REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPILAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE « Abbés LAGHROUR » DE KHENCHELA

FACULTE DES SCIENCES ET DE LA TECHNOLOGIE

Département de Mathématique et Informatique

Mémoire

Présenté pour l'obtention du diplôme de Master en informatique (L.M.D)

Par :

Chekilla Zanoubia

Guellab Salim

Option : Sécurité et Technologies Web (STW)

Vers un nouveau modèle organisationnel basé sur les Réseaux de Neurones pour les Systèmes Multi Agents

---

Soutenu le : 14/07/2021 devant les jurys :

- **Dr. Malik Mohamed Mehdi**
- **Dr. Makhlouf Ledmi**
- **Dr. SOUIDI Mohammed El Habib (encadreur)**

Année : 2020/2021

Résumé

Un système multi-agent est un ensemble d'entités capables d'interagir afin d'effectuer une tache complexe. Les Systèmes Multi-Agents (SMA) doivent souvent poursuivre un but commun et pour y parvenir ils doivent former des coalitions efficaces. Une approche pour la formation de coalition concernant la poursuite multi-agent basée sur les Réseaux de Neurones (NN) et le modèle organisationnel Overlapping Agent Group Role Membership Function (OAGRMF) est proposée dans ce mémoire. Ce nouveau modèle nommé OAGRMF-NN est composé de deux parties, l'extraction de fonctionnalités ainsi que la génération des groupes. Tout d'abord, les couches d'extraction des éléments peuvent résumer les caractéristiques OAGRMF pour tous les groupes. Deuxièmement, ces caractéristiques seront alimentées à la partie de génération de groupe basée sur la couche de carte autoorganisée (SOM) qui est utilisée pour regrouper les poursuivants avec des caractéristiques similaires.

Mots clés : Système multi-agents (SMA), Réseau de neurone (RN), modèle organisationnel Overlapping Agent Group Role Membership Function (OAGRMF), Poursuite-évasion.

Abstract

Multi-agent Systems (MAS) often must pursue a standard goal and so as to realize that they have to form an efficient formation. An approach for coalition formation of multi-agent pursuit supported neural network (NN) and agent group role membership function based on a limited overlapping of the dynamic groups (OAGRMF) model is proposed and thus making a completely unique algorithm called OAGRMF-NN. This new algorithm consists of two parts i.e. feature extraction and group generation. First, the layers of feature extraction can abstract the OAGRMF feature for all of the groups. Secondly, those features are going to be fed to the group generation part supported self-organizing map (SOM) layer which is employed to group the pursuers with similar features.

Keywords Multi-agent system (MAS), Neural network (NN), Multi-agent pursuit coalition formation based on a limited overlapping of the dynamic groups (OAGRMF), Pursuit-evasion.

Remerciement

Tout d'abord nous remercions ALLAH le tout puissant, le miséricordieux

Qui nous donnés la force, le courage et la volonté

Pour achever ce modeste travail

Nous remercions nous parents qui sans leurs sacrifices et leurs efforts

, nous ne saurai jamais arrivée où nous sommes aujourd'hui.

NOUS tenons à exprimer nous profonde gratitude à notre encadreur

Dr. Mohammed El habib souidi *pour sa disponibilité ses aides précieuses et ses conseils judicieux et ses contrôles et orientations. Et pour avoir accepté de diriger ce travail. Son soutien, ses compétences et sa clairvoyance nos été d'une aide inestimable.*

Tous les enseignants de la Département de informatiques à l'université de Khenchela, pour leurs enseignements de qualité et leurs conseils qui nous ont permis de poursuivre notre itinéraire académique jusqu'à présent ;

Dédicace

A mes très chers parents

J'ai toujours attendu avec une grande impatience ce jour où je vous Témoignerai toute la gratitude

D'une fille qui s'est toujours vantée de Vous avoir comme père et mère.

Aucune dédicace n'est susceptible de vous exprimer la profondeur

de mon amour, de mon estime et l'infinie reconnaissance pour tous les

sacrifices consentis avec dévouement pour mon éducation et mes

longues années d'études

Vous avez guetté mes pas et vous m'avez couvert de tendresse, vos prières et vos

bénédictions m'ont été d'un grand secours pour mener

a bien mes études.

Je serai votre dévouée pour tout le restant de mon existence et nulle

déclaration ne m'allégerais de la lourde responsabilité dont je me

sens investie à votre égard.

Que Dieu le tout puissant, vous comble de santé, de prospérité et

vous accorde une longue vie afin que je puisse vous combler à mon

tour...

A mes très chers frères ***: Nedjmeddine, Mahdi*** *qui n'ont cessé d'être pour moi des exemples de persévérance, de courage et de générosité*

A ma très chère sœur : ***Asma*** *, et ma très cher neveu* ***islam***

Je remercie ALLAH de nous avoir unis dans une si belle famille.

Puisse ce travail être le témoignage de ma profonde affection. Que

Dieu vous comble de bonheur, de santé, de succès et de prospérité dans votre vie et vous protège.

À tout ma famille sans exception ;

Qu'ALLAH, le Très-Haut, fasse que le meilleur reste à venir.

ZANOUBIA

Dédicace

A mon Homme ***Salim***

Pour l'amour et l'affection qui nous unissent.

Je ne saurais exprimer ma profonde reconnaissance

pour le soutien continu dont tu as toujours fait preuve.

Tu m'as toujours encouragé, incité à faire de mon mieux,

ton soutien m'a permis de réaliser le rêve tant attendu.

Je te dédie ce travail avec mes vœux de réussite,

de prospérité et de bonheur.

Je prie Dieu le tout puissant de préserver notre attachement mutuel,

et d'exaucer tous nos rêves.

ZANOUBIA

Dédicace

Dès les premières lignes de ce document

je ressens une grande gratitude et une gratitude illimitée envers Dieu

Créateur du Ciel et de la Terre, Maître des temps et des circonstances

A la personne la plus parfaite qui fut en ce monde,

je dédie ce travail, il est vrai qu'il n'est pas avec nous pour récolter les fruits de son sacrifice,

mais il reste toujours le plus présent, à l'âme de mon père ***Radjeb*** *qui a fait moi ce que je suis.*

Et à ma chère maman qui me fortifie par ses mots et m'encourage et me pousse à aller

le plus loin possible pour réaliser mes ambitions.

A mon frère ***Ammar***

Ces quelques lignes, ne sauraient traduire le profond

amour que je te porte.

Ta bonté, ton précieux soutien,

ton encouragement tout au long de mes années d'étude,

ton amour et ton affection, ont été pour moi l'exemple de persévérance.

Je trouve en toi le conseil du frère et le soutien de l'ami.

Que ce travail soit l'expression de mon estime pour toi et que Dieu te protège,

t'accorde santé, succès et plein de bonheur dans ta vie.

Et à mes sœurs, je leur souhaite à tous bonheur et réussite.

Mes salutations spéciales à Chourouk , Djalal et Wassim.

SALIM

Dédicace

Je Didier ce travail à petite famille

À ma femme ***Zanoubia***

malgré que Aucune dédicace ne pourrait exprimer mon amour et mon attachement à toi.

Depuis que je t'ai connu, tu n'as cessé de me soutenir et de m'épauler.

Tu me voulais toujours le meilleur.

Ton amour ne m'a procuré que confiance et stabilité.

Tu as partagé avec moi les meilleurs moments de ma vie, aux moments les plus difficiles de ma vie, tu étais toujours à mes cotés,

Je te remercie de ne m'avoir jamais déçu.

Aucun mot ne pourrait exprimer ma gratitude, mon amour et mon respect.

Je remercie le bon dieu qui a croisé nos chemins.

Puisse le bon dieu nous procure santé et longue vie.

Qu'Allah nous aide et nous rassemble pour le bien.

SALIM

Table des matières

Résumé i
Abstract ii
Remerciement iii
Dédicace iv
Table des matières viii
Table des abréviations x
Liste des figures xi
Liste des Tableaux xii
INTRODUCTION GENERALE xiii
Chapitre 01
1.1 Introduction 1
1.2 Les Systèmes Multi-Agents (SMA) 1
1.2.1 Définition d'un système multi-agents 1
1.2.2 Agent 2
1.3 Modèles d'organisation dans les Systèmes Multi-Agents 4
1.3.1 AGR (Agent Group Rôle) 5
1.3.2 AGRMF (Agent Group Rôle Membership function) 5
1.3.3 OAGRMF (Overlapping Agent Group Rôle Membership function) 6
1.4 Processus de décision markovien MDP 7
1.5 Les réseaux de neurones 8
1.5.1 Historique 8
1.5.2 Le neurone biologique 9
1.5.2.1 Fonctionnement du neurone biologique 10
1.5.3 Le Réseau de neurone artificiel 10
1.5.3.1 Du neurone biologique au neurone formel 10
1.5.3.2 Caractéristiques des Réseaux de Neurones Artificiels 12
1.5.4 L'apprentissage de réseaux de neurones 15
1.5.4.1 Apprentissage supervisé 15
1.5.4.2 Apprentissage par renforcement 15
1.5.4.3 Apprentissage non supervisé 15
1.5.5 Architecture des Réseaux de neurones 15

Table des matières

1.5.5.1 Les réseaux Feed-forward 16
1.5.5.2 Les réseaux Feed-back 18
1.5.6 Avantages et inconvénients des réseaux de neurones artificiels 20
1.6 Conclusion 21
Chapitre 02 .....
2.1 Introduction : 22
2.2 Travail connexe 22
2.3 Description du problème de Multi-agent Poursuite 24
2.3.1 Poursuivants : 24
2.3.2 Évader 26
2.4 Structure de l'OAGRMF-NN 26
2.5 Extraction des caractéristiques des poursuivants 29
2.6 La couche SOM pour la génération de groupe 31
2.7 Algorithme de formation d'OAGRMF-NN 34
2.7.1 Fonction d'attractivité du groupe 34
2.7.2 Rétropropagation basée sur CEF 35
2.8 Conclusion 37
Chapitre 03 .....
3.1 Introduction 38
3.2 Plateformes multi-agents 38
3.2.1 MACE 38
3.2.2 ZEUS 39
3.2.3 MADKIT 40
3.2.4 SWARM 40
3.3 Plateforme de simulation NetLogo 41
3.4 Résultats de la simulation 43
3.5 Conclusion 49
Conclusion générale 50
Références 51

Table des abréviations

ABM *: modèles basés sur les agents*

AGR *: Agent- groupe- rôle*

AGRMF *: Agent Group Rôle Membership function*

BLE *: Broadcast of Local Eligibility*

CEF *: coalition evaluation function*

DAI *: intelligence artificielle distribuée*

GAF *: La fonction d'attractivité du groupe*

GUI *: Une interface utilisateur graphique*

MAS *: Multi-agent Systems*

NN *: Réseaux de Neurones*

OAGRMF *: Overlapping Agent Group Role Membership Function*

PE*: poursuite-évasion*

PMC *: perceptron multicouche*

RBF *: Le réseau à fonctions radiales*

SMA *: Les Systèmes Multi-Agents*

SOM *: Self organizing map*

Liste des figures

Figure 1.3.1 : *Modèle AGR*.....5
Figure 1.3.2: *OAGRMF méta-model.* 6
Figure 1.5.1 : *Un neurone avec son arborisation dendritique*.....9
Figure 1.5.2 : *Passage du neurone biologique vers le neurone formel*.....12
Figure 1.5.3 : *Perceptron monocouche*.....16
Figure 1.5.4: *Perceptron multicouche* 17
Figure 1.5.5 : *Schéma d'une carte auto-organisatrice* 18
Figure 1.5.6 : *Architecture d'un Réseau de Hopfield*.....20
Figure 2.5 : *Dispositif d'extraction pour les poursuivants* 30
Figure 2.6: *Réseau de neurones pour la formation d'une coalition* 34
Figure 3.2: *L'architecture MadKit* 40
Figure 3.4.1 : *Capture des évadés* 43
Figure 3.4.2 : *L'écran d'interface NetLogo pour notre simulation* 44
Figure 3.4.3 : *Evolution moyenne des récompenses au cours d'une épisode de la poursuite* 45
Figure 3.4.4 : *Récompense moyenne obtenue par itération pendant un épisode complet de la poursuite*.....47
Figure 3.4.5: *Durée moyenne de la poursuite*.....48

Liste des Tableaux

Tableau 1: *Les agents cognitifs vs réactifs* *4*
Tableau 2: *passage du neurone biologique vers le neurone formel.* *11*
Tableau 3: *Les fonctions d'activations* *14*

INTRODUCTION GENERALE

Les modèles basés sur les agents (**ABMs**) cherchent à cloner le comportement humain dans une réalité virtuelle ou un environnement artificiel.

Dans ce dernier, les décideurs des agents travaillent ensemble, dans le but de produire des phénomènes possibles compréhensibles par les spécialistes des sciences sociales. Cette définition n'est peut-être pas suffisamment claire pour ceux qui ne sont pas des experts en intelligence artificielle distribuée (DAI), mais elle offre une signification claire si l'on tient compte du fait que nous connaissons tous les jeux vidéo, qui reflètent un type particulier d'ABM

En fait, les jeux vidéo intègrent les joueurs dans un monde virtuel dans lequel certains personnages (les agents) interagissent : monstres, poursuivants, évadés, astronautes, soldats, etc. Chacun de ces agents entreprend un protocole de comportement spécifique de règles (un algorithme de comportement) concernant le comportement dans le cas individuel et, surtout, le cas où l'agent interagit avec d'autres agents rencontrés. Cependant, il existe tellement de possibilités que les rencontres aléatoires peuvent facilement transformer le jeu en un certain nombre de résultats très différents. Les jeux vidéo produisent des événements possibles à partir des interactions entre les agents mis en œuvre, dont certaines pourraient être assez improbables.

Multi-agent poursuite-évasion (PE) dans un environnement inconnu devient l'un des défis les plus intéressants dans différents domaines basés sur des agents tels que la coordination des tâches et la planification des trajectoires.

L'objectif général de cette mémoire est de proposer un nouveau modèle organisationnel base sur les Réseaux de Neurones pour les Systèmes Multi Agents le but des simulations effectuées est de montrer comment ces différents mécanismes affectent le temps et le développement interne des poursuivants pendant les poursuites. Cette mémoire comprend trois chapitres :

Le chapitre 1 présente les concepts liés à notre travail, parmi lesquels nous pouvons trouver les systèmes multi-agents ainsi que les agents, les modèles organisationnelles, et aussi les définitions essentielles et les principaux titres liée à Les réseaux de neurones.

Le chapitre 2 présente la proposition d'un algorithme de formation de coalition basé sur le modèle organisationnel Agent-Groupe-Rôle-membership_function-Overlapping (OAGRMF). Cet algorithme est appliqué au problème de poursuite-évasion, Dans ce modèle, les poursuivants

intègrent les groupes via l'utilisation de réseaux de neurone Pendant la formation de la coalition le RN va permettre aux groupes d'agents de regrouper d'une manière automatique et intelligents.

Dans le dernier chapitre nous allons étudier le temps d'exécution ainsi que le nombre d'itération utilise pour capturer un évader dans différents cas. Les résultats de la simulation reflètent les avantages apportés par ce nouvel algorithme par rapport à les autres travail reliée.

Chapitre 01
Introduction

Contenue

Chapitre 01 .....

