Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mohamed Khider Biskra

Faculté des Sciences Exactes et des Sciences de la Nature et de la Vie

Département d'Informatique

N°d'ordre :……
Série :……

Mémoire

Présenté en vue de l'obtention du diplôme de Magister en Informatique

Option**: Génie Logiciels** (Ecole doctorale)

Une approche automate cellulaire pour L'apprentissage d'un agent

Par :

M^{elle} Merabet Rabiya

Soutenu le :17 /06 /2012

Devant le jury :

Mr. Mohamed Benmohamed,	**Pr. Université de Constantine**	**Président**
Mr. Kazar Okba,	**Pr. Université de Biskra**	**Rapporteur**
Mr. Kamel Eddine Melkemi,	**M.C.A Université de Biskra**	**Examinateur**
Mr. Nacer Khelil,	**M.C.A Université de Biskra**	**Examinateur**

Remerciements

Je tiens à remercier très sincèrement mon encadreur, Professeur Kazar Okba, Professeur à l'Université de Biskra, pour m'avoir mis sur la voie de la recherche et d'avoir accepté de diriger ce modeste projet, pour les conseils qu'elle m'a prodigué et pour les efforts qu'elle a consenti tout au long de la réalisation de ce travail. Pour sa patience quasi sans limites à mon égard, et son soutient sans lequel ce travail n'aurait pas vu le jour. Merci beaucoup.

Comme je remercie Monsieur Mohamed Benmohamed, Professeur à l'Université de Mentouri de Constantine, pour avoir accepté la présidence du jury.

Je remercie vivement Monsieur Kamel Eddine Melkemi, Maître de Conférence à l'Université de Biskra, ainsi que Monsieur Nacer Khelil, Maître de Conférence à l'Université de Biskra, pour avoir accepté de participer à ce jury.

Je remercie aussi Monsieur ali abbassene, Monsieur laid kahloul, et Monsieur Zakaria Boumerzoug.

Enfin, que toutes les personnes qui ont participé de prés ou de loin à la réalisation de ce travail trouvent ici l'expression de mes sincères remerciements.

Résumé

L'apprentissage automatique (machine - learning en anglais) est un des champs d'étude de l'intelligence artificielle. L'apprentissage automatique fait référence au développement, à l'analyse et à l'implémentation de méthodes qui permettent à une machine (au sens large) d'évoluer grâce à un processus d'apprentissage, et ainsi de remplir des tâches qu'il est difficile ou impossible de remplir par des moyens algorithmiques plus classique. On peut concevoir un système d'apprentissage automatique permettent à un robot, ayant la capacité de bouger ses membres, mais ne sachant rien de la coordination des mouvements permettant la marche, d'apprendre à marche. Le robot commencera par effectuer des mouvements aléatoire, puis, en privilégiant les mouvements lui permettant d'avancer, mettra peu à peu en place une marche de plus en plus efficace …

L'objectif du travail consiste à proposer une approche d'apprentissage à base d'automate cellulaire pour un agent.

Mots clés : apprentissage, automate cellulaire, système multi–agent.

Abstract

The machine learning is a field of study of artificial intelligence. The machine learning refers to the development, analysis and implementation of methods that allow a machine(in the broad sense) to evolve through a learning process , and so fulfill the tasks which are difficult or impossible to perform in more conventional algorithmic ways. We can design a machine learning system that enables a robot which has the ability to move its limbs, but does not know anything about the coordination of movement for walking, to learning to walk. The robot will begin by performing random motion, then, by privileging the movements that enable it to go forward, will gradually put in place a more and more efficient step...

The objective of this work is to propose a learning approach based on cellular automaton for agent.

Keywords: learning, cellular automata, multi-agent.

Table des matières

Table des Figures

Liste Des Abréviations

Abrev	**Description**
(IA)	**I**ntelligence **A**rtificielle
(ML)	**M**achine **L**earning
(AG)	**A**lgorithmes **G**énétique
(RN)	**R**éseaux **D**e **N**eurones
(CBR)	**R**aisonnement **F**ondé **S**ur **D**es **C**as
(AD)	**A**rbre **D**e **D**écision
(MMC)	**M**odèle **D**e **M**arkov **C**achés
(HMM)	**H**idden **M**arkov **M**odels
(K-ppv)	**K**-**P**lus **P**roche **V**oisin
(RB)	**R**éseau **B**ayésien
AC	**A**utomate **C**ellulaire
CA	**C**ellular **A**utomata

Introduction Générale

Introduction Générale

Dans les années 50, il n'y avait pas de différence entre intelligence artificielle et informatique. Depuis, l'IA est devenue la branche de l'informatique qui se préoccupe surtout du raisonnement, de l'apprentissage, de la perception, de la motricité, la créativité et du langage (ce qui est plutôt humain).

L'apprentissage artificiel s'intéresse à l'écriture de programmes d'ordinateur capables de s'améliorer automatiquement au fil du temps, soit sur la base de leur propre expérience, soit à partir de données antérieures fournies par d'autres programmes.

Dans le domaine scientifique relativement jeune de l'informatique, l'apprentissage artificiel joue un rôle de plus en plus essentiel. Au début de son existence, dans les années 1950, l'informatique se résumait principalement à programmer des machines en leur spécifiant ligne après ligne la séquence d'instructions que l'ordinateur aurait à suivre. Autour des années 1990, les logiciels étaient devenus si complexes qu'une alternative s'imposait naturellement: développer des techniques pour que les programmes puissent s'entraîner à partir d'exemples. Le résultat est qu'il existe aujourd'hui de nombreux domaines d'application de l'informatique dans lesquels les méthodes de l'apprentissage artificiel sont employées pour entraîner les logiciels. Mieux, le code résultant dépasse de beaucoup en performance les réalisations les plus abouties de programmation manuelle (ligne après ligne). C'est ainsi que tous les meilleurs logiciels commercialisés de reconnaissance de la parole sont fondés sur l'entraînement de leurs programmes à la reconnaissance des différents sons et mots. La plupart d'entre eux permettent même à l'utilisateur d'accoutumer le système aux caractéristiques de sa voix. D'autres exemples existent dans des domaines tels que la vision par ordinateur, le traitement automatique du texte et la robotique.

La discipline de l'apprentissage artificiel peut donc déjà revendiquer des succès dans un grand nombre de domaines d'application.

Au moment où nous passons des cinquante premières années de l'informatique au les cinquante prochaines, il semble certain que le rôle de l'apprentissage artificiel ne cessera de croître au centre de cette science.

Pourquoi cette progression ? La réponse fondamentale est que nous possédons désormais la compréhension de plusieurs principes calculatoires qui guident tout processus d'apprentissage, qu'il soit implémenté sur une machine ou sur un humain. La discipline de l'apprentissage artificiel possède désormais de riches fondements théoriques : on commence à savoir répondre a de question comme : (Quelles sont les différents types d'apprentissage automatique ?).

Une notion importante en intelligence artificielle et la notion d'agent et son extension aux systèmes multi-agents. À la base, un agent se définit comme une entité qui agit de façon rationnelle. Cependant, en informatique, un agent se distingue d'un programme par ses attributs supplémentaires comme le fait « *d'agir de façon autonome, de percevoir son environnement, d'être persistant pendant de longues périodes et de s'adapter aux changements* ». Pour sa part, un agent apprend à agir toujours de façon à obtenir le « meilleur résultat ». Ces définitions peuvent sembler abstraites, mais l'important est de savoir que le but des agents est d'agir le mieux possible selon la situation et le but souhaité.

Les agents ou systèmes intelligents prendront de plus en plus de place dans nos vies à mesure que les domaines de l'informatique, de l'électronique et de la mécanique se perfectionneront.

En outre, il ne suffit pas à un agent d'être capable de choisir des actions, il faut aussi qu'il soit capable de bien cerner son environnement. Le défi est donc de trouver de nouvelles approches pour l'apprentissage d'un agent pour qu'il puisse apprendre l'expérience d'un comportement. Une approche a été développée appelée Automates Cellulaires "Cellular Automata" (CA).

Les automates cellulaires ont été inventés par Stanislaw Ulam et John von Neumann à la fin des années 40. En 1948, von Neumann a proposé un article, publié dans [von Neumann, 1951], dont une des questions centrales était de savoir s'il était possible de concevoir une machine capable de s'auto-reproduire : En effet, de manière générale, les machines produisent des objets plus simples qu'elles-mêmes. Stanislaw Ulam, collègue de von Neumann, s'intéressait aux « objets géométriques définis de façon récursive » : dans

un monde divis´e en cellules par une grille, l'état d'une cellule évolue en fonction de celui de ses voisines. Il suggéra alors à von Neumann d'utiliser
un tel monde abstrait pour pallier les difficultés pratiques qui se posaient pour la construction de l'automate auto-reproducteur.

Les automates cellulaires sont une classe de systèmes dynamiques non linéaires discrets construits à partir d'un grand nombre d'automates identiques d'états-finis dont chacun reçoit des entrées d'autres automates à l'intérieur d'un quartier prédéfini. Chacun utilise l'entrée modèle et ses informations d'état actuel pour mettre à jour son propre état ainsi que pour déterminer la sortie qu'il enverra à d'autres automates au sein de son voisinage [Bac 64][Krl 63] . Les automates sont généralement agencés selon un modèle cellulaire, d'où leur nom. Ce qui rend ces systèmes mathématiques non-linéaires distribués intéressants est le fait que même avec un automate relativement simple et des règles logiques simples, le système de l'AC est généralement capable de générer une grande variété de comportements complexes. Von Neumann et Ulam, entre autres, ont été en mesure de démontrer que certaines catégories d'AC sont capables d'autoreproduction [Neu 66]. Il a également été largement conjecturé que même les AC assez simple sont en mesure d'effectuer le calcul universel [Smi 68] [Cod 65].

Afin d'exploiter toute la puissance des ACs, nous devrons utiliser comme approche d'apprentissage pour agent.

Dans ce travail, l'accent sera mis sur l'application de cette notion d'automates cellulaires pour l'apprentissage d'un agent. Particulièrement, nous nous sommes intéressés à utiliser cette approche pour le domaine de la robotique, le robot commencera par effectuer des mouvements aléatoire, puis, en privilégiant les mouvements lui permettant d'avancer, mettra peu à peu en place une marche de plus en plus efficace. L'idée principale est de proposer un nouveau modèle de conception pour l'agent.

L'objectif de ce mémoire consiste donc à proposer une approche automate cellulaire pour l'apprentissage d'un agent.

Ce mémoire est constitué de cinq chapitres répartis comme suit :

Dans le premier chapitre, nous présentons un état de l'art sur l'apprentissage artificiel ainsi que ses différentes approches d'apprentissage en IA. Nous allons tout d'abord proposer quelques définitions de base de l'apprentissage artificiel et de l'apprentissage automatique. Ensuite, nous allons présenter les différents types d'apprentissage automatique et les paradigmes d'apprentissage. Enfin nous terminons ce chapitre par une synthèse sur les différentes approches d'apprentissage automatique en IA.

Dans le deuxième chapitre, nous présentons un état de l'art sur l'apprentissage chez les agents. Il contiendra premièrement une étude du concept d'agent et les différentes typologies d'agent, l'agent apprenant et l'apprentissage mono agent et l'apprentissage muli-agents. Finalement nous présentons quelques travaux d'apprentissage chez les agents existants.

Dans le troisième chapitre, nous présentons l'automate cellulaire et l'apprentissage : Etudes techniques et formelles. Il contiendra premièrement les définitions de base d'automate cellulaire et la différente classe d'automate cellulaire, finalement quelque travail d'automate cellulaire et son utilisation pour l'apprentissage.

Dans le quatrième chapitre nous allons décrire notre conception d'une architecture d'agent apprenant basé sur l'automate cellulaire déterministe pour formaliser le problème d'apprentissage et utiliser un nouveau mécanisme d'apprentissage par renforcement pour apprendre les évaluations de l'environnement.

Dans le cinquième chapitre, nous poursuivons avec une étude de cas afin de valider notre proposition. On a choisi comme cas à étudier le déplacement d'un agent dans un labyrinthe considéré comme un exemple typique pour l'agent qui cherche la sortie en temps plus court (ou bien cherche le chemin de sortie le plus court dans un délai plus court). Nous présentons ensuite la plate-forme NETLOGO et comment les différents composants de notre architecture, peuvent être implémentés en utilisant cette plate-forme.

Enfin, nous terminons ce mémoire par une conclusion générale, qui récapitule les travaux réalisés et fait le point sur un ensemble de perspectives envisagées.

Chapitre I

Etat De L'Art Sur Les Approches D'Apprentissage En IA

I.1. Introduction

A quoi sert l'apprentissage artificiel ? La plupart des programmes d'intelligence artificielle possèdent aujourd'hui un module d'apprentissage et tous les programmes de reconnaissance des formes sont fondés sur des algorithmes d'apprentissage. Et que font ces programmes? Ils sont capables de reconnaître la parole humaine et de l'interpréter. Ils réalisent une analyse automatique de photos satellites pour détecter certaines ressources sur la Terre. Ils assistent les experts pour prendre des décisions dans des environnements complexes et évolutifs, par exemple le marché financier ou le diagnostic médical. Ils fouillent d'immenses bases de données hétérogènes comme les millions de pages Web accessibles à tous. Ils analysent les données clientèles des entreprises pour les aider a mieux cibler leurs compagnes de publicité. Ils participent aussi à des tournois : le 11 mai 1997, le tenant du titre de champion du monde du jeu d'échecs, Gary Kasparov, a été battu en match par un programme.

On sait donc programmer les ordinateurs pour leur faire exécuter des tâches considérées comme intelligentes, de multiples façons et de manière de plus en plus efficace. Ce chapitre s'intéresse à un aspect particulier de cette intelligence artificielle: la faculté d'apprentissage.

L'objectif de ce chapitre est présenter quelques approches d'apprentissage automatique, qui sont connues dans le domaine de l'IA. Nous allons tout d'abord proposer quelques définitions de base de l'apprentissage artificiel et de l'apprentissage automatique. Ensuite, nous allons présenter les différents types d'apprentissage automatique et les paradigmes d'apprentissage. Enfin nous terminons ce chapitre par une synthèse sur les différentes approches d'apprentissage automatique en IA.

I.2. Généralités et définitions

Le terme Intelligence Artificielle (IA) a été introduit par McCarthy en 1956 [Mcc 56] pendant une conférence au Dartmouth College, et depuis, ce terme a été retenu pour représenter le domaine. L'IA est directement liée aux concepts de systèmes à bases de connaissances, systèmes experts, systèmes intelligents, acquisition de connaissances, apprentissage automatique, parmi d'autres sujets d'étude et d'application pratique.

I.2.1. Quelques applications de l'apprentissage artificiel

Voyons maintenant comment rendre un programme plus efficace en le dotant d'une possibilité d'apprentissage. Reprenons pour cela les applications de l'intelligence artificielle et de la reconnaissance des formes citées ci-dessus.

- Un programme de reconnaissance de la parole augmente ses performances au fur et à mesure de son utilisation par la même personne: c'est une expérience qu'il est aujourd'hui facile de faire en pratique si on achète un logiciel personnel de dictée vocale.
- Un programme de détection des ressources terrestres apprend à reconnaître une zone de pollution au milieu de la mer, a partir d'une base de données d'exemples d'images de zones connues comme propres ou comme polluées : cette base de données lui sert d'expérience pour déterminer sa décision sur une zone inconnue.
- Un programme de diagnostic sur un ensemble d'informations évolutives prises sur un patient doit avoir été pourvu de connaissances, à partir de diagnostics de praticiens et d'experts sur des situations types. Mais il doit aussi avoir été doté d'un module de généralisation, de façon à réagir correctement à des situations auxquelles il n'a jamais été confronté exactement.
- Les moteurs de recherche sur le Web pourraient être munis d'un module d'adaptation au style de navigation de l'usager: c'est une faculté souhaitable pour augmenter l'ergonomie de leur utilisation. Les programmes ne sont pas encore réellement enrichis de cette propriété, mais il est clair que c'est une condition nécessaire pour franchir certains obstacles de communication si évidents actuellement.
- L'exploitation des fichiers client d'une entreprise est souvent faite par un expert ou un programme expert qui utilise des règles explicites pour cibler un segment de clientèle susceptible d'être intéressé par un nouveau produit. Mais ces règles peuvent être acquises automatiquement, par un apprentissage dont le but est de fournir de nouvelles connaissances expertes, à la fois efficaces et intelligibles pour l'expert.
- Un programme de jeu d'échecs possède en général une très bonne efficacité à priori; mais il est naturel d'essayer de le doter d'un module où il peut analyser ses défaites et ses victoires, pour améliorer ses performances moyennes dans ses parties futures. Ce module d'apprentissage existe dans un certain nombre de programmes de jeux.

I.2.2. Quelques définitions de base

Nous allons rappeler ici les notions de base qui permettent de comprendre l'apprentissage artificiel [Lau 03].

- **Apprentissage (sous entendu: artificiel, automatique):** (Machine Learning) Cette notion englobe toute méthode permettant de construire un modèle de la réalité à partir de données, soit en améliorant un modèle partiel ou moins général, soit en créant complètement le modèle. Il existe deux tendances principales en apprentissage, celle issue de l'intelligence artificielle et qualifiée de symbolique, et celle issue des statistiques et qualifiée de numérique.

- **Précision vs Généralisation:** Le grand dilemme de l'apprentissage. La précision est définie par un écart entre une valeur mesurée ou prédite et une valeur réelle. Apprendre avec trop de précision conduit à un (sur-apprentissage), comme l'apprentissage par cœur, pour lequel des détails insignifiants (ou dus au bruit) sont appris. Apprendre avec trop peu de précision conduit à une (sur-généralisation) telle que le modèle s'applique même quand l'utilisateur ne le désire pas. Les deux types d'apprentissage, numérique et symbolique, ont défini des mesures de généralisation et c'est à l'utilisateur de fixer le seuil de généralisation qu'il juge optimal.

- **Connaissances Empiriques:** les connaissances empiriques, c'est-à-dire les connaissances expérimentales, sont représentées par l'ensemble des cas pratiques observés sur un sujet (ensemble d'exemples). Ce sont des connaissances "pures" qui n'ont pas été traitées, analysées ou modifiées. Ces connaissances représentent les résultats d'expériences ou les exemples de cas pratiques; elles n'ont pas encore subi de transformations en vue d'obtenir une théorie plus générale sur le domaine. On peut dire que ce sont des connaissances de bas niveau.

- **Connaissances Théoriques:** les connaissances théoriques modélisent les connaissances sur un sujet à l'aide d'une théorie correspondant au problème posé. Elles sont des connaissances "traitées" qui ont été obtenues à partir de l'analyse des connaissances de base. Ce type de connaissances représente une généralisation du savoir. Ce sont des connaissances dites de haut niveau. Elles sont habituellement

représentées par des structures symboliques, telles que les règles de production, les modèles mathématiques, les réseaux sémantiques, les objets structurés.

- **Fouille de données:** (Data Mining) ou Extraction de connaissances a partir des données (Knowledge Discovery in Data). La fouille de données prend en charge le processus complet d'extraction de connaissances: Stockage dans une base de données, sélection des données à étudier, si nécessaire: nettoyage des données, puis utilisation des apprentissages numériques et symboliques afin de proposer des modèles à l'utilisateur, enfin validation des modèles proposés. Si ces modèles sont invalides par l'utilisateur, le processus complet est répété.

- **Classification, classement et régression:** la classification, telle qu'elle est définie en analyse de données, consiste à regrouper des ensembles d'exemples non supervisés en classes. Ces classes sont souvent organisées en une structure (clustering). Si cette structure est un arbre, alors on parle de taxonomie ou de taxinomie (taxonomy). Sous l'influence du mot anglais classification, on a tendance à confondre classification et classement. Ce dernier mot désigne le processus de reconnaissance en intension (par leurs propriétés) de classes décrites en extension (par les valeurs de leurs descripteurs). Lorsque les valeurs à prédire sont des classes en petit nombre, on parle de classification.

I.2.3. L'Acquisition de Connaissances

L'acquisition de connaissances est un processus de rassemblement des connaissances nécessaires à la résolution d'un problème lié à une application spécifique. Ce processus inclut la codification (structuration) des connaissances dans un formalisme adapté aux outils informatiques utilisés. Il y a plusieurs façons de coder les connaissances afin de créer la base de connaissances, on peut diviser les processus d'acquisition de connaissances en deux grands groupes [Lau 03] : obtention par explicitation de connaissances (Knowledge Elicitation) et obtention par apprentissage automatique ou semi-automatique (Machine Learning).

L'explicitation de connaissances est réalisée par l'ingénieur de connaissances, qui va disposer de différents moyens pour la réalisation de cette tâche. Les connaissances sont obtenues à partir d'interviews avec des experts et d'informations disponibles dans des livres, manuels, rapports, etc. Cette approche est très intéressante, puisqu'on profite des

connaissances déjà acquises et bien élaborées. Malheureusement, on constate un certain nombre d'inconvénients :

- Le processus d'extraction de connaissances d'un expert est extrêmement ennuyeux, il prend beaucoup de temps et sa mise en place présente de sérieuses difficultés,
- Les experts humains utilisent souvent des activités mentales d'un niveau "sub-cognitif" pour résoudre les problèmes : ils ne savent pas verbaliser ce type de connaissances,
- Parfois les experts ne sont même pas conscients d'avoir utilisé certaines connaissances dans la résolution d'un problème et par conséquent ne les explicitent pas.

L'explicitation de connaissances par des ingénieurs de connaissances et des experts a un problème très connu des systèmes experts: il s'agit du goulot-d'étranglement du processus d'acquisition de connaissances (The Knowledge Acquisition Bootleneck) [Ala 01] [Nik 97][Hay 93]. Ce processus d'acquisition de connaissances, qui est indispensable au bon fonctionnement d'un système expert, est aussi le responsable des plus grands problèmes qui surviennent lors de la construction d'un tel système. Par conséquent, les chercheurs du domaine de l'IA. ont beaucoup investi dans l'étude et l'implémentation de systèmes d'acquisition automatique de connaissances.

Les recherches sur l'acquisition automatique de connaissances sont à l'origine des méthodes d'apprentissage automatique, l'un des domaines de l'IA. en plus grande expansion actuellement.

I.2.4. Apprentissage Automatique

L'apprentissage automatique est une partie très important de l'Intelligence Artificielle et doit être l'une des principales caractéristiques des systèmes intelligents.

La définition la plus générale de l'apprentissage peut se trouver dans un dictionnaire sous la forme suivante :

I.2.4.1. Définition de l'Apprentissage

Apprendre, c'est acquérir un savoir, une compréhension ou une aptitude par l'étude, l'instruction ou l'expérience.

Il est question d'apprentissage dans de nombreux domaines, chacun ayant une définition propre de cette notion. On peut par exemple citer les domaines suivants :

- philosophie : le problème posé par l'apprentissage est celui de la possibilité même du raisonnement par induction dont les enjeux épistémologiques sont importants.
- linguistique : l'apprentissage est l'acquisition de la langue maternelle (en particulier apprentissage de la syntaxe en priorité sur celui de la morphologie et la sémantique).
- psychologie: la psychologie distingue mal la mémoire de l'apprentissage et s'intéresse à l'apprentissage des catégories de base ou des concepts.
- biologie : la biologie s'intéresse surtout au substrat matériel de l'apprentissage et assimile l'apprentissage à une tache de contrôle (réglage d'une boucle sensori-motrice).

I.2.4.2. Définition d'apprentissage automatique

Une des meilleures définitions de l'apprentissage automatique est celle donné par H. Simon [Sim 83]: "*L'apprentissage dans un système est indiqué par les changements qu'il subit. Ces changements sont adaptatifs dans le sens où ils rendent possible au système de réaliser une même tâche, ou des tâches tirées d'une même population, d'une façon plus efficace et plus efficiente la prochaine fois qu'elle sera réalisée*".

I.2.4.3. Formulation d'apprentissage

Un algorithme d'apprentissage A est un algorithme qui a pour fonction d'apprendre à effectuer une tâche à partir d'un ensemble *S* de données (exemples). Chaque donnée Z_i est constituée d'un objet d'entrée x_i et une valeur de sortie y_i

$$Z_i = (x_i, y_i)$$

$$S = \{Z_i\}_{i=1}^{m} = \{Z_1, Z_2 \ldots\ldots\ldots Z_m\}$$

L'objectif de l'algorithme d'apprentissage est de construire pour tout couple (x, y) une bonne fonction ***h(x)*** appelé hypothèse qui représente la solution du problème en discussion, tel que ***h(x) = y.***

L'entrée de l'algorithme d'apprentissage A est un ensemble *S* et la sortie est une fonction appelée ***h***.

On écrit : $A(S) = h$

I.2.4.4. Caractéristiques d'apprentissage

Parmi les caractéristiques primordiales de l'apprentissage, nous citons les trois suivantes [Lau 03].

L'abstraction des données ; savoir si la méthode utilisée engendre des données proches de grandeur physique ou plutôt de symbole.

a. l'élément temporel ; il est important de déterminer si le système apprenant est en mesure de s'adapter « en ligne » ou il considère de façon différé toutes nouvelles modifications de l'environnement.

b. le rôle de l'enseignant ; il s'agit de savoir si le système apprend de façon autonome ou il a besoin d'être supervisé.

