logique entre la relation R et son équivalente R'.

Exemple1 : Soit la tâche de formation de Tas T1, et la tâche de la répartition équitable T2. Il est clair que T2 est l'inverse de T1, donc il existe entre les deux tâches une relation R(T1, T2) = Inverse(T1, T2). On peut aussi donner la table de consultation de T1 comme représentant toutes les règles sensori-motrices conduisant à la formation de tas, et la table de consultation de T2 comme représentant toutes les règles sensori-motrices conduisant à réaliser T2. Notre problème dans un premier temps consistera à essayer d'établir une relation R' entre les règles sensori-motrices des deux tables de consultation, et de trouver ensuite une correspondance entre R et R'.

Ceci dans le but de pouvoir prédire à partir d'une tâche émergente déjà accomplie, une autre tâche émergente qui n'a pas encore eu lieu.

Exemple2 : Soit un groupe de robots peintres, qui collaborent pour teinter un mur circulaire sous forme d'un cylindre (voir fig a). Ce mur est subdivisé en carrés, et la tâche à faire émerger est que le mur soit peint en une couleur donnée (autre que la couleur blanche). Les comportements de base que les robots peuvent adopter sont :

- Se déplacer à droite, si l'espace à droite est libre, sinon à gauche, sinon en haut, sinon en bas, sinon attendre.
- Peindre le carré visionné devant, s'il est de couleur blanche, sinon changer de position (verticalement ou horizontalement).

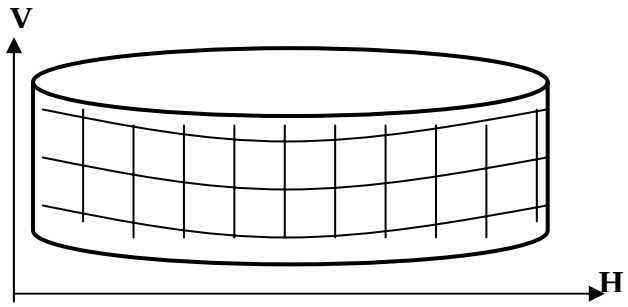


Fig a : Le mur circulaire, divisé en carrés, représentant chacun, le champ de vision d'un robot donné. Le robot peut se déplacer verticalement selon l'axe V et horizontalement selon l'axe H.

Initialement, les robots se positionnent à des emplacements aléatoires, autour du mur, et prennent des hauteurs aléatoires pour peindre, (l'unité étant l'arrête du carré à peindre). La tâche inverse est d'effacer le mur, autrement de le peindre avec la couleur blanche. Notre objectif dans un premier temps est de trouver une relation entre les règles sensori-motrices des deux tâches.

En ce qui concerne la première tâche T1 : « Peindre le mur », les règles sensori-motrices qu'on propose, et qui restent à vérifier expérimentalement, sont :

Sm11: carré visualisé de couleur blanche → peindre carré en couleur spécifiée,
Sm12 : carré visualisé de couleur spécifié → se déplacer à droite sinon à gauche sinon en haut sinon en bas, sinon attendre.

En ce qui concerne la tâche T2 : « Effacer mûr », les règles sensori-motrices sont :

Sm21 : carré visualisé de couleur blanche → se déplacer à droite sinon à gauche sinon en haut sinon en bas sinon attendre.
Sm22 : carré visualisé de couleur spécifique → peindre carré en couleur blanche.

La tâche T2 est l'inverse de T1, et les règles sensori-motrices {sm1i} sembles être le contraire des règles sensori-motrices {sm2j} de T2. A ce stade, on peut énoncer une règle, qui stipule que :

Deux règles sensori-motrices (s, m) concernant T1 et (s', m') concernant T2 sont l'une l'inverse de l'autre, si pour la même vision locale d'entrée s = s', la sortie m' selon T1 est l'inverse de la sortie m selon T2.

En appliquant cette règle à notre exemple on obtient ce qui suit :

- pour la même entrée « carré blanc » on a pour T1 peindre, c-à-d peindre en couleur désignée et ne pas se déplacer, et on a pour T2 se déplacer, c-à-d ne pas peindre et se déplacer.
- Pour la même entrée « carré de couleur désignée » on a pour T1 se déplacer, c-à-d ne pas peindre et se déplacer, et on a pour T2 Peindre c-à-d peindre et ne pas se déplacer.

On remarque que ceci vérifie bien la règle énoncée ci-avant. Dans cet exemple R = R'. Mathématiquement parlant, on peut dire qu'entre R et R' existe une fonction d'identité qu'on note Id : Id (R) = R'.