1.1 Introduction 1

1.2 Les Systèmes Multi-Agents (SMA) 1

1.2.1 Définition d'un système multi-agents 1

1.2.2 Agent 2

1.3 Modèles d'organisation dans les Systèmes Multi-Agents 4

1.3.1 AGR (Agent Group Rôle) 5

1.3.2 AGRMF (Agent Group Rôle Membership function) 5

1.3.3 OAGRMF (Overlapping Agent Group Rôle Membership function) 6

1.4 Processus de décision markovien MDP 7

1.5 Les réseaux de neurones 8

1.5.1 Historique 8

1.5.2 Le neurone biologique 9

1.5.2.1 Fonctionnement du neurone biologique 10

1.5.3 Le Réseau de neurone artificiel 10

1.5.3.1 Du neurone biologique au neurone formel 10

1.5.3.2 Caractéristiques des Réseaux de Neurones Artificiels 12

1.5.4 L'apprentissage de réseaux de neurones 15

1.5.4.1 Apprentissage supervisé 15

1.5.4.2 Apprentissage par renforcement 15

1.5.4.3 Apprentissage non supervisé 15

1.5.5 Architecture des Réseaux de neurones 15

1.5.5.1 Les réseaux Feed-forward 16
1.5.5.2 Les réseaux Feed-back 18
1.5.6 Avantages et inconvénients des réseaux de neurones artificiels 20
1.6 Conclusion 21

1.1 Introduction

Les systèmes multi-agents (**SMA**) représentent actuellement un domaine très actif et largement appliqué. Ce domaine est une branche de l'intelligence artificielle distribuée (IAD) quand nous parle d'intelligence artificielle en peux mentionner Les réseaux de neurones, les **RN** formels sont des systèmes de traitement de l'information dont la structure s'inspire de celle du système nerveux. Leurs deux grands domaines d'application sont d'une part la modélisation biologique, dont il ne sera pas question ici, et d'autre part, la réalisation des machines destinées à effectuer des tâches auxquelles les ordinateurs et les outils traditionnels semblent moins bien adaptés que les êtres vivants, telles que la **classification**.

Nous commençons donc ce chapitre, en premier lieu, par les principes de base des concepts liés à notre travail, parmi lesquels nous pouvons trouver **les systèmes multi-agents** ainsi que les agents, les modèles organisationnelles tell que AGR… , ensuite nous rappellerons l'historique des **RN**, et les définitions essentielles, nous expliquons ce qu'est un neurone formel, un réseau de neurones, et l'apprentissage des **RN** (nous précisons notamment les différences entre l'apprentissage supervisé et l'apprentissage non supervisé).

1.2 Les Systèmes Multi-Agents (SMA)

Avant de décomposer le concept de système multi-agents en ses composants de base, il semble important de déterminer le concept du système à étudier

1.2.1 Définition d'un système multi-agents

Ferber a défini les **systèmes multi-agents** Dans son livre « Les Systèmes Multi-Agents : Vers une Intelligence Collective » comme « un ensemble composé de :

- Un environnement **E** : c'est-à-dire un espace disposant généralement d'une métrique.
- Un ensemble d'objets **O** situés : c'est-à-dire 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 d'agents **A**, qui sont des objets particuliers (c'est-à-dire), 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 d **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. »

1.2.2 Agent

Dans le passé, le terme « agent » a été suivi d'une variété de sous-disciplines de l'intelligence artificielle et de l'informatique. Le concept d'agents est devenu courant en robotique, en communication de données et en recherche sur les systèmes concomitants, ainsi qu'en génie logiciel, en IA et en IA distribuée. Un article remarquable publié dans un quotidien national britannique fait l'hypothèse suivante : « L'informatique base agent (ABC) est susceptible d'être la prochaine percée importante dans le développement de logiciels. [1].

Le dictionnaire définit un agent comme « celui qui, ou celui qui, exerce un pouvoir ou produit un effet » [2]. Bien que cette définition ne soit pas très utile, elle indique au moins que l'action est en quelque sorte mise en cause et, en fait, il semble à première vue que la notion d'action est inextricablement liée à celle d'agence : « Les agents font des choses, ils agissent, c'est pourquoi on les appelle agents » [3].

Ferber définit un agent comme [4] : « Un agent est une entité autonome, réelle ou abstraite, qui est capable d'agir sur elle-même et sur son environnement, qui, dans un univers multi-agents, peut Communiquer avec d'autres agents et dont le comportement et la conséquence de ces observations, De ses connaissances et des interactions avec les autres agents ».

Russel définit un agent comme [5] : « Un agent est une entité qui perçoit son environnement et agisse sur celui-ci ».

1.2.2.1 Caractéristiques des agents

A partir de définitions précédentes, nous Pouvons identifier des caractéristiques essentielles pour les agents

- **La situation** : toutes les définitions notent qu'un agent doit être situé dans un environnement. Un agent peut donc percevoir son environnement et agir sur les changements de cet environnement.
- **L'autonomie** : cette caractéristique est, peut-être, la caractéristique la plus importante des agents. On désigne par cette caractéristique la capacité de l'agent d'agir sans l'intervention d'un tiers. En d'autres termes, un agent peut atteindre ses objectifs sans demander l'aide d'un autre agent, système ou utilisateur.
- **La réactivité** : la capacité de l'agent de percevoir son environnement et répondre aux différents changements apparus dans le temps adéquat
- **La pro-activité** : la capacité de l'agent de prendre l'initiative en démontrant des comportements orientés objectifs.
- **La sociabilité** : la capacité de l'agent d'interagir avec les autres agents en utilisant un langage de communication.

1.2.2.2 Les types d'agents

- **Les agents cognitifs** : cette classe des agents fait partie de l'intelligence artificielle symbolique[1]. Ainsi, l'agent possède une représentation symbolique de son environnement, des autres agents, des actions et des états internes de l'agent. Sachant que les buts de l'agent sont modélisés comme des états internes, le raisonnement de l'agent consiste à trouver l'ensemble des actions possibles pour satisfaire ses buts. C'est à dire il a un objectif à

  Atteindre, un plan à réaliser. Capable de planifier ses tâches, de Mémoriser, de raisonner de manière approfondie.

[1] Désigne l'ensemble des approches et techniques en IA qui sont fondées sur des représentations « symboliques » (lisibles par l'homme)

- **Les agents réactifs** : les agents réactifs sont basés sur l'intelligence artificielle réactive[2]. Les agents réactifs sont des agents qui ne possèdent ni une représentation de leur environnement ni un vrai mécanisme de raisonnement. En plus, leur comportement est modélisé par des règles de type stimulus-réponse qui produisent des actions à partir de perceptions.
- **Les agents hybrides** : la division des agents en deux classes ne représente qu'une vue idéaliste. En fait, ces deux classes ne présentent que les cas extrêmes. En réalité, un agent nécessite des capacités réactives et des capacités cognitives. Ainsi, il est important de combiner les deux approches citées ci-dessus. Un agent hybride est un agent conçu en couches. Les couches inferieurs assurent des comportements réactifs. Par contre, les couches supérieures sont responsables de capacités cognitives complexes comme les aspects sociaux.

AGENTS COGNITIFS	AGENTS REACTIFS
Représentation explicite de l' environnement	Pas de représentation explicite
Peut tenir compte de son passé	Pas de mémoire locale
Agents complexes	Fonctionnement stimulus/action
Nombre d'agents réduit	Nombre d'agents élevé

Tableau 1: Les agents cognitifs vs réactifs

1.3Modèles d'organisation dans les Systèmes Multi-Agents

Un modèle organisationnel est mécanisme de coordination de tache pour les systèmes multi-agent. Ce modèle permet aux agents de coopérer et de se partager les taches afin de pouvoir résoudre un problème complexe. Tout cela se fait dans le cadre d'un métamodèle qui permet de définir les relations entre les agents ainsi que le rôle de chacun.

[2] École de l'intelligence artificielle est basée sur la possibilité de concevoir des comportements intelligents à partir de comportements simples

1.3.1 AGR (Agent Group Rôle)

Le modèle **AGR (Agent, Groupe, Rôle)** [Gutknecht et al., 2004], est l'évolution du modèle AALAADIN. Comme le décrit la Figure 1, l'organisation dans ce modèle s'articule autour des notions d'agent, de groupe et de rôle.

- **Agent** : Dans ce modèle, un agent est une entité capable d'agir et de communiquer, qui peut jouer un ou plusieurs rôles dans un ou plusieurs groupes. Il n'existe aucune contrainte sur la structure interne de l'agent.
- **Groupe** : le moyen de regroupement des agents « le groupe est avant tout un terme générique pour qualifier la communauté d'agent en relation » [6].
- **Un rôle** : on peut lui donner la définition : « représentation abstraite d'une fonction du groupe pouvant contraindre le comportement de l'agent, et incarnée dans un ou des comportements spécifiques par l'entité » [6].

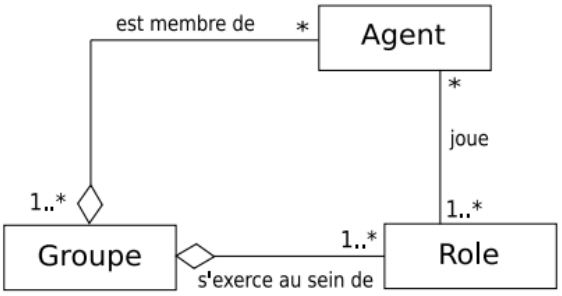


Figure 1.3.1 : Modèle AGR

1.3.2 AGRMF (Agent Group Rôle Membership function)

AGRMF est une extension du modèle agent group-rôle. Contrairement à d'autres institutions, cette approche permet aux agents de considérer le groupe comme flou. Par conséquent, chaque groupe peut être géré au moyen d'une fonction d'adhésion. Au lieu de déterminer

explicitement si un agent spécial est censé ou non appartenir à l'un des groupes, l'**AGRMF** admet les degrés d'adhésion à un groupe donné. Ce changement offrira plus d'espace pour utiliser différents algorithmes afin d'optimiser le résultat pour l'organisation et plus de flexibilité pour la réorganisation par rapport au modèle AGR ordinaire. [7]

1.3.3 OAGRMF (Overlapping Agent Group Rôle Membership function)

Est un Nouveau cadre de modélisation organisationnelle, L'objectif de ce modèle est d'optimiser l'exécution des tâches multi-agents ainsi que le développement des agents au cours de ce processus. Afin d'obtenir un Rôle, l'Agent doit appartenir au Groupe proposant le Rôle

concerné par un diplôme d'Affilié spécifique. En comparaison avec AGRMF, OAGRMF permet le chevauchement des groupes existants [8] :

$$\forall\, gr1\,, gr2 \in\ GR \begin{cases} si\ gr1\ \cap gr2 = \emptyset\ alors\ AGRMF \\ sinon \qquad\qquad\qquad\quad OAGRMF \end{cases} \qquad \textbf{(1)}$$

gr1 , gr2 : groupes composés d'agents.

GR : l'ensemble des groupes existants.

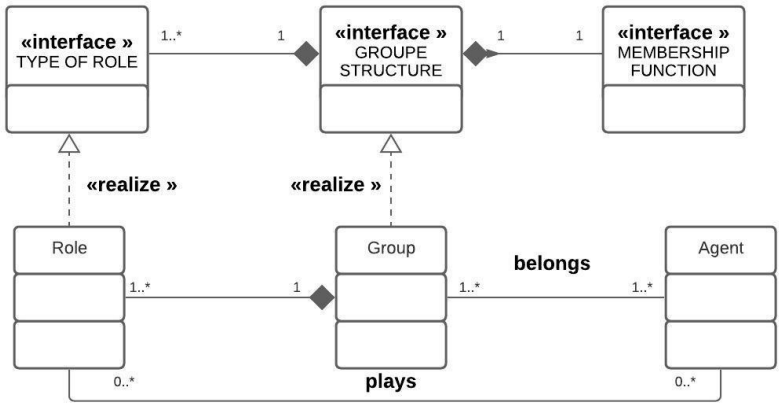


Figure 1.3.2: OAGRMF méta-model.

1.4 Processus de décision markovien MDP

Un processus stochastique $\{X_t, t \in T\}$ est un ensemble de variables aléatoires indexées par un paramètre t (temps discret) et définies sur le même espace de probabilité. La variable X_t représente l'état du processus au temps t. De plus, l'ensemble de toutes les valeurs possibles pour cette variable est appelé l'espace des états du processus et sera noté S.

Une chaîne de Markov de temps discret est un processus stochastique de temps discret $\{X_t, t = 0, 1...\}$, défini sur un espace fini d'Etats S et vérifiant la propriété Markov suivante :

$$\forall i \in S, \forall t > 1 [\ X_t = i \mid X_0, \dots, X_{t-1}] = P[\ X_t = i \mid X_{t=i} \mid X_{t-1}] \qquad (2)$$

Dans un tel processus, connu sous le nom de processus sans mémoire, prédire l'avenir à partir du présent ne nécessite pas la connaissance du passé. Les chaînes de Markov sont utilisées pour modéliser les systèmes stochastiques mais ne permettent pas à un agent d'intervenir et d'agir sur l'évolution du système.

Les processus de décision de Markov sont une extension des chaînes de Markov pour modéliser la dynamique d'un système sous le contrôle d'un agent par des actions et attribuer des récompenses aux transitions entre les états

Le processus de décision de Markov (MDP) fournit un formalisme de modèle et résout les problèmes de planification et d'apprentissage dans l'incertitude. Supposons que nous ayons d'une part, un système qui change au fil du temps comme un automate probabiliste et d'autre part, une sorte de contrôleur externe qui examine l'état actuel du système. Ce contrôleur peut utiliser un certain nombre d'actions pour objectif. Son objectif est de construire un plan optimal pour maximiser la récompense qu'il peut obtenir. L'évolution du système est considérée comme un processus de Markov, c.-à-d. l'évolution d'une séquence temporelle d'États distincts selon les probabilités de transition, uniquement basée sur les phénomènes de l'état précédent. Puterman [42] décrit ces modèles situés à la confluence des théories de la décision et des probabilités. Les PDM permettent de prendre des décisions séquentielles dans l'incertitude.

Le MDP est composé de deux éléments distincts : d'une part, un processus de Markov et un élément contrôleur de l'autre. À chaque étape, le contrôleur du système observe le processus et a

la capacité d'agir sur lui, en faisant une action éventuellement nulle. Un raisonnement basé sur le MDP peut mener au discours suivant : "Je sais que je suis dans une telle situation et si je fais une telle action, il y a tellement de chances de se retrouver dans cette nouvelle situation en obtenant une telle récompense". Le MDP repose sur quatre composantes, à savoir les états, les actions, les transitions et les récompenses.

1.5 Les réseaux de neurones

1.5.1 Historique

1890 : La loi de fonctionnement pour l'apprentissage est présentée par W. James.

1943 : Warren Mc Culloch et Walter Pitts proposent le premier modèle du neurone formel.

1949 : Règle de Donald Hebb qui décrit que si les neurones d'une synapse sont activés d'une façon synchrone et répétée, la force de connexion synaptique est croissante .

1958 : Création du premier réseau de neurones par Rosenblatt, le "perceptron", ce dernier est inspiré du système visuel. Il permet d'apprendre et d'identifier des formes simples et aussi de

Calculer certaines fonctions logiques. 1969 : Marvin Minsky et Seymour Papert montrent les limites du perceptron, surtout à l'incapacité de résoudre des problèmes non linéairement séparables (exemple du XOR). Période noire des Réseaux de neurones (≈ 15 ans) et beaucoup de déception chez les amateurs de l'intelligence artificielle.

1982 : John Hopfield réactive l'intérêt de l'utilisation de ce domaine grâce à sa découverte sur l'utilisation des réseaux récurrents (feed-back) après la première classe du perceptron .

1975 : Werbos propose l'idée d'une possible utilisation d'une rétropropagation du gradient.

1985-1986 : Le Cun et Parker (1985) continuent leurs recherches du pricipe de Werbos (1975), mais les travaux de Rumelhart (1986) furent le vrai départ de l'apprentissage des réseaux de neurones multicouche avec la méthode de rétropropagation du gradient

1.5.2 Le neurone biologique

Le neurone est une cellule composée d'un corps cellulaire et d'un noyau. Le corps cellulaire ramifie pour former ce que l'on nomme les dendrites. Celles-ci sont parfois si nombreuses que l'on parle alors de chevelure dendritique ou d'arborisation dendritique. C'est par les dendrites que l'information est acheminée de l'extérieur vers le soma, corps du neurone. L'information traitée par le neurone chemine ensuite le long de l'axone (unique) pour être transmise aux autres neurones. La transmission entre deux neurones n'est pas directe. En fait, il existe un espace intercellulaire de quelques dizaines d'Angströms (10-9m) entre l'axone du neurone afférent et les dendrites (on dit une dendrite) du neurone efférent. La jonction entre deux neurones est appelée la synapse (**Figure** 1.5.1) [9].