I.2.4.5. Conditions d'apprentissage

Pour toutes formes d'apprentissage, un ensemble de conditions sont nécessaires pour assurer le progrès adéquat du système apprenant [Yan 04].

Condition 1 : *« Toute forme d'apprentissage nécessite la répétition des décisions dans le temps. ».*La première condition consiste à permettre au système en phase d'apprentissage d'effectuer plusieurs essais, car tant qu'il y a plus de répétition, il y aura plus d'expériences.

Condition 2 : *« Toute forme d'apprentissage nécessite un mécanisme de rétroaction environnemental. ».*Afin qu'un système abouti à un apprentissage, il faut qu'il reçoit un feedback (rétroaction) de son environnement à la suite de ses propres décisions et les décisions des autres.

Condition 3:*« Toute forme d'apprentissage nécessite un mécanisme d'adaptation des décisions ».*Selon la troisième condition, non seulement l'acquisition de nouvelles informations par le système apprenant mène à un apprentissage, mais aussi les renseignements sur lesquels ces informations peuvent être utilisées sont aussi nécessaires.

Condition 4 :*« Toute forme d'apprentissage nécessite l'existence d'un mécanisme de stockage de l'information ; la mémoire ».*En effet, s'il n'y a pas une sauvegarde de conséquences obtenues dans le passé, le système ne bénéficie pas de ses décisions antérieures. Le stockage d'informations offre au système la possibilité de modifier sa décision sur la base de l'expérience passée et d'améliorer ses performances. En revanche, la mémorisation doit être guidée par des stratégies adéquates en tenant compte de l'espace mémoire comme un facteur critique.

I.2.4.6. Classification d'apprentissage automatique

Dans la littérature, on dénombre une grande variété de forme d'apprentissage. Il y a cependant différents critères importants pour classifier ces méthodes, dans ce qui suit un survol sur les plus récurrents.

I.2.4.6.1. Selon les connaissances manipulées

On peut distinguer deux types d'apprentissage artificiel, caractérisés par le type des données utilisées :

- **L'apprentissage symbolique :** l'apprentissage symbolique, comme son nom l'indique, manipule des symboles. Il fonctionne grâce à la mise en place de relations entre ces symboles par le biais de jugements. L'idée est donc d'élaborer des méthodes permettant d'extraire des connaissances structurelles ou décisionnelles à partir d'instances peu structurées. L'avantage principal de l'apprentissage symbolique est sa portée sémantique forte. Un expert qui analyse le système apprenant peut comprendre

la façon dont celui-ci fonctionne et les résultats fournis sont facilement interprétables [Lau 03] [Mic 90] [Gia 92].

- **L'apprentissage numérique :** l'apprentissage numérique ne manipule pas de symbole, il traite uniquement des valeurs numériques quantitatives qui vont être manipulées afin de réaliser l'apprentissage. Les méthodes d'apprentissage numérique se révèlent être portables et permettent une grande adaptabilité, car non dépendant de symboles. Par contre, le fonctionnement interne du système est opaque. Il est très difficile, voir impossible, de comprendre comment le système apprend [Lau 03] [Mic 90].

I.2.4.6.2. Selon le type d'information disponible

En apprentissage, le fait qu'intervienne (ou non) un enseignant afin d'aider l'apprentissage, en fonction des informations dont il dispose, on peut distinguer trois grandes familles :

- Apprentissage supervisé,
- Apprentissage non-supervisé,
- Apprentissage par renforcement.

- **Apprentissage supervisé :** Ici, un expert fournit la ou les solutions attendues, ou évalue l'erreur commise par le système. Durant l'apprentissage, le système utilise les informations données par l'expert pour améliorer ses performances. Le but est de généraliser l'association apprise à des situations nouvelles. Pour plus de détail sur ce genre d'apprentissage, le lecteur peut consulter les références : [Lau 03] [Mic 90] [Rus 95] .

- **Apprentissage non-supervisé :** Dans l'apprentissage non-supervisé, le système ne reçoit aucune information extérieure concernant les résultats attendus. On dispose seulement d'un nombre fini de données d'apprentissage, constituées "d'entrées", sans qu'aucun label n'y soit rattaché [Lau 03]**.**

- **Apprentissage par renforcement :** En apprentissage par renforcement, le système va essayer d'apprendre « seul » la façon d'améliorer ses résultats. La seule information dont il dispose est une critique sur son choix précédent (cette critique peut venir de

l'environnement, d'une critique extérieure ou être une auto-évaluation). En fonction des récompenses ou punitions, il doit modifier ou non son comportement afin d'améliorer ses performances. L'exemple typique est celui d'un robot autonome qui évolue et effectue des actions dans un environnement totalement inconnu initialement. [Lau 03] [Mic 90] [Rus 95].

I.2.4.6.3. Selon l'objectif attendu du processus d'apprentissage

Une autre classification proposée qui distingue entre trois types d'apprentissage classés selon l'objectif attendu du processus d'apprentissage. Dans chaque type, le résultat final (solution) supposé connu d'avance, la différence se résume dans la méthode appliquée pour atteindre la solution désirée ; par évaluation, optimisation ou bien un entraînement.

- **Apprentissage par évaluation:** consiste à déterminer pour un problème donné et un ensemble de méthodes de résolutions possibles, quelle est celle qui convient le mieux à la solution [Lau 03].

- **Apprentissage par optimisation:** l'objectif dressé par ce type est de déterminer la méthode de résolution d'un problème qui a donné lieu à la solution. Sachant que cette méthode doit être optimale (vérifier un critère d'optimalité) [Lau 03].

- **Apprentissage par entrainement :** consiste à adapter les connaissances actuelles d'un problème donné afin de maximiser la probabilité d'avoir la solution attendue [Lau 03].

I.2.4.6.4. Selon la stratégie utilisée

L'apprentissage peut être encore envisagé selon quatre stratégies adoptées pour qu'un système apprenne. Ces stratégies sont exposées ci-dessous [Lau 03]:

- **Apprentissage par cœur :** consiste en l'acquisition de connaissances et d'aptitudes sans recourir à une modification du système apprenant ou des inférences de sa part. Il existe une absence de généralisation, les connaissances mémorisées ne peuvent être réutilisées dans des nouvelles situations, le résultat stocké qui convient le mieux à la nouvelle situation est fourni sans aucune modification.

- **Apprentissage par instruction :** consiste en l'acquisition de connaissances et d'aptitudes qui nécessitent l'intégration des nouvelles représentations de son environnement. L'apprentissage est caractérisé par une forme locale de généralisation. Ces instructions sont mémorisées et utilisées si nécessaire dans le contexte qui leur est propre pour résoudre certaines tâches.

- **Apprentissage par analogie :** l'apprentissage par analogie désigne les méthodes dites paresseuses ; c'est-à-dire n'effectuant pas de généralisation des données disponibles. Il consiste en l'acquisition de connaissances pour les stocker comme référence de cas. Ce mécanisme permet au système de générer un résultat à l'aide des situations plus familières adaptées à la nouvelle ou bien adopter une attitude dans une situation inconnue.

- **Apprentissage par l'exemple :** consiste en l'acquisition de connaissances pour la caractérisation et la discrimination d'un concept. Ce genre d'apprentissage se caractérise par généralisation globale qu'il nécessite. Les exemples initiaux sont oubliés et une nouvelle connaissance générale comprime ces exemples est mémorisée.

I.3. Les approches d'apprentissage en IA

Dans ce qui suit, nous allons présenter quelques approches d'apprentissage automatique, qui sont connues dans le domaine de l'IA.

I.3.1. Les Algorithmes génétiques (AG)

Les AG forment une famille très intéressante d'algorithmes d'optimisation. Ils ont été développés en premier par John Holland [Hol 75] à l'université de Michigan. Le principe des AG [Gol 89] est de coder chaque solution potentille d'un problème par un «*chromosome* ». L'ensemble des chromosomes ou « *individus* » forment alors la « *population* » qui va évoluer au cours du temps.

Une «*génération* » est l'état de la population à un instant *t*. La population évolue au cours des générations en suivant des lois de sélection, de croisement et de mutation. En informatique, nous parlons d'opérateur génétique.

Dans ce qui suit, nous allons définir les opérations possibles à effectuer lors de l'utilisation des algorithmes génétiques [Gol 89][Rom 03][Rou 03].

A. **La création :** La première étape de l'algorithme génétique consiste à créer la population initiale. Cette première génération est choisie aléatoirement parmi les valeurs possibles de la population.
B. **L'évaluation :** La fonction d'évaluation est généralement appelée "*fitness*". Elle permet d'attribuer une note à chaque individu en fonction de son aptitude à résoudre le problème posé. Les éléments de la population peuvent ainsi être différenciés pour reproduire les meilleurs et éliminer les moins efficaces.
C. **La Sélection :** L'opérateur de sélection donne plus de chance aux « bons » individus de participer à la génération suivante en fonction d'un certain critère, la fitness.
D. **Le Croisement :** L'objectif de croisement est de combiner des chromosomes à partir de parents présentant des qualités, pour obtenir de meilleurs individus. Le croisement consiste à mélanger, au niveau du génome, des gènes des deux individus parents pour générer deux enfants. La taille de la population reste constante.
E. **La mutation :** Elle effectue une modification des gènes des chromosomes des enfants. En fonction du problème à résoudre, à chaque individu est associé un 'degré' d'adaptation à son environnement ; appelé fitness. Après croisements et mutation, une nouvelle génération est construite en conservant les individus de la population précédente ayant une propriété de fitness particulière, jusqu'à la convergence vers une solution optimale.

L'utilisation des algorithmes génétiques dans les systèmes d'apprentissage automatique s'est largement développée depuis les années quatre-vingt. Plusieurs applications des AG à la construction de classifieurs ont été déjà expérimentées avec des bons résultats. La première application a été réalisée par celui qui est à l'origine même des AG, J. Holland [Hol 86] [Hol 87]. Plusieurs autres expérimentations ont fait suite à ces premiers travaux [Gol 89] [Gre 93] [Dej 88][Dav 91][Dej 90][Dej 93b][Eic 96].

Pour pouvoir réaliser un apprentissage de règles avec des AG, il est nécessaire d'avoir une représentation interne de l'espace de recherche. Le codage utilisé par les AG est donc très important vis-à-vis de l'implémentation des outils d'apprentissage automatique. Le choix d'un type de codage spécifique doit essayer de garder les propriétés des AG, sans imposer des restrictions qui peuvent nuire ou restreindre la convergence de l'algorithme vers une bonne solution.

Deux approches assez connues pour générer automatiquement un classifieur à partir d'un AG sont : l'approche de Pitt et l'approche de Michigan [Dej 93a]. L'approche de Pitt, développé initialement à l'Université de Pittsburgh, postule qu'un élément (chromosome) correspond à une base de règles et qu'une population de chromosomes est un ensemble de bases de règles. L'autre approche, celle de Michigan, à été développé à l'Université de Michigan par l'équipe de J.Holland. Cette approche postule qu'un chromosome représente une règle et qu'une population correspond donc à une base de règles. L'algorithme, au cours de ses itérations cherche à construire et améliorer une seule base de connaissances.

I.3.2. Les Réseaux Neuronaux artificiels (RN)

I.3.2.1. Le neurone biologique

Le neurone : est une cellule composée de trois parties, vis-à-vis des transferts d'information, ont un rôle fonctionnel bien définit, à savoir :

- Le corps cellulaire,
- Les dendrites,
- L'axone.

Et une jonction entre deux neurones est appelée synapse. Un neurone est donc caractérisé par : Son état (x), le niveau d'activation qu'il reçoit en entrée (u), sa fonction de transition (f), sa fonction d'entrée (une somme en générale) [Rus 95] [Mcc 43].

La Figure I.1 représente un neurone biologique et ses composantes.

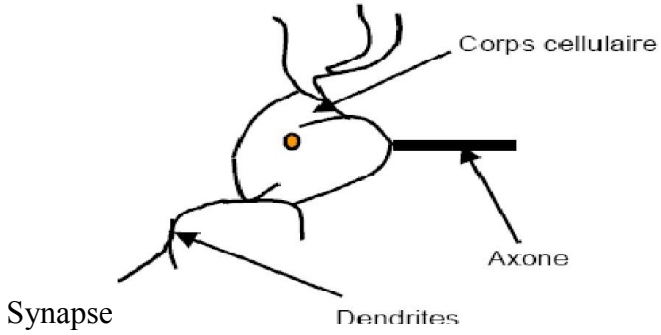

Figure I.1 : Neurone biologique.

I.3.2.2. Le neurone formel

Un neurone formel est un modèle très simplifié du neurone biologique, c'est un automate possédant plusieurs entrées et une seule sortie capable d'effectuer une opération mathématique [Tou 92].

Figure I.2 montre une correspondance entre le neurone naturel et celui formel.

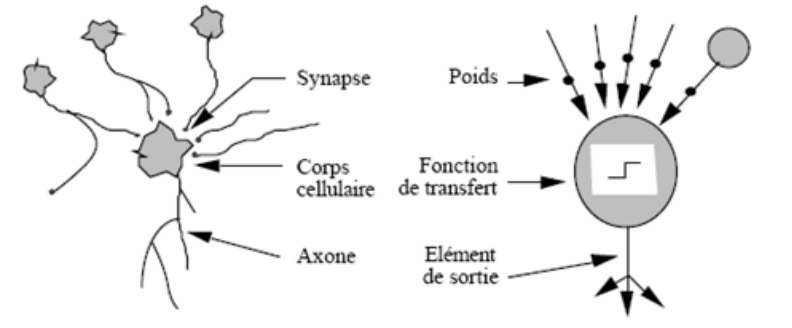

Figure I.2 : Mise en correspondance neurone biologique/neurone artificiel.

On appelle RN (réseaux de neurones) un ensemble de calculateurs numériques qui agissent comme des unités élémentaires, ils sont reliés entre eux par un ensemble d'interactions pondérées qui transmettent des informations numériques d'un neurone formel à l'autre.

L'apprentissage des réseaux de neurones est généralement un processus graduel, itératif et où les poids du réseau sont modifiés plusieurs fois selon une règle d'apprentissage avant d'atteindre leurs valeurs finales.

Cet apprentissage se fait suivant deux approches [Fer 98]: Soit on fixe l'architecture et on apprend les poids comme le fait la règle de Hebb, la règle de Windroff et l'algorithme de rétro propagation. Soit on apprend de manière constructive le nombre d'unités et les poids.

En effet, les RN ont été appliquées avec succès à l'apprentissage de tâches de classification et d'approximation de fonctions. Leur utilisation permet de passer des données au prédiquant, sans intermédiaire, sans codage et sans interprétation sujette à caution. Ils possèdent également une grande résistance au bruit ou au manque de fiabilité des données [Lau03].

Les informations à traiter sont codées sous la forme d'un vecteur appelé patron d'entrée, qui est communiqué aux neurones d'entrée du réseau. La réponse du réseau s'interprète à partir de la valeur d'activation de ses neurones de sortie, dont le vecteur s'appelle patron de sortie.

Généralement, l'apprentissage se fait sur une période relativement longue, et comprend quatre étapes de calcul :

1. Initialisation des poids synaptiques du réseau,
2. Présentation du patron d'entrée et propagation d'activation,
3. Calcul de l'erreur,
4. Calcul du vecteur de correction.

I.3.3. Le Raisonnement fondé sur des cas – CBR

Le raisonnement fondé sur des cas (CBR) [Lau 03] [Mal 95][Mal 96a][Mal 96b] [Kol 93].recouvre un ensemble de méthodes de résolution de problèmes qui exploite les expériences passées, plutôt que les connaissances générales d'un niveau supérieur, telles que les règles de production. Les CBR sont des méthodes d'apprentissage totalement fondées sur les connaissances empiriques, au lieu de faire usage des connaissances théoriques d'un domaine.

Un système CBR est capable d'utiliser la connaissance spécifique contenue dans son expérience passée pour résoudre les nouveaux problèmes. Cette expérience est représentée normalement sous la forme de cas. Ces cas, qui ont été corrigés et assignés par l'expert aux classes auxquelles ils appartiennent, constituent ainsi la mémoire d'un système CBR.

Quand un nouveau problème est présenté à un système CBR, il va se rappeler des cas passés stockés dans son mémoire, similaires au problème courant. Ensuite, le système adapte la meilleure solution mémorisée et la transfère au problème actuel. Le cas nouveau, qui a été traité par le système et reconnu, peut être à son tour mémorisé et donc ajouté comme une nouvelle expérience du système.

En général, un système de raisonnement fondé sur des cas contient les phases suivantes :

1. *Remémoration* des cas les plus similaires par rapport au cas posé en question;

2. *Réutilisation* de la connaissance du(ou des) cas remémore(s) pour la résolution du problème;
3. *Révision* de la solution donné afin de la valider;
4. *Mémorisation* de cette nouvelle expérience, pour une utilisation future.

I.3.4. Arbre de décision (AD)

Les arbres de décision [Bre 84] [Qui 86] [Qui 79] [Lew 94][Apt 94] sont composés d'une structure hiérarchique en forme d'arbre. Cette structure est construite grâce à des méthodes d'apprentissage par induction à partir d'exemples. L'arbre ainsi obtenu représente une fonction qui fait la classification d'exemples, en s'appuyant sur les connaissances induites à partir d'une base d'apprentissage. En raison de cela, ils sont aussi appelés arbres d'induction (Induction Decision Trees).

Une définition un peu plus formelle des arbres de décision est la suivante: un arbre de décision est un graphe orienté, sans cycles, dont les nœuds portent une question, les arcs des réponses, et les feuilles des conclusions, ou des classes terminales.

Traditionnellement, un arbre de décision se construit à partir d'un ensemble d'apprentissage par raffinements de sa structure. Un ensemble de questions sur les attributs est construit afin de partitionner l'ensemble d'apprentissage en sous-ensembles qui deviennent de plus en plus petits jusqu'à ne contenir à la fin que des observations relatives à une seule classe. Les résultats des tests forment les branches de l'arbre et chaque sous-ensemble en forme les feuilles.

Les AD ont prouvé leur qualification dans le domaine d'apprentissage automatique. Leur succès est notamment dû à leur aptitude à traiter des problèmes complexes de classification. En effet, ils offrent une représentation facile à comprendre et à interpréter ainsi qu'ils possèdent une capacité à produire des règles logiques de classification. Leur rapidité en classement dépend tout simplement du nombre majeur des nœuds en partant de la racine aux feuilles.

I.3.5. Les chaînes de Markov et les modèles de Markov cachés (MMC)

Les chaînes de Markov [Lau 03] sont un cas particulier de modèles graphiques. Elles représentent en effet les dépendances temporelles dans une séquence de variables S_0, …,

S_t. Lorsque chaque variable ne dépend que de la variable précédente dans la séquence, on dit que l'on a une chaîne de Markov :

$$P(S_t|S_0, \ldots, S_{t-1}) = P(S_t|S_{t-1})$$

Le formalisme des chaînes de Markov est particulièrement adapté pour la représentation de séquences, qu'elles soient de nature temporelle, comme par exemple des cours boursiers ou un signal acoustique, de nature spatiale, comme par exemple une chaîne d'ADN, ou d'autres types de dépendances linéaires.

Une généralisation des chaînes de Markov s'appelle les modèles de Markov cachés (*Hidden Markov Models*). Formellement, un modèle de Markov caché (d'ordre un) est un modèle génératif de séquences défini par un ensemble d'états, un alphabet discret de symboles, une matrice de probabilités de transitions entre états et une matrice de probabilité d'émissions de chaque symbole de l'alphabet à partir de chaque état. Le système évolue aléatoirement d'un état à l'autre suivant les probabilités de transition en émettant des symboles de l'alphabet.

Seuls les symboles émis sont observables, et non les transitions entre états, qui sont internes au modèle. La séquence d'états est donc une séquence de variables cachées ou latentes expliquant les observations.

I.3.6. La méthode des k-plus proches voisions(Kppv)

C'est une approche dite « memory-based », elle suit un raisonnement par analogie, utilisée principalement pour les problèmes de classification.

Cette méthode diffère des autres méthodes d'apprentissage car aucun modèle n'est induit à partir d'exemple. Les données restent telles quelles; elles sont stockées en mémoire. Quand on parle de voisin, cela implique la notion de distance ou de dissemblance. La décision consiste à chercher les k échantillons les plus proches de l'observation et d'affecter la classe la plus fréquentée dans ces K instances. Différents outils statistiques sont utilisés, la plus populaire est la distance eucleudienne .

Les expériences menées avec les Kppv montrent qu'ils résistent bien aux données bruitées, ainsi qu'ils présentent l'avantage de la simplicité**.** Par contre ils requièrent de nombreux exemples, ce qui rend le processus de décision complexe. Bien que le temps d'apprentissage soit inexistant, la classification d'un nouveau cas est coûteuse puisqu'il faut comparer ce cas à tous les exemples déjà classé [Lau 03] [Yan 94].

I.3.7. Les réseaux bayésiens (RB)

Les RB (réseaux bayésiens) sont connus comme un formalisme apte à assurer la prise en compte de l'incertain en modélisation des problèmes [Lau 03] [Jea 06] [Ste 03].

La structure de ce type de réseau est simple: un graphe dont lequel les nœuds représentent des variables aléatoires correspondant aux données et les arcs reliant ces derniers sont attachés à des probabilités conditionnelles. Ils exploitent le théorème de bayes afin d'apporter des solutions.

L'apprentissage d'un RB peut être réalisé selon deux manières. La première fait un apprentissage des paramètres où on suppose que la structure du réseau à été fixée et il faudra estimer les probabilités conditionnelles de chaque nœud du réseau. Tandis que la deuxième sert à apprendre la structure; le but est de trouver le meilleur graphe représentant les tâches à résoudre [Lew 92a][Lew 94][Joa 97][Sah 98].

I.4. Synthèse

Dans ce chapitre nous avons présenté quelques approches d'apprentissage existant dans l'IA. L'objectif de cette synthèse est de présenter une discussion sur les différents avantages et inconvénients de chaque approche.

- Les algorithmes génétiques se présentent comme une approche très intéressante. Ils sont capables d'exploiter l'espace de recherche sans avoir à imposer de restrictions (*biais*) liées au type d'algorithme choisi, ce qui n'est pas toujours le cas des autres algorithmes d'apprentissage automatique. Par contre, les choix faits au niveau du codage employé, du type des règles manipulées, ou du type de méthodes de sélection et de transformation utilisées, vont beaucoup influencer l'apprentissage. Une des tâches les plus difficiles va être la définition de ces choix d'implémentation des AG [Dej 93a]. Ce type de problème est d'autant plus délicat qu'il n'existe pas de méthodologie générale à suivre.

La difficulté du choix d'un codage et d'une bonne fonction de sélection et d'adaptation font des AG une technique très particulière qui doit être bien maîtrisée pour donner des bons résultats.

Dans le cas contraire, ces algorithmes n'arriveront pas à trouver de bonnes solutions aux problèmes posés. Le manque d'une garantie de convergence vers une bonne solution

représente l'un des principaux désavantages des AG. De plus, les AG sont des algorithmes très lourds à exécuter, d'où l'intérêt d'exploiter le parallélisme dans ce type d'approche.

Comme d'autres algorithmes d'optimisation, ils connaissent les problèmes liés à la convergence et au blocage dans des minima-locaux. Par contre, si les fonctions de mutation, de reproduction et de croisement sont bien définies, les AG peuvent constituer une solution assez intéressante pour s'échapper des minima-locaux. Un autre avantage des AG est le fait qu'ils admettent l'introduction de connaissances théoriques préexistantes dans la population initiale à traiter. Donc, cette population initiale nous permet d'avoir un point de départ pour l'apprentissage qui peut être plus proche de la solution finale recherchée [Eic 96].

Pour conclure cette partie, il nous reste à souligner que la majorité des AG sont des approches qui manipulent des informations purement symboliques, d'où la difficulté à travailler avec des variables continues et des données approximatives. C'est le type de représentation des règles symboliques qui va définir le pouvoir d'un tel algorithme d'apprentissage.

- La deuxième approche s'est appuyée sur Les réseaux neuronaux. Les réseaux de neurones, fabriqués de structures cellulaires artificielles, constituent une approche permettant d'aborder sous des angles nouveaux les problèmes de perception, de mémoire, d'apprentissage et de raisonnement. Ils s'avèrent aussi des alternatives très prometteuses pour contourner certaines des limitations des ordinateurs classiques. Grâce à leur traitement parallèle de l'information et à leurs mécanismes inspirés des cellules nerveuses (neurones),

Deux avantages majeurs à l'utilisation des réseaux neuronaux [Adl 03,Cla 92]: (i) Capacité de représenter n'importe quelle dépendance fonctionnelle. Le réseau découvre la dépendance lui-même sans avoir besoin qu'on lui "souffle" quoi que ce soit. Comportement moins mauvais en cas de faible quantité de données. Et on passe directement des données au prédicateur, sans intermédiaire, sans recodage, sans discrétisation, sans simplification ou interprétation sujette à caution; (ii) Pour l'utilisateur novice, l'idée d'apprentissage est plus simple à comprendre que les complexités des statistiques multi-variables. Cependant les réseaux de neurones artificiels ont besoin de cas réels servant d'exemples pour leur apprentissage (on appelle cela la *base d'apprentissage*). Ces cas doivent être d'autant plus nombreux que le problème est complexe et que sa topologie est peu structurée. Par exemple, on peut optimiser un système neuronal de

lecture de caractères en utilisant le découpage manuel d'un grand nombre de mots écrits à la main par de nombreuses personnes. Chaque caractère peut alors être présenté sous la forme d'une image brute, disposant d'une topologie spatiale à deux dimensions, ou d'une suite de segments presque tous liés. La topologie retenue, la complexité du phénomène modélisé, et le nombre d'exemples doivent être en rapport. Sur un plan pratique, cela n'est pas toujours facile car les exemples peuvent être soit en quantité absolument limitée ou trop onéreux à collecter en nombre suffisant [Mar 06] ; Ainsi Le problème de la "boîte noire" : la validation du modèle neuronal, La validation se fait par test sur un certain nombre de dossiers passés, dont on connaît l'issue, que l'on a écartés de l'échantillon d'apprentissage et réservés à cet effet. Ainsi que l'ajustement du paramètres (poids, unités) devenu de plus en plus une tâche ardue autant que les problèmes sont complexes, et par conséquent le réseau atteint un état de sur apprentissage, et perd ses pouvoirs à résoudre les problèmes en discussion.