6.1. Introduction

Netlogo et un environnement fort intéressant pour comprendre les phénomènes d'émergence et les systèmes collectifs. On rappelle que les phénomènes « émergents » (voir chapitre 1) sont des caractéristiques apparaissant de manière inattendue (le terme consacré est « contre-intuitive ») au sein de systèmes complexes composés de nombreux éléments. La pensée, la conscience, se manifestant par l'interconnexion de milliards de neurones est un phénomène émergent. Un vol d'oiseaux, des centaines de lucioles synchronisant leur éclat, l'apparition brusque de la pauvreté au sein d'un marché libre qui devrait en théorie répartir également les richesses, en sont d'autres exemples. L'urbanisme n'est pas non plus avare en phénomènes émergents : par exemple, lorsqu'une ville est encombrée, le bon sens dicte d'ajouter des voies perpendiculaires pour dégager les grosses artères. Ce que firent par exemple les urbanistes de Stuttgart, qui constatèrent, à leur grande surprise, un résultat exactement inverse.

Pour comprendre comment fonctionne cette " émergence ", on doit disposer d'une nouvelle catégorie d'outils intellectuels et matériels qui nous permettent d'appréhender cette réalité extraordinaire. Il s’agit d’une classe d’outils, dont le premier à été crée par Mitchel Resnick, du MIT : Starlogo, à destination des enfants. Le mot " logo " accolé au programme dit tout sur son origine. C’est une variante du langage logo de Seymour Papert, créé dans les années 80 pour initier les plus jeunes à l'informatique. Resnick, ayant en effet constaté l'incapacité dans laquelle nos contemporains se trouvaient de comprendre les systèmes collectifs et décentralisés avait conclu à la nécessité de créer un système leur permettant de « toucher du doigt » une réalité aussi abstraite. Naturellement les jeunes générations étaient les premières visées par son projet, car plus tôt on apprend à comprendre ces concepts difficiles, plus on sera capable de les manier arrivé l'âge adulte. Resnick, qui après Starlogo, s'est dirigé vers la création de " briques programmables " qui inspirèrent la gamme Lego Mindstorms, s'est d'ailleurs spécialisé en matière ludique. Mais ces jouets ne sont pas pour autant limités : on peut faire avec eux des choses très compliquées.

Avec Starlogo Resnick a déclenché une petite révolution : il a inventé la notion d'outil philosophique ; Jusqu'ici, la pensée s'exprimait essentiellement par l'écriture, ou par la parole. Elle va maintenant pouvoir s'incarner dans un programme. Car un code Starlogo n'a pas pour but de produire une « application » qui doit être utilisée, mais à aider à penser, à découvrir, voire à convaincre. Dans la foulée, Resnick a inventé une nouvelle sorte d'écriture : « l'essai actif ». Il s'agit d'un mélange de texte écrit et de programmes informatiques (par exemple une page web dans laquelle sont inscrites des applets java) permettant au lecteur de tester et comprendre les idées exprimées en jouant de manière interactive avec les programmes.

Pourquoi Netlogo ?

Puisque Starlogo est le premier langage du genre, pourquoi se consacrer plutôt à Netlogo ? Lorsque Resnick a crée son outil, dans les années 90 et qu'il a sorti son manuel de base [1] : « Turtles, Termites and Traffic Jams », Starlogo ne fonctionnait que sur des machines très puissantes dites « parallèles » : Etant donner qu’elles étés capables d'effectuer de nombreuses opérations simultanément. Finalement, Starlogo est passé sous Macintosh, puis est devenu accessible pour toutes les machines grâce à sa version java. Il est à noter que Starlogo est gratuit, et peut être téléchargé. Mais au fur et à mesure que Starlogo devenait plus accessible, les versions évoluaient et du coup, bon nombre des instructions données dans le livre de Resnick furent modifies, rendant son « turtles, termites... » quasi inutilisable. Face à Starlogo, à la Northwestern University, on créa un clone, le Netlogo, possédant lui aussi son set spécifique d'instructions, modifié par rapport au Starlogo originel. Netlogo a la réputation d'être plus rapide et plus stable.

La version 3.0.2 de *NetLogo*, offre entre autres un affichage en trois dimensions permettant de mieux distinguer les agents mobiles des cellules du maillage spatial sur lequel ils se déplacent et avec lequel ils interagissent. La nouvelle version intègre d'autre part une interface de modélisation pour les Dynamiques de Système.

Le logiciel est livré avec une bibliothèque étendue d'exemples de projets dont un bon nombre implémentant des modèles dynamiques bien connus des sciences sociales ou autres.

6.2. Les Acteurs principaux

Un agent est une entité qui a la possibilité d'interagir avec son environnement et avec d'autres agents. Cette entité peut être une personne, un animal, un insecte, une cellule, un pays, etc… Les agents ont la particularité de pouvoir suivre des instructions ou des lois. En effet, le comportement de chaque agent est déterminé par un ensemble de lois, en général des lois stimulus-réponse typiques et simples.