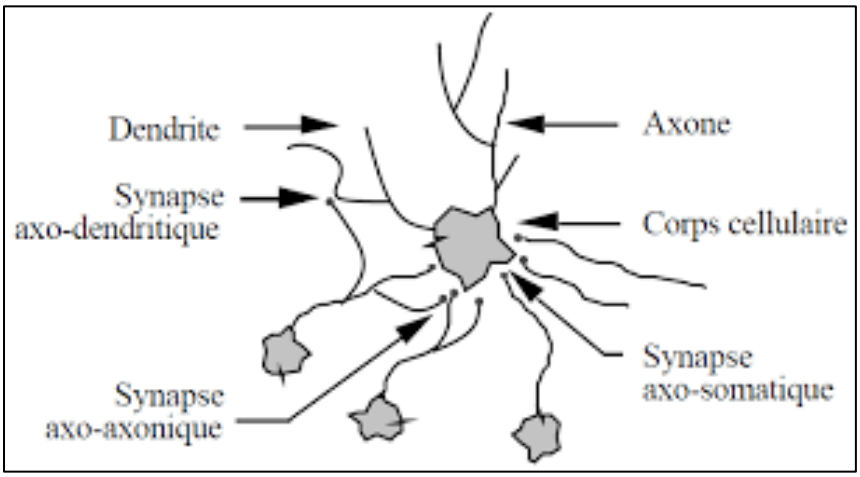


Figure 1.5.1 : Un neurone avec son arborisation dendritique

- **Corps Cellulaire** : Souvent appelé " Le Soma" effectue la somme des informations qui lui parviennent, il traite ces mêmes informations et renvoie le résultat sous forme de signaux électriques vers les autres neurones à travers l'axone.
- **Axone** : C'est un long prolongement fibreux du neurone, il fait la liaison entre les neurones et leurs conduit l'influx nerveux.
- **Dendrite** : Joue le rôle de capteur, car il reçoit les informations venant de l'extérieur vers le corps cellulaire. Certaines dendrites peuvent avoir un effet moteur favorisant la

transmission d'une information dans l'axone, d'autres au contraire ont un effet inhibiteur qui bloque la transmission de l'influx dans l'axone.

- **Synapses** : c'est une zone située entre deux neurones (cellules nerveuses) et assurant la transmission des informations de l'une à l'autre.
- **Influx Nerveux** : c'est une activité électrique qui parcourt les axones sous la forme d'une séquence de potentiel d'action, provocant la libération des neurotransmetteurs au niveau des synapses. Ces potentiels d'action sont générés par des échanges d'ions entre l'intérieur et l'extérieur des neurones. La vitesse de propagation de l'influx nerveux est de 100m/s

1.5.2.1 Fonctionnement du neurone biologique

Les dendrites reçoivent des autres neurones, des informations sous forme de signal électrique (influx nerveux), elles seront envoyées vers le corps cellulaire (Soma) et plus précisément le noyau qui fera la somme des influx arrivant aux dendrites tout en affectant des poids dits "poids synaptiques".

Le résultat transite le long de l'axone jusqu'aux terminaisons synaptiques ; et avec l'arrivé du signal, les vésicules synaptiques se fusionnent avec la membrane cellulaire pour libérer les neurotransmetteurs.

1.5.3 Le Réseau de neurone artificiel

D’une manière générale le RN formel est un Ensemble de neurones formels interconnectés permettant la résolution de problèmes complexes tels que la reconnaissance des formes ou le traitement du langage naturel, grâce à l'ajustement des coefficients de pondération dans une phase d'apprentissage

1.5.3.1 Du neurone biologique au neurone formel

Si nous essayons d'interpréter autrement le fonctionnement précédent du neurone biologique, on pourrait dire que le noyau fait la somme du produit des informations acquis par les dendrites et les poids attribués à ces mêmes informations, ensuite le corps cellulaire analyse le résultat, si ce dernier est supérieur à un seuil, il sera envoyé vers l'axone qui, soit ce subdivisera

pour alimenter d'autres neurones, soit à attaquer un élément moteur (muscle par exemple); Mc Culloch et Pitts [10], furent les premiers à mettre en œuvre les réseaux de neurones, ils ont résumé cette approche mathématique par l'équation du neurone formel [11] :

$$S = F\left(\sum_{i=1}^{n} w_i E_i\right) \qquad \textbf{(3)}$$

Où :

- E_i : Représente l'entrée i du neurone.
- w_i : Représente le poids attribué à l'entrée i.
- F : Représente la fonction d'activation (Seuil) du neurone

  {1 si F(x) > θ ; 0 sinon }, avec θ : Seuil.
- S : Sortie du neurone.

On pourra résumer une modélisation de tel neurone par le tableau 2 et la **Figure 1.5.2**, qui nous permettra de voir clairement le passage du neurone biologique vers le neurone formel.

Neurone biologique	**Neurone formel**
Synapses	Poids de connexions
Axones	Signal de sortie
Dendrite	Signal d'entrée
Somma	Fonction d'activation

Tableau 2: passage du neurone biologique vers le neurone formel.

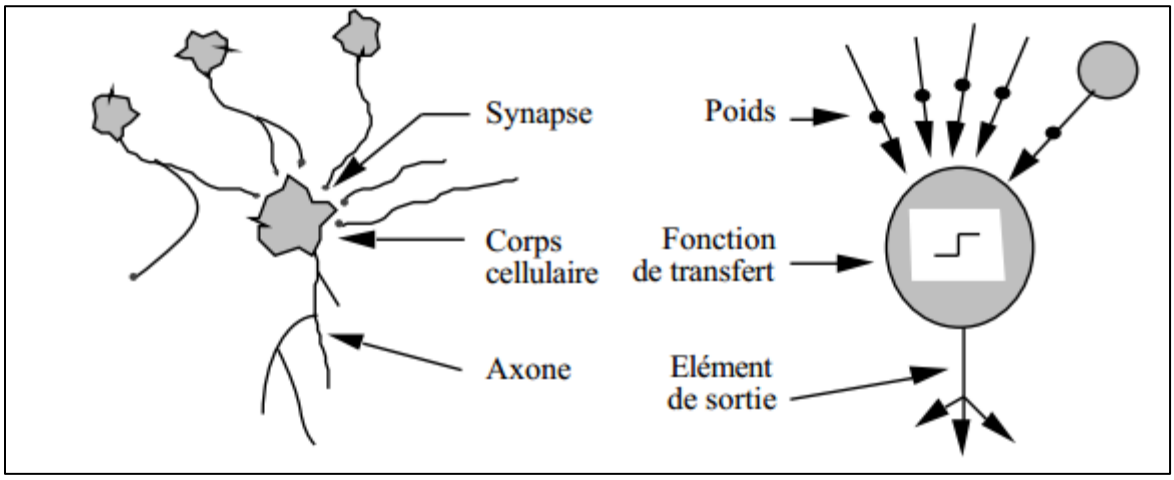


Figure 1.5.2 : Passage du neurone biologique vers le neurone formel

1.5.3.2 Caractéristiques des Réseaux de Neurones Artificiels

a) **Présentation** : Les réseaux de neurones artificiels sont donc des modèles inspirés du fonctionnement du cerveau humain, dans le but est de concevoir des machines capables de, non d'imiter, mais de se rapprocher au maximum du comportement de l'intelligence dont disposent les êtres humains, d'où la notion de "Intelligence Artificielle".

b) **Les Entrées** : Chaque neurone possède plusieurs entrées, on note E_i ($1 \leq i \leq n$) les entrées du réseau, avec n : nombre de neurones d'entrées ; et à chacune d'entre elle on affecte un poids ωi, on y ajoute un coefficient dit "biais" ω0 supposé lié à une entrée $E_0 = 1$, la i ème information qui parviendra au neurone est le produit ωi x E_i . Le neurone calculera la somme : ωi x $E_i n$ i=1, celle-ci sera le potentiel du neurone, la sortie sera générée par une fonction d'activation, qui sera très importante car elle déterminera par la suite le fonctionnement du réseau [12].

c) **Les fonctions d'activations** [16] :

La fonction d'activation (ou fonction de seuillage, ou encore fonction de transfert) sert à introduire une non linéarité dans le fonctionnement du neurone. Les fonctions de seuillage présentent généralement trois intervalles :

- En dessous du seuil, le neurone est non actif (souvent dans ce cas, sa sortie vaut 0 ou - 1).
- Aux alentours du seuil, une phase de transition.
- Au-dessus du seuil, le neurone est actif (souvent dans ce cas, sa sortie vaut 1) [16].

Dans sa première version, le neurone formel était implémenté avec une fonction à seuil, mais de nombreuses versions existent. Ainsi, le neurone de McCulloch et Pitts a été généralisé de différentes manières, en choisissant d'autres fonctions d'activations, comme les fonctions énumérées dans le tableau (3). Les trois fonctions les plus utilisées sont les fonctions « seuil » « linéaire » , « sigmoïdes ».

d) **La sortie :** peut-être utiliser en tant que résultat final, ou pour alimenter d'autres neurones.

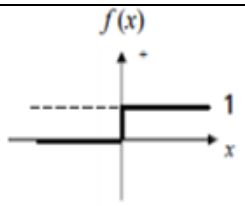
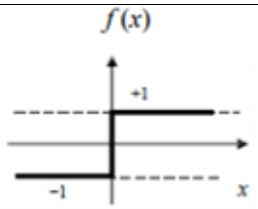
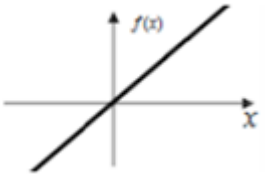
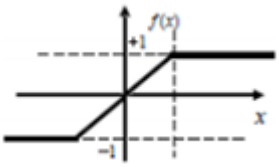
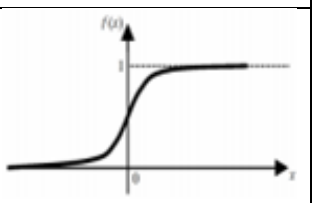
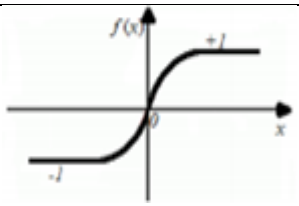
Catégories	Type	Equation	Allure
Seuil	Heaviside	$f(x) = \begin{cases} 0 \; si \; x < 0 \\ 1 \; si \; x \geq 0 \end{cases}$	
	Signe	$f(x) = \begin{cases} -1 \; si \; x \leq 0 \\ 1 \; si \; > 0 \end{cases}$	
Linéaire	Identité	$f(x) = \{x$	
	Saturé symétrique	$f(x) = \begin{cases} -1 \; si \; x \leq -1 \\ 1 \; si \; x \geq 1 \\ x \; sinon \end{cases}$	
Non linéaire	Sigmoïde	$f(x) = \frac{1}{1 + e^{-x}}$	
	Tangente hyperbolique	$f(x) = \frac{2}{1 + e^{-x}} - 1$	

Tableau 3: Les fonctions d'activations

1.5.4 L'apprentissage de réseaux de neurones

L'apprentissage est la caractéristique principale des réseaux de neurones ; c'est le processus d'adaptation des paramètres d'un système pour remplir au mieux la tache pour la quelle le réseau est destiné. Le type d'apprentissage est déterminé par la manière dont les paramètres sont adaptés, et il existe plusieurs méthodes et algorithmes pour adapter ces paramètres [14].

1.5.4.1 Apprentissage supervisé

Un Expert fournit au réseau des couples de données (entrée, sortie désirée correspondante). Les paramètres du réseau sont ajustés de manière à minimiser une certaine norme de l'erreur de sortie constituée par la différence entre la sortie réelle du réseau et la valeur désirée correspondante (fournie par l' Expert) [15].

1.5.4.2 Apprentissage par renforcement

Est une approche utilisée dans les problèmes de planification à travers le temps. Elle utilise deux réseaux : un réseau d'action et un réseau d'évaluation qui joue le rôle d'un superviseur et qui génère un signal d'avertissement à chaque fois que les actions prises sont mauvaises. Ce signal sert à entraîner le réseau d'action. Les poids du réseau d'évaluation sont altérés dans le sens de renforcer les bonnes actions et de sanctionner les mauvaises.

1.5.4.3 Apprentissage non supervisé

En absence de tout professeur, le réseau organise lui-même les formes d'entrée en classes de façon à minimiser un critère de performances. Ceci peut être fait, par exemple, en désignant un certain nombre de neurones gagnants dans une compétition d'activation ou en désignant un certain nombre de bassins d'attraction dans l'espace d'état [17]**.**

1.5.5 Architecture des Réseaux de neurones

Après avoir vu que l'on pouvait classifier les Réseaux de neurones selon le mode d'apprentissage, on verra maintenant la classification par architecture qui elle aussi se Subdivise en deux grandes familles : Les réseaux Feed-forward et les réseaux Feed-back.

1.5.5.1 Les réseaux Feed-forward

Ce sont des réseaux dans lesquels les informations se propagent successivement de couche en couche sans retour en arrière. On y trouve :

Le perceptron monocouche

C'est le réseau le plus simple en vue de l'architecture et au niveau de sa mise en œuvre, il se compose d'une couche d'entrée et une couche de sortie. Ce réseau est capable de résoudre des problèmes linéairement séparables (ex : fonction logique 'OU' ou 'AND') ; à noter que tous les neurones de la couche d'entrée sont liés à tous les neurones de sortie, comme le montre la figure suivante :

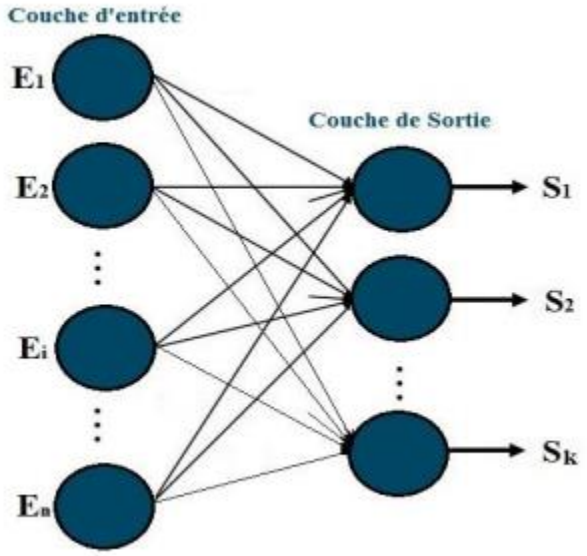


Figure 1.5.3 : Perceptron monocouche

$\boldsymbol{E_1}$: Entrée 1 de la couche d'entrée.

$\boldsymbol{E_i}$: Entrée i de la couche d'entrée.

$\boldsymbol{E_n}$: Entrée n de la couche d'entrée.

$\boldsymbol{S_1}$: Première sortie du réseau.

S_k : Sortie k du réseau (k étant le nombre de sortie).