- La troisième approche s'est appuyée sur le Raisonnement Fondé sur des Cas. Les systèmes de raisonnement fondé sur des cas font appel à des algorithmes d'apprentissage dits paresseux [Aha 97]. Ils diffèrent des autres algorithmes par le fait qu'ils retardent le traitement des informations, c'est-à-dire qu'ils ont un faible coût calculatoire pendant la phase d'apprentissage et des calculs plus intenses pendant la phase de test. Les systèmes CBR n'essaient pas de traiter les exemples fournis, alors que les autres algorithmes d'apprentissage font une compilation des exemples d'apprentissage et les remplacent par des abstractions concises (e.g.: arbres de décision, ensembles de règles). Les CBR sont des systèmes purement fondés sur les connaissances empiriques et ne permettent pas l'utilisation de connaissances théoriques [Alt 97]. Donc, on ne peut pas profiter des connaissances disponibles sur le domaine d'application sauf si celles-ci sont représentées par des cas pratiques [Ors 95].

  Enfin, ce type de système n'est pas très performant en ce qui concerne l'explication de la solution trouvée puisqu'au mieux il va donner la liste des cas les plus semblables au cas présenté sans faire une analyse plus profonde de la solution fournie.

- La quatrième approche s'est appuyée sur les arbres de décisions. Ils sont largement utilisés, car ce type d'approche est assez stable, facile à comprendre et donne des bons résultats dans un grand nombre d'applications car il y a une classification très efficace, et aussi la possibilité d'ajouter de la connaissance du domaine et représentable en logique

d'ordre un. Cependant, dans certains types d'applications, les arbres de décision donnent des résultats décevants [Qui 86], si la taille de l'ensemble d'apprentissage augmente, ou si il y'a du bruit dans les données.

- La cinquième approche est appuyée sur Les chaînes de Markov et les modèles de Markov cachés. Les HMM sont des modèles probabilistes d'émission de séquences, discrètes ou continues. Pendant l'étape d'apprentissage, il arrive que les MMC se trouvent au point de risque à converger vers un minimum local au lieu d'un minimum global, ce qui affaiblit le résultat obtenu. Une autre limite à citer concerne le passage à l'échelle, l'impossibilité de la prise en compte efficace de l'influence de phénomènes indépendants et la difficulté de généralisation [Lau 03].

- La sixième approche est appuyée sur les k-plus proches voisions. C'est une approche dite « memory-based », elle suit un raisonnement par analogie, utilisée principalement pour les problèmes de classification [Blin 02]. Les expériences menées avec les Kppv montrent qu'ils résistent bien aux données bruitées, ainsi qu'ils présentent l'avantage de la simplicité**.** Par contre, ils requièrent de nombreux exemples, ce qui rend le processus de décision complexe. Bien que le temps d'apprentissage soit inexistant, la classification d'un nouveau cas est coûteuse puisqu'il faut comparer ce cas à tous les exemples déjà classés.

- Finalement, la structure l'approche des réseaux bayésiens (RB) est simple, un graphe dont lequel les nœuds représentent des variables aléatoires correspondant aux données et les arcs reliant ces derniers sont attachés à des probabilités conditionnelles. Les RB offrent un mécanisme efficace de résolution du problème [Lau 03]. Ainsi que leur pouvoir de gérer des données incomplètes, comme ils permettent d'apprendre la relation causale qui peut aider la prise de décision. De plus, ils donnent la possibilité de rassembler des connaissances de diverses natures dans un même modèle. En revanche, les RB ont pas mal d'inconvénients surtout la complexité des algorithmes. La plus parts des algorithmes développés pour l'inférence et l'apprentissage utilisent des variables discrètes et la manipulation des variables continues est en cours de recherche. Un autre défi est que la généralité du formalisme des RB aussi bien en terme de représentation que d'utilisation les nœuds les rend difficile à manipuler à partir d'un certain taille**.**

I.5. Conclusion

Ce chapitre a présenté un état de l'art sur l'apprentissage en IA, pour cela nous avons présenté quelques définitions de base associées à l'apprentissage, ainsi que les caractéristiques d'apprentissage et aussi les différentes classifications d'apprentissage automatique, pour les approches d'apprentissage en IA, il existe plusieurs types, nous avons présenté quelques approches d'apprentissages, qui sont les plus connues dans le domaine de l'IA. Nous avons terminé cet état de l'art par une synthèse récapitulative en montrant les avantages et les inconvénients de ces approches.

L'objectif de ce chapitre était d'introduire la notion de l'apprentissage automatique, ainsi que ses différentes approches. L'objectif de ce travail est plutôt d'étudier l'apprentissage chez les agents artificiels. Pour cette raison, l'objectif du prochain chapitre serait de présenter l'apprentissage chez les agents.

Chapitre II

ETAT DE L'ART :

L'apprentissage Chez Les Agents

II.1. Introduction

Un agent qui agit dans un environnement incertain pour y accomplir un but est généralement vu comme un problème de décisions séquentielles. Le but visé par un tel agent est souvent spécifié par le problème alors que la stratégie pour arriver à ce but est très souvent inconnue. Dans ces conditions, lorsque le modèle de l'environnement est inconnu, il est nécessaire d'utiliser des algorithmes d'apprentissage qui vont permettre à l'agent d'accomplir son but. Selon [Lau 03], l'apprentissage «*Consiste à construire un modèle de la réalité à partir de données, soit en améliorent un modèle partiel, soit en créant complètement le modèle.*» C'est pourquoi, depuis le début de l'informatique, de nombreuses méthodes d'apprentissage ont été développées pour construire automatiquement le modèle de l'environnement physique réel. Cependant, un tel apprentissage ne peut se faire sans hypothèse sur le domaine de la solution.

Les méthodes d'apprentissage automatiques dépendent des connaissances qu'a le concepteur sur le domaine. Ainsi, s'il connait des informations relatives au but, il peut utiliser l'apprentissage par renforcement et faire apprendre à l'agent, son comportement optimal en interaction avec l'environnement. Si, en revanche, le concepteur connait une partie de la solution, il peut utiliser l'apprentissage supervisé pour la généraliser à l'ensemble du problème. Si, de plus, le problème comporte plusieurs agents, les méthodes à utiliser vont dépendre de l'interaction qui existe entre les agents. Ainsi, dans les cas où les agents ont les mêmes buts, le concepteur peut utiliser des extensions d'algorithmes mono-agents. Si, par contre, les agents ont des buts différents (sans être obligatoirement opposés), les méthodes à base de théorie des jeux sont le plus souvent utilisées.

L'objectif de ce chapitre est d'initier le lecteur novice au domaine des agents qui constituent un des piliers de notre travail. Nous nous intéressons tout d'abord à des considérations générales sur les notions d'agent et de systèmes multi-agents. Par la suite, nous procédons par une étude sur l'apprentissage d'agents. Enfin nous terminons ce chapitre par une discussion générale sur le chapitre I et ce chapitre.

II.2. Étude du concept d'agent

Avant de d'exhiber les caractéristiques de l'apprentissage chez l'agent, il est essentiel de se pencher d'abord sur la notion d'agent. Il existe, à l'heure actuelle, une pléthore de définitions de l'agent. Mais diffèrent selon le type d'application pour laquelle est conçu l'agent [Jar 02].

Nous proposons une première définition suivante « Un agent est tout ce qui peut être vu comme percevant son environnement au travers de capteurs et agissant sur cet environnement au travers d'effecteurs. » [Rus 95].

Une deuxième définition proposée par [Per 97] « *un agent est une entité logicielle ou matérielle, à laquelle est attribuée une certaine mission qu'elle est capable d'accomplir de manière autonome, disposant d'une connaissance partielle de ce qui l'entoure (son environnement), et agissant par délégation pour le compte d'une personne ou d'une organisation*» .

Cependant, Jacques Ferber [Fer 95] propose une définition plus précise et plus contraignante des agents puisqu'il fixe neuf caractéristiques pour ces derniers :

« Un agent est un système :

- *Qui est capable d'agir dans un environnement ;*
- *Qui peut communiquer directement avec d'autres agents ;*
- *Qui est mue par un ensemble de tendances (sous forme d'objectifs individuels ou d'une fonction de satisfaction, voire de survie, qu'elle cherche à optimiser) ;*
- *Qui possède des ressources propres ;*
- *Qui est capable de percevoir (mais de manière limitée) son environnement ;*
- *Qui ne dispose que d'une représentation partielle de cet environnement (éventuellement aucune) ;*
- *Qui possède des compétences et offre des services ;*
- *Qui peut éventuellement se reproduire ;*
- *Dont le comportement tend à satisfaire ses objectifs, en tenant compte des ressources et des compétences dont elle dispose, et en fonction de sa perception, de ses représentations et des communications qu'elle reçoit. »*

II.2.1. Caractéristiques des agents

Les définitions ci- dessous émergent les caractéristiques suivantes :

- ❖ L'agent est menu d'une tâche à effectuer (résolution des problèmes) et possède des techniques pour atteindre son objectif.
- ❖ L'agent est une entité autonome ; son comportement est déterminé par sa propre expérience exercé dans son propre environnement.
- ❖ L'agent dispose d'une connaissance, même partielle, de son environnement courant. Ceci lui permet de prendre les décisions appropriées.
- ❖ L'agent est caractérisé par un comportement flexible; il répond d'une manière opportune aux changements qu'il perçoit dans son environnement, il prend l'initiative d'adopté la décision appropriés pour atteindre son but, comme il est capable d'interagir avec les autres agents.

Étant qu'un agent possédant ces caractéristiques, il est habilitant d'apprendre et d'évoluer en fonction de cet apprentissage en modifiant son comportement selon ses expériences passé. La figure II.1 représente une architecture d'un agent selon Russell et Norving .

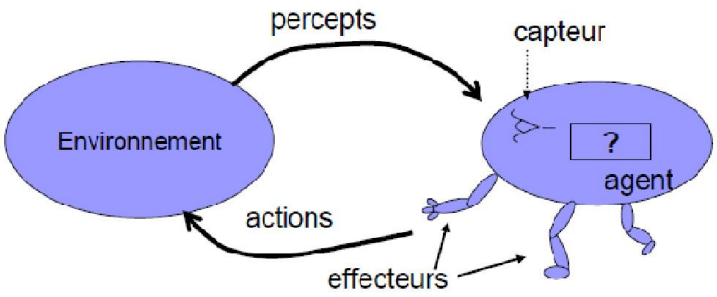

Figure II.1: Architecture d'un agent *(Russell and Norvig).*

II.2.2. Apprentissage ou rationalité ?

Russell et Norvig ont étendu la définition d'un agent en introduisant le concept de rationalité [Russ 95].

Un Agent rationnel idéal choisi toujours l'action qui l'amène au but final en maximisant la valeur de la mesure de performance. Il agit sur la base de ses connaissances

du monde, pour chaque séquence de perceptions La perception active est un aspect important d'un comportement rationnel.

Alors quelle est la différence entre la rationalité et l'apprentissage d'un agent? Pour qu'on puisse répondre à cette question, il faut citer les hypothèses en quoi s'appuie la rationalité [New 82].

- **L'information complète:** Tous les agents sont supposés être entièrement informés du problème.
- **L'anticipation rationnelle :** l'agent sait au préalable les résultats attendus d'une action et il dispose des critères d'évaluation de leurs actions lui permette de prendre la bonne décision.
- **La rationalité parfaite:** tous les agents sont supposés être parfaitement capables de déduire leur comportement optimal quelque soit la complexité du problème.
- **Les espérances communes:** tous les agents agissent selon les mêmes principes de l'information complète et de la rationalité parfaite. Ils savent que les autres savent qu'ils savent.

Le problème posé et que l'on peut aisément remarquer est le rencontre des difficultés dans les cas complexes, justement parce que ces trois suppositions ne sont pas toujours satisfaites.

- Les agents doivent s'informer du contexte quand le jeu a commencé. Et les problèmes de contexte peuvent eux mêmes ne pas être complètement définis initialement.
- Les agents ne sont pas toujours assez intelligents ou n'ont pas de capacité d'action pour calculer un optimum.
- Les agents utilisent différentes méthodes d'approches et ne peuvent compter sur les autres pour dupliquer leur raisonnement.
- Les conséquences des actions choisis ne sont pas toujours logiques.
- Ces difficultés mènent à des prévisions qui ne sont pas toujours escomptées.

C'est alors qu'une théorie évolutionnaire a apparu. Son principe est de considérer le problème exactement à l'opposé de la rationalité, avec des agents qui ont initialement une très petite rationalité et connaissances spécialisées de leur domaine d'action. Nos agents sont alors amenés à s'adapter et à apprendre, devenant petit à petit experts dans leur propre

domaine. L'un des précurseurs de cette approche est J. Holland [Hol 86] dans ses travaux sur l'inférence et les algorithmes génétiques, en introduisant la notion d'agents artificiels adaptés.

II.2.3. Apprentissage ou adaptation ?

La définition d'un agent s'enrichie à chaque fois qu'on lui attribue une nouvelle caractéristique, (figure II.2) L'adaptation aussi semble un point fort pour la persistance de l'agent dans son entourage, car l'agent est situé dans un environnement qui change souvent, par conséquent il doit être en mesure à répondre aux dynamiques produits. Cela nécessite une capacité d'adaptation pour qu'il puisse survivre et atteindre ses objectifs.

Le thème de l'apprentissage et de l'adaptation pour les agents est fréquemment évoqué. L'importance de l'apprentissage devrait s'accroître dans les prochaines années tant l'adaptation est primordiale chez les agents [Fer 97].

Quelle relation existe-t-il entre l'apprentissage et l'adaptation ? les propos de G.Weiss,(99) [Wei 99] ne fait aucune distinction explicite entre les deux concepts, l'adaptation est couverte par l'apprentissage. Il s'agit ici de s'interroger sur la question suivante : comment faire évoluer le comportement des agents de manière à ce qu'ils puissent tirer parti des expériences passées?

Cependant, on peut tirer les points suivants :

- L'adaptation est l'acte résultat face aux changements des situations.
- L'adaptation n'octroie pas des nouvelles connaissances ou comportements. Elle ne sert qu'à modifier les anciens pour les réajuster.

En récapitulation, on peut retenir la définition qui vient d'un agent apprenant, elle correspond le plus mieux au cadre de notre étude.

La figure II.2 représente une nouvelle caractéristique d'un agent.

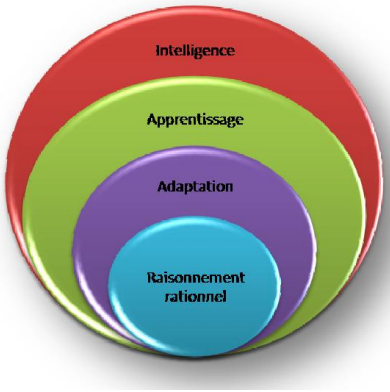

Figure II.2 : Caractéristiques d'un agent.

II.2.4. Agent apprenant

On dit qu'un agent est entrain d'apprendre si pendant l'exécution de ses tâches il cherche à améliorer ces performances d'exécution et à acquérir des nouvelles connaissances en fonction de ce qu'il perçoit et la connaissance dont il dispose, par modification de techniques et la prise en compte des changements survenus dans son environnement.

Auprès de ce qui était énoncé auparavant, l'apprentissage est caractérisé par :

- L'apprentissage est une action autonome.
- Apprendre pour s'évoluer en acquérant des nouvelles connaissances (Représentation et comportement).
- Apprendre pour raisonner avec rationalité.
- Réciproquement, Apprendre pour assurer son adaptation, et s'adapter par Apprentissage.
- Apprendre pour procurer de l'intelligence.

↳ Si un agent est capable de raisonner face aux situations rencontrées dans son environnement (choisir la meilleure action à exécuter), d'avoir une faculté à s'adapter aux changements produits, et disposé à apprendre des nouvelles procédés pour s'améliorer, alors il est sur le point d'être un agent évolue qualifier d'intelligence figure II.3.

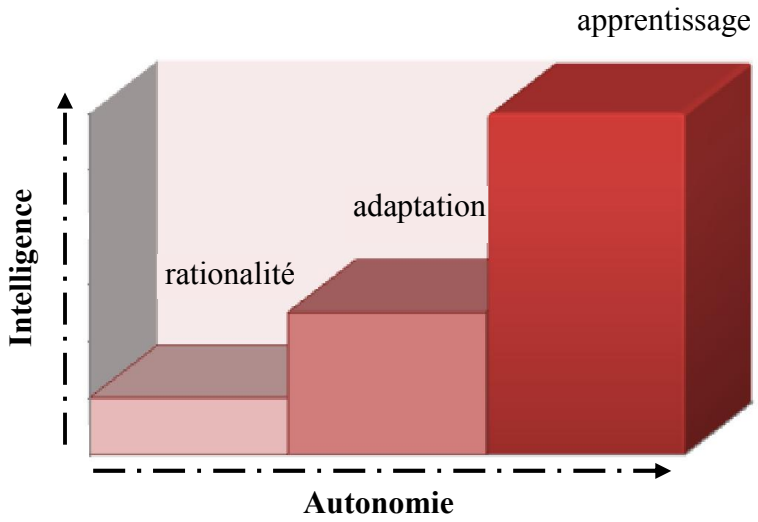

Figure II.3 : Degré d'autonomie et d'intelligence par rapport les caractéristiques d'un agent.

Pour que l'agent puisse faire évoluer, il a besoin de deux types d'information. Tout d'abord, il doit avoir des informations sur la manière dont le monde évolue, indépendamment de l'agent, comme il doit avoir ensuite des informations sur la manière dont ses propres actions affectent le monde autour de lui.

II.2.5. Typologie des agents

Il existe un grand nombre de typologies d'agents [Fer 95, Hu 98, Woo 95], classées en trois types d'agents : réactifs, cognitifs ou délibératifs et hybrides.

- **Les agents réactifs :** Les agents réactifs qui ne représentent pas explicitement son environnement (ainsi des autres agents) et pas de mémoire de leur histoire. Leurs comportement se basent sur le principe : stimulus = action (par exemple dans la robotique : si mur = demi-tour).Ce type d'agent nécessite l'utilisation d'un grand nombre d'agents pour effectuer les opérations. L'interaction de plusieurs agents permet l'émergence d'une intelligence. Nous pouvons citer les sociétés d'insectes comme les fourmis, les termites qui représentent des meilleurs exemples d'agents réactifs.

- **Les agents Cognitifs ou délibératifs :** Les agents cognitifs représentent explicitement l'environnement (et les autres agents) qui peuvent tenir compte de leur passé apprentissage. Ces agents peuvent représenter bien l'homme et sont dits intentionnels

car ils ont des buts. Ils sont intelligents, donc intéressants individuellement et collectivement. Ils peuvent communiquer par tableau noir et /ou par échange de messages et ils n'ont qu'une vue partielle de l'environnement.

- **Les agents hybrides** : intègrent l'aspect cognitif et réactif.

II.3. Les systèmes multi agent

Les chercheurs en intelligence artificielle sont orientés vers une vision distribuée des systèmes pour plusieurs motivations. Plutôt que de considérer un agent unique, compliqué, difficile à maintenir et apparaissant finalement comme une ressource critique, ils ont appliqué le principe "diviser pour régner". Les systèmes multi-agents s'appuient sur le principe suivant: au lieu d'avoir un seul agent en charge de l'intégralité d'un problème, on considère plusieurs agents qui n'ont chacun en charge qu'une partie de ce problème. La solution au problème initial est alors obtenue au travers de l'ensemble des comportements individuels et des interactions, c'est à dire par une résolution collective [Den 91].

Définition

Dans [Fer 95], un système multi-agent est défini de la façon qui suit: On appelle système multi-agent (ou SMA), un système composé des éléments:

- Un environnement E, c'est-à-dire un espace disposant généralement d'une métrique.
- Un ensemble d'objets O. Ces objets sont situés, c'est-à-dire que, pour tout objet, il est possible, à un moment donné, d'associer une position dans E. Ces objets sont passifs, c'est-à-dire qu'ils peuvent être perçus, créés, détruits et modifiés par les agents.
- Un ensemble A d'agents, qui sont des objets particuliers ($A \subset O$), lesquels représentent les entités actives du système.
- Un ensemble de relations R qui unissent des objets (et donc des agents) entre eux.
- Un ensemble d'opérations Op permettant aux agents de A de percevoir, produire, consommer, transformer et manipuler des objets de O.
- Des opérateurs chargés de représenter l'application de ces opérations et la réaction du monde à cette tentative de modification, que l'on appellera les lois de l'univers.

II.4. Apprentissage chez les agents

Pour la plupart des tâches que doit accomplir les agents et même si l'environnement paraît plus ou moins stable, il est extrêmement difficile et même impossible à priori de déterminer correctement le comportement et les activités concrètes de ce système au moment de sa conception. Cela nécessite de connaître à priori la nature de l'environnement émergent et le genre d'agents qui pourront s'adapter à ce nouvel environnement. Ce genre de problèmes qui résultent de la complexité des systèmes multi-agents peuvent être évités ou du moins réduits en octroyant aux agents la possibilité de s'adapter et d'apprendre avec la possibilité d'améliorer leur propre performance ou/et celle du système dans son ensemble [Wei 96].

II.4.1. Apprentissage mono agent

L'idée derrière l'apprentissage, c'est que les perceptions de l'agent ne devraient pas être utilisées uniquement pour choisir des actions, mais également pour améliorer l'habilité de l'agent à agir dans le futur. Une telle amélioration est une forme d'apprentissage et est très importante car c'est ce qui lui permet d'évoluer, de s'adapter et de s'améliorer [Rus 95].

Par exemple, on peut penser à la conduite automobile. Au début, la conduite est très difficile, car on doit penser à toutes les actions que l'on fait, mais plus on se pratique, moins on réfléchit et plus la conduite devient un réflexe.

Pour les agents, on peut penser à appliquer la même chose. C'est à dire, lorsqu'un agent fait face à une situation pour la première fois, il doit délibérer plus longtemps pour choisir ses actions. Mais, avec un module d'apprentissage, plus l'agent effectue des tâches similaires, plus il devient rapide. Son comportement passe graduellement d'un état délibératif, à un état réactif. L'agent a donc appris à exécuter une tâche. D'un point de vue plus technique, on peut dire que l'agent *a*, en quelque sorte, compilé son raisonnement dans une certaine forme d'ensemble de règles qui lui permettent de diminuer son temps de réflexion. Ce type d'apprentissage peut être très utile pour des agents hybrides.

Ce n'est qu'une façon dont les agents peuvent apprendre, il en existe plusieurs autres. En fait, on considère que toute technique qui permet à un agent d'améliorer son efficacité est une technique d'apprentissage.

L'apprentissage individuel consiste qu'un agent peut apprendre de façon solitaire et indépendante des autres agents [Wei 93]. Il n'est pris en considération que lorsque tous les éléments du processus d'apprentissage sont exécutés par le même agent et n'incluse que ses propres expériences [Wei 96]. À ce stade plusieurs travaux sont considérés intéressants, ils implémentent différents techniques d'apprentissage dans divers domaines d'application.

Entre autre nous citons Baird & Moore 1998 [Bai 98] , Kearns & Singh 1998[Kea 98], Satinder Singh 2000 [Sat 00] .

II.4.2. Apprentissage multi agents

Jusqu'ici on a fait état des systèmes où évoluait un seul agent. Dans la plupart des situations réelles toutefois, l'agent n'est pas seul dans son environnement et il y a bien d'autres agents autour de lui. Il nous faut donc aborder des systèmes où plusieurs agents doivent interagir entre eux. De tels systèmes sont appelés « systèmes multiagents » (SMA).

L'apprentissage est une composante importante des systèmes multi-agents.

Ces systèmes évoluent généralement dans des environnements complexes (c'est-à-dire larges, ouverts, dynamiques et non prévisibles) [Sen 99].

Pour de tels environnements, c'est très difficile et même quelquefois impossible de définir correctement et complètement les systèmes à priori, c'est-à-dire lors de la phase de conception, bien avant leur utilisation. Ceci exigerait de connaître à l'avance toutes les conditions environnementales qui vont survenir dans le futur, quels agents seront disponibles à ce moment et comment les agents disponibles devront réagir et interagir en réponse à ces conditions. Une manière de gérer ces difficultés est de donner à chaque agent l'habileté d'améliorer ses propres performances, ainsi que celles du groupe auquel il appartient.