NetLogo est un logiciel permettant de simuler les interactions entre un grand nombre d'agents, puis de voir ce qui se passe lorsque les agents sont programmés pour suivre des règles spécifiques. NetLogo permet de montrer que lorsque des agents suivent des loissimples, le résultat est parfois complexe et inattendu. Les agents vivent sur un support 2D constitué de cellules appelées « patches ». Ces cellules ont des coordonnées. Il y a trois types d'agents : Les tortues (turtles), Les patches, et L'observateur.

6.2.1. Les tortues : Dans NetLogo on peut manipuler des entités, qu'on peut appeler fourmis, termites, loups, etc. Ce sont des sortes de races. On fait remarquer que par défaut ces entités sont dénommées tortues. Ils sont appelés « tortues » en hommage au langage de programmation dont dérive NetLogo dans lequel le programmeur contrôle des tortues sur un écran. Les tortues répondent à des commandes ou des fonctions. Elles peuvent être des centaines, voire des milliers, sur l'écran au même moment. Les tortues ont pour coordonnées xcor et ycor. Ces coordonnées peuvent être des flottants.

6.2.2. Les patchs : Les patchs constituent l'environnement des tortues, ce sont les cases de l'écran principal de NetLogo. Grâce à la programmation des patchs, on peut donc faire intervenir l'environnement des tortues; ceci permet d'établir des modèles beaucoup plus proches de la réalité, quand on sait l'importance des interactions entre les individus et leurs environnements dans l'établissement des comportements. Ils vous permettront par exemple, de mettre de la nourriture dans l'environnement des tortues, ou de laisser des phéromones.

Chaque patch est un carré de sol (une case) sur lequel les tortues peuvent se déplacer. Le patch central a pour coordonnées (0,0). Les coordonnées d'un patch sont appelées pxcor et pycor. Le nombre total de patches est déterminé par les variables screen-edge-x et screenedge-y.

Screen-edge-x : valeur maximale en abscisses.
Screen-edge-y : valeur maximale en ordonnées.

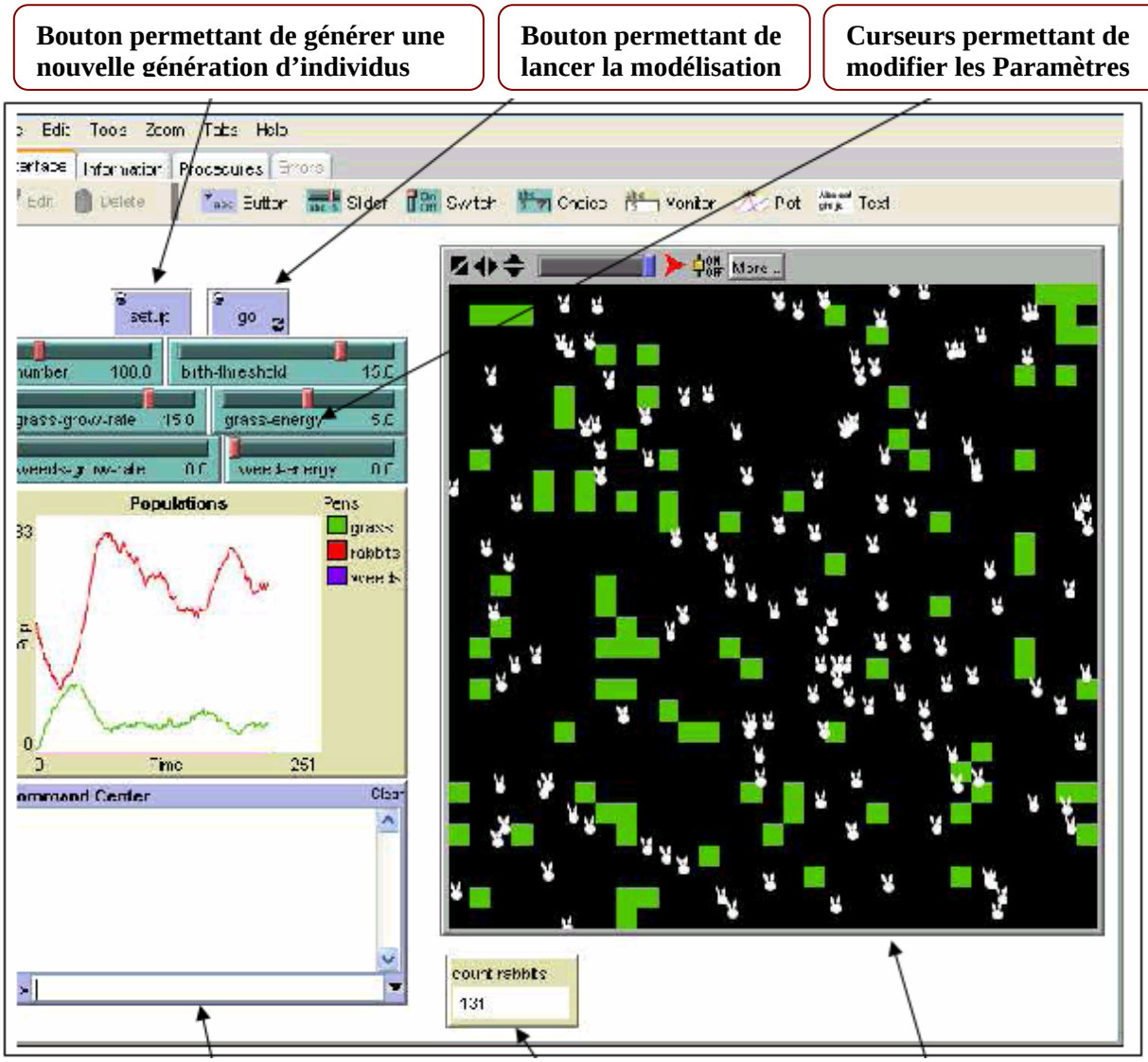
Par défaut, ces deux variables sont à 17. Il y a donc 1225 patches au total sur l'écran. On peut changer ce nombre dans la fenêtre NetLogo's Graphics. Les coordonnées des patches sont des entiers. Notons que le support des patches n'est pas borné. Chaque patch a le même nombre de voisins. Ainsi si une tortue disparaît d'un côté de l'écran, elle réapparaît du côté opposé.