Le perceptron multicouche (PMC)

Contient des couches cachées entre la couche d'entrée et celle de sortie, ce réseau résout des problèmes non linéairement séparables (ex : fonction logique 'XOR') ; et là aussi, chaque neurone des couches cachées est connecté à tous les neurones de la couche qui la précède et celle qui la suit. La figure ci-dessous représente un PMC :

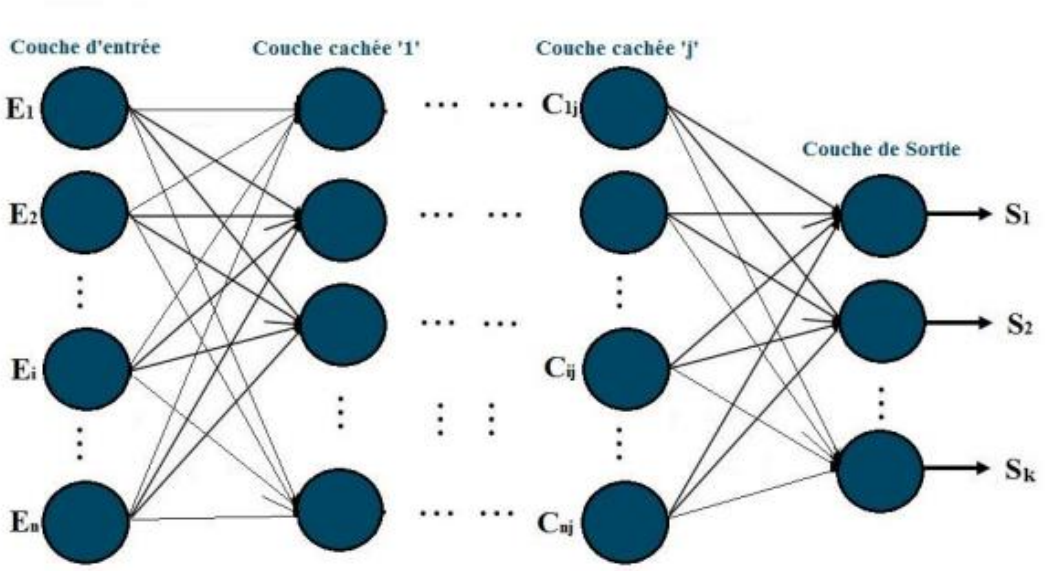


Figure 1.5.4: Perceptron multicouche

E_1 : Entrée 1 de la couche d'entrée.

E_i : Entrée i de la couche d'entrée.

E_n : Dernière entrée de la couche d'entrée.

C_{1j} : Premier neurone de la couche cachée j.

C_{ij} : Neurone i de la couche cachée j.

C_{nj}: Dernier neurone de la couche cachée j (m : nombre de neurones dans la couche cachée j).

S_1 : Première sortie du réseau.

S_k : Sortie k du réseau (k étant le nombre de sortie).

Le réseau à fonctions radiales (RBF)

Possède la même architecture que le PMC, la différence réside au niveau de la fonction d'activation des neurones, car les RBF utilisent les fonctions gaussiennes.

1.5.5.2 Les réseaux Feed-back

Appelés aussi "Réseaux récurrents", sont des réseaux dans lesquels le retour en arrière est possible. On y trouve :

Cartes auto-organisatrices de Kohonen

C'est un réseau à apprentissage non supervisée, il est sous forme d'une grille dont chaque nœud représente un neurone associé à un vecteur de poids ; la figure ci-dessous explique mieux son architecture :

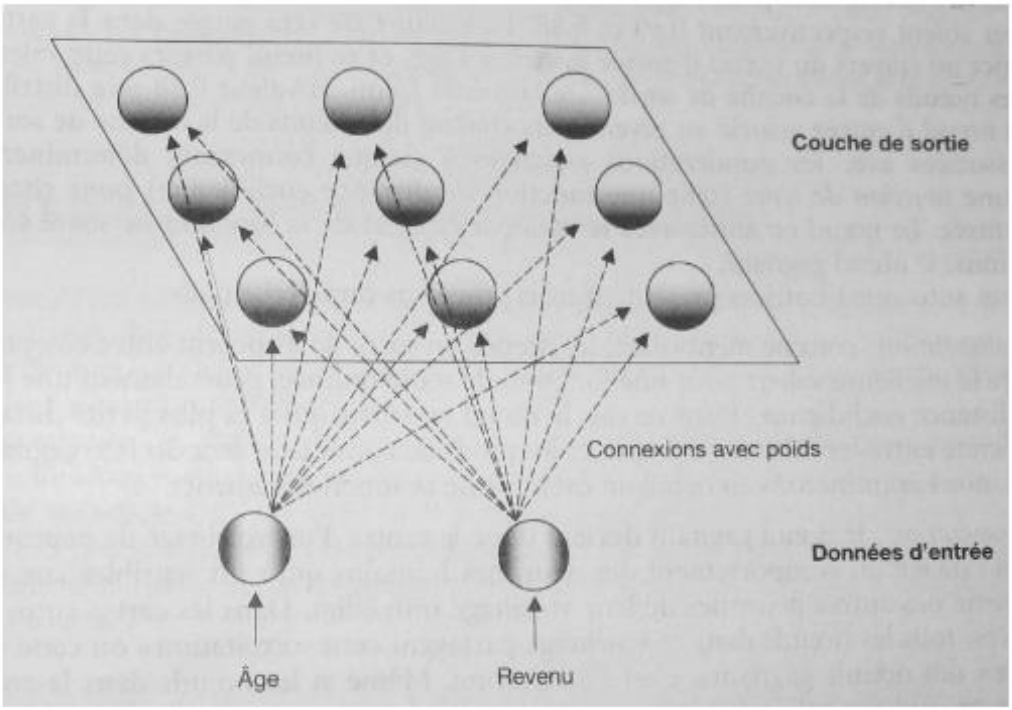


Figure 1.5.5 : Schéma d'une carte auto-organisatrice

Une carte auto-organisatrice est un procédé qui convertit un signal d'entrée complexe (Plusieurs variables par exemple) en une nouvelle variable catégorielle : c'est donc un procédé de classification (modélisation non-supervisée). Les SOM sont une généralisation de l'analyse en composantes principales. Elle fonctionne comme un réseau de neurones sans variable cible et avec plusieurs nœuds dans la couche de sortie. La carte structure les nœuds en sortie en classes de nœuds. Contrairement aux réseaux de neurones, SOM n'a pas de couches cachées. La couche de sortie contient plusieurs nœuds, représentés par un réseau rectangulaire

. Le nombre de nœuds dans la couche de sortie est défini arbitrairement par l'utilisateur. Il définit le nombre maximum de classes[18].

a) - Principes de fonctionnement

Fonction de score : Les valeurs des nœuds de la couche d'entrée (valeurs normalisées des variables prises en compte par le modèle) sont distribuées dans les nœuds de la couche de sortie après transformation en fonction des pondérations du réseau : on parle de « fonction de score ». Cette fonction est généralement une fonction de distance euclidienne. Le nœud de sortie qui a le meilleur résultat (on dit le «meilleur score ») est le « nœud gagnant » : il Reçoit l'individu en question. Le meilleur score c'est la plus petite distance entre les poids de connexion et les données d'entrée [18].

b) -Principe de fonctionnement : liaison de voisinage des nœuds de la couche de sorties

Comme tous les réseaux de neurones, les nœuds d'une même couche, notamment les nœuds de la couche de sortie, ne sont pas liés les uns aux autres. Cependant, dans le cas de données similaires, les poids des nœuds voisins du nœud gagnant sont ajustés pour favoriser leur victoire. C'est ce qu'on appelle la coopération et l'adaptation des nœuds de la couche de sortie. S'adapter, c'est apprendre. Comme tous les réseaux de neurones, les nœuds de la même couche, en particulier les nœuds de la couche de sortie, n'ont aucun lien entre eux

Réseaux de Hopfield

Ce sont des réseaux entièrement connectés. Dans ce type de réseau, chaque neurone est connecté à chaque autre neurone et il n'y a aucune différenciation entre les neurones d'entrée et de sortie. Ils sont capables de trouver un objet stocké en fonction de représentations partielles ou

bruitées. L'application principale des réseaux de Hopfield est l'entrepôt de connaissances mais aussi la résolution de problèmes d'optimisation. Le mode d'apprentissage utilisé est là aussi le mode non-supervisé.

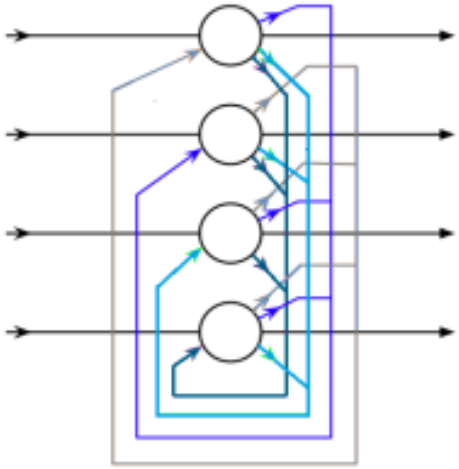


Figure 1.5.6 : Architecture d'un Réseau de Hopfield

1.5.6 Avantages et inconvénients des réseaux de neurones artificiels

On dénombre plusieurs avantages du réseau de neurone artificiel : sa capacité à apprendre, à modéliser les systèmes, de plus ils sont adaptés au traitement des données non linéaires, et prennent compte de variables qualitatives en attribuant des données binaires, mais aussi la possibilité de travailler avec des données bruités ou incomplètes, et bien d'autres encore. Cependant il présente quelques inconvénients telle que l'apparition du problème de sur-apprentissage, l'apprentissage n'est pas toujours évident, la non interprétation des poids ou encore l'indétermination directe du nombre de couches cachées et des neurones.

1.6 Conclusion

Dans ce chapitre nous avons présenté les concepts de base des systèmes multi-agents et nous avons aussi exposé l'un des puissants outils de l'IA, à savoir les réseaux de neurones, qu'on a étudié

d'une manière globale, et on a montré leurs historiques, leurs différents modes et architectures dont ils disposent, on a aussi vu leur classification selon l'architecture ou le mode d'apprentissage, ainsi que leurs avantages et inconvénients.

Dans le chapitre suivant en va présentera notre nouvel algorithme OAGRMF base sur les réseaux de neurones (OAGRMF-NN) .

Chapitre 02
Proposition d'un un nouveau modèle organisationnel base sur les Réseaux de Neurones

Contenue

Chapitre 02 .....
2.1 Introduction : 22
2.2 Travail connexe 22
2.3 Description du problème de Multi-agent Poursuite 24
2.3.1 Poursuivants : 24
2.3.2 Évader 26
2.4 Structure de l'OAGRMF-NN 26
2.5 Extraction des caractéristiques des poursuivants 29
2.6 La couche SOM pour la génération de groupe 31
2.7 Algorithme de formation d'OAGRMF-NN 34
2.7.1 Fonction d'attractivité du groupe 34
2.7.2 Rétropropagation basée sur CEF 35
2.8 Conclusion 37

2.1 Introduction :

La résolution de certains problèmes d'optimisation et de classification restent un souci considérable chez les informaticiens, c'est pour cela que plusieurs algorithmes ont fait surface pour pallier à ce genre de problèmes, et parmi ces algorithmes, on trouve le **Réseau De Neurones artificiel** qui en a appliqué. Le principe des Réseau de neurones se base sur une approche mathématique du fonctionnement du cerveau humain.

Dans ce deuxième chapitre, nous allons introduire notre nouvel algorithme nommé **OAGRMF-NN**. Au début, nous allons discuter quelques travaux reliés à le nôtre concernent la formation de coalition et les système multi agent. Aussi, nous allons expliquer le problème **Multi-agent Poursuite**.

Ce chapitre contient aussi l'application du Réseau de nuerons qui va permettre aux groupes d'agents de regrouper d'une manière automatique et intelligents.

2.2 Travail connexe

Beaucoup de travail a été fait et appliqué pour la formation de coalition de système multi-agents tels que la bionique, la sociologie et la théorie des jeux. En gros, ces méthodes peuvent être divisées en deux types : basées sur le comportement et basées sur le plan. Les systèmes comportementaux sont inspirés par certains systèmes biologiques comme l'essaim, la colonie de fourmis et d'autres animaux sociaux. Ce système multi-agents est composé d'un grand nombre de robots, dont chacun interagit avec l'environnement et d'autres agents obéissant à des règles de comportement simples. Bien que chacun d'eux n'ait que des capacités et des ressources limitées, tout comme les abeilles en essaim, des règles de comportement bien conçues permettent à ces systèmes d'accomplir les tâches spécifiques. Les algorithmes de présentation de cette méthode sont abordés dans ALLIANCE **[19],** Broadcast of Local Eligibility (BLE) **[20],** l'optimisation des essaims **[21]** et le recrutement émotionnel **[27].** La recherche sur la coordination distribuée des agents dynamiques en réseau a également donné des résultats et a été appliquée à des MAS pratiques. Denis Budaev et Konstantin Amelin ont proposé un système multi-agents centré sur le

réseau pour la programmation des tâches en temps réel pour le groupe d'UAV **[22].** La littérature **[23]** a proposé un

troupeau décentralisé de plusieurs coptères qui effectue des vols extérieurs qui naviguent eux-mêmes en fonction de la informations dynamiques reçues d'autres agents du voisinage.

Certaines approches basées sur la théorie des jeux peuvent également être considérées comme de Ce type, la littérature **[26]** est venu avec une formation d'équipe décentralisée basée sur l'élimination itérative des stratégies dominées. Adel. Ghazikhani a proposé un algorithme de formation de coalition basé sur la valeur de Shapley **[24].**

Une approche pour calculer la stratégie optimale pour un poursuivant qui maximise la possibilité de capturer un évaseur est présenté dans la littérature **[40].** la littérature **[26]** a proposé une solution utilisant une planification stratégique multi-agents basée sur théorie des jeux qui font le meilleur de tout plan de voyage partagé trouvé pour les services de transport.

Il est difficile d'analyser les méthodes basées sur le comportement car la précision du comportement macro ne peut pas être garantie efficacement. En d'autres termes, il est difficile de conduire le comportement de l'agent directement et les utilisateurs ne peuvent que définir les règles simples auxquelles les agents doivent obéir. Au contraire, la méthode explicite basée sur le plan est conforme à l'objectif et a une mise en œuvre relativement facile. Il est plus efficace dans l'exécution des tâches spécifiques en raison du plan conçu. Baofu Fang et LuChen ont proposé un mécanisme d'enchères en deux étapes pour recruter des équipes en fonction du facteur de coopération émotionnelle **[28].**

A.Rauniyar, PK. Muhuri a appliqué des immigrants basés sur l'algorithme génétique des immigrants aléatoires et l'élitisme à la formation de coalition optimale en tenant compte des variantes dynamiques **[32].** Le modèle de protection de la vie privée est proposé pour assurer la compétitivité et l'autonomie des agents lorsque des méthodes fondées sur des plans sont appliquées à la formation de coalitions **[29].** Vaibhav Katewa et Fabio Pasqualetti ont conçu un mécanisme d'ajout de bruit selon le cadre de confidentialité différentiel classique mis en œuvre par les agents **[31].** Littérature **[30]** propose une classe d'algorithmes itératifs pour résoudre le problème de

l'optimisation distribuée privée afin d'atteindre la confidentialité différentielle et la convergence vers une valeur commune.

En outre, il y a une nouvelle branche ces années appelée graphique de synergie qui utilise des graphiques pour représenter la compatibilité basée sur les tâches et la capacité de travail d'équipe de tous les agents. Il applique la théorie des graphes pour trouver la formation optimale de l'équipe **[33].** Ensuite, ils combinent la théorie de l'agent apprenant et des graphiques de synergie pour promouvoir les instances d'apprentissage de l'agent ; au lieu de considérer leurs capacités qui sont statiques **[34].** Il sera difficile pour la méthode fondée sur le plan de déterminer les indicateurs de capacité **[35]** qui représentent la mesure dans laquelle chaque attribut contribue à la tâche lorsque l'échelle de la MAS est trop grande. L'objectif est donc de combler cette lacune.

2.3 Description du problème de Multi-agent Poursuite

2.3.1 Poursuivants :

L'ensemble des poursuivants est dénoté par :

$P = \{p_1, p_2 \dots p_n\}$

Chaque agent devrait être décrit par deux paramètres de capacité supplémentaires afin de déterminer la fonction d'adhésion du modèle AGR :

Self-conference Degree

Il existe de nombreuses façons qui peuvent représenter un poursuivant correctement, mais l'une des représentations les plus intuitives est le taux de réussite des tâches. Ce paramètre peut être considéré comme degré d'auto-référence qui peut être calculé comme suit :

$\forall Conf \in [\lambda, 1] : conf = \max\left(\lambda, \frac{C_s}{C_t}\right)$ **(4)**

C_s : Est le nombre de tâches que l'agent a terminées avec succès.