Il est à noter que l'apprentissage dans un système multi-agent comprend l'apprentissage dans un système mono-agent parce qu'un agent peut apprendre en solitaire et de façon complètement indépendante des autres agents. Mais aussi, il l'étend bien au delà dans la mesure où les activités d'apprentissage d'un agent peuvent être influencées considérablement par les autres agents et que plusieurs agents peuvent apprendre de manière distribuée et interactive comme une seule entité cohérente.

Dans un environnement multi-agent, les agents peuvent apprendre grâce aux autres. Par exemple, un agent A, qui voudrait savoir comment se rendre à un certain endroit,

pourrait demander à un autre agent B s'il connaît un bon chemin. Si B connaît un bon chemin, il peut le transmettre à A, permettant ainsi à l'agent A d'apprendre un bon chemin grâce à l'agent B.

D'un autre côté, les agents peuvent aussi apprendre à propos des autres. Par exemple, un agent peut regarder un autre agent agir dans certaines situations et, à l'aide de ces informations, il pourrait construire un modèle du comportement de l'autre agent. Ce modèle pourrait lui servir pour prédire les actions de l'autre agent dans le futur. Cette information pourrait l'aider à mieux se coordonner ou à mieux collaborer avec l'autre agent.

Plusieurs modèles ont été proposés pour représenter l'apprentissage multi-agent, [Tan 93], [Lit 94], [Wel 98a], [Wel 98b], [Sin 00], [Lit 01], [Pow 04], [Huh 98], [Ima 96], Sen 96], [Sen 98], [Wei 97], [Wei 98], [Wei 96], [Cla 98], [Dar 00] , [Bra 01].

Pendant notre étude, nous sommes intéressés par l'apprentissage mono agent.

II.5. Quelques travaux traitant l'apprentissage chez les agents

Nous exposerons quelques travaux d'apprentissage pour l'agent et suivi d'une discussion de chaque travail isolé.

- On constate un nombre croissant de travaux qui implémentent les algorithmes génétiques, la figure II.4, résume les publications énoncées depuis 1990 jusqu'à 2004 avec leurs énumération pendant chaque période. On observe une croissance très forte de l'utilisation des AG depuis 1994, il y on a 200 articles publiés avant le premier janvier 2004.

.

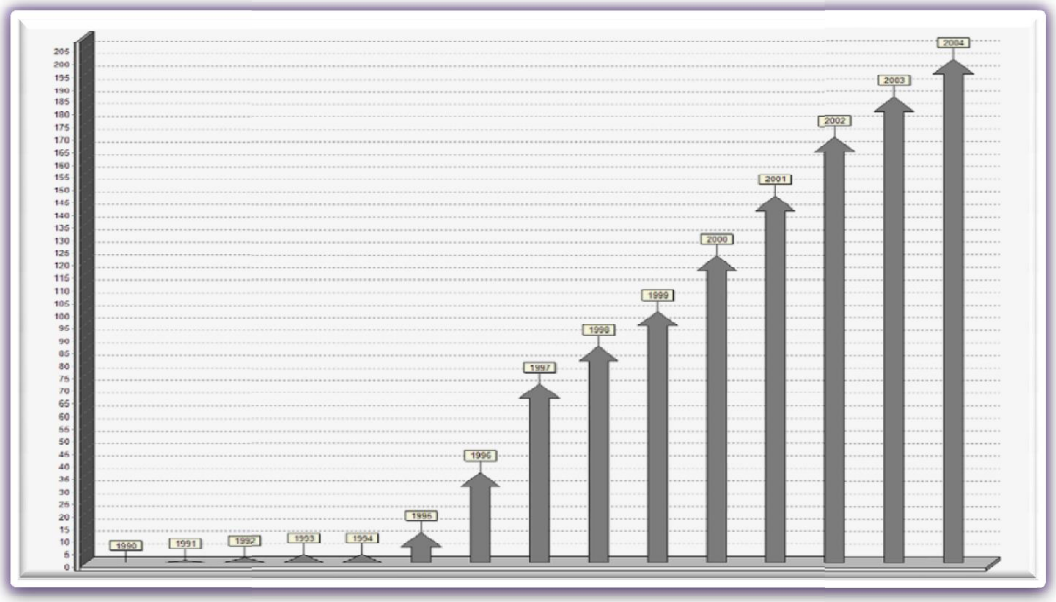

Figure II.4 : Évolution cumulée du nombre de publications utilisant les AG.

Les algorithmes génétiques (AGs) ont été appliqués avec succès dans des problèmes d'optimisation et d'apprentissage automatique [Lau 03], Ceux-ci sont particulièrement efficaces pour parcourir des espaces de recherche à la fois vastes et complexes. Ils ne nécessitent notamment pas de connaissances spécifiques du domaine an de converger vers une bonne solution, mais au contraire mettent en œuvre un cadre général permettant d'évaluer et de recombiner des solutions partielles.

De Jong [Jon 88] , rapporte que les AG sont plus performantes que d'autres techniques d'apprentissage pour la conception de systèmes d'apprentissage. Cette performance est également comparée dans les travaux de Grenfenstte[Gre 93] qui présente une rétrospective sur l'application des algorithmes génétiques dans le domaine des algorithmes d'apprentissage. Plusieurs auteurs ont également employé les AG spécifiquement pour l'acquisition de connaissances dans les systèmes de classificateurs Knight et Sen[Kin 95], Fàbrega et Guiu,[Fab 99], Holmes,[Hol 02] présente un bon exposé sur le sujet dans lequel de nouveaux modèles sont discutés et plusieurs applications intéressantes sont présentées. Des travaux de recherches concernant les systèmes multi-agents proposent aussi d'employer les AG comme système d'apprentissage à l'intérieur des agents intelligents. Cette approche permet aux agents d'être complètement autonomes d'un point de vue génétique [Cad 00]. Cette autonomie génétique permet un MA au

niveau, par exemple, de la détermination du meilleur jeu d'heuristiques au cœur même des agents.

Précisément les recherches récentes en économie pour adopter les AGs comme outil d'apprentissage sont encourageant.

Schulenburg et Ross [Sch 00] établent la performance des agents artificiels face aux données provenant d'un marché financier réel. Les agents sont d'abord entraînés sur des données des neuf années et ensuite ils utilisent les règles qu'ils ont développées pendant cette période pour gérer leurs transactions pendant la dixième année. Les simulations montrent que les agents artificiels développent une large diversité de stratégies innovantes qui ont une performance supérieure à celle des stratégies de base de type buy and-hold.

Vriend [Vri 00] propose l'utilisation des algorithmes génétiques pour l'étude des dynamiques d'apprentissage et il obtient la convergence vers la solution de Cournot avec l'apprentissage individuel.

Vallée [Val 00] utilise les AGs pour étudier les ajustements dans un modèle de jeu répété consacré à la crédibilité de la politique monétaire du Gouvernement. Après l'annonce du gouvernement, l'évolution de l'AG correspond à l'apprentissage, par les agents, de la fonction de réaction du gouvernement en fonction de l'évolution de l'inflation réalisée. Cet article teste différentes structures possibles pour un AG avec codage réel et cherche à mesurer l'information générée par chaque structure et son impact sur l'évolution du jeu. L'intérêt est qu'on doit porter à la structure de l'AG retenu et la nature de l'information générée et traité par le AG.

Jasmina Arifovic, Michael K. Maschek [jas 06], éprouvent l'apprentissage individuel des firmes pour mettre à jour leurs decisions sur la quantité de productions. Ils ont illustré la convergence du modèle. Leur conclusion était que cette convergence est dû grâce aux deux points : la manière d'évaluation des performances de la production des règles, plus la fonction du coût.

II.6. Discussion

Ces dernières années, il ya eu une croissance explosive de la recherche sur différents types de systèmes d'apprentissage artificiel allant du systèmes d'apprentissage automatique le plus traditionnelle qui s'appuie fortement sur les règles explicites symboliques et la manipulation des symboles, aux algorithmes génétiques qui a insistent fortement sur la simulation de la sélection naturelle et l'évolution génétique, aux réseaux neuronaux artificiels qui tentent de modéliser l'apprentissage biologique naturel par la modification synaptique.

L'apprentissage automatique a ses racines dans la recherche sur l'intelligence artificielle traditionnelle qui est construite sur l'hypothèse que l'intelligence réelle peut être simulée par la manipulation des symboles abstraits [Mic 83]. La caractéristique la plus distinctive entre l'approche classique d'apprentissage automatique et les deux autres est l'ancienne utilisation extensive de soi-disant connaissance préalable pour combattre l'explosion combinatoire typique de tout problème d'apprentissage non négligeable. En ce sens, l'apprentissage automatique peut être considéré simplement comme un outil pour accélérer les tâches d'apprentissage que les êtres humains performent déjà bien. Par conséquent les techniques d'apprentissage automatique sont plus une extension de compétence humaine inhérente d'apprentissage qu'un phénomène émergent» indépendant.

En outre, il est souvent difficile pour les chercheurs de décider où est ce que les aide de connaissance humaine doivent s'arrêter et les machines doivent prendre en charge la tâche de l''apprentissage, la conséquence étant que la machine de que beaucoup de problèmes d'apprentissage donnent le sentiment d'être excessivement artificiel. En toute équité cependant, il y a des techniques inspirée de l'apprentissage automatique des qui ne dépendent pas exagérément sur les connaissances antérieures, mais plutôt sont basés sur les techniques vraies et éprouvée de l'inférence statistique.

Les exemples de cette dernière catégorie sont les approches de l'arbre de décision théorique de l'information et l'approche de regroupement conceptuel, qui tous les deux ont connu un succès considérable lorsqu'elles ont été utilisées comme l'extension d'acquisition de connaissances d'un expert système. Les algorithmes génétiques, d'autre part, tentent d'aborder la question de la nature combinatoire du problème de recherche qui accompagne invariablement tous, sauf les tâches les plus triviales d'apprentissage sans

l'aide des connaissances antérieures [Hol 75] [Gol 89]. Les partisans de cette ligne de recherche estiment que le processus de la sélection naturelle, la reproduction et la modification génétique peuvent être imités artificiellement dans un ordinateur, permettant ainsi que des recherches soient effectuées sur des voies parallèles et potentiellement non rentables élagués alors que de nouvelles voies potentiellement rentable sont ouvertes à l'exploration. On peut se demander si les algorithmes génétiques peuvent en effet trouver de bonnes voies en général, et il ya des exemples de paysages qui sont GA-dur, ce qui signifie que pour ces paysages, l'algorithme génétique ne trouvera même pas de bonnes solutions, sauf par pur hasard. Un autre inconvénient majeur est qu'une grande population est généralement nécessaire pour l'algorithme génétique pour fonctionner efficacement, ce qui signifie généralement que le système d'apprentissage donné devrait être reproduit pour générer une assez grande population afin que l'algorithme génétique fonctionne correctement. Par exemple, les chercheurs de l'algorithme génétique ont tenté de former un réseau de neurones simples pour apprendre en utilisant plus de dix mille réseaux de neurones identiques [Hol 87]. Considérant le genre de tâches que le réseau de neurones a été demandé d'effectuer, cela est manifestement excessif.

L'objective de réseau de neurones artificiels est que par l'utilisation d'un élément de réseau de calcul simples qui communiquent un `a autre via des connexions pondérées, on peut imiter la fonction interne du cerveau suffisamment pour permettre au réseau de se comporter comme un apprentissage automatique massivement parallèle [Hop 84] [Blo 62]. C'est en raison de la vaste utilisation de représentations distribuées qui sont moins familières et plus difficile à penser que des représentations symboliques locales les plus coutumier, cette approche prend certainement l'air d'être nettement non-algorithmique et plus "semblable au cerveau-» dans la façon dont les réseaux apprennent à effectuer certaines tâches. Ce n'est pas vraiment le cas. Il n'ya rien de nouveau sur les représentations distribuées; la représentation bien connue de Fourier ne peut évidemment être considérée comme une sorte de représentation distribuée, mais elle existe depuis plus d'une centaine d'années. Les méthodes d'apprentissage utilisés en général pour la formation des filets neuronaux sont soit d'un type correcteur d'erreur ou d'un type de renforcement, dont les deux peuvent être mis dans le cadre de la traditionnelle techniques d'optimisation de fonction, en les plaçant clairement dans la même catégorie que tout autre technique algorithmique.

Cependant, ce que la recherche de réseau neuronal artificiel a montré, est que beaucoup de tâches «intelligentes» que les ordinateurs peuvent effectuer avec l'aide de l'IA de programmation de style AI peuvent être faites par les réseaux de neurones artificiels simulés sur un ordinateur, avec l'avantage que ces dernier n'ont pas besoin d'être programmés pour des tâches spécifiques, mais ont plutôt appris à les effectuer en grande partie de la même façon que les êtres humains et les animaux sont instruits à faire certaines choses.

Les algorithmes d'apprentissage des réseaux neuronaux, étant étroitement liés aux techniques d'optimisation classiques, se sont révélés être tout à fait capables de réduire l'erreur de performance même pour des tâches relativement complexes.

Étant principalement des techniques en ligne, qui sont souvent sous-optimales, les algorithmes d'apprentissage neural ne peuvent pas correspondre aux méthodes d'optimisation hors-ligne traditionnelle, soit dans la vitesse d'apprentissage ou dans la précision, surtout pour les petites tâches.

Cependant, la nature en ligne des techniques d'apprentissage neural rend possible de former les réseaux neuronaux pour les tâches complexes qui nécessitent l'utilisation de très grandes séries de formation. Ainsi, les réseaux neuronaux tiennent la promesse comme une puissante alternative aux techniques symboliques traditionnelles de l'IA pour les tâches intelligentes.

Malgré de nombreux succès récents, l'approche biologiquement inspirée de générer des modèles d'apprentissage intelligent a toujours des limites pratiques en raison des plus faibles capacités informatiques du réseau relatives aux dispositifs informatiques généraux telles les machines d'état finies, des automates cellulaire, et des machines de Turing . Bien que récemment, plusieurs groupes, aient commencé à examiner le problème d'obtenir des réseaux neuraux pour apprendre à se comporter comme une machine à états finie ou un automate cellulaire, les problèmes testés sont restés majoritairement dans la catégorie «jouet». Par conséquent, il peut être raisonnablement déclaré qu'à l'heure actuelle il est encore trop tôt de dire si la génération actuelle de réseaux neuraux récurrents a suffisamment de puissance de calcul pour faire face à des problèmes d'inférence grammaticale assez compliqués.

Les automates cellulaires sont une classe de systèmes dynamiques non linéaires discrets construits à partir d'un grand nombre d'automates identiques d'états-finis dont chacun reçoit des entrées d'autres automates à l'intérieur d'un

quartier prédéfini. Chacun utilise l'entrée modèle et ses informations d'état actuel pour mettre à jour son propre état ainsi que pour déterminer la sortie qu'il enverra à d'autres automates au sein de son voisinage [Bac 64][Krl 63] . Les automates sont généralement agencés selon un modèle cellulaire, d'où le nom. Ce qui rend ces systèmes mathématiques non-linéaires distribués intéressant est le fait que même avec un automate relativement simple et des règles logiques simples, le système de l'CA est généralement capable de générer une grande variété de comportements complexes. Von Neumann et Ulam, entre autres, ont été en mesure de démontrer que certaines catégories de CA sont capables d'autoreproduction [Neu 66] .Il a également été largement conjecturé que même les CA assez simple sont en mesure d'effectuer le calcul universel [Smi 68] [Cod 65].

Les automates cellulaires, en d'autres termes, sont des systèmes de calcul puissants bien adaptés pour les problèmes d'inférence complexes.

Notre travail est concerne l'apprentissage d'un agent à base d'automate cellulaire.

II.7. Conclusion

L'agent comme entité logicielle ou matérielle a besoin, dans la majorité des cas, d'un processus d'apprentissage, pour qu'il devienne plus puissant et performant à résoudre les problèmes dont il rencontre dans son environnement et d'agir efficacement.

Selon les recherches en apprentissage, plusieurs méthodes ont été développées, malgré de nombreux succès récents, les modèles d'apprentissage intelligent ont toujours des limites pratiques en raison des faibles capacités informatiques. Pour cette raison les machines d'état finies, les automates cellulaires, et les machines de Turing, aient commencé à examiner le problème d'apprendre à se comporter.

Notre recherche est basée sur l'apprentissage à base d'automate cellulaire, donc nous avons besoin de détailler la notion d'automate cellulaire dans le chapitre suivant.

Chapitre III

L'automate Cellulaire Et L'apprentissage :

Etudes

Techniques Et Formelles

III.1. Introduction

Ces dernières années, il ya eu une croissance explosive dans la recherche sur différents types de systèmes d'apprentissage artificiel. Les automates cellulaires constituent l'archétype du système apparemment simple au comportement global complexe. Les automates cellulaires sont des réseaux de cellules à une, deux trois ou n dimensions. Dans un automate cellulaire le temps et l'espace sont représentés de manière discrète. Chacune des cellules contient un automate à états finis qui calcule son état en fonction d'un voisinage, par exemple ses deux cellules adjacentes dans le cas d'un automate à une dimension, ou bien ses huit cellules adjacentes dans le cas d'un automate à deux dimensions. Chacun des automates est défini par un ensemble de règles locales qui vont spécifier son comportement, en fonction de son voisinage, des voisinages différents pouvant conduire à des états différents de l'automate considéré.

Dans ce chapitre nous ferons un tour d'horizon sur les automates cellulaires en présentant leur historique, les concepts, les propriétés et quelques travaux sur l'apprentissage à base d'automate cellulaire.

III.2. Une brève histoire des automates cellulaires

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. Par exemple, si une cellule donnée était en contact avec deux cellules allumées elle s'allumait sinon elle s'éteignait. 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. 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?

En parallèle, John von Neumann -- fort des 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'instruction, 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. Ce mécanisme correspond de fait à l'interprétation actuelle du fonctionnement de la molécule d'ADN découverte après les travaux de von Neumann. A.C. Clarke a rendu les machines de von Neumann célèbre avec la série « 2001 Odyssée de l'espace. » Pour transformer Jupiter en étoile, un premier monolithe se reproduit, les descendants font de même, la population croît ainsi de manière exponentielle pour atteindre rapidement la taille nécessaire à la réalisation d'une aussi gigantesque tâche [Bon, 1994].

C'est S. Ulam qui a suggéré à von Neumann d'utiliser ce qu'il appelait les « espaces cellulaires » (cellular spaces) 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.

Vers les années quatre vingts avec les travaux de [Ste 02], les automates cellulaires sont devenus un outil à vocation générale. Ils deviennent actuellement dans plusieurs disciplines allant des mathématiques à l'informatique en passant par la physique. Par ailleurs les recherches sur les automates cellulaires concernent de nombreuses applications parmi lesquelles les systèmes biologiques et écologiques constituent les domaines où l'on compte un grand nombre de publications.

III.3. Définition des automates cellulaires

Un automate cellulaire (cellular automata (CA)) consiste en une grille régulière de « cellules » contenant chacune un « état » choisi parmi un ensemble fini et qui peut évoluer au cours du temps. L'état d'une cellule au temps t+1 est fonction de l'état au temps t d'un nombre fini de cellules appelé son « voisinage ». À chaque nouvelle unité de temps, les mêmes règles sont appliquées simultanément à toutes les cellules de la grille, produisant une nouvelle « génération » de cellules dépendant entièrement de la génération précédente [Kpre 84].

Nous retrouvons une grande source d'informations dans le littérature sur les ACs[Wol 94, Wol 83, Gan03], ce qui nécessite de libeller notre recherche sur ce qui nous intéresse dans notre travail [How 90].

III.3.1. Etude formelle

Un Automate Cellulaire est défini par la donnée d'un quadruplet $A = (T ; v ; \varepsilon; f)$ où :

- T est un treillis de dimension d formé par des cellules c disposées selon un arrangement qui dépendra de la dimension de l'espace et de la forme choisie pour les cellules. C'est T qui définit l'espace Cellulaire.
- v est un ensemble de cellules avec les quelles une cellule donnée c peut interagir, appelé voisinage. On peut distinguer, dans le cas d'un Automate Cellulaire bidimensionnel, plusieurs types de voisinages à savoir Von Neumann, Moore, voire la figure III.1.
- ε est un ensemble fini de valeurs susceptibles d'être prises par l'état d'une cellule.C'est en général un anneau cyclique avec card. $\varepsilon = k$.
- f est une fonction de transition d'état qui définit la dynamique de l'Automate Cellulaire donnée par : $f(e_t(v(ce)))=e_{t+1}(c)$

Où $e_t(c)$ désigne l'état de la cellule c à l'instant t *et* $e_t(v(c))$ l'état de son voisinage.

Nous parlons également de configuration d'un AC donnée par l'application :

$e : T \rightarrow \varepsilon$ qui fait correspondre à chaque cellule c de T une valeur prise dans ε qui Constituera l'état de la cellule c à l'instant t.

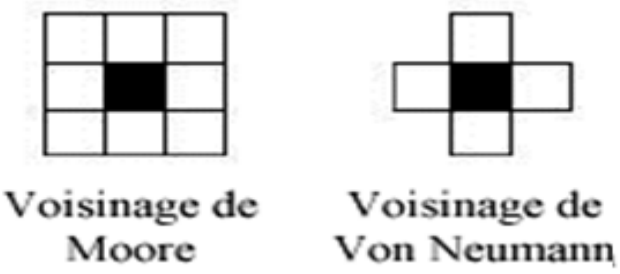

Figure III.1: Différents voisinages possibles dans un automate à 2 dimensions.

Le fonctionnement d'un automate cellulaire peut être déterministe ou probabiliste.

III.3.2. Différents types d'Automates cellulaires

Il existe différents types d'AC dont nous citons quelques uns [Ras 92]:

III.3.2.1. Les automates cellulaires Déterministe

Les automates cellulaires Déterministe sont été initialement introduits par Von Neumann [Neu 66, Wol 84] .Un automate cellulaire déterministe ***A*** est spécifié par une espace discrète cellulaires ***U***, quatre ensembles finis ***X, Y, Q, N,*** et quatre fonctions ***£ , δ, β*** et **Ω** où[Howar 90]:

1) L'espace cellulaire ***U*** est habituellement une ***N*** torus discrets tels que chaque point de U est indexée par un Z n-uplet, où :

$$z = (\mu_1, \mu_2, \ldots, \mu_N) \quad \mathbf{mod}(n_1, n_2, \ldots, n_N),$$

2) ***X*** : est l'ensemble des symboles d'entrée (input),

3) ***Y*** : est l'ensemble des symboles de sortie (output),

4) ***Q*** : est l'ensemble des symboles de l'Etat (interne).

5) ***N***: est la liste des relations de voisinage qui déterminent la position relative des sites voisins en treillis d'un site donné dans ***U,***

6) ***£*** , la fonction de configuration d'état de voisinage, est telle qu'il cartes tout ***u*** site donné dans le réseau de la voisinage de ***u*** tel que spécifié par la relation de voisinage définie par ***N ,***

£ :

$$Q \to \otimes_N Q.$$

7) ***δ*** , la fonction d'état suivant, qui utilise le courant d'entrée reçus à n'importe quel site avec l'état de la configuration actuelle du quartier tel que défini par ***N*** pour déterminer l'état ***q*** que ce site sera en particulier à l'étape prochaine fois.

δ :

$$X \times \otimes_N Q \to Q.$$

8) ***β*** : est la fonction de sortie de telle sorte que l'état de la configuration de voisinage ***£ (u)*** de tout *u* site du réseau donnée dans *U* donne toujours de sortie ***β(£ u)***.

β :

$$\otimes_N Q \to Y.$$

9) Ω , la transition mondiale et la fonction de sortie, agit sur le treillis ***U*** entier, transformant toute mondiale état de configuration ***S*** à l'instant ***t*** actuelle pas de temps dans une nouvelle configuration globale ***S '*** au pas de temps suivant ***t + 1*** avec la sortie mondiale ***O***

Ω :

$$\otimes_N Q \to \otimes_N (Q \times Y).$$

Notez que la définition ci-dessus des automates cellulaires déterministes diffère de la description d'habitude dans la dynamique de l'automate cellulaire dépendent désormais sur

un flux continu de données éventuellement d'entrée en plus d'informations sur l'état mondial actuel. La fonction de sortie est également introduite pour des raisons pratiques, car il est nécessaire dans la plupart des applications pratiques. Toutefois, cela n'affecte pas la dynamique d'ensemble.