6.2.3. L'observateur : C'est une sorte de superviseur dans le monde des tortues. L'observateur est extérieur au monde dans lequel vivent les tortues mais il peut superviser ce qu'il s'y passe et ordonner aux tortues de faire certaines choses. C'est lui que l'on utilise lorsque l'on veut gérer un programme dans NetLogo.

Il permet également d'intervenir alors même que le programme est déjà lancé. L'observateur peut être utilisé pour attribuer des ordres spécifiques à des patches ou à des tortues. Il collecte également des données pour créer des graphiques.

6.3. Le logiciel NetLogo

6.3.1. L'interface graphique : l'interface graphique de NetLogo se présente comme suit :



Le code de la simulation est accessible dans l'onglet « procédures ». Les boutons, les curseurs, les fenêtres d'affichage de courbes et les fenêtres d'affichage de paramètres sont créés à partir de la barre d'outils de l'interface graphique.

6.3.2. Reporters : Les Reporters exécutent une opération, et retournent un résultat soit à une commande soit à un autre reporter (un peu comme une fonction). Il existe deux types de reporters :

- Reporters primitives : prédéfinis dans NetLogo
  Exemple : la fonction Random.
- Reporters procédures : définis par le programmeur.

6.3.3. Le langage de programmation

6.3.3.1. Les différents types de variables

Il y a trois types de variables :

- variable globale
- variable tortue
- variable patch.

Une variable globale n'a qu'une seule valeur pour cette variable, et tous les agents peuvent y accéder. Chaque tortue a sa propre valeur pour toutes les variables tortues et chaque patch a sa propre valeur pour toutes les variables patches. Les variables patches commencent par un « p » pour ne pas les confondre avec les variables tortues.

Exemple : Chaque tortue a une variable « color » et chaque patch a une variable « pcolor ». Il y a des variables prédéfinies : xcor, ycor, heading, pxcor, pycor, etc. On peut également définir nos propres variables.

Définition des variables :

Les différentes variables se définissent de la manière suivante :

- Variable globale : **globals** [clock]
- Variable tortue : **turtles-own** [energy speed]
- Variable patch: **patches-own** [friction]

On utilise la commande « **set** » pour les initialiser. Par défaut, elles sont initialisées à 0. Les variables globales peuvent être lues et modifiées par n'importe quel agent à tout moment. Une tortue peut lire et modifier les variables patches du patch sur lequel elle agit. Lorsque l'on veut qu'un agent lise ou modifie la variable d'un autre agent, il faut mettre «**–of** » après le nom de la variable et spécifier, après, de quel agent il s'agit.

Exemples :

Set color-of turtle 5 red	qui veut dire : *la tortue n° 5 devient rouge.*
Set pcolor-of patch 2 3 green	qui veut dire : *le patch de coordonnées (2,3) devient vert.*

6.3.3.2. Les variables locales

C'est une variable exclusive à une fonction. On la définit de la manière suivante : **locals** [var1 var2 …]. Ces variables ne sont utilisables qu'à l'intérieur de la fonction. Elles doivent être définies au début de la fonction, avant toute commande.

6.3.3.3. Les variables prédéfinies

Il existe plusieurs variables prédéfinies relatives aux variables « tortues » :

breed : (= race)
Elle permet de sélectionner l'ensemble des tortues de la même race.
Exemple : if breed = chat [show « miaou ! »]

breeds : Ce mot clé permet de définir l'ensemble des races. Il ne peut être utilisé qu'en début de programme, avant toute définition de fonction.
Exemple : breeds [souris grenouille]

Color : Elle définit la couleur d'une tortue.

heading : Elle indique vers quelle direction la tortue est dirigée. Heading appartient à [0, 360].
0 = Nord, 90 = Est, 180 = Sud, 270 = Ouest.

hidden ? Elle contient un booléen indiquant si la tortue est visible ou non. On peut modifier cette variable pour faire apparaître ou disparaître une tortue.

label : Elle permet d'associer une valeur à une tortue. Cette valeur peut être de n'importe quel type.