C_t : Est le nombre de tâches auxquelles l'agent a participé.

Λ : Est pour contrôler l'ampleur du changement de la gamme Conf comme une fonction linéaire rectifiée.

Credit

Crédit mesure la capacité de l'agent à travailler avec d'autres agents. Si ce crédit est faible, cela signifie que l'agent pourrait ne pas être en mesure d'exécuter correctement les services demandés par les autres agents. Parce que l'échec de la tâche affectera également le rendement des autres agents. Le crédit d'un agent est indiqué comme suit :

$$\forall Credit \in [0\,,1] : credit = \min\left(1\,, \frac{1-C_b}{C_t - C_s}\right) \quad \textbf{(5)}$$

C_b : est le nombre de tâches que l'agent doit abandonner.

Distance

La distance entre le poursuivant et l'évadé dans l'environnement est un critère crucial pour la poursuite-évasion. La distance Dist entre le poursuivant P et E est calculée comme suit :

$$Dist_{PE} = \sqrt{(CC_{Pi} - CC_{Ei})^2 + \left(CC_{Pj} - CC_{Ej}\right)^2} \quad \textbf{(6)}$$

$CC_{Pi}\,, CC_{Pj}$: sont les coordonnées cartésiennes du poursuivant.

CC_{Ei}, CC_{Ej} : sont les coordonnées cartésiennes de l'évader.

Récompenses (R)

Les PDM visent à automatiser le processus de prise de décision, aussi, nous devons être en mesure d'avoir une mesure de l'utilité des actions possibles. C'est pourquoi nous spécifions les valeurs gagnées correspondant au choix d'une telle action dans un tel État. Une récompense, dite immédiate, est perçue pour chaque transition, mais elle peut être nulle si l'État concerné contient un obstacle.

La récompense de chaque cellule est inversement proportionnelle à la distance entre la cellule concernée et la cellule contenant la cible poursuivie. La dispersion entre le poursuivant et l'acquéreur de l'Evader dans l'environnement implique un impact considérable sur le traitement de la poursuite. Il est calculé à l'aide des coordonnées cartésiennes :

$$Dist(s,s') = \sqrt{(CO_{xi} - CO_{s'x})^2 + \left(CO_{sy} - CC_{s'y}\right)^2} \qquad \textbf{(7)}$$

$$R(s) = \mathfrak{m} - Dist(s,s')$$

(CO_x, CO_y) : Coordonnées cartésiennes de la cellule contenant l'agent dans.

$\mathfrak{m}$:la récompense maximale ou la récompense de la cellule contenant la cible.

s: l'état contenant le poursuivant.

s': l'état contenant l'évadé.

2.3.2 Évader

L'ensemble des évadés est dénoté par :

$$E = \{e_1, e_2 \dots e_n\}$$

Chaque évadé a sa propre difficulté de poursuite, ce qui signifie combien de poursuivants sont nécessaires pour attraper cet évadé, l'ensemble de la difficulté de poursuite est dénoté par

$$D = \{d_1, d_2 \dots d_n\}$$

Le poursuivant qui attrape l'évadé obtiendra la récompense qui est égale à d_e

2.4 Structure de l'OAGRMF-NN

Une nouvelle variable est nécessaire pour représenter l'aptitude d'un agent à exécuter une tâche dans un groupe. La fonction d'adhésion au groupe d'agents (AGRMF) est une extension du modèle agent-group-rôle. Contrairement à d'autres institutions, cette approche permet aux agents de considérer le groupe comme flou. Par conséquent, chaque groupe peut être géré au moyen d'une fonction d'adhésion. Au lieu de déterminer explicitement si un agent spécial est censé ou non appartenir à l'un des groupes, l'AGRMF admet les degrés d'adhésion à un groupe donné. Ce changement offrira plus d'espace pour utiliser différents algorithmes afin d'optimiser le résultat pour l'organisation et plus de flexibilité pour la réorganisation par rapport au modèle AGR ordinaire.

Par rapport au modèle organisationnel AGRMF, **OAGRMF** est plus adéquate dans le cas où le nombre d'agents existants est inférieur à l'exigence. Par ailleurs, ce modèle élimine le problème lié aux externalités négatives émergeant des regroupements d'agents via le principe du chevauchement. Dans le modèle OAGRMF, un groupe signifie une seule tâche.

Lorsque la tâche est terminée, le groupe doit être renvoyé, donc (μ_t(p)) = μ_g (p) ici **t** est la tâche que le groupe exécute.

Dans notre algorithme, OAGRMF de P au groupe g μ_g (p) déterminé par $Credit_p, Conf_p, Dist_p$ distpt est utilisé pour représenter l'aptitude du poursuivant p à rejoindre le groupe g et exécuter la

tâche t comme Eq. 8. Les séquences de base de l'algorithme de génération de coalition basé sur le modèle OAGRMF qui est proposé par la littérature [8] sont montrées comme algorithme 1

$$\mu_g(\mathrm{P}) = \frac{Coef_1^t * Pos_{pt} + Coef_2^t * Conf_p + Coef_3^t * Credit_p}{\sum_{i=1}^{3} Coef_i^t} \qquad \textbf{(8)}$$

L'idée principale de cet algorithme est de sélectionner le poursuivant avec le degré maximal d'adhésion de la cible pour rejoindre le groupe.

Le poursuivant sera attribué à l'agent après vérification de ces deux conditions :

- Si l'agent appartient au groupe concerné avec le degré de membership plus élevé. Savoir cela représente le nombre de poursuivants nécessaires pour capturer l'évadé.
- La deuxième condition concerne le degré de chevauchement des groupes en train de déterminer combien de poursuivants appartiennent à plus d'un groupe de poursuite En fait, si ce degré dépasse 50 % et l'agent concerné appartient déjà à un ou plus autres groupes de poursuite alors le rôle ne sera pas attribué

Algorithm 1

OAGRMF

Inputs: Agents

positionOutputs: Evaders capture

x←number of evaders detected;

y←number of pursuers;

Create x Groups;

Broadcast (offer);

For each-evader **do**

While (Groupk/=Complete) **do**

Repeat

T1[i]←(Group, Ai);

T2[I]←Ai;

i←i+1;

Until I=y

Repeat

t←index (max(T1));

Agent-Selected←T2[];

If overlapping-degree () < 50% **then** Role-attribution(Agent-Selected);

Delete (max(T1));

Delete (T2[]);

j←j+1;

nbr-role (Agent-Selected)++;

If nbr-role (Agent-Selected) >1 **then** overlapping-degree () ++;

End **if**;

Until j=Rek

k←k+1;

end **For**;

While (Time > 0) **do**

If obstacle() = true **then** RBA();

Else Move-to-target();

End **if**;

If (Capture() = true) **then** Update (Rewards);

End **while**;

If (Time = 0 and Capture() = false) **then** Update (Fines);

End.

Après la détection des évadés existants dans l'environnement, un groupe de poursuite concernant chaque évadé sera créé. Comme expliqué en [35], chaque groupe de recherche est doté d'une fonction d'adhésion qui détermine le degré d'adhésion de chaque agent par rapport au groupe concerné selon l'équation 8.

2.5 Extraction des caractéristiques des poursuivants

Extraction de fonctionnalités pour les poursuivants d'OAGRMF-NN. Le Pursuer i peut être représenté par un vecteur de caractéristique $x_i = \{Credit_i, Conf_i, Dist_{i1}, Dist_{i2}..., Dist_{im}\}$ où le $Dist_{im}$signifie la distance entre pursuer i et evader m. Par Eq. 8, nous pouvons voir que le numérateur de μt(p) peut être exprimé comme l'excitation du vecteur d'entrée $[Conf^p, Credit^p, Dist^p]$ à un neurone dont le vecteur de poids est $[Coef_1^t, Coef_2^t, Coef_3^t]$.

Chaque neurone t peut être considéré comme un filtre qui sélectionne l'agent approprié à la tâche t. Par conséquent, le vecteur de poids du neurone t, $w_t = [Coef_1^t, Coef_2^t, Coef_3^t]$ a le même objectif que le vecteur principal d'indicateur de capacité de la tâche t.

Tous les filtres composent d'une couche de filtre, c'est-à-dire d'une couche de neurones. La couche est une couche convolutionnelle au lieu de la couche de connexion complète parce qu'un neurone de cette couche n'est pas connecté avec tous les neurones.

Dénominateur de la partie droite de la formule (8) $\sum_{i=1}^{3} Coef_i$ est conçu pour normaliser le vecteur $w_i = [Coef_1^i, Coef_2^i, Coef_3^i]$.

Le processus de régularisation, la limitation des poids initiaux, et l'ajustement de la fonction d'activation peuvent faire la couche de convolution atteindre cet objectif. Pour résumer, la sortie de la couche convolution peut représenter ut(p) pour chaque poursuivant p et tâche t de sorte que la mise à jour des poids de cette couche représente également le changement de la cognition du système aux attributs de la tâche selon le processus de poursuite.

La couche d'entrée est suivie par cette couche à convolution. Ensuite, la couche cachée peut améliorer la capacité d'approximation du réseau neural [36]. La structure de cette partie est montrée dans la ***F*igure 2.5** La couche d'extraction de caractéristique produit finalement la caractéristique de chaque poursuivant qui peut être appelée comme OAGRMF-caractéristique.

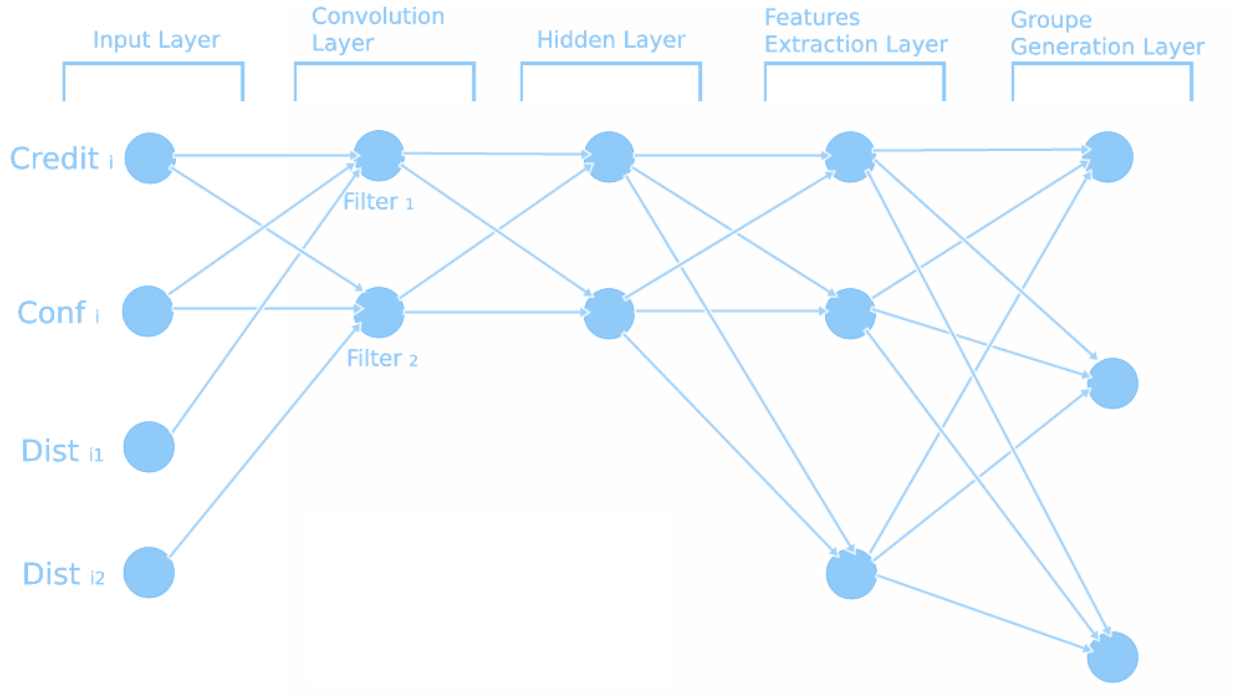


Figure 2.5: Dispositif d'extraction pour les poursuivants

2.6 La couche SOM pour la génération de groupe

L'extraction de fonctionnalités n'est pas suffisante pour la génération d'équipes. OAGRMF-NN devrait apprendre à assigner des poursuivants aux groupes appropriés pour utiliser pleinement chaque agent. Évidemment, les poursuivants ayant des caractéristiques OAGRMF similaires sont censés être assignés à un groupe avec une grande probabilité. Par conséquent, une couche Self Organized Map (SOM) est ajoutée après la partie d'extraction de la fonction. La SOM est l'une des modèles de réseaux qui constituent un type d'approche d'apprentissage non supervisé [37].

La SOM est largement utilisée pour analyser les caractéristiques intrinsèques des données [38, 39]. La couche SOM est une couche unique et entièrement connectée. Ensemble de formation fournit

X = $\{X^1, X^2, X^3 \dots X^P\}$, pour chaque x dans l'ensemble d'entraînement et le réseau produit un noeud gagnant dont la réponse est le maximum parmi tous les nœuds comme Eq. 8 représente.

$$y = argmax_{k=1.2.3\dots m}(\overrightarrow{W_k^T} * \overrightarrow{X}) \qquad \textbf{(9)}$$

L'algorithme d'apprentissage de SOM est l'apprentissage compétitif qui va promouvoir le vecteur de poids connecté au nœud gagnant de X pour représenter le plus caractéristique du X dans la couche entière. Les mises à jour des poids représentent l'Eq. 9.

$$\begin{cases} w_{ji}(t+1) = w_{ji}(t) + \alpha\left(x_i(t) - w_{ji}(t)\right) & y_i = winner(x) \\ w_{ji}(t+1) = w_{ji}(t) & y_i \neq winner(x) \end{cases} \qquad \textbf{(10)}$$

Algorithm 2

Groupe generation

```
FillInputVect
ComputeFeatures
SOMLayerTraining
Show SOMOutput_Winner

pg1 ← empty list
pg2 ← empty list
pgOv ← empty list

for ( g ← 0 ; g < 3 ; g++ )
    s ← 0
    foreach agent in AgentList
        if (g == SOMOutput_Winner)
         s ← s + FeaturesVect
        end
    end
    I ← getminIndex s

    foreach Agent in AgentList
        if (g == SOMOutput_Winner)
         if (I == 0)
            addtolist(Agent,pg1)
         else if (I == 1)
            addtolist(Agent,pg2)
         else if (I == 2)
            addtolist(Agent,pgOv)
         end
        end
      end

    while (sizeof pg1 > 2)
        ag ← find agent in pg1 with highest distance from evader1
        remove ag from pg1
          add ag to pgOv
    end
    while (sizeof pg2 > 2)
        ag ← find agent in pg2 with highest distance from evader2
        remove ag from pg2
```

```
            add ag to pgOv
    end
    while (sizeof pg1 < 2)
        ag ← find agent pg Ov with lowest distance from evader1
        add ag to pg1
        remove ag from pgOv
    end
    while (sizeof pg2 < 2)
        ag ← find agent in pgOv with lowest distance from evader2
        add ag to pg1
        remove ag from pgOv
    end

    while (sizeof pg1 < 4 and sizeof pg1 < 4)
        ag ← find agent in pgOv with lowest distance from evader1 or evader2
        add ag to pg1
        add ag to pg2
        remove ag from pgOv
    end
End.
```

Les étapes de l'algorithme de formation de la SOM est inspiré par les vecteurs de fonction OAGRMF et d'obtenir le résultat final du groupe sont présentés comme **algorithme 2**.

Les étapes de l'algorithme sont expliquées de la manière suivante : premièrement initialiser les paramètres et SOM, deuxièmes le processus d'apprentissage de la SOM

La construction complète du réseau de neurones pour la formation de coalitions est montrée ***F*igure 2.6** L'erreur est passée de la couche de sortie à la couche cachée. Lorsque le réseau neuronal converge, le vecteur de poids des neurones i aide représente le centre des caractéristiques OAGRMF des poursuivants dont le neurone gagnant est i. L'algorithme de première partie c.-à-d. la formation de SOM et l'autre partie c.-à-d. obtenir le résultat final du groupe sont tous deux représentés dans l'**algorithme 2**.