III.3.2.2. Les automates cellulaires Stochastiques

Un automate cellulaire stochastique [Howar 90] A et définit par ***SCA*** = ***(U,X,Y,Q,N, £, F, G) ;***

(***U, X, Y, Q, N***) d'ensembles est définie par l'état ***£*** fonction de voisinage de configuration ainsi que par deux fonctions de probabiliste ***F*** et ***G***, où ***U, X, Y, Q, N*** et ***£*** conservent leur signification respective dans le cas d'automate cellulaire déterministe:

F :

$$\mathbf{X} \times \otimes_{\mathbf{N}} \mathbf{Q} \rightarrow \mathbf{Q}.$$

Dans cette fonction on a ajouté une fonction de probabilité que l'on notera par

$$f(x, n, q)$$

Vérifie:

$$\forall x \in X, \quad \forall n \in N, \quad \sum_{q \in Q} f(x, n, q) = 1$$

G :

$$\otimes_{\mathbf{N}} \mathbf{Q} \rightarrow \otimes_{\mathbf{N}} (\mathbf{Q} \times \mathbf{Y}).$$

$$\forall n \subset N, \quad \sum_{y \in Y} g(n, y) = 1.$$

Les automates cellulaires Stochastiques avec des valeurs d'entrée fixes sont appelés automates cellulaires stochastiques autonomes. Il est clair que tout automate avec des entrées constantes est équivalent à un automate sans symboles d'entrée. Il est intéressant de noter que la formulation d'une automate cellulaire stochastiques autonomie peut être utilisée pour décrire le comportement dynamique d'un automate cellulaire déterministe avec des entrées qui sont des variables aléatoires indépendantes, ou, plus généralement, un processus de Markov finies [Wol 84] .

Notre choix s'est orienté vers Les automates cellulaires déterministes.

III.4. Caractéristiques des automates cellulaires

Un automate cellulaire (AC) se définit à l'aide de deux types de caractéristiques : structurelles et fonctionnelles [Fat 01]. Les premières concernent l'aspect topologique du réseau cellulaire, les secondes concernent l'aspect dynamique de l'évolution du réseau au cours du temps. En règle générale, le réseau des cellules peut prendre corps dans des espaces à *D* dimensions, *D* étant une, deux ou trois dimensions ou encore plus. Théoriquement, il n'y a pas de limite à la dimension d'un automate, si ce n'est la puissance de calcul des machines sensées le reproduire. Par exemple, dans le cas d'une matrice à *2D*, la structure du réseau cellulaire peut-être de deux types :

- **La structure en île :** Les cellules du réseau sont virtuellement entourées par des cellules mortes.
- **La structure en tore :** Les cellules du bord haut de la matrice peuvent contacter celles du bord bas, et de même pour les cellules des bords gauche et droit.

Le fonctionnement de chaque cellule du réseau peut être caractérisé par l'*automate fini (V, vo, f)* [Fat 01] :

- *V* est l'ensemble des états cellulaires.
- *vo* un état particulier appelé état de repos.
- *f* est la *fonction de transition* qui à chaque voisinage qui est *n-tuple* d'éléments De *V* associe un élément de *V*.

V étant l'ensemble des états que peut prendre une cellule. Le plus souvent limité à *2*, mais il n'y a aucune limite à ce nombre, c'est le cas par exemple pour l'automate de Von Neumann qui était à *29* états. Dans la pratique, les états cellulaires sont souvent représentés par des couleurs ce qui permet de suivre l'évolution de l'automate. Comment l'évolution de tel automate aura lieu ?

La fonction de transition *f* est un ensemble de règles qui permettent de déterminer le nouvel état d'une cellule en fonction de son état précédent et de l'état précédent de son voisinage. Généralement, ces règles ne sont pas explicitées, mais résumées sous la forme de métarègles de type *« si...alors »* [Fat 01] par exemple figure III.2.

111	110	101	100	011	010	001	000
0	1	0	1	1	0	1	0

Figure III.2 : Représentation d'une règle d'un automate à une dimension.

L'automate évolue discrètement par génération où le temps s'écoule par à-coups. Ceci signifie qu'à la génération ***t,*** chaque cellule décide d'après sa fonction de transition que devient son état au futur c'est-à-dire au temps ***t+1***. Une fois chaque cellule à déterminé son prochain état, et seulement à ce moment là, elle passe au nouvel état calculé d'une façon à ce que le réseau cellulaire sera entièrement remis à jour de manière synchrone. On parle alors d'une simulation d'un traitement parallèle. Formellement, on peut exprimer l'évolution de l'automate par la Figure III.3.

$$Si\ (t+1) = f(\{ Sj\ (t) \}) \quad \text{où} \quad j \in Vi \ \Lambda \ Vi \text{ désigne le voisinage de la cellule } i$$

Figure III.3 : Règle de mise à jour d'un AC [Fat 01].

Un AC pouvait donc être défini par un aspect topologique décrivant la façon par laquelle les cellules sont arrangées sur le réseau cellulaire et un autre fonctionnel caractérisé par le voisinage, l'ensemble d'états et la fonction de transition qui décrit son évolution.

III.5. Terminologie

Nous avons choisi quelques concepts clé que se partagent la plupart des AC [Htt] :

- **Voisinage :** Pour chaque cellule le passage d'un état à un autre est déterminé en examinant les états des cellules voisines. Seules les interactions locales sont permises entre les cellules.

- **Parallélisme :** Toutes les cellules constituant le réseau cellulaire de l'automate sont mises à jour de manière simultanée et synchrone.

- **Déterminisme :** Un AC est dit déterministe si sa fonction de transition est une fonction classique c'est-à-dire une même entrée donne lieu toujours à la même sortie, donc la donnée des états des cellules voisines détermine à elle seule le nouvel état d'une cellule. Certains AC dits stochastiques introduisent un facteur de probabilité dans leurs fonctions de transitions en faisant intervenir des variables aléatoires, donc une même entrée peut donner lieu à plusieurs sorties différentes, en d'autre terme l'évolution d'une configuration par une même fonction de transition n'est pas toujours la même.

- **Homogénéité :** On dit qu'un AC est homogène s'il satisfait les conditions suivantes :
  - La topologie du réseau cellulaire est régulière.
  - La fonction qui calcule le voisinage doit être uniforme pour toutes les cellules.
  - L'évolution de l'automate se définit par une seule règle de transition qui s'applique à toutes les cellules.

- **Discrétisation :** Un AC s'évolue dans le temps de manière discrète, c'est-à-dire génération après génération, ce qui s'oppose avec nombreux phénomènes physiques continus.

III.6. Quelques travaux traitant l'apprentissage chez les agents a base d'automate cellulaire

Peu de travaux existent pour l'étude d'apprentissage d'un agent à base d'automate cellulaire.

Nous exposerons les travaux qui nous avons trouvé pour l'apprentissage d'un agent a base d'automate cellulaire.

P. Kachroo[Kac 99], Cet article suggère un contrôleur intelligent pour un système automatisé véhicule de planification de sa trajectoire propre, fondée sur le capteur et de la communication des données. Le contrôleur intelligent est conçu en utilisant l'apprentissage stochastique automates théorie. En utilisant les données reçues à partir de capteurs embarqués, deux automates (un pour les actions latérales, l'une des actions longitudinales) peut apprendre la meilleure action possible pour éviter les collisions. Le système a l'avantage d'être capable de travailler dans des environnements non modélisés stochastiques, à la différence méthodes de contrôle adaptatifs ou les systèmes experts. Simulations pour simultanée le contrôle latéral et longitudinal d'un véhicule de fournir des résultats encourageants.

Cheng-Yuan Liou [Che 98], basée sur le système multi agent, utilise le modèle de Blumbergs comme le noyau pour le comportement d'un agent et aussi modèle de calcul, mais le modèle de Blumbergs est construit à la main mais les développeurs ont à concevoir et à ajuster les paramètres à la main. Pour effectuer cette tâche délicate, alors vous besoin d'un mécanisme d'apprentissage pour donner à l'agent une capacité d'apprentissage. En outre, il est important d'ajuster l'agent lui-même pour faire d'adaptation aux changements dans l'environnement.

Cet article utiliser un algorithme d'apprentissage par automates cellulaires stochastiques (SCA) pour donner la capacité d'apprentissage à l'agent désiré.

III.7. Apprentissage par renforcement (AR)

Nous n'allons pas présenter en détail l'apprentissage par renforcement, mais nous rappelons les grands principes qui ont guidé notre démarche.

L'importance de l'apprentissage par renforcement dans l'apprentissage des animaux est reconnue depuis longtemps par les psychologues. Ceci a inspire aussi, les chercheurs sur les RN artificiels et les algorithmes génétiques intéressés par l'intégration des paradigmes général d'apprentissage dans les systèmes de IA. Pratiquement, le renfoncement prend forme de punitions et récompenses, et les probabilités de la réponse du système serait modifié en direct proportionnellement aux fréquences selon lesquelles les punition/récompenses sont administrées.

Développée depuis les années 1980, la théorie de l'apprentissage par renforcement a pour objectif de reproduire ce mécanisme. C'est une méthode de contrôle adaptative qui ne nécessite pas de connaitre le système mais simplement un critère de satisfaction.

L'apprentissage par renforcement consiste, pour un agent, à apprendre un comportement optimal (c'est à dire une politique optimale) à partir d'interactions avec son environnement et ce en ne connaissant ni la fonction de transition, ni sa fonction de récompense.

Un agent interagissant avec un environnement teste différent actions et doit apprendre un comportement grâce à un système de récompenses et de punitions. Ainsi, on ne dit pas explicitement à l'agent ce qu'il doit faire mais une évaluation de sa situation lui est donnée.

Les domaines d'application de l'apprentissage par renforcement sont par exemple la robotique, les jeux vidéo où l'ordinateur va progressent en même temps que le joueur. Ou encore la micro robotique et plus particulièrement la micromanipulation.

III.8. Conclusion

Un automate cellulaire est un système dynamique discret, de conception simple basé sur la donnée d'un réseau uniforme qui définit l'espace cellulaire. Chaque élément de cet espace, appelé cellule, peut prendre plusieurs états possibles dans un ensemble discret donné. L'état d'une cellule est susceptible de changer à des moments fixes, au moyen d'une règle de transition fondée sur la configuration de l'espace d'échange entre cette cellule et ses voisins.

Dans ce chapitre, nous avons fait une étude détaillée sur l'automate cellulaire en présentant son fonctionnement, ses caractéristiques principales, et ses types. On va exploiter cette étude, dans le modèle d'apprentissage de notre conception d'agent, qui sera décrite dans le chapitre suivant.

Chapitre IV

Conception Du Modèle D'un Agent

Apprenant Basé

Automate Cellulaire

IV.1. Introduction

Plusieurs méthodes ont été développées par l'intelligence artificielle pour reproduire certains aspects de l'intelligence humaine. Ces méthodes permettent de simuler les processus d'apprentissage en s'appuyant sur les connaissances de base disponibles.

Chaque méthode comporte des points forts, mais aussi des limitations. La réalisation d'automates cellulaires est une nouvelle démarche pour l'apprentissage d'un agent.

L'agent doit apprendre son comportement en s'adaptant à son milieu d'opération. Souvent, cet environnement n'est pas abordable, au sens qu'il peut être mal connu: l'automate cellulaire est alors une solution pour modéliser l'apprentissage dans le comportement de cet agent. L'un des points faibles chez l'automate cellulaire est que peu de travaux existent pour l'étude d'apprentissage chez les agents.

Dans ce chapitre nous allons décrire notre conception d'une architecture d'agent apprenant basée sur l'automate cellulaire déterministe pour formaliser le problème d'apprentissage et utiliser un nouvel mécanisme d'apprentissage par renforcement pour apprendre les évaluations de l'environnement.

Nous spécifions les éléments intervenant dans l'architecture proposée, ainsi que les étapes à inclure dans le processus d'apprentissage. Finalement, l'algorithme général de comportement sera présenté.

IV.2. Description de notre travail

L'apprentissage est une aptitude nécessaire pour amener un système à évoluer de manière efficace tenant compte les circonstances de son environnement.

Les agents autonomes ont été appliqués dans de nombreux domaines .Un agent est appelé autonome s'il décide de lui-même, comment relier les données du capteur à des commandes du moteur de telle sorte que ses objectifs sont pris en charge avec succès [Mae 95].

Nous nous attendons à un agent autonome leur permettant de s'adapter, robuste et effective dans son environnement dynamique un automate cellulaire qui donne la capacité de l'agent d'apprentissage.

Plus précisément, notre travail se situe dans une branche de la machine learning nommé reinforcement Learning. Ce dernier est une classe d'algorithmes qui déterminent la façon dont un agent autonome, pouvant observer et interagir avec son environnement, apprend à choisir de manière rationnelle les actions optimales pour atteindre son but. Suite à chaque action, l'agent peut recevoir une récompense ou une pénalité qui lui indique si l'action est désirable ou non. Par exemple, un agent qui apprend à jouer aux échecs reçoit une récompense positive quand il gagne la partie, négative lorsqu'il la perd et nulle dans tous les autres états. L'agent doit donc apprendre, à partir de ces informations étalées dans le temps, à choisir la séquence d'actions produisant la plus grande récompense totale.

Le choix du reinforcement Learning est motivé par la propriété de cette classe d'algorithmes qui permet à l'agent d'apprendre par lui-même un modèle de l'environnement.

Avant de commencer, l'agent ne connait ni son but ni les actions à éviter. Au fil de ses expériences, il apprendra les actions à faire pour maximiser ses récompenses.

Le reinforcement Learning est également souhaitable dans le cas d'un environnement dynamique. En effet, les récompenses varient au cours du temps et l'agent s'adaptera progressivement pour apprendre les règles de transition de l'environnement. Dans le cadre de ce travail, la méthode d'apprentissage d'un agent est basée sur l'intégration entre l'automate cellulaire déterministe et l'apprentissage par renforcement.

L'apprentissage par renforcement définit un type d'interaction entre l'agent et l'environnement. Depuis une situation réelle « s » dans l'environnement, l'agent choisit et exécute une action « a » qui provoque une transition vert l'état « s' ». Il reçoit en retour un signal de renforcement « r » négatif de type pénalité si l'action conduit à un échec ou positif de type récompense s'il l'action est bénéfique ; un signal nul signifie une incapacité à attribuer une pénalité ou une récompense.

L'agent utilise alors ce signal pour améliorer sa stratégie, c'est-à-dire la séquence de ses actions afin de maximiser le cumul de ses récompenses futures.

Nous nous attendons qu'en simulant les comportements des êtres humains, un agent peut traiter plus sophistiqués ce problèmes. Nous croyons que cela est raisonnable, parce que, si un agent peut simuler le comportement des gens, puis l'agent peut être en mesure de résoudre les mêmes problèmes que les gens peuvent résoudre, au moins à un degré significatif.

Cependant, en plus de la simulation des comportements, des caractéristiques de modélisation interne est important. Par exemple, si une personne mange un morceau de pizza, Il sera plus satisfait s'il a faim que s'il est repu. Si nous voulons que de simuler la situation ci-dessus nous aurons à traiter avec deux choses. La première est que, il doit y avoir une façon de dire que le degré de la faim (état interne); la seconde est que, dans le cadre de différents états de fonctionnement internes, la valeur de la pizza (conviction de monde extérieur) peut être différente.

De nombreux comportements des êtres humains montrent ces gens ont aussi des propriétés de motivation pour faire quelque chose.

Par conséquent, nous devons concevoir cet agent afin qu'il possède de nombreuses caractéristiques personnelles. Cet agent sera des états mentaux comme les émotions, motivation et les croyances en ce qui concerne le monde extérieur. Il serait peut-être heureux si elle termine une tâche à temps. Il serait peut-être impatient si elle consacre trop de temps à faire la même chose. Cet agent choisit son action en fonction de son état mental interne et des stimuli externes. Par exemple, si l'agent est faim, il y aura plus de chances de sélectionner une action à nourriture d'une action sportives.

Pour cette raison nous allons proposée l'unité des variable interne dans notre modèle de conception d'un agent apprenant.

Les automates cellulaires sont définit dans le chapitre précédant on peut donnée la définition suivante *« Cellular automata are discrete dynamical systems whose behavior is completely specified in terms of a local relation, much as is the case for the large class of continuous dynamical systems defined by partial differential equations. In this sense, cellular automata are the computer scientist's counterpart to the physicist's concept of field »*[Tof 88].

Notre choix s'est orienté vers Les automates cellulaires déterministes.

Alors les automates cellulaires assistent l'agent par l'optimisation des paramètres dans le modèle de sélection d'action. Il utilise un mécanisme d'apprentissage par renforcement pour apprendre les évaluations de l'environnement.

Dans la section suivante, nous allons proposée un nouvelle architecture d'un agent apprenant a base d'automate cellulaire déterministe et qui intègre le mécanisme d'apprentissage par renforcement.

IV.3. Architecture Globale du Système Proposée

Russell et Norvig définissent donc un agent comme étant n'importe quelle entité qui perçoit son environnement au travers des ses capteurs et sur qui elle agit via ses effecteurs (comme nous avons vu au chapitre II).

Cette définition nous permet d'introduit d'un nouveaux modèle de conception d'agent apprenant composé des modules décris comme suit (Figure IV.1):

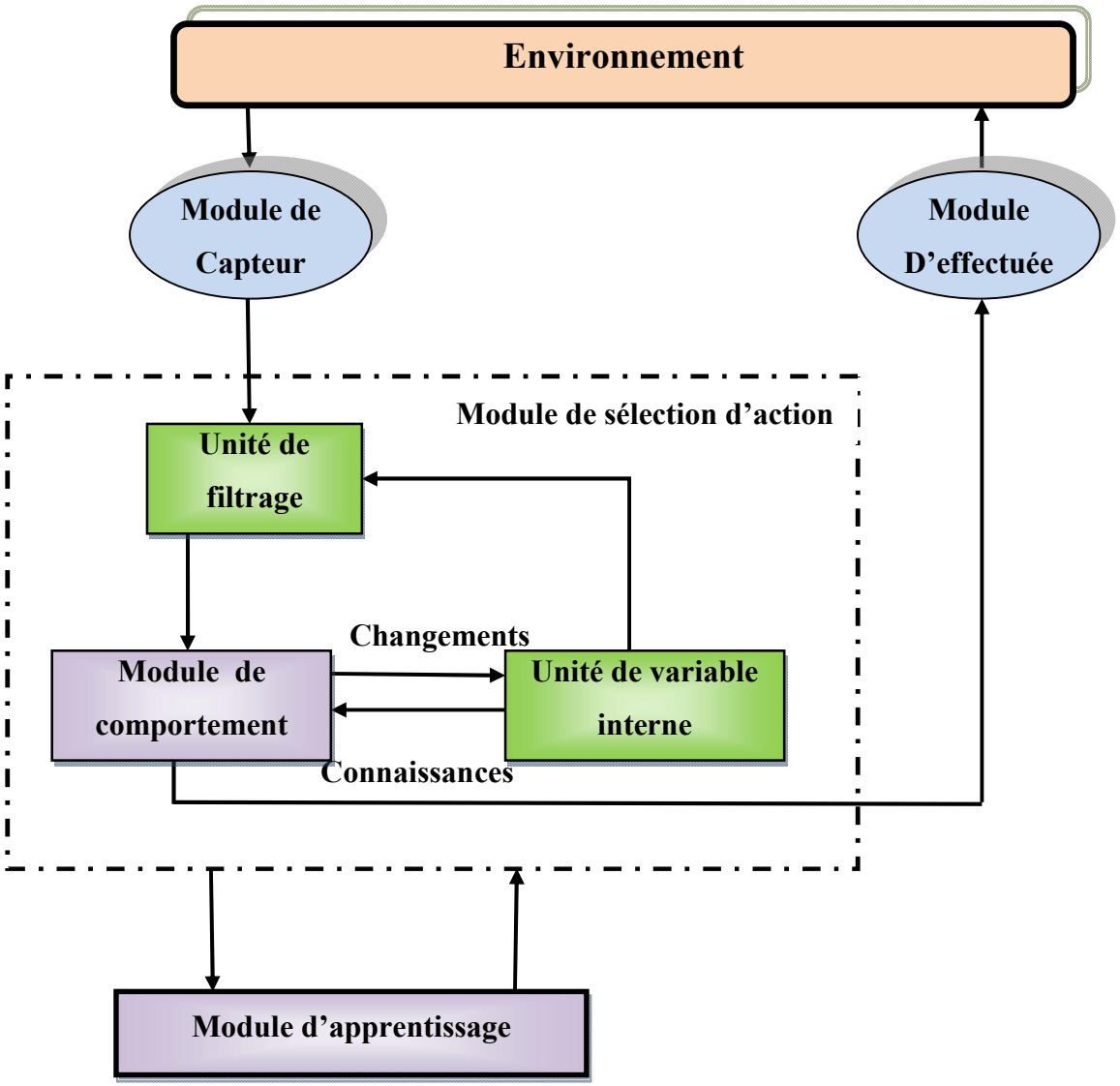

Figure IV.1 : Architecture globale d'un agent apprenant.

IV.3.1. Description des différents modules

Nous allons décrire brièvement ici les différents modules qui composent cette architecture d'un agent apprenant.

IV.3.1.1. Module d'environnement

Un agent a besoin d'un monde, il importe de tracer des limites entre ce qui est l'agent et ce qui est le monde de l'agent, l'environnement dans lequel il est situé. Donc un agent situé dans un monde délimité et qui "se pose des questions" sur ce monde. La métaphore des questions veut dire que l'agent a une sorte de projet personnel qu'il veut mener à terme (sa mission) et à la réalisation du quel l'état du monde peut être favorable ou pas ; le monde est alors une entité différente de l'agent. D'autre part, ce module consiste à interpréter les décisions prises de l'agent sous forme d'actions effectuées dans le monde extérieur où il se situe.

L'environnement dans lequel se déplacent les agents est important. Celui-ci fournit en effet les informations nécessaires aux capteurs du robot et ses propriétés déterminent les effets des actions des agents. On peut le définir comme un système duquel l'agent doit apprendre, au fil des expériences, les éléments nécessaires à la réalisation de son objectif. L'agent fait partie de ce système et peut le modifier.

IV.3.1.2. Module de Capteur

Est utilisé pour l'agent rassemble les informations externe du monde (environnement), et envoi les informations vert le module de sélection d'action.

IV.3.1.3. Module d'effectuée

L'agent changer l'état, un ensemble des opérations que peut effectuer un agent afin de modifier son environnement.

IV.3.1.4. Modules de sélection d'action

Son rôle demeure toujours de choisir des actions à effectuer, en se basant sur les perceptions en entrée, ainsi que les informations retournées par le module d'apprentissage.

Son choix est effectué sur quelle règle à appliquer. Ce module est composé de deux unités et un module (voir figure IV.1).

IV.3.1.4.1. Unité de filtrage

Cette unité reçoit des connaissances du module de capteur ensuit filtre de bruit selon les valeurs interne d'un agent. Le rôle de cette unité pour nettoyer les données, en particulier en éliminant les attributs (connaissances) inutiles.

Par exemple :

Un agent dans un labyrinthe, il y'a des murs et des couloire vide; alors l'unité de filtrage élimine les cas des murs.

IV.3.1.4.2. Unité des variables internes

Est utilisé pour modéliser les états internes d'un agent, cette variables peut changer au fil du temps selon le module de comportement, pour cette raison il faut stocker dans une mémoire sous formes d'une table permettant d'enregistrer pour chaque valeur d'un état interne d'un agent $S_i(t)$ les valeurs de gain $g_i(t)$. La mémoire offre à l'agent de s'évoluer en tenant compte les critiques reçus dans ses expériences passées.

Premièrement l'état interne d'un agent est vide parce que l'agent premièrement ne reconnait pas le monde externe et après l'apprentissage il est apprend et modifier la table de mémoire.

Premièrement la valeur de $g_i = 1$; pour tous état initiaux.

Procédure de Gain ; dans l'unité des variables internes

Si (cet état utilisé plus) alors g_i augment ;

;; l'agent réalisée l'état qui contient d'un valeur de gain (g_i) plus grand ;

Alors nous pouvons dire que cette valeur de gain (g_i) indique comme un signal de renforcement (fonction de renforcement dans cette unité), pour réaliser d'un état interne d'un agent.

IV.3.1.5. Module de comportement

Son rôle est de choisir des actions a(t) à effectuer, en se basant sur l'unité de libération (filtrage) en entrée et l'ensemble des variables internes, ainsi que les informations retournées par le module d'apprentissage. Son choix est effectué sur quelle règle à appliquer. Il inclut une base de règle contenant celles appris suite aux cycles d'apprentissage.

Dans le module de comportement il y a certain cas des systèmes forcé pour l'agent sortie cette partie parce que le système travaille avec un délai fixe au début de système.

Par exemple : un agent dans un labyrinthe est forcé pour terminer si le délai a exprimé.

Pour cette raison nous avons ajouté le délai au module de comportement (Figure IV.2).