Label-color : Elle détermine la couleur du label d'une (si elle en a un). Label-clor appartient à [0,140].

shape : Elle contient une chaîne de caractères représentant le nom de la configuration courante de la tortue.

size : La taille par défaut est 1.0. La tortue a la même taille qu'un patch. Toutes les tortues ont la même taille sauf si on coche la case « turtle-sizes ».

who : Elle contient l'identificateur de la tortue (>= 0). On ne peut pas modifier cette variable.
Exemple : show values-from (turtles with [color = red]) [who]
liste les identificateurs des tortues rouges.

xcor, ycor : Ce sont les coordonnées de la tortue.

Il existe également plusieurs variables patches prédéfinies : pcolor, plabel, plabel-color, pxcor, pycor. Leur fonction est la même que pour les variables tortues.

6.3.3.4. Les fonctions

Une fonction se définit de la manière suivante :

```
 to nom_fonction
    corps_de_la_fonction
end
```

Les fonctions prédéfinies du langage NetLogo sont répertoriées dans une page du logiciel appelée *The Primitives Dictionary*.

Il y a deux fonctions à définir obligatoirement dans un programme NetLogo : la fonction **setup** et la fonction **go**. C'est à l'utilisateur de définir ces deux fonctions. La fonction setup définit l'état initial du modèle. La fonction go démarre le processus de modélisation.

6.3.3.5. Les structures

La structure if :
Syntaxe : if condition [traitement]
Exemple : ask patches [if pxcor > 0 [set pcolor blue]

La structure ifelse :
Syntaxe: ifelse condition [traitement_si_la_condition_est_vérifiée] [traitement_sinon]
Exemple : ask patches [ifelse pxcor > 0 [set pcolor blue] [set pcolor red]]

La structure ask :
Cette structure permet d'appliquer une ou plusieurs instructions à toutes les variables d'un même type. En effet, la structure suivante :

ask turtles
[
…..]

permet d'appliquer à toutes les variables de type tortue les instructions qui se situeront dans le corps de la structure. On peut également affiner la sélection en précisant les spécificités de ces variables. En effet, la structure suivante :

ask turtles with [malade?]
[
…..
]

permet d'appliquer à toutes les variables de type tortue étant malades les instructions qui situeront dans le corps de la structure.

6.3.3.6. La gestion des fenêtres d'affichage

En ce qui concerne l'affichage d'une courbe : après avoir créer une fenêtre permettant l'affich age d'une courbe dans l'interface graphique, il suffit de mettre à jour à chaque pas de temps la valeur de la courbe (ou des courbes) que l'on souhaite afficher. Considérons les lignes de code suivantes :

set-current-plot "population"
set-current-plot-pen "total"
plot count turtles

La fonction set-current-plot permet de spécifier que l'on va mettre à jour la fenêtre d'affichage appelée « population ». On spécifie alors que l'on va considérer la courbe appelée « total » de cette fenêtre à l'aide de la fonction set-current-plot-pen. Finalement on met à jour la valeur de la courbe en fournissant en paramètre de la fonction plot l'opération à effectuer. Dans notre exemple, on compte le nombre total d'individus.

En ce qui concerne l'affichage d'une fenêtre contenant un paramètre (qui est une variable globale) : après avoir créer une fenêtre permettant l'affichage d'un paramètre, il suffit de remettre à jour à chaque pas de temps la valeur de ce paramètre.

6.4. Exemple de code NetLogo

```
To setup                          ; Procédure setup
  Ca
  Crt 100

  Ask turtles
          [  set color red          ; colorer les tortues en rouge
             rt random 360          ; orienter chacune au hasard
    fd 50 ]                         ; les faire avancer de 50 pas

  Ask patches
          [if (pxcor > 0)           ; colorer les patches du coté droit
              [set pcolor green] ]  ; de l'écran en vert
End
```

6.5. Conclusion

La simulation basée agent fait depuis longtemps parti des moyens que la science s'est donnée pour saisir et représenter des phénomènes dynamiques complexes étudiés dans tous les domaines de recherche. C'est en sciences humaines que ce type d'approche se montre particulièrement riche, permettant d'explorer d'une façon rigoureuse les liens existant entre les diverses échelles du phénomène social ; d'aller, par exemple, au bout de nombreuses hypothèses portant sur le lien entre des stratégies de comportement individuelles et les phénomènes sociaux globaux qui en émergent. Parmi les nombreux (madkit, …) logiciels permettant de faire usage de tels modèles se distingue notamment « NetLogo ».