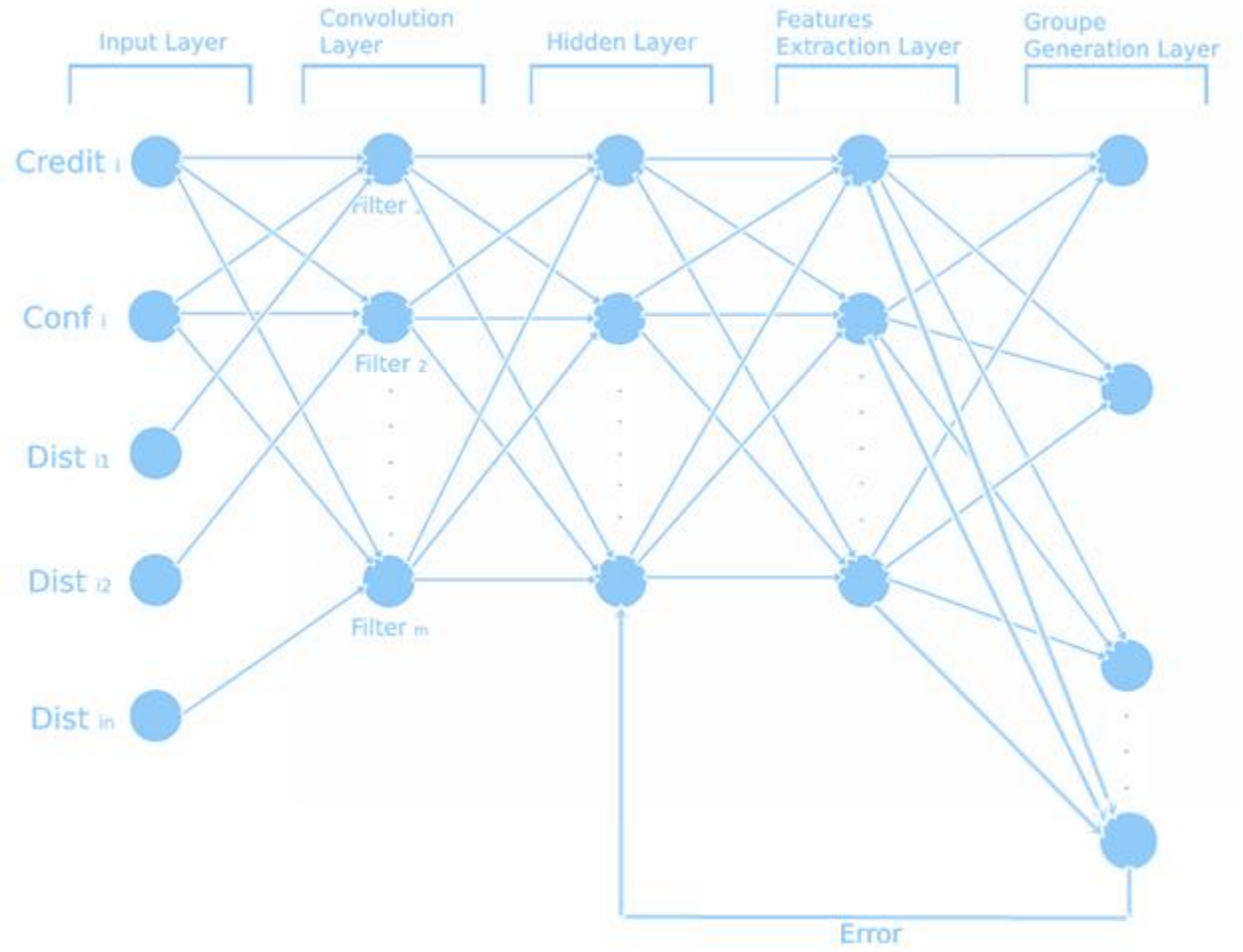


Figure 2.6 : Réseau de neurones pour la formation d'une coalition

2.7 Algorithme de formation d'OAGRMF-NN

2.7.1 Fonction d'attractivité du groupe

La fonction d'attractivité du groupe (GAF) est proposée pour évaluer l'exactitude de la formation de coalition obtenue par OAGRMF-NN. Assigner un poursuivant au groupe qui est attiré par lui et il sera considéré comme une décision correcte.

$GAF_g(p')$ peut être exprimé comme le rapport entre la volonté de p d'accomplir la tâche qui est exécutée par le groupe g et la difficulté existante de cela. La volonté est le taux de réussite

$U_t(p')$ qui est calculé en fonction du vecteur de capacité principal de p'. La difficulté existante peut être représentée par la distance moyenne entre l'évadé et les poursuivants qui sont responsables de l'attraper. Le $GAF_g(p')$ est indiqué dans Eq. 11

$$\mathrm{GAFg}(p') = \frac{\mu g(p') * d}{\sum_{p \epsilon g} dis(p, E)} \quad (11)$$

Il est bénéfique pour le SMA que chaque poursuivant quitte le groupe d'origine pour poursuivre plus d'intérêts [40], c'est aussi le moyen de réorganisation. Chaque agent doit prendre une décision entre les intérêts du groupe d'origine $GAF_g(p)$ et les intérêts des autres groupes $\frac{\sum_{g \epsilon \{G-g\}}(GAF(p))}{|G-g|}$ Les résultats sont également déterminés par le facteur d'inertie Cp qui représente la nature des poursuivants à garder dans le groupe d'origine.

Ce facteur entre 0 et 1 peut être utilisé pour contrôler la vitesse de la réorganisation. La fonction d'évaluation de la coalition (CEF) est présentée dans l'Eq. 12.

$$\begin{cases} CEF(p) = 1\ C_p * GAF_g(p) > (1 - C_p) \frac{\sum_{g \epsilon \{G-g\}}(GAF(p))}{|G-g|} \\ CEF(p) = 0\ C_p * GAF_g(p) < (1 - C_p) \frac{\sum_{g \epsilon \{G-g\}}(GAF(p))}{|G-g|} \end{cases} \quad (12)$$

2.7.2 Rétropropagation basée sur CEF

L'algorithme d'apprentissage de l'extraction de la fonction OAGRMF met à jour les poids de NN dont le but est de promouvoir NN et de mieux comprendre les propriétés de toutes les tâches, ce qui permet d'assigner les tâches de manière plus optimale. L'algorithme d'apprentissage est un nouvel algorithme de rétropropagation [44] basé sur le CEF. L'étiquette de chaque échantillon d'entraînement du poursuivant p peut être définie selon le CEF'(p) qui est calculé dans la dernière itération. CEF (p) = 0 indique que p ne satisfait pas à la tâche assignée par OAGRMF-NN. Ainsi, l'étiquette du vecteur d'entraînement x^p devrait être le vecteur caractéristique du groupe g'qui est le plus attiré par lui. En d'autres termes, le poids vecteur w_g du neurone représente g'. Le résultat de la mise à jour des poids est que le vecteur caractéristique OAGRMF de p est plus proche de $w_{g'}$

Au contraire, lorsque le CEF(p) = 1, l'étiquette x^p est le vecteur de poids du gagnant neurone w_g. La fonction de coût est montrée dans Eq. 13.

$$E(\vec{w}) = \frac{1}{2}\left(CEF(p)\sum_{d\epsilon D}\sum_{k\epsilon output}^{q}(win_k(d) - o_k(d))\right)^2 + (1 - CEF(p))\left(\sum_{d\epsilon D}\sum_{k\epsilon output}^{q}(win'_k(d) - o_k(d))\right)^2 \quad (13)$$

d est le vecteur caractéristique de pursuer p, $o_k(d)$ est le vecteur de sortie de la partie d'extraction de la fonction AGRMF ainsi que le vecteur d'entrée de la couche SOM. $w_o(k)$ est le poids du neurone gagnant dans la couche SOM, $w'_o(k)$ est le poids du neurone du groupe g′qui est $max_{arg}GAF(p)$, CEF′(p) peut être traité comme une constante parce qu'il fait partie de l'étiquette et ne peut pas être modifié à l'itération actuelle. E ($\vec{w}$) est une fonction dérivée continue qui permet d'utiliser l'algorithme de rétro-propagation pour mettre à jour les poids de la partie extraction d'entités d'OAGRMF-NN sous forme d'Eq. 14

$$\delta(w) = -\eta\nabla_\omega \frac{1}{2}\left(CEF(p)\sum_{d\epsilon D}\sum_{k\epsilon output}^{q}(win_k(d) - o_k(d))\right)^2 + (1 - CEF(p))\left(\sum_{d\epsilon D}\sum_{k\epsilon output}^{q}(win'_k(d) - o_k(d))\right)^2 \quad (14)$$

2.8 Conclusion

Après avoir bien détaillé le problème ainsi que notre solution proposée dans ce chapitre, nous allons pouvoir l'implémenter en comparaison avec des travaux connexes dans le chapitre suivant via l'utilisation de la plateforme Netlogo [41].

Chapitre 03
Expérimental

Contenue

Chapitre 03 .....
3.1 Introduction 38
3.2 Plateformes multi-agents 38
3.2.1 MACE 38
3.2.2 ZEUS 39
3.2.3 MADKIT 40
3.2.4 SWARM 40
3.3 Plateforme de simulation NetLogo 41
3.4 Résultats de la simulation 43
3.5 Conclusion 49

3.1 Introduction

Dans ce chapitre, nous allons lever l'ambiguïté sur quelques plateformes multi-agent telles que MACE, ZEUS, Madkit et SWARM. Aussi nous allons argumenter notre choix concernant la sélection de la plateforme Netlogo pour effectuer nos simulations. Durant ces simulations, nous allons étudier le temps d'exécution ainsi que le nombre d'itération utilise pour capturer un évader. Tous ces paramètres seront étudiés en comparaison avec des travails connexe de Formation de coalition comme OAGRMF, AGRMF, AGRMF-NN.

3.2 Plateformes multi-agents

Il existe plusieurs types des plateformes multi-agents : « Une liste assez complète de ces plates-formes se trouve à l'adresse « 170H170H191H170H170Hhttp://www.agentlink.org/ ». Parmi les plates-formes fournies comme logiciels libres, il y a quelques-unes plus connues pour avoir été utilisées dans le développement de plusieurs applications : JADE, MACE, ZEUS, et MADKIT … SWORM ... Il faut noter que cette liste n'est pas unique, et qu'il y a aussi d'autres platesformes qui ont été utilisées avec beaucoup de succès pour bâtir diverses applications » [43].

3.2.1 MACE

MACE (Gasser e.a., 1987) est le premier environnement de conception et d'expérimentation de différentes architectures d'agents dans divers domaines d'application. Dans **MACE**, un agent est un objet actif qui communique par envoi de messages. Les agents existent dans un environnement qui regroupe tous les autres agents et toutes les autres entités du système. Un agent peut effectuer trois types d'actions : changer son état interne, envoyer des messages aux autres agents et envoyer des requêtes au noyau **MACE** pour contrôler les événements internes. Chaque agent est doté d'un moteur qui représente la partie active de l'agent. Ce moteur détermine l'activité de l'agent et la façon dont les messages sont interprétés. **MACE** a été utilisé pour développer des simulations d'applications distribuées.

3.2.2 ZEUS

ZEUS (Nwama e.a., 1999) est une plate-forme multi-agents conçue et réalisée par British Telecom (Agent Research Programme of BT Intelligent Research Laboratory) pour développer des applications collaboratives. **ZEUS** est écrit dans le langage Java et il est fondé sur les travaux de la FIPA. L'architecture des agents **ZEUS** est similaire à la majorité des agents collaboratifs. Elle regroupe principalement les composantes suivantes :

- Une boîte aux lettres et un gestionnaire de messages qui analyse les messages de la boîte aux lettres et les transmet aux composantes appropriées ;
- Un moteur de coordination ;
- Un planificateur qui planifie les tâches de l'agent en fonction des décisions du moteur de coordination, des ressources disponibles et des spécifications des tâches ;
- Plusieurs bases de données représentant les plans connus par l'agent, les ressources et l'ontologie utilisée ;
- Un contrôleur d'exécution qui gère l'horloge locale de l'agent et les tâches actives.

L'environnement comporte trois bibliothèques : une avec des agents utilitaires, une avec des outils pour la construction des agents, et une avec des composants agents.

ZEUS met un fort accent sur la méthodologie de développement, fondée sur la notion de rôle (voir aussi la section précédente). **ZEUS** a été utilisé pour développer plusieurs applications réelles comme les ventes aux enchères et la simulation de la fabrication des ordinateurs. Les caractéristiques des domaines d'applications de **ZEUS** ont été définies par les concepteurs ; parmi ces caractéristiques, on peut mentionner :

- Chaque agent crée un plan qui nécessite un raisonnement explicite pour atteindre son but ;
- La résolution de problèmes nécessite une coopération entre agents ;
- Le rôle de chaque agent consiste à contrôler un système externe qui réalise une tâche du domaine d'application, la résolution de problème est ainsi effectuée par ce système externe et contrôlée par les agents.

3.2.3 MADKIT

Est une plate-forme développée par le Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier (LIRMM) de l'Université Montpellier II. **MADKIT** est libre pour l'utilisation dans l'éducation. **MADKIT** est écrit en Java et est fondé sur le modèle organisationnel Alaadin. Il utilise un moteur d'exécution où chaque agent est construit en partant d'un micro-noyau. Chaque agent a un rôle et peut appartenir à un groupe. Il y a un environnement de développement graphique qui permet facilement la construction des applications.

La **Figure 3.2** résume l'architecture MadKit. Cette plateforme est constituée d'un composant principal appelé Le micro-noyau agent c'est l'infrastructure ou l'environnement qui affecte et gère les rôles, responsable à la communication. Les agents sont des Threads JAVA, qui évoluent d'une manière asynchrone. Ils sont devisés en Agents applicatifs. Et en Agents système.

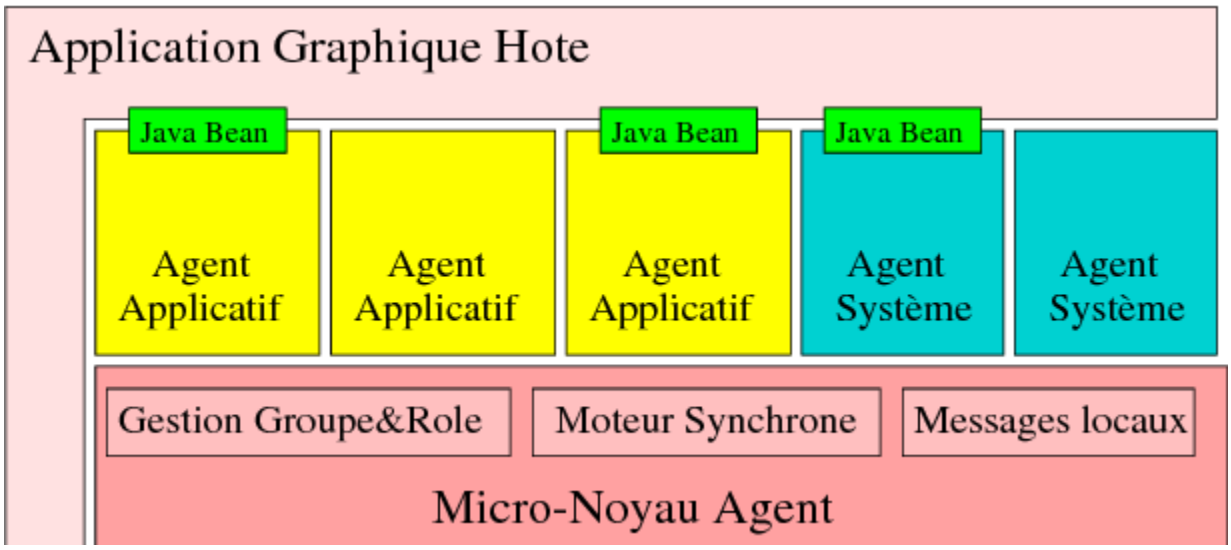


Figure 3.2: L'architecture MadKit

3.2.4 SWARM

SWARM (Minar e.a., 1996) est une plate-forme multi-agents avec agents réactifs. L'inspiration du modèle d'agent utilisé vient de la vie artificielle. **SWARM** est l'outil privilégié de la communauté américaine et des chercheurs en vie artificielle. L'environnement offre un ensemble

de bibliothèques qui permettent l'implémentation des systèmes multi-agents avec un grand nombre d'agents simples qui interagissent dans le même environnement. De nombreuses applications ont été développées à partir de **SWARM** qui existe aujourd'hui implémenté en plusieurs langages (Java, Objective-C).