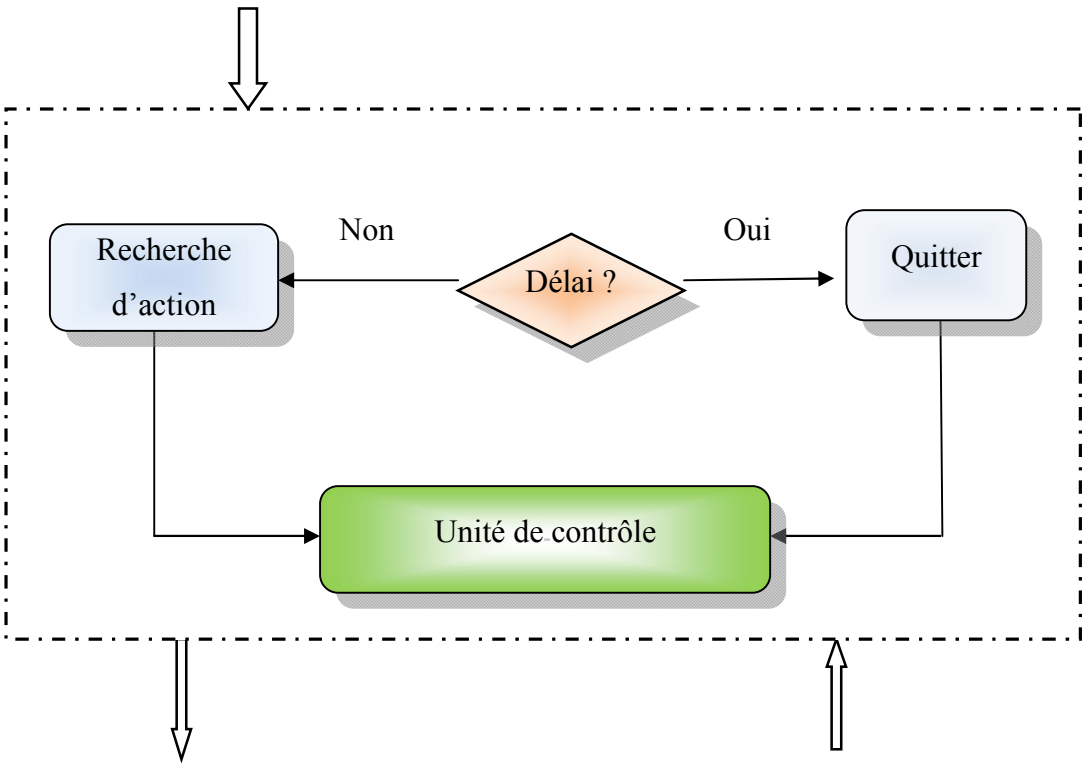

Figure IV.2 : Module de comportement.

- **Unité de contrôle :** dans cette unité est exécutée l'action pour réagir au monde externe et l'agent lui-même.

On peut écrire un algorithme simple sans module d'apprentissage pour bien reconnaitre la raison de délai :

Algorithme de comportement simple

Etat d'un agent ($e_i(t)$) ; return par l'unité de filtrage.

Ensemble des états internent ; return par l'unité des variables interne qui Contient dans une table.

Si (délai est terminer) alors fin ; agent fin de recherche.

Sinon (l'agent recherche d'une action possible pour exécutée).

IV.3.1.6. Module d'apprentissage

L'apparition de nouveaux comportements ou l'amélioration du comportement se fait via l'exploration des nouvelles règles. Il reçoit des connaissances du module de sélection d'action. Il décide à partir de ces données de la manière dont le module de sélection d'action devrait modifier les états internes afin que l'agent fasse mieux dans le futur.

Il est aussi responsable des améliorations du comportement via l'exploration des nouvelles règles.

Ce module est constitué de deux unités : une unité de base des règles et une unité de noyau d'apprentissage (figure IV.3). Dans la suivant on a exposé une description plus raffinée.

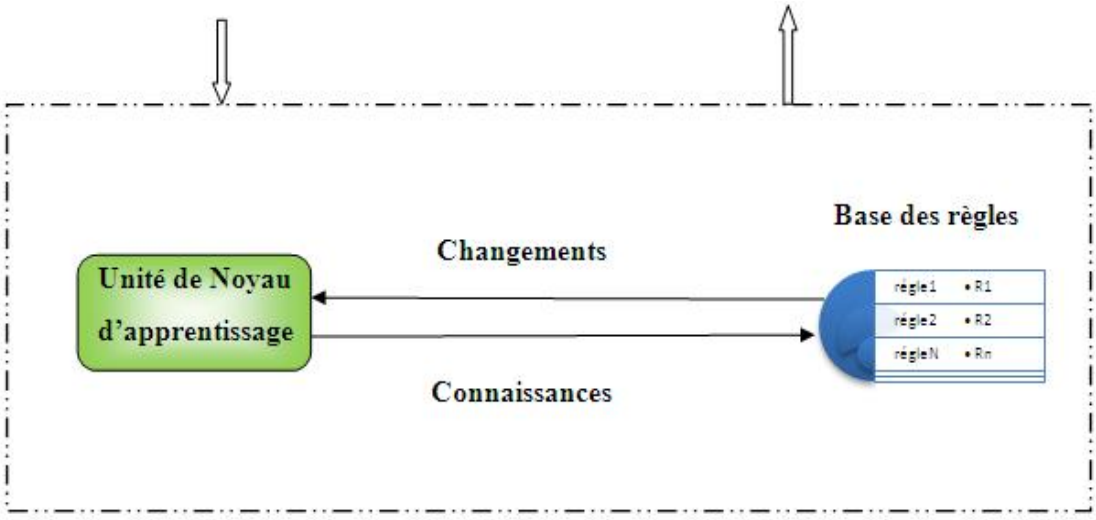

Figure IV.3: Module d'apprentissage.

- **Base de règle :** mémorise un ensemble des règles de transition et chaque règle possède une valeur de renforcement R_i pour renforce la réalisation d'un règle.

IV.3.1.6.1. Composants de l'unité de noyau d'apprentissage

Elle contient deux autres sous unités :

- **Unité de mémorisation :** les renforcements obtenus seront stockés dans une mémoire sous forme d'une matrice permettant d'enregistrer pour chaque règle $regl_i(t)$ sa valeur de renforcement correspondante R. La mémoire offre à l'agent de s'évoluer en tenant compte les critiques reçus dans l'unité de noyau d'apprentissage (sous unité d'évaluation).

- **Une unité d'évaluation :** fait appel à un processus d'apprentissage par renforcement. Elle sert à associer à un instant t, une valeur de renforcement r(t) d'une action a(t), correspondante à un état perçu e(t). Un renforcement estimé positif signifie que l'agent a bien agit et son processus d'apprentissage fonctionne d'une bonne manière. En revanche, le renforcement négatif indique un échec, et par conséquent la stratégie adaptée au niveau d'apprentissage doit être modifiée.

Dans la section suivante, il sera définit notre approche d'apprentissage (le processus d'apprentissage).

IV.3.2. Approche développementale de l'apprentissage

Chaque automate cellulaire est définit par trois caractéristiques principales :

1) **un ensemble des états** → chaque état S_i au temps t, prenne comme entré un ensemble des variables d'entrée X_i et donne comme résultat un ensemble des variables de sortie Y_i. Le diagramme suivant résume quoi dit au-dessus (figure IV.4):

X_i, Y_i et S_i sont respectivement les entrées- sorties et l'état i au pas de temps t.

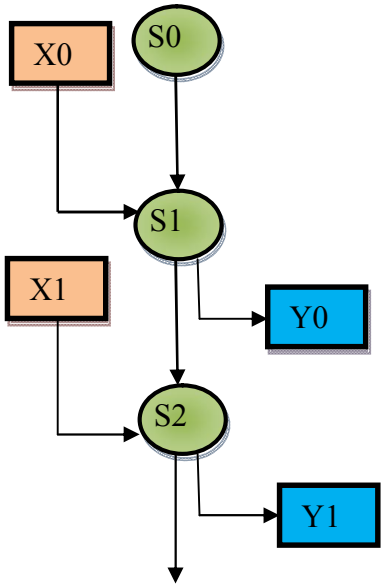

Figure IV.4: le diagramme de transition d'état d'un AC.

2) **Un concept de voisinage :** la taille de la forme du voisinage est fixée suivant le type d'application a traitée.
3) **Un ensemble des règles de transition :** partie très importante dans l'automate cellulaire parce qu'elle définie la dynamique locale ou globale d'un automate cellulaire.

Les règles de transition sont défini selon le problème d'une application mais dans notre mémoire les règles de transition peuvent être modifié selon le signale de renforcement. Maintenant on peut dire que l'apprentissage par renforcement dans l'automate cellulaire est plus important, parce que le signale de renforcement permet de modifier la table de règle de AC avec le type de signale :

Si (le signale (r(t)) = récompense) alors (modifier la tables de AC avec récompense (R augment)).
Sinon (modifier la table de AC avec pénalisé (R diminue)).

IV.3.2.1. Méthode d'apprentissage par renforcement des automates cellulaires déterministe

Nous avons proposée une stratégie d'apprentissage pour notre conception qui est basée sur la nouvelle méthode d'apprentissage par renforcement des automates cellulaires déterministe :

Si, au moment *t*, l'automate est à l'état $s_t = s^1$ et la sortie est $y_t = y^1$, puis

- lorsque le signal de renforcement $r_t = 1$ (pénaliser)

  $Q \rightarrow \otimes_N Q$

  et de préserver la normalisation, pour toutes les autres sorties $y \neq y^1$

  $\otimes_N Q \rightarrow Y$

- lorsque $r_t = 0$ (non pénaliser),

  Ne rien faire !,

Si, au moment *t*, l'automate est à l'état $s_t = s^{1,}$ son état précédent est $s^{0,}$ et l'entrée est $x_t = x^1$,

Puis

- lorsque le signal de renforcement $r_t = 1$ (pénaliser)

  $X \times \otimes_N Q \rightarrow Q$

et de préserver la normalisation pour tous les autres états $s \neq s^1$,

$\otimes_N Q \rightarrow \otimes_N (Q \times Y)$

- lorsque $r_t = 0$ (non pénaliser)

  Ne rien faire !

IV.3.2.2. L'automate cellulaire global adapté

Dans cette partie on décrit les étapes d'apprentissage :

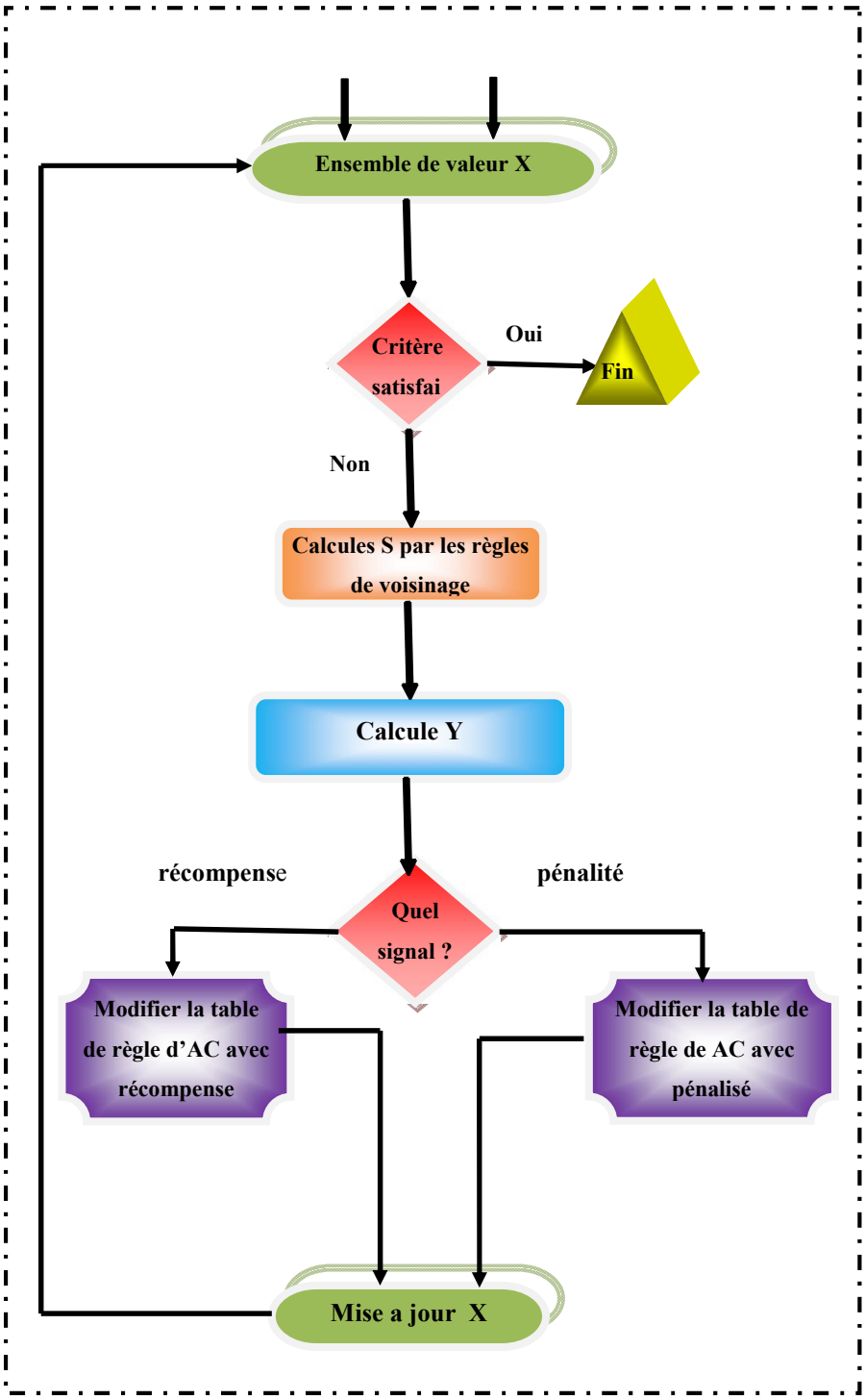

Figure IV.5 : Organigramme d'apprentissage (l'automate cellulaire adapté par apprentissage par renforcement).

Nous présentons dans cette partie notre algorithme d'apprentissage, nous avons pré en compte l'apprentissage par renforcement comme une technique permettant à un agent de modifier la table de règle de AC, par essai/erreur à choisir la meilleure règle selon la situation dans laquelle il se trouve.

Premièrement, l'agent exécute d'une façon aléatoire, ensuite nous présentons notre mécanisme d'apprentissage par automate cellulaire déterministe des actions.

1. Un ensemble X : C'est un ensemble des variables interne d'un agent qui existent dans le module de sélection d'action.
2. Critère satisfait : si (atteint l'objectif (atteint le but d'un problème) d'un agent) alors
   (Fin de programme) ;
   Sinon (aller vert l'étape 3).
3. Calcul S : ensembles des états qui associe les variables ; il calcule S en recherchant l'état le plus proche de deux voisins autour de chaque cellule utilisé les formules d'automate cellulaire déterministe qui définit dans le chapitre 3.
4. Calcules-Y : on recherche dans la table de règle d'AC.
5. **5.a)** Si (le signal de renforcement = récompense) alors (modifier la table de règle de AC avec la récompense).
6. **5.b)** Si (le signal de renforcement = pénalité) alors (modifier la table de règle de AC avec la pénalité).
7. Mise à jour X : mise à jour les variables interne d'un agent allé vert l'étape 1.

IV.3.2.3. L'algorithme de comportement de l'agent sans module d'apprentissage

La séquence d'exécution de l'algorithme est la suite :

1. Capte les informations externes de monde.
2. Filtre les bruits s'il est existe selon l'état interne.
3. Cherche dans le comportement l'action appropriée a(t) (module de comportement)
   →Cette recherche basée sur les états internes d'un agent :
   * Si (trouve e(t) dans le mémoire d'un état interne) alors (prend l'action a(t)).
   * Sinon (l'action exécute d'une façon aléatoire).
4. Mise à jour leur mémoire d'un état interne (unité des variables internes).
5. Exécuter l'action dans l'environnement.
6. Reboucler l'algorithme.

En regardant dans l'algorithme ci-dessus, il est construit à la main, les développeurs doivent concevoir et d'ajuster les paramètres à la main. Pour effectuer cette tache d'une façon délicate, nous avons besoin d'un mécanisme d'apprentissage pour donner à l'agent une capacité d'apprentissage, il est important que l'agent ajuste lui-même à l'ordre pour le rendre adaptatif à changement dans l'environnement.

IV.4. Conclusion

Dans ce chapitre, nous avons présenté une nouvelle conception d'un agent apprenant, fait intégrer l'automate cellulaire comme un outil d'apprentissage, lui permettant l'amélioration des performances c'est l'apprentissage par renforcement qui assure le bon fonctionnement d'apprentissage, à travers l'assignement positif ou négatif des décisions effectuées dans l'environnement, auprès des solution choisis. Ces renforcements permettent de diriger l'élément d'apprentissage vers la bonne décision.

Néanmoins, la durée de l'apprentissage peut être longue, en effet, l'agent adapte son comportement en fonction des récompenses ou punitions reçues. Mais, au début de sa phase d'apprentissage, il ne possède généralement pas de connaissances. Il explore alors de manière plus ou moins aléatoire son environnement.

Dans le chapitre suivant, nous allons décrire une étude de cas avec sa validation.

Chapitre V

Etude De Cas Et Implémentation

V.1. Introduction

Dans ce chapitre nous allons spécifier la démarche décrite dans le chapitre précédent pour une étude de cas dans un environnement réel. En s'inspirant des principes d'apprentissage automatique qui permettent à un robot, ayant la capacité de bouger ses membres, mais ne sachant rien de la coordination des mouvements permettant la marche, d'apprendre à marches. Le robot commencera par effectuer des mouvements aléatoires, puis, en privilégiant les mouvements lui permettant d'avancer, mettra peu à peu en place une marche de plus en plus efficace. Le cas d'étude est un problème récurrent de marche dans un labyrinthe pour trouver la sortie du labyrinthe.

Ce chapitre a pour objectif d'implémenter le modèle proposé, tout en analysant les résultats obtenus. Nous illustrons notamment l'effet des paramètres choisis sur la qualité des solutions trouvées.

Pour cela, nous procédons comme suit: nous commençons par présenter le problème à résoudre. Par la suite, nous décrivons brièvement la plate-forme que nous avons adoptée pour l'implémentation de notre problématique. Enfin nous terminons ce chapitre par des résultats obtenus à partir de l'implémentation de notre étude de cas.

V.2. Présentation de l'étude de cas

V.2.1. Description du problème à résoudre

La problématique de mon travail est de faire une simulation de déplacement d'un robot dans un labyrinthe grâce à un processus d'apprentissage. Ce robot est identifié par sa position en x et y, il n'est pas simplement une machine à associer des évènements à des situations. Il va mémoriser l'ensemble du parcours d'un labyrinthe. Les actions peuvent être les mouvements haut, bas, gauche et droite, pour sortir le labyrinthe, dans le cadre du projet, nous avons simulé l'environnement de labyrinthe, modélise le plans de labyrinthe avec les dimensions (longueur, hauteur), et le robot qui existe dans labyrinthe pour trouver la sortie avec un temps plus petit.

Les comportements que nous avons modélisés sont : d'apprendre à marche, éviter les obstacles (mur), trouver le chemin pour sortir du labyrinthe.

V.2.2. Travail Théorique

D'abord il faut étudier la modélisation et la simulation à base d'agent. Les connaissances de ce domaine qui sont nécessaires pour ce cas d'étude comportent:

La modélisation des comportements des agents, la modélisation de l'environnement, l'architecture de l'agent. Puis, pour modéliser le système d'apprendre à marche d'un robot dans un labyrinthe pour trouver la sortie avec un temps plus petit et sans parcourir tous les chemins d'un labyrinthe inutile.

V.2.2.1. L'environnement "Labyrinthe"

Nous avons besoin de la présentation du labyrinthe d'une façon plus claire. Il s'agit comme d'un damier à géométrie torique (les bords nord-sud et est-ouest sont joints), dans lequel évolue un « robot ou bien agent » qui peut se déplacer d'une case à la fois dans les quatre directions cardinales. Chaque case du labyrinthe est d'un certain type de case, défini par les caractéristiques suivantes :

La couleur : une information visuelle pour l'utilisateur.

L'accessibilité : l'agent peut-il pénétrer sur la case ?

Le caractère terminal : lorsque le robot est sur une case terminale.

La récompense : la récompense que l'agent obtient en essayant de pénétrer sur la case.

Nous avons proposés une table qui représente l'environnement de labyrinthe :

Coul.	**Acces.**	**Termin.**	**Description**
Blanc	Oui	Non	Une case vide 'normale'
Noire	Non	___	Un mur (case inaccessible)
Vert	Oui	Oui	La sortie de labyrinthe

Table V.1 : Tableau qui représente les types de cases possibles d'un labyrinthe.

V.2.2.2. Comportements de l'agent (robot)

Un agent est un système complexe dont les caractéristiques sont susceptibles de se modifier au cours du temps. L'agent n'a pas une connaissance complète de l'environnement qui l'entoure et peut ne pas être en mesure de détecter un élément important.

On doit d'abord commencer par identifier les différente règles de déplacement d'un agent les regl(i), pour nous il y'en a quatre :

regl (1) = aller au nord.
regl (2) = aller au sud.
regl (3) = aller à l'est.
regl (4) = aller à l'ouest.

Les voisinages d'état d'un agent dans un labyrinthe sera {S(1), S(2), S(3), S(4), S(5)} ; S(5) pour l'état centrale et les quatre états adjacentes.

Pour les variables internes v_i on pourrait utiliser 2 variables :

$v_i(1)$ = le temps.
$v_i(2)$ = sortie de labyrinthe.

V.2.2.3. Pour le stimulus

Dans la réalisation de notre système est le choix des différentes procédures nécessaire pour l'implémentation des différents composants.

- **Dessi-labyrinth ( ):** Pour construire un labyrinthe, nous avons crée 4 type de Labyrinthe.
- **Crée-agent ( ): C**rée l'agent dans le labyrinthe.
- **Filtrer ( ):** 0 pour les murs, 0.5 pour les couloirs et 1 pour la sortie.
- **Effectuer ( ):** Déplacement d'un agent vert nouvelle état.
- **Sortie ( ):** donne 1 si sortie et 0 si non.
- **Renforcement ( ):** Pour le renforcement, on peut choisir r = 1 lorsque le robot atteint le but (le robot a atteint le sortie), r = −1 lorsque le robot se cogne dans un mur (au temps t=1 ou t=2 par exemple) et r = 0 autrement.

- **Comportement ( ):** Cherche la sortie dans chaque direction du voisinage d'états selon l'approche qui proposé au chapitre présidant.
- **Voisin ($s_i(t)$) :** Chercher les voisins de cette état s_i à l' instant t.
- **Apprentiss-AC ( ):** Applique l'approche qui définit dans le chapitre précédant.
- **Variab-inter ( ):** Nous avons utilisée deux variables interne.
- **Aléatoire ( ):** Circulation d'un agent (robot) d'une façon aléatoire (réalisation d'action d'une façon aléatoire.

V.2.3. Travail Pratique

Le travail pratique s'agit de modéliser le modèle proposé sur une plateforme choisie.

Puis faire des expérimentations pour vérifier le modèle et pour donner des analyses du système.

V.2.3.1. Langage de programmation utilisé

Il existe beaucoup de logiciels et de plateformes qui permettent de faire la simulation à base d'agents, comme NetLogo, GAMA, JACK, JADE, Swamp, Repast, etc. Nous avons choisi NetLogo comme une plateforme appropriée pour notre problème.

V.2.3.1.1. Présentation de Net Logo (version 4.)

NetLogo est entièrement écrit en java, il est utilisé pour modéliser les systèmes décentralisés, afin de permettre aux étudiants et aux mieux concevoir et connaitre ce type de systèmes.

NetLogo nous permet manipuler des dizaines, voir des milliers de « tortus » sur un écran d'ordinateur. L'avantage de NetLogo est qu'il permet également de programmer les « patches » sous les tortus, c'est-à-dire leur environnement. Avec NetLogo, on peut donc modéliser des phénomènes aussi différents, la propagation d'un feu de foret, un trafic de voitures ou encore la construction d'un nid par des termites.

Dans notre mémoire, nous utilisée la version de NetLogo (version 4.1) [Zeg 05].

V.2.3.1.2. Motivation de choix du Net Logo

Nous avons utilisé le langage Net Logo dans notre travail puisque :

- Il permet de concevoir l'interface utilisateur de manière particulière et interactive.
- C'est un langage spécifique pour la simulation agent et multi-agent.
- C'est un langage pour modéliser des phénomènes aussi différents d'une agrégation de cellules.
- C'est un langage pour modéliser les systèmes décentralisés.
- C'est un langage adapté à modéliser les systèmes complexes développant le temps fini.
- C'est un environnement modelant programmable pour simuler des phénomènes normaux et sociaux [Zeg 05].

V.2.3.1.3. L'interface graphique de NetLogo :

La Figure V.1 représente l'interface graphique de la plate forme NetLogo.

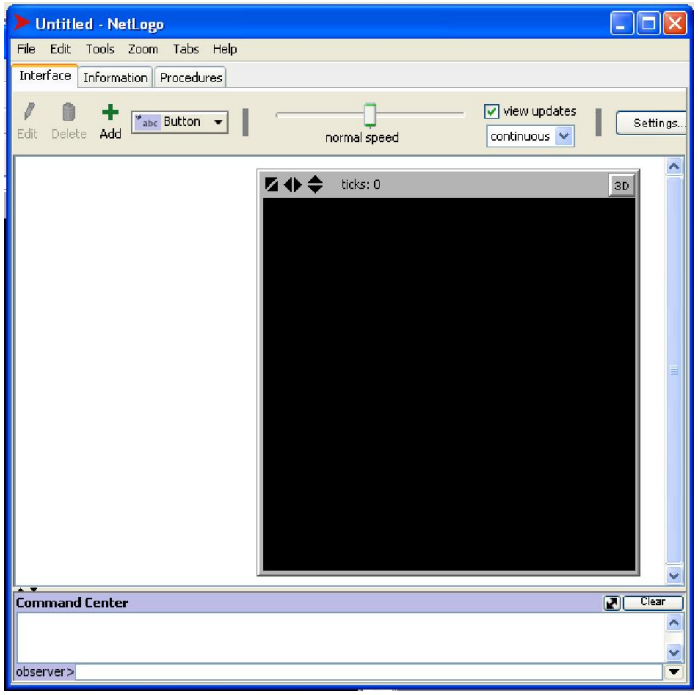

Figure V.1 : Interface graphique de la plate-forme NetLogo.