Donc, NetLogo est un logiciel de modélisation d'environnement pour simuler des phénomènes naturels et sociaux. Il est particulièrement adapté pour modéliser des systèmes complexes de développement à long terme. Les utilisateurs de NetLogo peuvent appliquer des instructions à des centaines ou des milliers d'agents opérant en parallèle. Le logiciel permet ainsi d'étudier la connexion qui existe entre le comportement d'individus isolés et d'en déduire des modèles qui découlent de l'interaction entre de nombreux individus. Ces simulations s'appliquent (entre autres) aux sciences naturelles et sociales, à savoir la biologie et la médecine, la physique et la chimie, les mathématiques et l'informatique, et l'économie.

6.5. Références

Références bibliographiques

[1] Axtell, R., Axelrod, R., Epstien, J. M., and Cohen, M. D. (1996). "Aligning simulation models: A case study and results. Computational and Mathematical Organization Theory", 1:123–141.
[2] Resnick, M. (1997). " Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds". MIT Press.
[3] Manuelle d'utilisation de Netlogo 3.02
[4] Stéphane P. & Elodie K. « Simulations individus-centrées de phénomènes d'épidémiologie avec NetLogo » Université du Havre, Faculté des Sciences et Techniques, 2003/2004.

Références sur le Web

http ://ccl.northwestern.edu/netlogo/
http ://www.swarm.org/
http ://jasss.soc.surrey.ac.uk/
http ://www.cpm.mmu.ac.uk/
http ://www.alife.org
http ://animatlab.lip6.fr/
http://www.cbs.umn.edu/populus/T
http://www.avignon.inra.fr/internet/unites/biometrie/mobidyc_projet/version_index_html
http://lil.univ-littoral.fr/Mimosa/
http://cormas.cirad.fr/
http://wiki.swarm.org/wiki/Main_Page

Autres logiciels similaires

Madkit : MadKit http ://www.madkit.org/
Starlogo : http://www.media.mit.edu/starlogo/

Voir aussi :

Cormas développé par le CIRAD (2003), et qui est spécifique au domaine de la gestion des ressources renouvelables,
Swarm développé par le Swarm Development Group à Santa Fe (2004).

Glossaire

Biomimétique : La biomimétique est une technique qui étudie et s'inspire de la Nature pour résoudre des problèmes quotidiens. C'est une nouvelle discipline basée non pas sur ce qui peut être extrait de la nature mais sur ce qui peut être appris d'elle. Par exemple, l'araignée est capable de tisser une soie cinq fois plus résistante que l'acier, et arrive à cette « prouesse » sans température élevée (consommatrice d'énergie) et sans émettre de dioxine.

Essaim : Colonie d'abeilles sortant de la ruche mère pour fonder une nouvelle ruche. Groupe nombreux d'animaux de même espèce.

Fouille de données : C'est l'extraction d'informations originales, auparavant inconnues et potentiellement utiles, à partir de données. C'est aussi la découverte de nouvelles corrélations, tendances et modèles par le tamisage d'un large volume de données. C'est aussi un processus d'aide à la décision où les utilisateurs cherchent des modèles d'interprétation dans les données.

Couvain : Œuf d'insecte

Cybernétique : La cybernétique (du grec kubernân, qui signifie diriger) est une science fondée en 1948 par le mathématicien américain Norbert Wiener (dans Cybernetics, or Control and Communication in the Animal and the Machine). Son champ d'application ne s'applique pas qu'à l'artefact mais aussi au domaine animal et naturel. La cybernétique est la prise de conscience du processus vital qui maintient en équilibre l'ensemble des phénomènes. C'est la science de l'efficacité et du gouvernement par le contrôle organisé de toutes les informations y compris celles qui concernent les perturbations de toute nature, en vue de leur traitement pour parvenir à la régulation optimale de tout phénomène organique, physique ou esthétique. Il en résulte une permanence fluide, en équilibre souple, où chaque apparition d'une tendance à la périodicité ou à la stagnation provoque l'intervention des perturbations adéquates pour conserver l'ouverture et le caractère aléatoire de tout processus évolutif.

Empirique : qui procède plus par tâtonnement, par sentiment que par réelle connaissance scientifique. Ainsi, empiriquement l'homme a eu le sentiment que la Terre était un disque, et que le soleil tournait autour d'elle. Il en a déduit la suprématie de la Terre et des hommes, faits par Dieu à son image. Galilée a mis en évidence le système solaire, par une approche scientifique.

Entomologiste : Personne qui s'occupe d'entomologie, partie de la zoologie qui traite des insectes.

