INTELLIGENT MULTIAGENT COORDINATION BASED ON NEURO-FUZZY MODELS WITH HIERARCHICAL REINFORCEMENT LEARNING

Leonardo A. Forero, Marley M. B. R. Vellasco, Karla Figueiredo and Eugenio Silva

***Abstract*— This paper presents the research and development of a hybrid neuro-fuzzy model for the hierarchical coordination of multiple intelligent agents. The main objective of the models is to have multiple agents interact intelligently with each other in complex systems. We developed two new models of coordination for intelligent neuro-fuzzy multiagent systems that use MultiAgent Reinforcement Learning Hierarchical Neuro-Fuzzy with a market-driven coordination mechanism (MA-RL-HNFP- MD) and a MultiAgent Reinforcement Learning Hierarchical Neuro-Fuzzy with graph coordination (MA-RL-HNFP-GC). After setting options and using the MA-RL-HNFP_MA family of models, the coordination systems were tested in two case studies involving the implementation of a benchmark game of predator-prey. The tests showed that the new system has the ability to coordinate actions between agents with a convergence rate nearly 30% greater than that of the original version.**

I. INTRODUCTION

There are many advantages to using multiple agents. Through parallel computing, multiple agents can work together to better exploit the decentralized structure of a given task and accelerate its completion [1][2]. Additionally, agents can exchange experiences by communicating [3], observe and learn from the most skilled agents [4], and serve as teachers for other agents [5]. The multiagent system (MAS) can also provide a high degree of scalability because it can add new agents when needed and assign the activities of failed agents to other agents [6]. The MAS described in this study also demonstrates the understanding of intelligence [7]. Because intelligence is strongly linked to interaction, the best way to create intelligent machines could be to build social networks of machines. Coordination is a key feature of a MAS that performs some activity in a shared environment [8]. Coordination is closely related to knowledge sharing between agents, and its main objective is to coordinate the actions of each individual agent to achieve the ultimate goal of the MAS [9]. The coordination mechanisms of most MASs can be classified as implicit (centralized) or explicit (distributed) [10]. These mechanisms are useful in simple environments, in which the mechanism has a single goal [11][12]. However, there are environments in which the ultimate goal is achieved by fulfilling a number of conditions and sub-goals, with different agents playing different roles. These environments require elaborate coordination mechanisms and greater knowledge of the environment on the part of the operator [8][13].

Without coordination, the benefits of distributed execution of tasks disappear and the group of agents can degenerate into a chaotic and incoherent collection of individual behaviors [9]. Several problems can occur in a MAS without coordination, such as conflicts over resource access or misuse, redundancy, and increased waiting time, particularly when the activity of an agent depends on the completion of activities by other agents [13].

A good coordination system should avoid or minimize the occurrence of these problems by optimizing the use of resources and the time required to achieve the objectives [14]. Effective coordination between agents operating in a shared environment contributes to improving the quality of the solutions achieved and to improved performance in solving tasks [15]. The aim of this paper is to extend the MultiAgent Reinforcement Learning Hierarchical Neuro-Fuzzy (MA-RL-HNFP) model by combining its advantages with those of existing coordination strategies to address complex environments and multiple objectives, improve performance, and enhance communication between agents. The main objective of this work is the development and implementation of strategies for multiagent coordination using the hybrid neuro-fuzzy models of the MA-RL-HNFP family. These models enable efficient and optimal coordination, which leads to improved communication among agents in complex environments and allows them to complete multiple objectives.

II. RLF-HNFP MODEL

The RL-HNFP model is composed of one or various standard cells called RL-neuro-fuzzy politree partitioning (RL-NFP). These cells are arranged in a hierarchical structure in the form of a tree in accordance with the type of partition being used. Binary Space Partitioning (BSP) [15][16] divides the space in two repeatedly. Politree partitioning is a generalization of the quadtree method [15]. In this partitioning method, the subdivision of the n-dimensional space is accomplished in $m=2^n$ subdivisions.

Figure 1a shows an example of a two-dimensional input partitioned using BSP. Politree partitioning can be represented by a tree structure. Figure 1b shows a typical example of politree partitioning (with n inputs). Hierarchical partitioning is flexible and minimizes the exponential rule growth problem because it creates new rules locally

Leonardo Forero and Marley M.B.R. Vellasco are with the Department of Electrical Engineering, Pontifical Catholic University of Rio de Janeiro (PUC-Rio), Brazil (email: {mendonza, marley}@ele.puc-rio.br).

Karla Figueiredo and Eugenio silva with the Department of Computer Science, Universidade Estadual da Zona Oeste (UEZO), Rio de Janeiro, Brazil (email: karla.figueiredo@gmail.com eugenio@ele.puc-rio.br)

This work was supported by CNPq and FAPERJ.

according to its learning process. This type of partitioning is considered recursive because it uses a recursive process to generate partitions. With such partitioning, the resulting models have a hierarchy in their structure and thus have hierarchical rules. The outputs of the cells in the lower levels are the consequents of the cells in higher levels.