- **Interface:**
  - vue graphique du modèle.
  - position des agents.
  - interface utilisateur.
  - Paramètre, exécution.
  - Observation.
  - Courbe, reporter.
- **Procédure**
  - Code des tortues.
  - Variables.
  - Code de l'observateur.
  - partie du code correspondant au programme du projet.

V.2.3.2. Les concepts d'agent dans NetLogo

Nous rappelons que Netlogo est un langage de simulation d'agents, c'est donc un peu particulier. Bien sûr les notions classiques de programmation existent (les boucles, les tests, les variables, etc...) mais il existe en plus les concepts suivants :

- les agents : ce sont les entités qui vont interagir dans la simulation. Netlogo considère 3 types d'agents.
  - Les tortues : Netlogo les nomme « turtles », c'est typiquement tout ce qui « vit » dans la simulation (animaux, plantes, robot, tout ce qui va interagir et « réfléchir »...).
  - Les « patches » : c'est l'espace dans lequel les tortues vivent. Ce sont carrés qui constituent l'espace où peuvent évoluer les agents.
  - L' « observer » : l'observateur, c'est à dire nous.

Dans mon thèse, nous intéressons au « turtles » et nous avons crée un seul type: personne, cette agent possède un programme qu'il exécute, c'est ce qui lui donne son comportement dans la simulation. Par exemple, le personne qui existe dans le labyrinthe ont pour déplacer /programme dans le couloire de labyrinthe pour le trouve la sortie. La vitesse de déplacement d'un agent dans un labyrinthe est fixée par la simulation et vous ne pouvez pas la changer.

- « interactivité »: on peut interagir avec la simulation en cours d'exécution en écrivant des commandes via le « command center » (en bas de l'interface).

Cette dernière fonctionnalité du langage est vraiment un bon moyen d'aborder le problème, nous avons propose de l'exploiter dans la section suivante...

Le « command center »

Agir sur la simulation Le « command center » est la partie basse de la fenêtre de Netlogo (Voir Figure V.2).

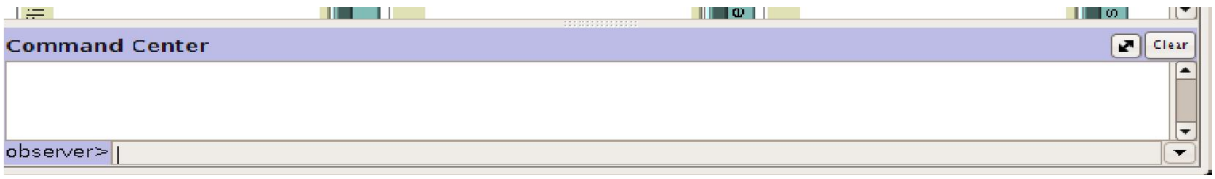

Figure V.2 : Le command center de la fenêtre de Netlogo.

On peut y donner les mêmes instructions que dans le programme qui définit la simulation. Cela fait du « command center » un très bon outil pour tester rapidement des idées et se faire la main avec le langage. Nous allons l'utiliser ici pour s'initier au langage Netlogo.

V.3. Résultat expérimentaux

Dans cette partie, nous allons présenter notre logiciel. Celui-ci concerne la circulation d'un agent (robot) dans un labyrinthe pour d'apprendre à trouver la sortie avec de temps plus court par la méthode d'apprentissage par renforcement à base d'automates cellulaires déterministe. Au début on va commencer par la présentation de l'interface et ensuite on conclura par d'exemple d'exécution d'un robot (agent) apprend dans un labyrinthe.

V.3.1. Présentation de l'interface

La manipulation de notre logiciel réalise à l'aide d'interface graphique plus claire et plus facile à traiter, cette interface est suivante:

V.3.1.1. Fenêtre principale

Nous avons utilisé dans notre logiciel une seule fenêtre principale qui contient toutes les variantes (les cas). Celles-ci sont représentées dans la figure suivante (Figure V.3):

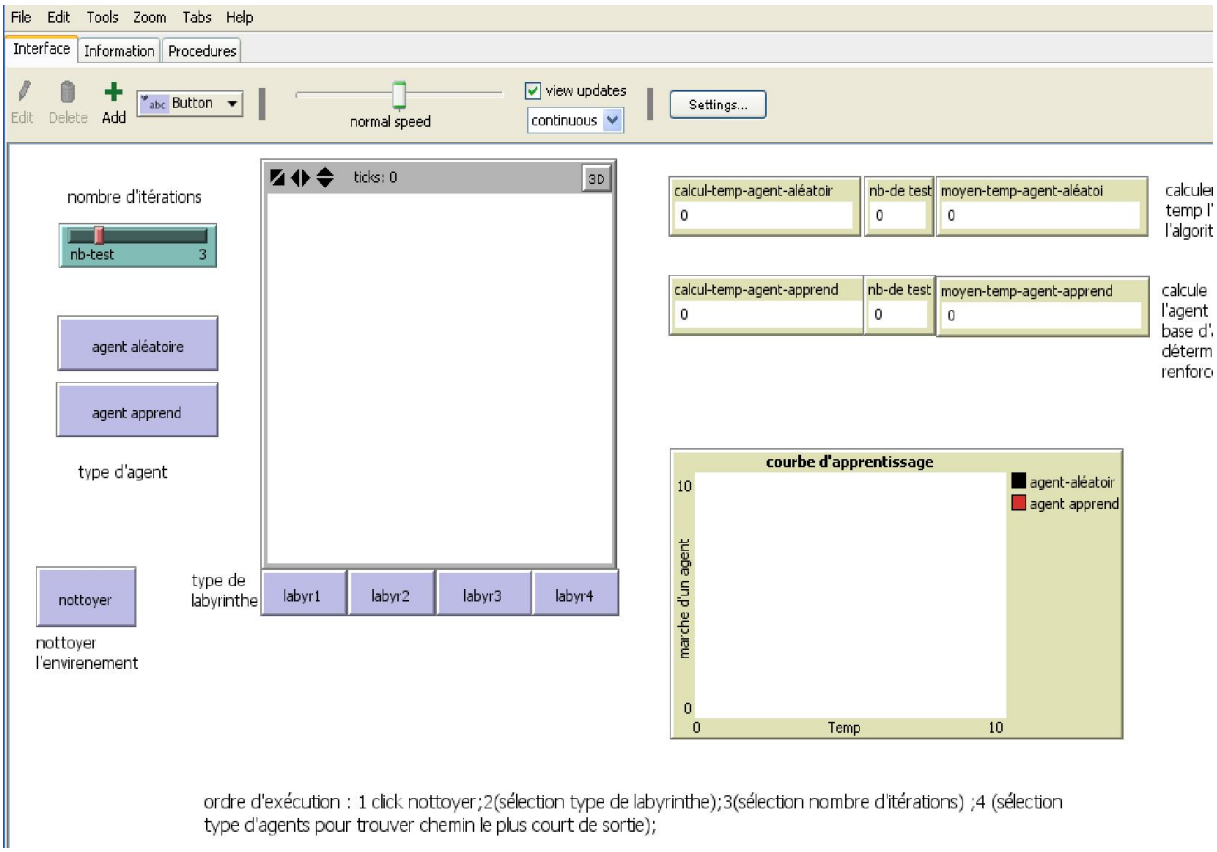

Figure V.3 : Interface de notre application.

La fenêtre principale de notre application dispose quatre (04) boutons pour accéder à toutes les fonctions disponibles :

V.3.1.2. Les bouton interviennent

Les boutons offrent à l'utilisateur un ensemble de fonction exprimant, la description de ces boutons est comme suit :

V.3.1.2.1. Labyrinthe

Permet de réalise un type de labyrinthe parmi les 4 type existe (Figure V.4, Figure V.5).

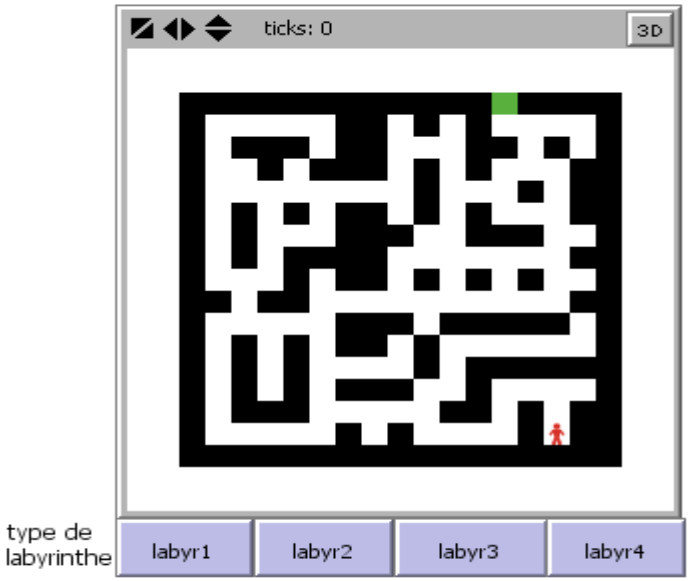

Figure V.4 : Labyrinthe de type 1 (labyr1)

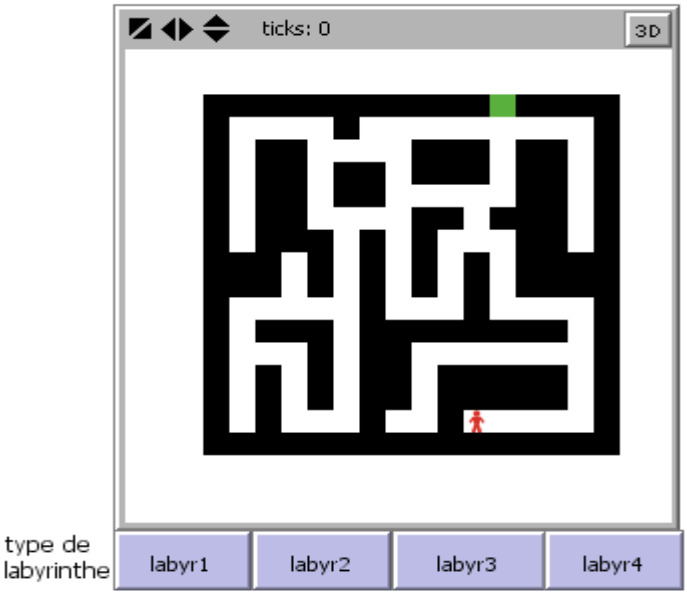

Figure V.5 : Labyrinthe de type 2 (labyr2).

V.3.1.2.2. Bouton d'itération

Possibilité de lancer le test 3 fois pour l'agent apprend (Figure V.6).

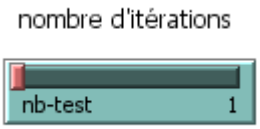

Figure V.6: Nombre d'itération.

V.3.1.2.3. Bouton de nettoyer

Possibilité de nettoyer notre environnement pour lancer une nouvelle exécution (Voir figure V.7).

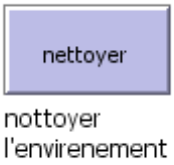

Figure V.7 : Nettoyer.

V.3.1.2.4. Bouton de réaliser le type d'agent

Ce bouton est composé de 2 sous-boutons suivants (Voir Figure V.9):

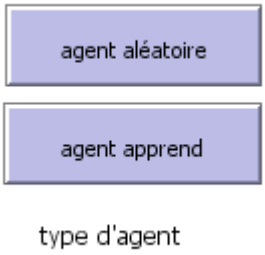

Figure V.9: Réalisation le type de programme à exécutée.

Comment réalisé ?

- Cliquer sur le bouton **nettoyer** pour initialisée le travail.
- Cliquer sur le bouton **labyr1 ou (labyrin2, labyr3, labyr4)** pour choisir un type parmi l'ensemble des type de labyrinthe existe et afficher sur le partie associer dans NetLogo.
- Cliquer sur le bouton **nb-test** pour choisirai le nombre de teste pour exécute.

- Cliquer sur le bouton **agent aléatoire** pour exécute le programme associée.
- Cliquer sur le bouton **agent apprend** pour exécute le programme associée.
- Dans ce projet, on affiche le résultat obtenus après sélectionné type d'agent a exécutée.

V.3.2. Résultats d'exécution

On choisies parmi les résultats obtenues un cas. Dans ce cas nous allons prendre labyrinthe de type 1 et le nombre de teste = 1, et le nombre de teste = 3. Appliquer sur les deux types des algorithmes (algorithmes aléatoire, algorithmes apprend).

Les figures suivant (Figure V.10, Figure V.12) présente les résultats obtenus par l'application de notre système d'apprend d'un agent dans la circulation dans un labyrinthe pour trouver la sortie. Nous avons appliqué notre système sur deux types d'algorithmes: l'algorithme aléatoire (n'existe pas l'apprentissage) et l'algorithme apprend (existe l'apprentissage).

V.3.2.1. Si le nombre d'itération = 1

Calcule le temps et le moyen de temps lorsque un agent exécute les deux algorithmes (l'algorithme aléatoire et l'algorithme apprend) (Figure V.10).

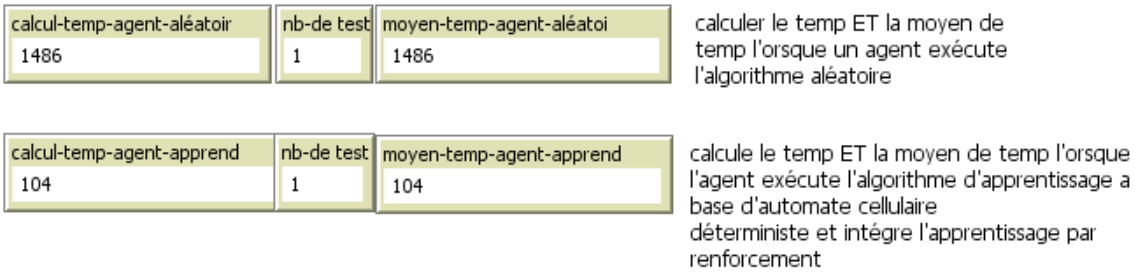

Figure V.10: Calcule le temps.

Graphe qui représente

La Figure V.11 représente une Graphe d'apprend à marche d'un agent dans un labyrinthe pour trouve la sortie avec le nombre de teste = 1 (il y'a deux types des graphes : graphe représente l'agent aléatoire et graphe représente l'agent apprend).

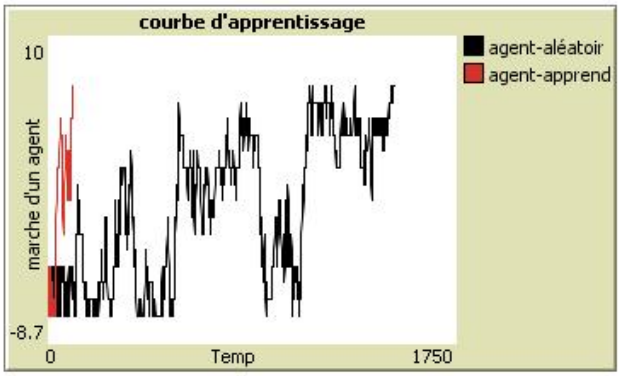

Figure V.11: Graphe d'apprend à marche d'un agent dans un labyrinthe pour trouve la sortie.

V.3.2.2. Si le nombre d'itération = 3

L'exécution de ces règles plusieurs fois (dans notre cas on a prend n= 3) permet de diminué le temps d'un agent apprend. Le nouveau résultat trouvé est alors la suivante :

La figure V.12 représente le résultat de Calcule le temps et le moyen de temps lorsque un agent exécute les deux algorithmes (l'algorithme aléatoire et l'algorithme apprend) (Figure V.12).

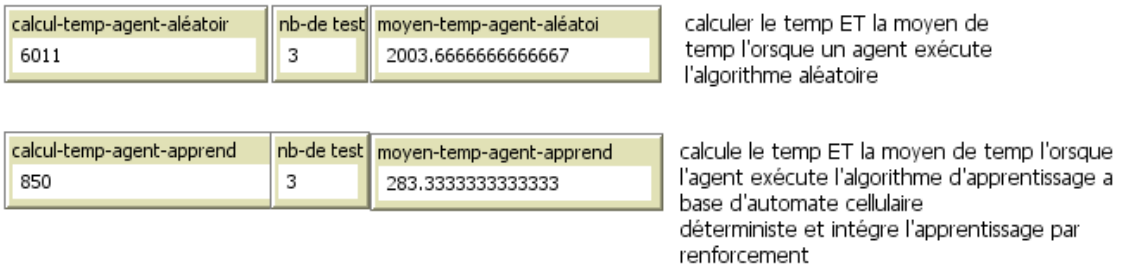

Figure V.12 : Montre qu'après 3 itérations.

Graphe qui représente

La Figure V.13 représente une Graphe d'apprend à marche d'un agent dans un labyrinthe pour trouve la sortie avec le nombre de teste = 3 (il y'a deux types des graphes : graphe représente l'agent aléatoire et graphe représente l'agent apprend).

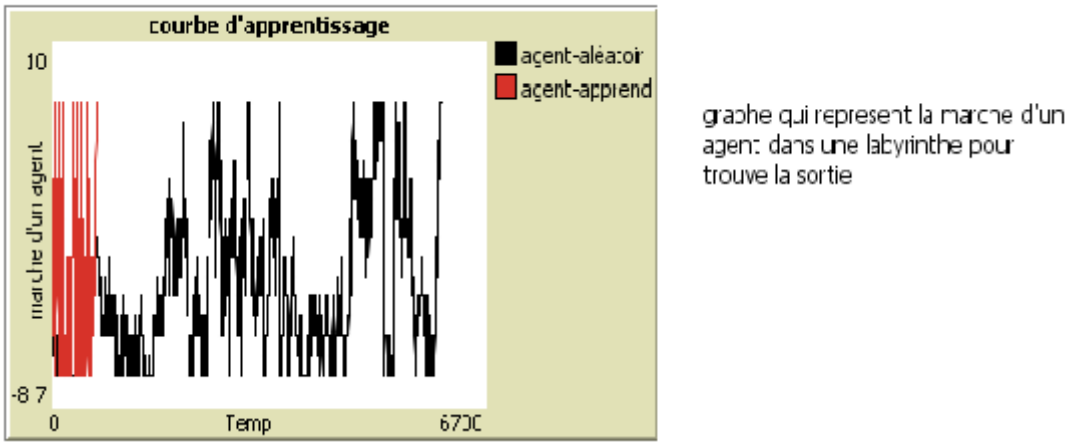

Figure V.13: Graphe d'apprend à marche d'un agent dans un labyrinthe pour trouve la sortie.

Nous avons représenté deux types d'algorithme d'agent, agent qui marche dans un labyrinthe d'une façon aléatoire et un agent apprenant à base AC déterministe et qui intègre l'apprentissage par renforcement.

Cette représentation d'agent pour comparer le temps et longueur de chemin entre les deux types d'algorithme d'agent.

L'agent aléatoire souvent ne trouve pas toujours la sortie d'un labyrinthe et des fois la trouvent à presque le temps max (max itérations).

L'agent avec approche d'apprentissage : il trouve toujours la sortie, dans ce type d'agent on n'oublie pas de parler de la phase d'apprentissage de l'agent par AC, disons qu'il a appris pendant n fois le chemin avant de le tester.

L'agent avec l'apprentissage le temps est diminué, même chose que la longueur du chemin parcouru qui est minimal.

V.4. Conclusion

Dans ce chapitre nous avons proposé une étude de cas comme application de notre conception. Nous avons présenté les différents modules de notre logiciel et les relations entre eux, avec l'implémentation des procédures utilisées.

Pour avoir une idée de cette implémentation, nous avons mis quelques résultats finaux à la fin de ce chapitre.

Avec ces résultats on peut dire que notre approche, qui est réalisé sur la plateforme de NetLogo, offre plusieurs fonctions pour le traitement d'apprentissage d'un agent à base d'automate cellulaire déterministe.

Nous avons décri et détaillé dans ce chapitre, l'implémentation d'apprendre à marche d'un agent dans un labyrinthe avec une étude de cas dans le domaine de la circulation d'un agent dans un environnement. La marche d'un agent dan un labyrinthe, ajoute l'approche d'apprentissage par renforcement à base d'automate cellulaire déterministe. Le comportement d'un agent est défini par le modèle de conception qui est développé dans le chapitre précédent.

Notre conception est bien implémentée en utilisant la plate-forme NetLogo, qui permet de faire usage de la simulation basée agents pour simuler les phénomènes dynamiques et complexe [Wil].

Conclusion Générale

Conclusion Générale

Tout au long de ce mémoire, nous avons présenté l'apprentissage en IA, pour cela nous avons présenté quelques définitions de base associées à l'apprentissage, ainsi que les caractéristiques d'apprentissage et aussi les différentes approches d'apprentissage automatique. Pour les approches d'apprentissage en IA, il existe plusieurs types, nous avons présenté quelques approches d'apprentissages, qui sont les plus connues dans le domaine de l'IA. L'objectif de ce travail est plutôt d'étudier l'apprentissage chez les agents artificiels.

L'agent comme entité logicielle ou matérielle a besoin, dans la majorité des cas, d'un processus d'apprentissage, pour qu'il devienne plus puissant et performant à résoudre les problèmes dont il rencontre dans son environnement et d'agir efficacement.

Selon les recherches en apprentissage, plusieurs méthodes ont été développées, malgré de nombreux succès récents, les modèles d'apprentissage intelligents ont toujours des limites pratiques en raison des faibles capacités informatiques. Pour cette raison, les machines d'états finis, les automates cellulaires, et les machines de Turing, aient commencé à examiner le problème d'apprentissage.

Les automates cellulaires sont des réseaux de cellules à une, deux trois ou n dimensions. Dans un automate cellulaire le temps et l'espace sont représentés de manière discrète.

Il existe différents types d'AC que nous avons cités. Par exemple, l'automate cellulaire déterministe et l'automate cellulaire stochastique.

C'est dans le cadre de cette problématique que notre travail s'est effectué. Après avoir présenté cette problématique et les verrous sous-tendus, ce manuscrit à décrit notre proposition. Celle-ci a consisté en l'étude et la conception d'une approche d'apprentissage basée automate cellulaire déterministe hybridé avec un apprentissage par renforcement.

Nous avons identifié les points intrinsèques à mettre en considération pour la conception d'une telle approche. Ensuite nous avons étudié les propriétés des deux

techniques d'apprentissage utilisées (automate cellulaire déterministe et apprentissage par renforcement), et l'apport de leur intégration.

La principale contribution de ce mémoire consiste en la proposition d'une architecture d'agent apprenant en utilisant l'AC déterministe et l'AR comme outil d'apprentissage. Valider une telle proposition a requis une implémentation qui simule le déroulement du processus d'apprentissage pour qu'on puisse évaluer sa pertinence à la résolution du problème, et mesurer la qualité des résultats obtenus.

Notre contribution offre l'avantage d'être une architecture caractérisée par les points suivants :

- ➔ Un mécanisme de mémorisation : pour garder trace des expériences passées,
- ➔ L'agent en exploitant les variables internes,
- ➔ Une auto-adaptation d'algorithme d'apprentissage,
- ➔ La possibilité de modifier la base des règles à appliquer au futur par le signale de renforcement.

Ce mémoire constitue une base de travail à partir du quelle, de nouvelles activités de recherche peuvent être lancées afin d'améliorer le travail présenté. Les perspectives que nous proposons peuvent donc s'orienter vers les directions suivantes:

- Au niveau des méthodes d'apprentissage de notre conception, utiliser la technique d'apprentissage par renforcement a base automate cellulaire stochastique définit dans l'article [Howar 90] et faire une étude comparative entre cette méthode et notre méthode.
- faire plus d'essais pour avoir plus de résultats.
- On peut ajouter (ou changement) les variables internes qui peuvent nous aider beaucoup pour l'apprentissage rapide.
- Le modèle de conception que nous avons définit au chapitre IV, peut être modifié de façon à intégrer l'unité de variable interne dans le module de comportement et définir donc un module de comportement.
- Le temps d'apprentissage est extrêmement lent. Vous pouvez remplacer les parties lentes apprentissage par perceptron multicouche.

- Dans le futur, nous espérons utilise dans un système multi agent, ce travail s'est concentré sur de cas simple, la suite naturelle est donc d'étudier d'autres cas plus évoluée.

Bibliographie

Bibliographie

[Ala 01] Alamargot, D. (2001). L'acquisition des connaissances. In C. Golder & D. Gaonac'h (Eds.). Enseigner à des adolescents. Manuel de Psychologie. Coll. Profession.