Entropie : L'entropie est une grandeur thermodynamique. C'est une quantité physique, mesurable, associée au degré de désordre d'un système macroscopique, ou au manque d'informations sur son état microscopique. En informatique on peut parler d'entropie d'une suite de bits et mesurer son degré de désordre : plus cette suite se rapproche d'une suite aléatoire et plus son entropie sera grande. Plus l'entropie de cette suite est élevée, plus il devient difficile de la compresser.

Ethologie : Science qui étudie, et décrit le comportement des animaux dans leur environnement.

Génotype : Ensemble des gènes d'un individu, qui caractérisent un être vivant, et qui constituent son hérédité.

Heuristique : Une heuristique est l'utilisation de règles empiriques (). C'est aussi une technique qui consiste à apprendre petit à petit, en tenant compte de ce que l'on a fait précédemment pour tendre vers la solution d'un problème. L'heuristique ne garantit pas du tout qu'on arrive à une solution quelconque en un temps fini. Opposé à algorithmique.

L'heuristique est essentiellement utilisée dans les antivirus, pour détecter des virus en les reconnaissant selon ce qu'ils sont capables de faire plutôt que selon une signature fixe.

Homéostasie : Tendance des êtres vivants à maintenir constants et en équilibre leur milieu interne et leur paramètres physiologiques.

Kénétique : consiste à pouvoir étudier, concevoir et réaliser des univers ou des organisations d'agents artificiels, capables d'agir, de collaborer à des tâches communes, de communiquer, de s'adapter, et de se représenter l'environnement dans lequel ils évoluent et de planifier leur actions

Néodarwinisme : Théorie de l'évolution qui n'admet pas l'hérédité des caractères acquis et considère la sélection comme résultant exclusivement des mutations génétiques et de l'influence du milieu

Organisation : Mode de constitution et de fonctionnement d'un système.

Perceptron : C'est un ensemble de neurones formels (modèle informatique visant à reproduire le fonctionnement des neurones du cerveau) interconnectés entre eux.

Il en existe deux types :

- le perceptron classique (deux couches de neurones)
- le perceptron multicouches (n couches de neurones)

Phéromone : Les phéromones sont des hormones émises par la plupart des animaux et qui agissent comme des messagers sur des individus de la même espèce. Extrêmement actives elles agissent en quantités infinitésimales, si bien qu'elles peuvent être détectées, même transportées à plusieurs kilomètres. Chez les mammifères et les reptiles, les phéromones sont détectées par l'organe voméro-nasal, tandis que les insectes utilisent généralement leurs antennes.

Phénotype : Aspect observable de l'individu, conditionné par son génotype et le milieu environnant.

Stigmergie : c'est un mode de communication particulier qui a été baptisé ainsi par le biologiste Pierre-Paul Grassé, en 1959. Celui-ci étudiait en particulier la construction du nid chez les termites. Voici son commentaire : « La coordination des tâches et la régulation des constructions ne dépendent pas directement des ouvriers, mais des constructions elles-mêmes. L'ouvrier ne dirige pas son travail, il est guidé par lui. C'est à cette stimulation d'un type particulier que nous donnons le nom de stigmergie ». En effet, les termites, comme les fourmis, laissent des traces de phéromones sur leur passage. Elles modifient donc leur environnement. Ces traces attirent leurs congénères, qui laissent des traces à leur tour, etc. Pour les araignées, ces traces sont les fils de soie.

Glossaire

Biomimétique : La biomimétique est une technique qui étudie et s'inspire de la Nature pour résoudre des problèmes quotidiens. C'est une nouvelle discipline basée non pas sur ce qui peut être extrait de la nature mais sur ce qui peut être appris d'elle. Par exemple, l'araignée est capable de tisser une soie cinq fois plus résistante que l'acier, et arrive à cette « prouesse » sans température élevée (consommatrice d'énergie) et sans émettre de dioxine.

Couvain : Œuf d'insecte

Cybernétique : La cybernétique (du grec kubernân, qui signifie diriger) est une science fondée en 1948 par le mathématicien américain Norbert Wiener (dans Cybernetics, or Control and Communication in the Animal and the Machine). Son champ d'application ne s'applique pas qu'à l'artefact mais aussi au domaine animal et naturel. La cybernétique est la prise de conscience du processus vital qui maintient en équilibre l'ensemble des phénomènes. C'est la science de l'efficacité et du gouvernement par le contrôle organisé de toutes les informations y compris celles qui concernent les perturbations de toute nature, en vue de leur traitement pour parvenir à la régulation optimale de tout phénomène organique, physique ou esthétique. Il en résulte une permanence fluide, en équilibre souple, où chaque apparition d'une tendance à la périodicité ou à la stagnation provoque l'intervention des perturbations adéquates pour conserver l'ouverture et le caractère aléatoire de tout processus évolutif.

Empirique : qui procède plus par tâtonnement, par sentiment que par réelle connaissance scientifique. Ainsi, empiriquement l'homme a eu le sentiment que la Terre était un disque, et que le soleil tournait autour d'elle. Il en a déduit la suprématie de la Terre et des hommes, faits par Dieu à son image. Galilée a mis en évidence le système solaire, par une approche scientifique.