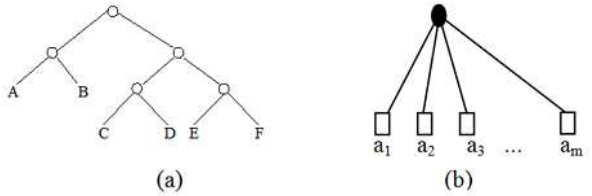

Figure 1. (a) BSP Tree representing BSP partitioning, (b) generic tree representation of politree partitioning.

An RL-NFP cell is a mini-neuro-fuzzy system that performs n-dimensional partitioning of a given space in accordance with the membership functions. The RL-NFP cell generates a precise (crisp) output after the defuzzification process [17][18]. Each cell receives all inputs considered in the problem and the value of each input variable (x_i) is read by one of the agent's sensors. It is then inferred in the antecedents' fuzzy sets (low - $\rho(x_i)$ - and high - $\mu(x_i)$). Figure 2 depicts an example of the resultant partition of a RL_NFP cell with two inputs – x_1 and x_2 , resulting in four sub-partitions (Quadtree partitioning).

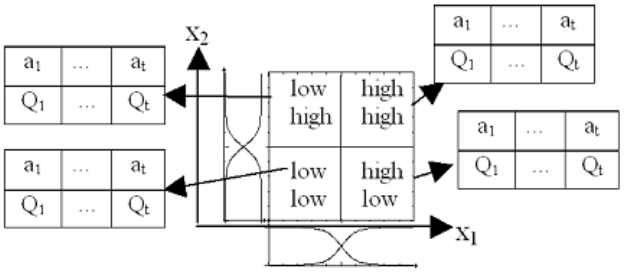

Figure 2 – Internal representation of the RL-NFP cells.

The consequents of the cell partitions may be of the singleton type (a constant) or the output from a stage of a previous level. Although a singleton consequent is simple, this consequent is not known in advance because each singleton consequent is associated with an action that has not been defined a priori. Each partition has a set of possible actions (a_1, a_2, ... a_1), in which each action is associated with a Q-value function. The Q-value is defined as the sum of the expected values of the rewards obtained by the execution of action a in state s, in accordance with a policy p. Further details of RL theory are presented in a previous study [19].

The linguistic interpretation of the mapping implemented by the RL-NFP cell is given by rules. Each rule corresponds to one of the polipartitions generated by the politree partitioning and has the following structure:

- Rule 1: If $x_1 \in \rho_1$and $x_2 \in \rho_2$ then $y = a_i$
- Rule 2: If $x_1 \in \rho_1$and $x_2 \in \mu_2$ then $y = a_j$
- Rule 3: If $x_1 \in \mu_1$and $x_2 \in \rho_2$ then $y = a_p$
- Rule 4: If $x_1 \in \mu_1$and $x_2 \in \mu_2$ then $y = a_q$

RL-HNFP models can be created based on the interconnection of the basic cells described above. The cells form a hierarchical structure that results in the rules that comprise the agent's reasoning. A neuro-fuzzy learning process is generally divided into two parts: structure identification and parameter adjustment. RL-HNFP performs these learning tasks in a single algorithm. It consists of six main steps: generate a root cell, calculate global reinforcement, back-propagate the reinforcement, select actions, update Q-values, and partitioning. A previous study provides details about how each step works [17].

III. MULTIAGENT RL-HNFP MODEL

The main idea of the multiagent version of the RL-HNFP family of algorithms is to ensure that all agents are able to explore different tasks or different state-action pairs simultaneously, thus speeding up learning and convergence for an optimal policy. There are many approaches that involve multiple agents, a goal to be learned, coordination among agents, and learning homogeneity. With the advent of multiagent RL-HNFP (MA-RL-HNFP), it became possible to broaden the RL-HNFP model in different ways to contemplate the greatest possible variety of applications.

In this paper, different types of MA-RL-HNFP systems derived from the original system are presented, and we describe their behavior, particularities, and objectives. Additionally, the differences between models in terms of coordination and learning dynamics are explained. The proposed MA-RL-HNFP systems differ in two main aspects: learning dynamics and the coordination mechanism among agents. For MASs, learning occurs one of two ways: with or without the sharing of the learning structure.

When the structure is shared, the learning structure is the same for every agent. The learning structure functions as a "group intelligence", which is used by all agents for decision making and serves as a knowledge repository. In other words, each agent can execute a specific action in a distinct state of knowledge. Subsequent agents can access the information provided by the previous agents in the single structure, choose the action to be performed, and update the Q values according to the received information.

This type of model, in which multiple agents use the same structure, is used when agents need to learn the same task and seek cooperation. If all of the agents compete using the exact same "intelligence", there would be no losers or winners.