[Adl 03] Abd-Krim SEGHOUANE, Gilles FLEURY, « Apprentissage de réseaux de neurones à fonctions radiales de base avec un jeu de données à entrée-sortie bruitées » , Learning radial basis function neural networks with noisy input-output data set. École Supérieure d'Électricité, Service des Mesures, plateau de Moulon, 3 rue Joliot Curie, 91192 Gif-sur-Yvette cedex, France, Abd-krim.Seghouane@supelec.fr, Gilles.Fleury@supelec.fr,2003.

[Apt 94] Apté, C., Damerau, F. J., and Weiss, S. M. (1994). Automated learning of decision rules for text categorization. ACM Transactions on Information Systems, 12(3) :233–251. Available from World Wide Web : http://www.acm.org/pubs/articles/journals/tois/1994-12-3/p233-apte/p233-apte.pdf.

[Alt 97] Althoff, K.-D.; Bergmann, R.; Wess, S.; Manago. M.; Auriol, E.; Larichev, O. I.;Bolotov, A.; Zhuravlev, Y. I.; Gurov, S. I. Integration of Induction and Case-Based Reasoning for Decision Support Tasks in Medical Domains: The Inreca Approach. AI in Medicine Journal, December 1997. Web: http://wwwagr.informatik.uni-kl.de/~althoff/althoff-pub.html.

[Alb 87] Albert J. and .ulik K. "A simple universal cellular automaton and its one-way and totalistic version" . Complex Systems. Vol. no.1: 1_16, 1987.

[Aha 97] Aha, David W. (Ed.) *Lazy Learning*. Reprinted from: Artificial Intelligence Review,Vol.11:1-5, 432p., Kluwer Academic Publishers, Dordrecht. June 1997.

[Bre 84] Leo Breiman, J. H. Friedman, R. A. Olshen, et C. J. Stone. Classification and Regression Trees. Wadsworth, 1984. Enseignant. Hachette Education. pp. 78-113.

[Bai 99] L.C. Baird et A.W. Moore. Gradient descent for general reinforcement learning. In *Advances in Neural Information Processing Systems 11*. The MIT Press, 1999.

[Bar 98] R.S. Sutton and A.G. Barto. Reinforcement Learning: An introduction. MIT Press, Cambridge, MA., USA, 1998.

[Bra 01] R. Brafma n and Mo she Tennenho ltz. R-max - a general polynomial time algorithm for near-optimal reinforcement lea rning. In IJCAI, 2001.

[Bac 64] G.C. Bacon, The decomposition of stochastic automata,Inform. Control 7 (1964) 320-339.

[Blo 62] H.D. Block, The perceptron: a model for brain functioning, Rev. Mod. Phys. 34 (1962) 123-135.

[Blin 02] Blin Laurent, « Apprentissage de structures d'arbres à partir d'exemples ; application à la prosodie pour la synthèse de la parole »,2002.

[Che 98] Cheng-Yuan Liou , Chung-Hao Tan, Hwann-Tzong Chen , and Jiun-Hung Chen , 1998 ;« Agents that Have Desires and Behave Adaptively » Department of Computer Science and Information Engineering ; National Taiwan University, Taipei,Taiwan, R.O.C.

[Cad 00] Cadon A., Galinho T.,Vacher J.-P., "Genetic Algorithm using Multi-Objective in a Multi-Agent System", *Robotic and Autonomous Systems*, Elsevier, 33 (2-3): 179- 190, 2000.

[Cla 92] Claude Touzet, "Les reseaux de neurones artificiels introduction au connexionnisme",1992.

[Cod 65] E.F. Codd, Propagation, computation and construction in two-dimensional cellular spaces, in: Cellular Automata (Academic Press, New York, 1965); J. von Neumann, The general and logical theory of automata, in: Cerebral Mechanisms in Behavior: The Hixon Symposium (Wiley, New York, 1951).

[Cla 98] C. Claus a nd C. Boutilier. The dynamics of reinforcement learning in cooperative multi-agent systems. In AAAI, pa ges 746–752, 1998.

[Den 91] Deneubourg J.-L., Goss S., Sendova-Franks A., Detrain C. et Chretien L. (1991) "The Dynamics of Collective Sorting Robot-like Ants and Ant-like Robots." In From Animals to Animats, Paris, J.-A. Meyer et S. W. Wilson (Ed.), p. 356-363, MIT Press.

[Dar 00] Darse Billings. Thoughts on roshambo. ICGA Journal, 23(1):3–8, 2000.

[Dav 91] Davis, Lawrence. Handbook of Genetic Algorithms. Van Nostrand Reinhold. 1991.

[Dej 88] De Jong K., "Learning with Genetic Algorithms: An Overview", Machine Learning, 3(2): 121-138, 1988.

[Dej 90] De Jong, Kenneth A. Genetic Algorithm-Based Learning. In: Machine Learning - Vol.3. Y. Kodratoff et al. (Eds.), Chapter 21. Morgan Kaufmann Publishers, San Mateo. 1990.

[Dej 93a] De Jong, Kenneth. Apprentissage à partir d'Algorithmes Génétiques. In: Apprentissage Symbolique - Une Approche de l'Intelligence Artificielle, Tome II. Y. Kodratoff, R. Michalski, J. Carbonell, T. Mitchell (Eds.), Editions Cépaduès, Toulouse - France. 1993.

[Dej 93b] De Jong, K.; Spears, W.; Gordon, D. Usign Genetic Algorithms for Concept Learning. Machine Learning, 13, pp.161-187. Kluwer Academic Publishers, Boston, MA - U.S.A. 1993.

[Eic 96] Eick, Christoph F. ; Kim, Yeong-Joon ; Secomandi, Nicola ; Toto, Ema**.** DELVAUX - An Environment that Learns Bayesian Rule-Sets with Genetic Algorithms. In : The Third World Congress on Expert Systems. Seoul, Korea, February, 1996. Web: http://www.cs.uh.edu/~yjkim/ or http://www.cs.uh.edu/~ceick/ceick.html

[Fer 95] Ferber J., Les systèmes multi-agents : vers une intelligence collective, InterEditions, ISBN : 2-72-96-0572-X, 1995.

[Fer 97] J. Ferber. *Les systèmes multi-agents:un aperçu général*. Techniques et sciences informatiques. Vol. 16, No. 8, pp. 979-1012, 1997.

[Fab 99] Fàbrega X.L., Guiu J.M.G.I., "GENIFER: A Nearest Neighbour based Classifier System using GA", *Proceeding of the Genetic and Evolutionary Computation Conference (GECCO99)*, 1999.,

[Fat 01] Fates Nazim, « Les automates cellulaires vers une nouvelle épistémologie ? », Mémoire de DEA, université de Paris I Sorbone, France, 2001.

[Fer 98] Fernando Santos Osório, « Inss: un système hybride neuro-symbolique pour l'apprentissage automatique constructif», L'Institut National Polytechnique de Grenoble - I.N.P.G.Laboratoire LEIBNIZ – IMAG, 1998.

[Gan 03] N. Ganguly, P. Maji and P.P. Chaudhuri, Cellular Automata Machine for Pattern Recognition. IEEE Trans. On System Machine Cybernetics Part-A, Juillet 2003.

[Gar 70] M. Gardner, "Mathematical Games: The fantastic combinations of John Conway's new solitaire game life », Scientific American, pp. 120-123, Octobre1970.

[Gre 93] Grefenstette, J. J., "Genetic Algorithms and Machine Learning", Invited talk, Proc. Sixth Annual ACM Conf. Computational Learning Theory (COLT 93), ACM, 1993.

[Gol 89] D.E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning (Addison-Wesley, Heading, MA, 1989).

[Gia 92] Giacometti, Arnaud. Modèles Hybrides de l'Expertise. Thèse de Doctorat en Informatique et Réseaux, Lab. LIFIA - IMAG, Grenoble / ENST Paris - France, Novembre 1992. Web: http://www-leibniz.imag.fr/RESEAUX/public. html. Ftp: ftp://ftp.imag.fr/ pub/LEIBNIZ/ESEAUX-D-AUTOMATES/giacometti.these.ps.tar.gz

[Hol 75] J.H. Holland**,** Adaptation in Natural and Artificial Systems, Ann Arbor, University of Michigan Press, (Second Edition : Cambridge, MA : MIT press, 1992), 1975.

[How 90] Howard Gutowitz, "cellular automata: theory and experiment ", 1990.

[Hay 93] Hayes-Roth B. et Collinot A. (1993) "A Satisficing Cycle for Real-Time Reasoning in Intelligent Agents". Expert Systems with Applications. 7, p. 31-42.

[Hol 87] John H. Holland. Genetic algorithms and classifier systems : Foundations and future directions. In J. J. Grefenstette, editor, Proccedings of the second international conference on genetic algorithms and their applications, pages 82-89, Hillsdale, New Jersey, 1987.Lawrence Erlbaum Associates.

[Hol 86] Holland, J. Escaping Brittleness: The Possibilities of General-Purpose Learning Algorithms Applied to Parallel Rule-Based Systems. In: Machine Learning - An Artificial Intelligence Approach, Volume II. R. Michalski et al. (Eds.), Morgan Kaufmann. 1986.

[Huh 98] M. Huhns and G. Weib, editors. Special Issue on Multiagent Learning of the Machine Learning Journal. Vol. 33(2-3), 1998.

[Hop 84] J.J. Hopfield, Neurons with graded response have collective computational properties like those of twostate neurons, in: Proc. Natl. Acad. Sci. US 81 (1984)3088-3092.

[Hol 87] K.J. Holyoak, R.E. Nisbett and P.H. Thagard, Classifier systems, q-morphisms, and induction, in: Genetic Algorithms and Simulated Annealing, ed. L. Davis (Artificial Neural Networks, 1987) pp. 116 128.

[Hug 98] Huget M.P., Pinson S. (1998) "Une typologie des Modèles d'Agents" Document du Lamsade, n° 110.

[Hol 02] Holmes J.H., Lanzi P.L., Stolzmann W. & Wilson S. (2002), "Learning classifier systems: New models, successful applications", *Information Processing Letters*, 82(1), 23-30.2002.

[Htt] http://www.futura-sciences.com/comprendre/d/dossier285-1.php

[Ima 96] I.F. Imam. Intelligent adaptive agents. Papers from the 1996 AAAI Workshop. Technical Report WS-96-04, AAAI Press, 1996.

[Jon 88] De Jong K., "Learning with Genetic Algorithms: An Overview", Machine Learning, 3(2): 121-138, 1988.

[Jas 06] JASMINA ARIFOVIC1 and MICHAEL K. MASCHEK , Revisiting Individual Evolutionary Learning in the Cobweb Model – An Illustration of the Virtual Spite-Effect, Accepted 10 July 2006.

[Jar 02] Imed Jarras et Brahim Chaib-draa : « Aperçu sur les systèmes multi-agents », Montréal, Juillet 2002.

[Jea 06] Jean-Marc Trémeaux , « Algorithmes génétiques pour l'identification structurelle des réseaux bayésiens»,Rapport de stage de Master Recherche en Informatique Spécialité Extraction de Connaissances à partir des Données Année 2005-2006, Université Lumière - Lyon II .

[Joa 97] Joachims, T. (1997). A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. In Fisher, D. H., editor, Proceedings of ICML-97, 14th International Conference on Machine Learning, pages 143–151, Nashville, US. Morgan Kaufmann Publishers, San Francisco, US. Available from World Wide Web : http://www-ai.cs.uni-dortmund.de/ Dokument/joachims_97a.ps.gz.

[Kol 93] Kolodner, J. Case-Based Reasoning. Morgan Kaufmann Publishers Inc. 1993.

[Kea 98] M. Kearns and S. Singh. Near-optimal reinforcement learning in polynomial time. In ICML, 1998.,

[Kac 99] P. Kachroo ,1999 « Multiple stochastic learning automata for vehicle path control in an automated highway system», University of Nevada Las Vegas, Department of Electrical & Computer Engineering, pushkin@unlv.edu

[Kin 95] Knight L., Sen S., "PLEASE: A Prototype Learning System Using Genetic Algorithms", *ICGA*, pp. 429-435, 1995.

[Krl 63] V.Y. Krylov, On one automaton that is asymptotically optimal in a random medium, Autom. Remote Control 24 (1963) 889-899.

[Kpr 84] K. Preston, M.J.B Duff, Modern Cellular Automata, Theory and Applications,Plenum Press, London, 1984.

[Kae 96] Leslie Pack Kaelbling et Michael L. Littman, Reinforcement Learning : A Survey, Journal of Artificial Intelligence Research 4, 1996

[Lau 03] Laurent Miclet , Antoine Cornuéjols , Avec la participation d'Yves Kodratoff . Apprentissage artificiel (concepts et algorithme) ,préface de Tom Mitchell , Deuxième tirage 2003.

[Lit 94] M.L. Littman. Markov games as a framework for multi-agent reinforcement learning. In Proceedings of the Eleventh International Conference on Machine Learning, pages 157{163, San Mateo, CA., USA, 1994.

[Lit 01] M. Littman et S. Singh R. Sutton. Predictive representations of state. In *Advances in Neural Information Processing Systems 14 (NIPS'01)*, 2001.

[Lew 94] Lewis, D. D. and Ringuette, M. (1994). A comparison of two learning algorithms for text categorization. In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval, pages 81–93, Las Vegas, US. Available from World Wide Web: http: //www.research.att.com/~lewis/papers/lewis94b.ps.

[Lew 92a] Lewis, D. D. (1992a). An evaluation of phrasal and clustered representations on a text categorization task. In Belkin, N. J., Ingwersen, P., and Pejtersen, A. M., editors, Proceedings of SIGIR-92, 15th ACM International Conference on Research and Development in Information Retrieval, pages 37–50, Kobenhavn, DK. ACM Press, New York, US. Available from World Wide Web : http://www.research.att.com/~lewis/papers/lewis92b.ps.

[Mic 83] R.S. Michalski, Carbonell and T. Mitchell, eds., Machine Learning, An Artificial Intelligence Approach Vols. I, II (Kaufmann, Los Altos, CA, 1983).

[Mal 95] Malek, M. & Labbi, A. Integration of Case-Based Reasoning and Neural Approaches for Classification. Rapport Technique - RT 131, LIFIA, IMAG. Grenoble, France. 1995. Web:http://www-leibniz.imag.fr/RESEAUX/public.html.

[Mal 96a] Malek M., Amy B. A Preprocessing Model for Integrating Case-Based Reasoning and Prototype-Based Neural Network. Connectionist Symbolic Integration, Lawrence, Erlbaum Associates. 1996. Web: http://www-leibniz.imag.fr/RESEAUX/public.html

[Mal 96b] Malek, Maria. Un modèle hybride de mémoire pour le raisonnement à partir de cas. Thèse de Doctorat en Informatique, Lab. LEIBNIZ - Université Joseph Fourier, IMAG, Grenoble - France, Octobre 1996. Web: http://www-leibniz.imag.fr/RESEAUX/public.html.Ftp: ftp://ftp.imag.fr/pub/LEIBNIZ/RESEAUX-D-AUTOMATES/malek.these.ps.gz.

[Mae 95] Maes P., 'Modeling Adaptive Autonomous Agents', Artificial Life Journal, Vol.1,. No.1&2, MIT Press, Cambridge, MA, 1995.

[Mic 90] Michalski, Kodratoff. Machine Learning-An artificial Intellingence Approach Vol-III, Michalski & Kodratoff Eds, Morgan Kaufmann, 1990.

[Mcc 56] McCarthy J , Shannon C, Kleene S. C. Representation of Events in Nerve Nets and Finite Automata. In: Automata Studies., (Eds.). pp.3-41, Princeton University Press,Princeton - NJ. 1956.

[Mcc 43] McCulloch W.S. et PittsW. A logical calculus of the ideas Imminent in Nervous Activity . Bulletin of math. Biophysics, Vol 5, 1943.

[Mar 06] Marc Lucea, " Modélisation dynamique par réseaux de neurones et machines à vecteurs supports : contribution à la maîtrise des émissions polluantes de véhicules automobiles". The de doctorat de l'universite paris, 2006.

[Nik 97] Nikolopoulos, Chris. Expert Systems - Introduction to First and Second Generation and Hybrid Knowledge Based Systems. Marcel Dekker Inc. Press, 1997.

[New 82] Newell A. (1982) "The Knowledge Level". Artificial Intelligence. 18(1), p. 87-127.

[Neu 66] J. yon Neumann, Theory of self-reproducing automata (edited and completed by A.W. Burks) (University of Illinois Press, Champaign, IL, 1966).

[Neu 66] John von Neumann, Theory of Self-Reproducing Automata, Urbana : University of Illinois Press, 1966.

[Ors 95] Orsier, B. *Etude et Application de Systèmes Hybrides Neurosymboliques*. Thèse de Doctorat en Informatique, Lab. LIFIA - UJF, Grenoble, France, Mars 1995.Web: http://wwwleibniz.imag.fr/RESEAUX/public.html. Ftp:ftp://ftp.imag.fr/pub/LEIBNIZ/RESEAUX-DAUTOMATES/orsier.these.ps.gz

[Perr 97] Stéphane Perret : « Agents mobiles pour l'accès nomade à l'information répartie dans les réseaux de grande envergure ». Thèse en Sciences Informatique, LSR-IMAG, Grenoble I, France, 19 Novembre 1997.

[Pow 04] R. Powers ,Y. Shoham, et T. Grenager : Multi-agent reinforcement learning : a critical survey. AAAI Fall Symposium on Artificial Multi-Agent Learning, 2004.

[Qui 86] J. R. Quinlan. Induction of decision trees. Machine learning 1, 1986.

[Qui 79] Quinlan, J. Ross. Discovering Rules by Induction from Large Collections of Examples. In: D. Michie (Ed.), Expert Systems in the Micro Electronic Age. Edimburgh, UK: Edimburgh University Press. 1979.

[Rom 03] Romaric Charton, Des Agents intelligents dans un environnement de communication multimédia : vers la conception de services adaptatifs, thèse de doctorat de l'université Henri Poincaré_ Nancy 1,Decembre2003.

[Rou 03] Jean_Pierre Géorgé, Résolution de problème par émergence , Etude d'un Environnement de programmation Emergente,Thèse de doctorat de l'Université du francois Rebelais de tours,2003-2003.

[Rus 95] Russell S. et Norvig P., Artificial Intelligence: A Modern Approach, The Intelligent Agent Book. Prentice Hall Series in Artificial Intelligence, 1995.

[Ras 92] S.Rasmussen, C. Knudsen, and R. Feldberg. "Dynamics of programmable matter". In C. G. Langton, C. Taylor, J. D. Farmer, and S. Rasmussen, editors, Artificial Life II, volume X of SFI Studies in the Sciences of Complexity, pages 211-254, Redwood City, CA. Addison-Wesly, 1992.

[Rus95] RUSSELL S. J., NORVIG P., Artificial Intelligence. A Modern Approach, Prentice-Hall, Englewood Cliffs, 1995.

[Sat 00] Satinder Singh , Learning Predictive State Representations, Computer Science and Engineering, University of Michigan,2000.

[Smi 68] A.R. Smith, Simple computational-universal cellular spaces and self-reproduction, in: Conference Record,IEEE 9th Annual Symposium on Switching and Automata Theory (1968) pp. 269-277.

[Sen 99] SEN S., WEISS G., « Learning in Multiagent Systems », WEISS G., Ed., *Multiagent Systems : A Modern Approach to Distributed Artificial Intelligence*, Chapitre 6, p. 259-298, The MIT Press, Cambridge, MA, 1999.

[Sch 00] Schulenburg, S. & Ross, P. (2000), An adaptive agent basel economic model, *in* P. L. Lanzi, W. Stoltzmann & S. W. Wilson, eds, "Learning classifier systems. From foundations to applications", Vol. 1813 of *Lecture notes in artificial intelligence*, Springer, Berlin, pp. 263–282.

[Sen 96] S. Sen. Adaptation, coevolution and learning in multiagent systems. Papers from the 1996 Spring Symposium. Technical Report SS-96-01, AAAI Press, 1996.

[Sen 98] S. Sen, editor. Special Issue on Evolution and Learning in Multiagent Systems of the International Journal of Human-Computer Studies. Vol. 48(1), 1998.

[Sing 00] Satinder Singh, Michael Kearns, and Yishay Mansour. Nash convergence of gradient dynamics in general-sum games. In UAI, pages 541–548, 2000

[Sim 83] Herbert A . Simon,Why should machines learn?, Machine Learning:An artificial intelligence approache (R.S.Michalsiki,J,G.Carbonell, and T.M.Mitchell, eds.),vol.I;Morgan Kaufmann,1983.

[Ste 03] Steven van Dijk, Linda C. van der Gaag, and Dirk Thierens. A Skeleton-Based Approach to Learning Bayesian Networks from Data. In PKDD, pages 132_143, 2003.

[Sah 98] Sahami, M., editor (1998). Proceedings of the 1998 Workshop on Learning for Text Categorization, Madison, US. Available as Technical Report WS-98-05.

[Tur 36] A.M Turing, "On Computable Numbers, with an application to the Entscheidungsproblem" , *Proceedings of the Mathematical Society*, Série 2, Vol. 42,p.230-265, 1936.

[Tan 93] M. Tan. Multi-agent reinforcement learning: Independent vs. cooper-ative agents. In Proceedings of the Tenth International Conference on Machine Learning, pages 330,337, 1993.

[Tou 92] Touzet, C., "Les réseaux de neurones artificiels : Introduction au connexionnisme ", Cours, exercices et travaux pratiques, 1992.

[Vri 00] Vriend, N. (2000), "An illustration of the essential difference between individual and social learning, and its consequences for computational analysis", *Journal of Economic Dynamics and Control* 24, 1–19.

[Val 00] Vallée, T. (2000), Heterogeneous inflation learning : communication versus experiments. Working Paper presenté à la conférence : *Complex behavior in economics : modeling, computing, and mastering complexity*, Aix en Provence, Marseille, France, 4-6 Mai.

[Van 97] G. VAN De VIJVER - *Emergence et explication* - Intellectica : Emergence and explanation, 1997/2 no25, ISSN no0984-0028 185-194, 1997.

[Wol 83] Stephen Wolfram, Statistical Mechanics of Cellular Automata, in Review of Modern Physics 55, P. 601-644, 1983.

[Wol 84] Stephen Wolfram, Universality and complexity in cellular automata, Physica D, 10 :1-35, 1984.

[Wol 94] Stephen Wolfram, Cellular Automata as simple self-organizing systems (1982), Addison-Wesley, 1994.

[Wei 93] G. Weiss," Learning to coordinate actions in multi-agent systems". In Proceedings of the 13th International Joint Conference on Artificial Intelligence, pages (311,316),1993.

[Wei 96] G. Weiss, "Adaption and learning in multi-agent systems", volume 1042 of Lecture Notes in Artificial Intelligence, chapter Adaptation and Learning in Multi-Agent Systems - Some Remarks and a Bibliography, pages 1_21. Springer-Verlag, 1996..

[Wei 99] Gerhard Weiss, éd. Multiagent Systems. A Modern Approach to Distributed Artificial Intelligence. MIT Press (1999).

[Wei 97] G. Weib, editor. Distributed artificial intelligence meets machine learning. Lecture Notes in Artificial in Artificial Intelligence, Vol. 1221. Springer-Verlag, Berlin, 1997.

[Wei 98] G. Weib, editor. Special Issue on Learning in Distributed Artificial Intelligence Systems of the Journal of Experimental and Theoretical Artificial Intelligence. Vol. 10(3), 1998.

[Wei 96] G. Weib and S. Sen, editors. Adaption and learning in multiagent systems. Lecture Notes in Artificial in Artificial Intelligence, Vol. 1042. Springer-Verlag, Berin, 1996.

[Wel 98a] M. Wellman et J. Hu. Multiagent reinforcement learning : theoretical framework and an algorithm. In *Proceedings of the 15th International Conference on Machine Learning (ICML'98)*, pages 242–250, 1998.

[Wel 98b] M. Wellman et J. Hu. Online learning about other agents in a dynamic multiagent system. In *Proceedings of the 2nd International Conference on Autonomous Agents (Agents'98)*, pages 239–246, 1998.

[Woo 95] Wooldridge, M. and Jennings, N., 1995. "Intelligent Agents: Theory and Practice", *The Knowledge Engneering Review*, 10(2):115-152, 1995.

[Wil] Wilensky Uri,Netlogo User Manual, Center for Connected learning and Computer-Based Modelling"Northwestern University".

[Yil 01a] Yildizoglu, M. (2001a), Connecting adaptive behaviour and expectations in models of innovation : The potential role of artificial neural networks, *European Journal of Economic and Social Systems*, Vol. 15, n°2, 203-220.

[Yil 01b] Yildizoglu, M. (2001b), Modelling adaptive learning : R&D strategies in the model of Nelson & Winter (1982). Working Papers IFREDE-E3i (http ://www.ifrede.org).

[Yan 94] Yang, Y. and Chute, C. G. (1994). An example-based mapping method for text categorization and retrieval. ACM Transactions on Information Systems, 12(3) :252–277. Available from World Wide Web : http://www.acm.org/pubs/articles/journals/tois/1994-12-3/p252-yang/p252-yang.pdf.

[Yan 04] Yann Braouezec, « Apprentissage et conflit exploration-exploitation : un essai»,Esilv-Dept Mathématiques et ingénierie Financière pole universitaire léonard de vinci,92916 paris la défense cedex,2004