Entomologiste : Personne qui s'occupe d'entomologie, partie de la zoologie qui traite des insectes.

Entropie : L'entropie est une grandeur thermodynamique. C'est une quantité physique, mesurable, associée au degré de désordre d'un système macroscopique, ou au manque d'informations sur son état microscopique. En informatique on peut parler d'entropie d'une suite de bits et mesurer son degré de désordre : plus cette suite se rapproche d'une suite aléatoire et plus son entropie sera grande. Plus l'entropie de cette suite est élevée, plus il devient difficile de la compresser.

Essaim : Colonie d'abeilles sortant de la ruche mère pour fonder une nouvelle ruche. Troupe nombreuse

Ethologie : Science qui étudie, et décrit le comportement des animaux dans leur environnement.

Fouille de données : C'est l'extraction d'informations originales, auparavant inconnues et potentiellement utiles, à partir de données. C'est aussi la découverte de nouvelles corrélations, tendances et modèles par la fouille et le tri dans un large volume de données. C'est aussi un processus d'aide à la décision où les utilisateurs cherchent des modèles d'interprétation dans les données.

Génotype : Ensemble des gènes d'un individu, qui caractérisent un être vivant, et qui constituent son hérédité.

Heuristique : Une heuristique est l'utilisation de règles empiriques (). C'est aussi une technique qui consiste à apprendre petit à petit, en tenant compte de ce que l'on a fait précédemment pour tendre

vers la solution d'un problème. L'heuristique ne garantit pas du tout qu'on arrive à une solution quelconque en un temps fini. Opposé à algorithmique. L'heuristique est essentiellement utilisée dans les antivirus, pour détecter des virus en les reconnaissant selon ce qu'ils sont capables de faire plutôt que selon une signature fixe.

Homéostasie : Tendance des êtres vivants à maintenir constant et en équilibre leur milieu interne et leurs paramètres physiologiques.

Kénétique : consiste à pouvoir étudier, concevoir et réaliser des univers ou des organisations d'agents artificiels, capables d'agir, de collaborer à des tâches communes, de communiquer, de s'adapter, et de se représenter l'environnement dans lequel ils évoluent et de planifier leur actions.

Mimétisme : Propriété qu'ont certaines espèces animales de se confondre avec le milieu environnant ou de ressembler à une autre espèce. Instinct d'imitation.

Néodarwinisme : Théorie de l'évolution qui n'admet pas l'hérédité des caractères acquis et considère la sélection comme résultant exclusivement des mutations génétiques et de l'influence du milieu

Organisation : Mode de constitution et de fonctionnement d'un système.

Perceptron : C'est un ensemble de neurones formels (modèle informatique visant à reproduire le fonctionnement des neurones du cerveau) interconnectés entre eux.

Il en existe deux types :

- le perceptron classique (deux couches de neurones)
- le perceptron multicouches (n couches de neurones)

Phéromone : Les phéromones sont des hormones émises par la plupart des animaux et qui agissent comme des messagers sur des individus de la même espèce. Extrêmement actives elles agissent en quantités infinitésimales, si bien qu'elles peuvent être détectées, même transportées à plusieurs kilomètres. Chez les mammifères et les reptiles, les phéromones sont détectées par l'organe voméro-nasal, tandis que les insectes utilisent généralement leurs antennes.

Phénotype : Aspect observable de l'individu, conditionné par son génotype et le milieu environnant.

Propriétés d'agent :

Autonome : La prise de décision sur son comportement est uniquement fonction de ses perceptions, connaissances, et représentation du monde (un agent peut être dépendant et autonomes).
Proactif : Génère ses buts, prise d'initiative pour satisfaire ses buts, pas dirigé seulement par les évènements.
Flexible : réaction aux changements dans l'environnement; adaptation aux ressources disponibles.
Social : capacité à interagir pour atteindre ses buts, pour aider d'autres agents dans leurs activités.
Situé : capacité à percevoir l'environnement et à y agir de façon limitée.

Stigmergie : c'est un mode de communication particulier qui a été baptisé ainsi par le biologiste Pierre-Paul Grassé, en 1959. Celui-ci étudiait en particulier la construction du nid chez les termites. Voici son commentaire : « La coordination des tâches et la régulation des constructions ne dépendent pas directement des ouvriers, mais des constructions elles-mêmes. L'ouvrier ne dirige pas son travail, il est guidé par lui. C'est à cette stimulation d'un type particulier que nous donnons le nom de stigmergie ». En effet, les termites, comme les fourmis, laissent des traces de phéromones sur leur passage. Elles modifient donc leur environnement. Ces traces attirent leurs congénères, qui laissent des traces à leur tour, etc. Pour les araignées, ces traces sont les fils de soie.