3.3 Plateforme de simulation NetLogo

Les plates-formes de simulation qui permettent un prototypage et des tests rapides des idées de conception initiales et des alternatives, peuvent améliorer le processus d'implémentation de systèmes complexes composés de milliers d'agents agissant en parallèle et multi-agents (**SMA**). Elles permettent aussi au développeur de concevoir son application sans avoir à une étude approfondie des **SMAs**.

Une telle plateforme idéale doit être caractérisée par la facilité d'apprentissage, une implémentation et affichage simples du **SMA**, Prendre en compte les fonctionnalités de programmation orientées agent qui permet de réaliser les alternatives de conception en un environnement de développement d'agent facilement. Cependant, ces exigences spécifiques font d'une telle plate-forme de simulation un outil d'apprentissage parfait.

NetLogo a été créé par le Dr Uri Wilensky au Centre d'apprentissage connecté et de modélisation informatique (CCL), situé à l'université de Tufts, Medford, MA. En 2000, le CCL a été transféré à l'université de Northwestern, Evanston, IL, où le traitement de **NetLogo** est poursuivi jusqu'à présent.

NetLogo est un langage de programmation multi-agents simple et primitif basé sur le langage de programmation Logo. Cette plateforme est distribuée en open source dans sa version actuelle.

La conception et le développement de **NetLogo** ont été réalisés dans le but de modéliser et simuler des systèmes complexes évoluant dans le temps. Il a été utilisé pour concevoir plusieurs systèmes complexes dans différents domaines tels que l'économie, la biologie, la physique, la chimie, la psychologie, la dynamique des systèmes et de nombreux autres phénomènes naturels et sociaux.

Le **NetLogo** peut modéliser Les agents mobiles et immobiles. Il traite plusieurs agents appartenant au même environnement physique et peut également inclure des milliers d'agents dans la même simulation. En plus, il existe d'autres capacités pour lier les agents et de les mettre en réseau, des options à la fois bidimensionnelles et tridimensionnelles (2D et 3D) concernant les modes d'affichage, et une commutation facile entre l'exécution en une seule étape et les simulations continues. Les caractéristiques qui distinguent NetLogo peuvent être résumées comme suit :

- **Documentation claire et complète** : contenant un ensemble large et riche d'exemples (exemples de modèles) concernant différents domaines pour guider les objectifs des utilisateurs.
- **Une interface utilisateur graphique (GUI)** : facile à comprendre, elle compose d'un ensemble d'outils simples pour produire un modèle, le superviser et contrôler son comportement.
- **Langage de modélisation** : Cette plateforme offre un langage de modélisation facile à apprendre, puissant et flexible pour la création de modèles.
- **Un outil de simulation participative «HubNet»** : permettant à plusieurs utilisateurs exécutant des programmes clients distincts d'interagir via un modèle NetLogo. En plus, il contient plusieurs outils qui facilitent l'exécution de plusieurs programmes en arrière-plan.
- **Espace de comportement** : un outil qui facilite l'exécution d'un modèle plusieurs fois grâce à l'utilisation de différentes entrées pour étudier les impacts de scénarios alternatifs.
- Éditeurs d'icônes et capacités d'importation pour les liens et les agents.
- **Un environnement distinct** : équipé d'outils appropriés pour développer graphiquement les modèles de systèmes dynamiques au lieu de modèles basés sur des agents.

Au cours de notre travail, nous avons vu l'utilité d'utiliser NetLogo dans le but de décrire l'environnement de poursuite-évasion ainsi que le processus de poursuite des coalitions des agents à la capture des évadés détectés. En fait, cette plate-forme offre la possibilité de créer différents types d'agents qui pourraient être distingués par plusieurs caractéristiques telles que le degré de mobilité et définir la vitesse du mouvement et les chemins disponibles à suivre, la forme et la couleur définissant le rôle attribué au cours du processus de poursuite.

De plus, l'environnement des agents proposé dans **NetLogo** plus connu sous le nom de (les Patches) correspond parfaitement à notre grille de cellules poursuite-évasion environnement. En d'autres termes, chaque patch caractérisé par des coordonnées cartésiennes est considéré comme une cellule qui peut être occupée par un agent ou un obstacle. Les couleurs des Patchs sont modifiables nous permettant de définir les cellules libres, les obstacles, ainsi que les bords de l'environnement.

3.4 Résultats de la simulation

Afin de résoudre le problème, on a vu l'utilité d'utiliser la plate-forme **NetLogo**. Cette plateforme orientée Agent propose des méthodes prédéfinies qui facilitent notre implémentation. Les expériences sont effectuées pour vérifier l'efficacité de notre algorithme global, qui comprend à la fois la partie d'extraction de caractéristiques et la partie de génération de groupe séparément. L'environnement de poursuite dans ces simulations est un « environnement fermé » avec 100 m * 100 m, avec 10 poursuivants et 2 évasures dont d est 4. « Environnement fermé » signifie que la carte consiste d'un couple d'obstacles de forme aléatoire que les poursuivants et les évadés ne peuvent franchir dans une région limitée. Tous les agents mesurent 20 cm * 20 cm. Une itération représente une seconde en environnement de simulation.

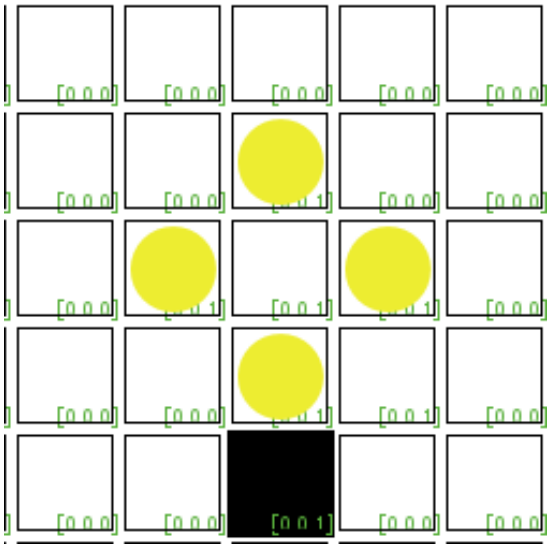


Figure 3.4.1 : Capture des évadés

Comme le montre la **figure 3.4.1**, dans laquelle il est spécifiquement détaillé comment un poursuivant de ce type pourrait être saisi. Chaque agent est marqué avec un numéro d'identification. Les poursuivants et les évadeurs ont une vitesse similaire (une cellule par itération) et un excellent système de communication. Les équipes des poursuivants sont totalement capables de déterminer leurs positions réelles, et les évadés ont disparu après la capture.

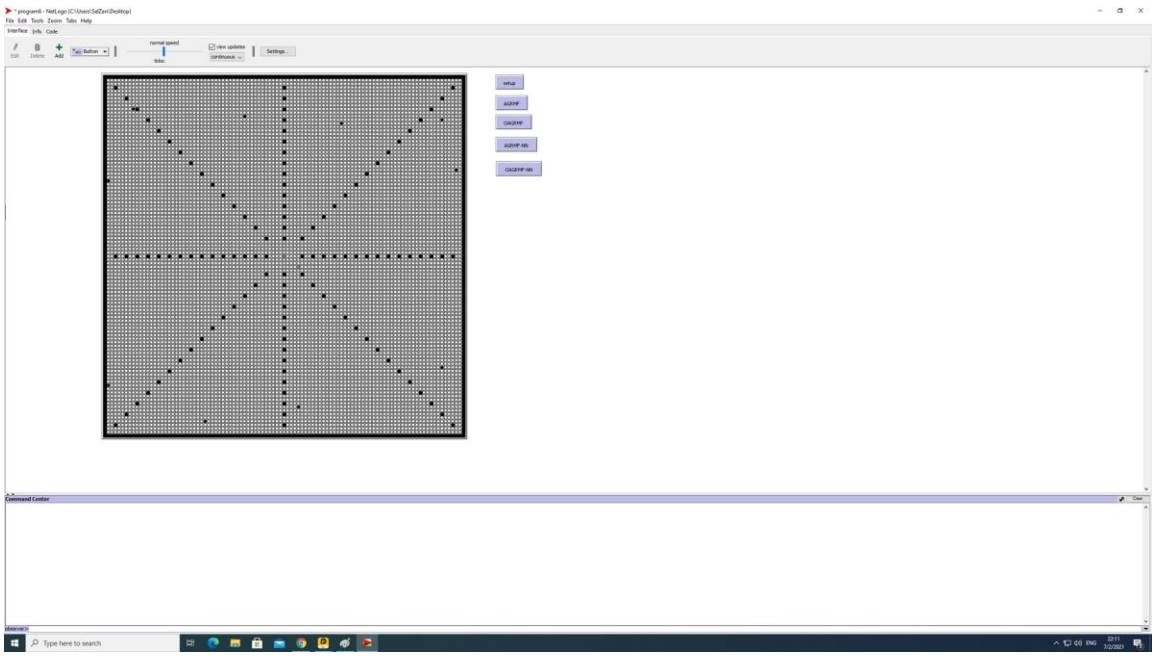


Figure 3.4.2 : L'écran d'interface NetLogo pour notre simulation

La **Figure 3.4.2** montre l'écran d'interface Net Logo pour notre simulation, en cliquant sur le bouton setup on obtient l'état initial du programme. Le bouton OAGRMF-NN renvoi la simulation de notre algorithme OAGRMF-NN. Le bouton OAGRMF renvoi la simulation d'algorithme modèle organisationnel Overlapping Agent Group Role Membership Function , Le bouton AGRMF renvoi la simulation d'algorithme modèle organisationnel Agent Group Role Membership Function , Le bouton AGRMF-NN renvoi la simulation d'algorithme modèle organisationnel Agent Group Role Membership Function base sur RN ,La partie (command center) montre les résultats de la simulation et les valeurs des variables, Le code de procédures est écrit dans l'onglet code.

Pour prouver l'efficacité de notre algorithme, nous avons vu l'utilité de les comparer avec des activités de recherche sur la poursuite et l'évasion :

- **Cas OAGRMF** : Formation de coalition de poursuite multi-agents basée sur un chevauchement limité des groupes dynamiques qui a été proposé dans la littérature [35]
- **Cas AGRMF** : Formation et reformation des groupes avec l'application de l'algorithme de coalition qui a été proposé dans la littérature [35]
- **Cas AGRMF-NN** : Formation et reformation des groupes avec l'application de l'algorithme de coalition qui a été proposé dans la littérature [35] base sur les réseaux de neurones
- **Cas OAGRMF-NN** : Formation de coalition avec l'application de notre algorithme de coalition

Les résultats de la simulation sont basés sur l'étude de capture des évadés où chacun utilise l'une des stratégies de Formation de coalition (cas OAGRMF-NN, cas OAGRMF,cas AGRMF, cas AGRMFNN) comme suit :

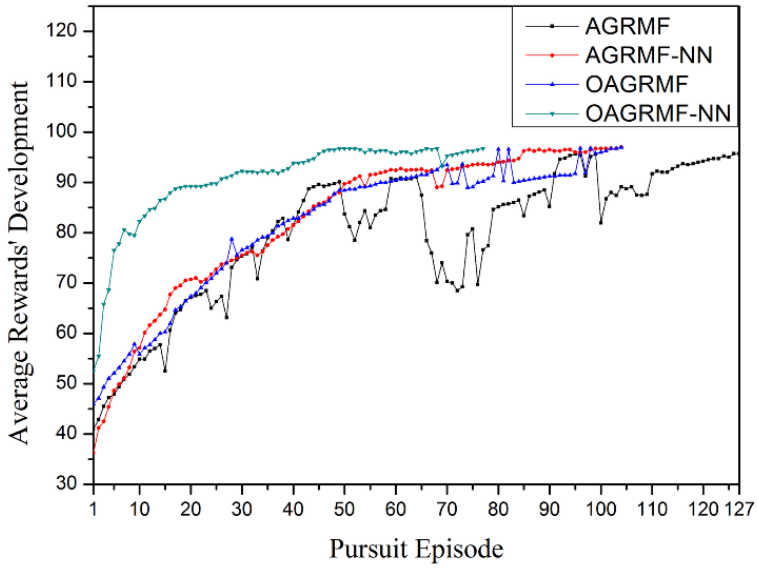


Figure 3.4.3 : Evolution moyenne des récompenses au cours d'une épisode de la poursuite

Résultat 1

La **Figure 3.4.3** représente le développement de la récompense moyenne des agents au cours d'un épisode de la poursuite (par rapport aux itérations), en comparant notre méthode OGRMF-NN par rapport aux méthodes existantes AGRMF, AGRMF-NN et OGRMF.

On peut noter que les agents, dans le cas de OGRMF-NN, atteignent la récompense moyenne maximale après 47 itérations de la poursuite. Cependant, dans les autres méthodes (AGRMF, AGRMF-NN et OGRMF) les agents atteignent la récompense moyenne maximale après 85, 96 et 125 itérations respectivement.

De plus, on note moins de diminutions de la récompense moyenne dans notre méthode puis dans la méthode AGRMF-NN en comparaison avec les autres méthodes. Ces diminutions de la récompense sont dues au fait que les agents changent leurs groups d'appartenance au cours de la poursuite.

Cette priorité que présente notre méthode la OGRMF-NN est due au comportement intelligent et collectif des agents fourni le réseau de neurones. Dans la OGRMF-NN, les groupes sont formés d'abord en grouppent les agents avec des vecteur de caractéristiques similaires, puis chaque groupe choisis l'évadé à poursuivre. En d'autres termes, les agents agissent en groupe et non pas individuellement, dû l'augmentation du taux de collaboration entre les agents, contrairement aux agents de la AGRMF et OGRMF qui choisissent les évadés à poursuivre d'une manière individuelle.

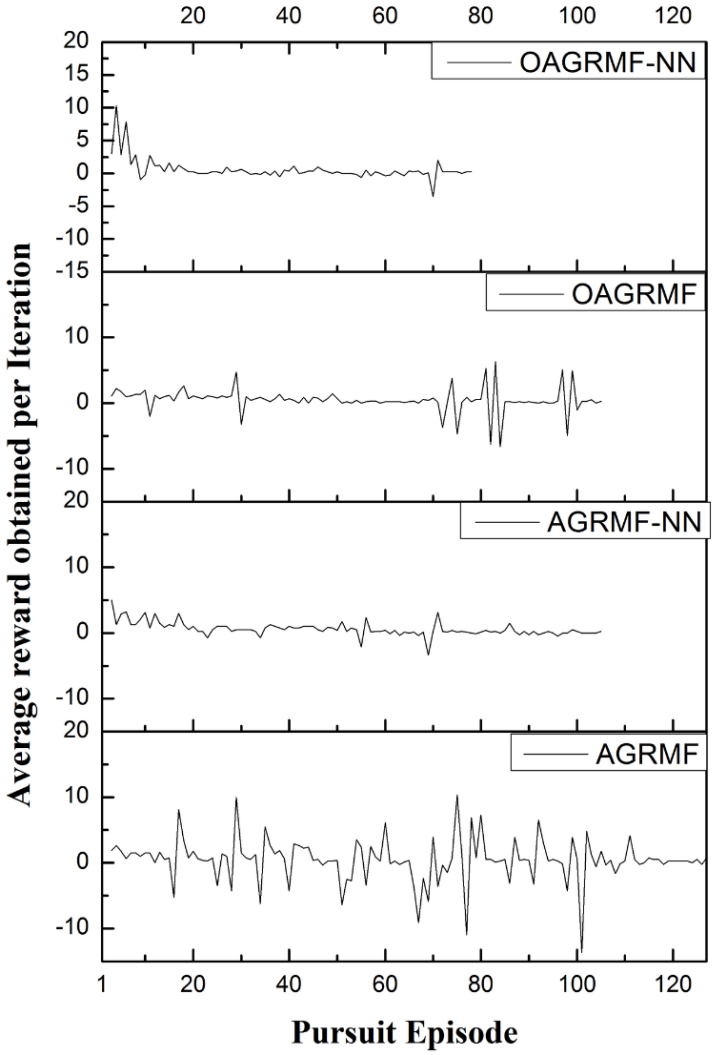


Figure 3.4.4 : Récompense moyenne obtenue par itération pendant un épisode complet de la poursuite

Résultat 2

La **Figure 3.4.4** montre les récompenses moyennes obtenues par les agents par itération au cours d'un épisode complet de la poursuite.

Dans le cas de notre méthode la OAGRMF-NN, on note uniquement deux diminutions négatives de la récompense moyenne, aux itérations 10 et 70. Puis, la méthode AGRMF-NN avec quatre

diminutions négatives de la récompense moyenne, aux itérations 23, 34, 55 et 70, puis la méthode OAGRMF est enfin la méthode AGRMF.

Les méthodes OAGRMF-NN et AGRMF-NN ont moins de diminution de la récompense au cours de la poursuite grâce à la stratégie de collaboration fournie par le réseau de neurones.

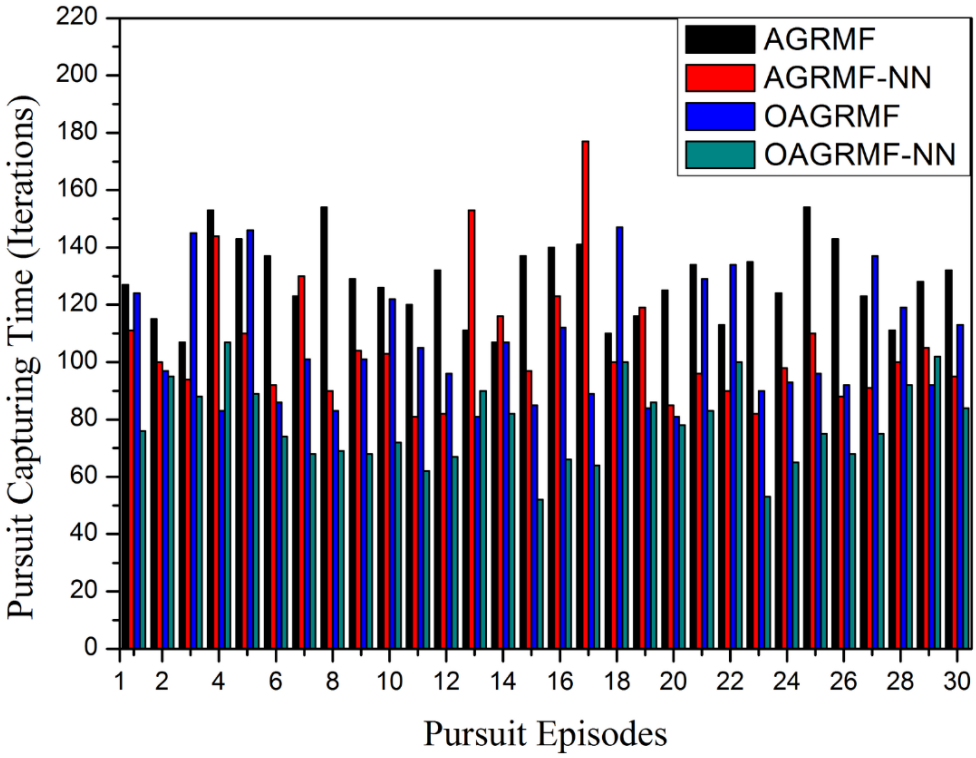


Figure 3.4.5: Durée moyenne de la poursuite

Résultat 3

La **Figure 3.4.5** montre la durée moyenne dans 30 épisodes de la poursuite, en comparant la méthode proposée OAGRMF-NN avec AGRMF, AGRMF-NN et OAGRMF. On note que dans 24 sur 30 épisodes sur, la méthode OAGRMF-NN représente la durée minimale de poursuite, en comparaison avec les autres méthodes. Cela montre la priorité de la méthode proposée par rapport aux autres méthodes. Cette priorité est le résultat de la stratégie de collaboration des agents offerte par l'utilisation du réseau de neurones

3.5 Conclusion

Grâce à notre implémentation, nous pouvons dire que Les RN sont un excellent outil dans la phase exploratoire de l'exploration de données et efficace par rapport au regroupement direct des données .et à travers les résultats reflétés dans ce chapitre, nous pouvons dire que notre nouvel algorithme Dun modèle organisationnel base sur les Réseaux de Neurones pour les Systèmes Multi Agents a permis l'obtention d'un temps d'exécution adéquat et de réduire le temps de calcul et de regroupements ainsi qu'un minimale nombre d'itération par rapport à les autres modèle durant l'exécution .

Conclusion générale

Ce mémoire avait pour ambition de réaliser un nouveau modèle basé sur les Réseaux de Neurones pour les Systèmes Multi Agents. Les solutions ne doivent pas forcément être optimales mais proposer un résultat convenable par rapport aux autres algorithmes propose dans la formation de coalition de système multi-agents.

Pour réaliser ce mémoire, nous avons d'abord rassemblé le plus de documents concernant le sujet. La formation de coalition étant un domaine relativement récent, malheureusement, il existe peu de livres liés à ce sujet, on a aussi parler des réseaux de Neurones et heureusement, on a trouvé des thèses, articles et autres papiers disponibles concernant ce sujet.

La deuxième étape consiste à introduire le sujet en précisant les grandes étapes pour pouvoir réaliser la troisième étape qui introduit le nouvel algorithme proposé basée sur les RN. Cette dernière vise regrouper les poursuivant avec l'utilisation de SOM pour la génération des groupes a fin d'extraire les fonctionnalités.

Cette application a été effectuée en comparaison avec les autres algorithmes (AGRMF, AGRMF-NN, OAGRMF).

Au cours des simulations, nous avons étudié la capture des évadés où chacun utilise l'une des stratégies de Formation de coalition, ainsi que la durée d'exécution et le nombre d'itérations. Les simulations sont créées sous la plateforme NetLogo 6.2.0.

Enfin, on représente les résultats. Ces derniers reflètent la différence entre les 3 algorithmes comparées, avantageusement pour le nouveau modèle organisationnel base sur les Réseaux de Neurones pour les Systèmes Multi Agents propose on a obtenue des résultats adéquat par rapport aux autres.

Pour conclure, notre perspective concernant le sujet étudié est l'application du nouveau modèle organisationnel proposé dans d'autres problèmes complexes tels que les systèmes distribués ainsi que la gestion du trafic aérien.

Références

[1] Abadi, M. (1987). *Temporal-logic theorem proving* (Doctoral dissertation, Stanford University).

[2] Agre, P. E., & Chapman, D. (1987, July). Pengi: An Implementation of a Theory of Activity. In *AAAI* (Vol. 87, No. 4, pp. 286-272).

[3] Shardlow, N. (1990). *Action and agency in cognitive science* (Doctoral dissertation, Master's thesis, Department of Psychlogy, University of Manchester, Oxford Rd., Manchester M13 9PL, UK).

[4] Ferber, J. (1997). *Les systèmes multi-agents : vers une intelligence collective*. InterEditions.

[5] Li, S., Wu, Y., Cui, X., Dong, H., Fang, F., & Russell, S. (2019, July). Robust multi-agent reinforcement learning via minimax deep deterministic policy gradient. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 33, No. 01, pp. 4213-4220).

[6] Gutknecht, O. (2001). *Proposition d'un modèle organisationnel générique de systèmes multi-agents et examen de ses conséquences formelles, implémentatoires et méthologiques* (Doctoral dissertation, Université Montpellier II-Sciences et Techniques du Languedoc).

[7] Qadir, M. Z., Piao, S., Jiang, H., & Souidi, M. E. H. (2020). A novel approach for multi-agent cooperative pursuit to capture grouped evaders. *The Journal of Supercomputing*, *76*(5), 3416-3426.

[8] Souidi, M. E. H., Siam, A., & Pei, Z. (2019). Multi-agent pursuit coalition formation based on a limited overlapping of the dynamic groups. *Journal of Intelligent & Fuzzy Systems*, *36*(6), 5617-5629.

[9] Ammar, M. Y. (2007). Mise en œuvre de réseaux de neurones pour la modélisation de cinétiques réactionnelles en vue de la transposition batch/continu (Doctoral dissertation).

[10] Hughes, J. R., Penney, D. W., & Stone, J. L. (1994). History of the Neuropsychiatric Institute of the University of Illinois Medical Center, Chicago, Illinois: Development of the Early EEG Laboratory and Epilepsy Clinic of Dr. Frederic A. Gibbs. Clinical Electroencephalography, 25(3), 99-103.

Références

[11] Kawaguchi, K. (2000). A multithreaded software model for backpropagation neural network applications. The University of Texas at El Paso.

[12] Mestan, A. (2008). Introduction aux Réseaux de Neurones Artificiels Feed Forward. alp. developper. com.

[13] Nerrand, O., Roussel-Ragot, P., Personnaz, L., Dreyfus, G., & Marcos, S. (1993). Neural networks and nonlinear adaptive filtering: Unifying concepts and new algorithms. Neural computation, 5(2), 165-199.

[14] Touzet, C. (1992). les réseaux de neurones artificiels, introduction au connexionnisme. EC2.

[15] Rizkalla, N. (2005). Nanoparticules et réseaux de neurones artificiels: de la préparation à la modélisation..

[16] : G. Dreyfur. « Réseaux de neurones : Méthodologie et application », édition Eyrolles, 2004.

[17] : B. Kosko. « Unsupervised Learning in Noise », IEEE Trans.Neural Net, Vol.1, N°1, pp. 44-57, Mars 1990

[18] Cours de Data Mining – 8 : Réseaux de neurone

[19]. Li, M., Cai, Z., Yi, X., Wang, Z., Wang, Y., Zhang, Y., & Yang, X. (2016, August). ALLIANCE-ROS: a software architecture on ROS for fault-tolerant cooperative multi-robot systems. In Pacific Rim International Conference on Artificial Intelligence (pp. 233-242). Springer, Cham.

[20]. Werger, B. B., & Matarić, M. J. (2000). Broadcast of local eligibility for multi-target observation. In Distributed Autonomous Robotic Systems 4 (pp. 347-356). Springer, Tokyo.

[21]. Yang, J., Zhang, H., Ling, Y., Pan, C., & Sun, W. (2013). Task allocation for wireless sensor network using modified binary particle swarm optimization. IEEE Sensors Journal, 14(3), 882-892.

[22]. Budaev, D., Amelin, K., Voschuk, G., Skobelev, P., & Amelina, N. (2016, April). Real-time task scheduling for multi-agent control system of UAV's group based on network-centric

technology. In 2016 International Conference on Control, Decision and Information Technologies (CoDIT) (pp. 378-381). IEEE.

[23]. Vásárhelyi, G., Virágh, C., Somorjai, G., Tarcai, N., Szörényi, T., Nepusz, T., & Vicsek, T. (2014, September). Outdoor flocking and formation flight with autonomous aerial robots. In 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (pp. 3866-3873). IEEE.

[24]. Ghazikhani, A., Mashadi, H. R., & Monsefi, R. (2010, May). A novel algorithm for coalition formation in multi-agent systems using cooperative game theory. In 2010 18th Iranian Conference on Electrical Engineering (pp. 512-516). IEEE.

[25]. Amigoni, F., & Basilico, N. (2012, May). A game theoretical approach to finding optimal strategies for pursuit evasion in grid environments. In 2012 IEEE international conference on robotics and automation (pp. 2155-2162). IEEE.

[26]. Hrnčíř, J., Rovatsos, M., & Jakob, M. (2015). Ridesharing on timetabled transport services: A multiagent planning approach. Journal of Intelligent Transportation Systems, 19(1), 89-105.

[27]. Delgado-Mata, C., & Ibanez-Martinez, J. (2006, November). An emotion affected Action Selection Mechanism for multiple virtual agents. In 16th International Conference on Artificial Reality and Telexistence--Workshops (ICAT'06) (pp. 57-63). IEEE.

[28]. Fang, B., Chen, L., Wang, H., Dai, S., & Zhong, Q. (2014). Research on multirobot pursuit task allocation algorithm based on emotional cooperation factor. The Scientific World Journal, 2014.

[29]. Torreno, A., Onaindia, E., Komenda, A., & Štolba, M. (2017). Cooperative multi-agent planning: a survey. ACM Computing Surveys (CSUR), 50(6), 1-32.

[30]. Huang, Z., Mitra, S., & Vaidya, N. (2015, January). Differentially private distributed optimization. In Proceedings of the 2015 International Conference on Distributed Computing and Networking (pp. 1-10).

[31]. Katewa, V., Pasqualetti, F., & Gupta, V. (2018). On privacy vs. cooperation in multi-agent systems. International Journal of Control, 91(7), 1693-1707.

Références

[32]. Rauniyar, A., & Muhuri, P. K. (2016, October). Multi-robot coalition formation problem: Task allocation with adaptive immigrants based genetic algorithms. In 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC) (pp. 000137-000142). IEEE.

[33]. Liemhetcharat, S., & Veloso, M. (2014). Weighted synergy graphs for effective team formation with heterogeneous ad hoc agents. Artificial Intelligence, 208, 41-65.

[34]. Liemhetcharat, S., & Veloso, M. (2017). Allocating training instances to learning agents for team formation. Autonomous Agents and Multi-Agent Systems, 31(4), 905-940.

[35]. Souidi, M. E. H., Songhao, P., Guo, L., & Lin, C. (2016). Multi-agent cooperation pursuit based on an extension of AALAADIN organisational model. Journal of Experimental & Theoretical Artificial Intelligence, 28(6), 1075-1088.

[36]. University of Illinois at Urbana-Champaign. Center for Supercomputing Research and Development, & Cybenko, G. (1988). Continuous valued neural networks with two hidden layers are sufficient.

[37]. Stephanakis, I. M., Chochliouros, I. P., Sfakianakis, E., & Shirazi, N. (2015, September). Anomaly Detection In Secure Cloud Environments Using a Self-Organizing Feature Map (SOFM) Model For Clustering Sets of R-Ordered Vector-Structured Features. In Proceedings of the 16th International Conference on Engineering Applications of Neural Networks (INNS) (pp. 1-9).

[38]. Vesanto, J., & Alhoniemi, E. (2000). Clustering of the self-organizing map. IEEE Transactions on neural networks, 11(3), 586-600.

[39]. Nakao, R., Abe, T., Funayama, S., & Sugimoto, C. (2016). Horizontally transferred genetic elements in the tsetse fly genome: an alignment-free clustering approach using batch learning self-organising map (BLSOM). BioMed research international, 2016.

[40]. Souidi, M. E. H., & Piao, S. (2016). A new decentralized approach of multiagent cooperative pursuit based on the iterated elimination of dominated strategies model. Mathematical Problems in Engineering, 2016.

[41] Tisue, S., & Wilensky, U. Center for Connected Learning and Computer-Based Modeling Northwestern University, Evanston, Illinois.

Références

[42] Puterman, M. L. (2014). Markov decision processes: discrete stochastic dynamic programming. John Wiley & Sons.

[43] Voir le site http://theses.univlyon2.fr/documents/getpart.php?id=lyon2.2008.nfaoui_eh&part=152187, consulté le : 03/07/2021 à 21 h 19 min.

[44] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. Neural computation, 1(4), 541-551.