Similar to the RL-HNFP model, learning also occurs cyclically. First, the agents access the structure to make a decision, explore the state-action pairs of the environment, receive the return, and transfer "what they have learned" to the knowledge structure, which processes the information of the explored state-action pair and received return value. A new cycle is then performed for another agent, which can explore a completely distinct space in the environment. The main difference is the agent's rotation in its exploration of

the environment. The full training process is divided into seven steps, as depicted in Figure 3.

However, in a competitive environment, there is no need for an agent to provide knowledge to its rivals. Then, the other method of learning involves each agent keeping its own structure separate from the other agents in the environment. In this case, each agent specializes itself to a specific task. This approach seeks to solve problems involving both cooperation and competition among multiple agents. To make agents cooperate, there must be a coordination mechanism or central agent that receives and combines the information from each RL-HNFP structure. In competitive problems, the agents may use their own structures independently, with the goal of learning a specific task and competing against other agents that are equally "equipped" with an RL-HNFP learning structure.

IV. COORDINATION IN MULTIAGENT SYSTEMS

Coordination is defined as "the act of working together harmoniously towards achieving an agreement and common goal" [13]. Working together, in turn, depends on the application or problem at hand. Coordination is important in competitive, cooperative, and mixed environments. In a competitive environment, coordination focuses on conflict resolution and negotiation. In a collaborative environment, coordination focuses on cooperation among agents, including ways to solve global problems in a distributed manner by creating teams and agents that work in groups to optimize the achievement of goals. Coordination in MASs seeks to efficiently organize the intelligent behavior of a set of autonomous agents. The goal in such a system is to define strategies or ways of coordinating the different skills and behaviors of the agents such that they can jointly take action and solve problems efficiently.

Coordination in MASs involves coordination between the agents and the coordination of actions of an agent. Many problems can occur in a MAS that lacks a coordinating mechanism or has one that is not appropriate:

- Conflicts between features or system resource demands;
- Redundancy in the tasks of agents;
- Increases in waiting time when the activity of an agent depends on the activities of other agents.

Thus, the main purpose of coordination is to avoid or minimize these problems while optimizing the resources and time to perform complex tasks. Such tasks can be completed though a single agent does not have all of the resources that are necessary to complete the task or meet several sub-goals to achieve the overall goal [1][2].

There are several coordination mechanisms. In this paper, two models are chosen that provide excellent results in the coordination of MASs in different applications [12] and are the most frequently used in the current literature. The first is market-driven coordination, and the second is graph coordination.

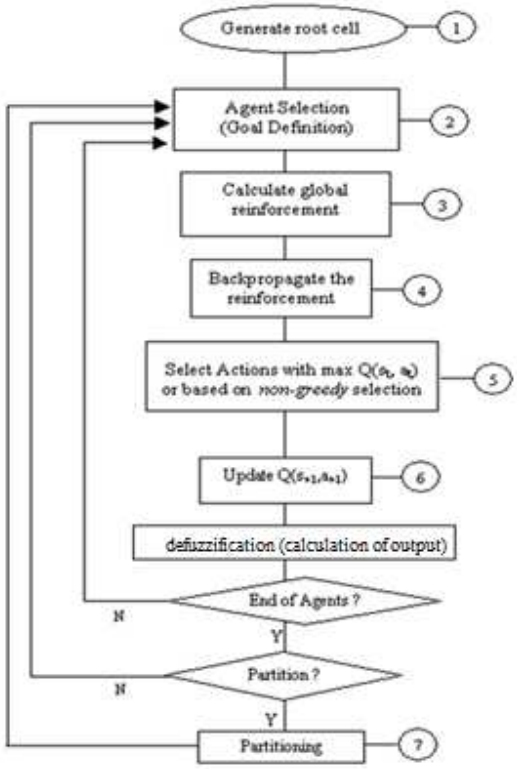

Figure 3 –Learning algorithm of the MA-RL-HNFP model.

A. *Market-driven coordination*

This approach considers a group of agents whose goal is to complete the tasks successfully while trying to minimize total costs. Each agent will seek to minimize costs and maximize its individual private profit. The idea of method-oriented MASs is based on the interaction of agents in a distributed manner, which allows agents to have greater bargaining power and information. Thus, agents must have a mechanism that allows for communication amongst them [20].

As soon as the MAS has a goal, the goal is decomposed into smaller tasks. Then, an auction is held for each of these tasks and is associated with a gain (reward) for that task. In each auction, the participating agents calculate the estimated cost to accomplish the task and offer a price to the auctioneer.

At the end of the auction, the bidder (agent) with the lowest price offered will be given the right of task execution and receive a gain. An agent can open another auction to sell the task that he won. Two or more agents can work together to accomplish a task that would be difficult for a single agent. To develop the market-driven strategy, a cost function must be defined for all of the resources needed to accomplish the task. Each of these resources can have a different weight depending on the importance of each to perform the task [20].

B. *Graph coordination*

This type of coordination uses graphs to relate agents to each other and coordinate their actions in a specific

situation. In most applications, it is rare that agents have interdependent actions, i.e., in a few situations, the actions of an agent require that the actions of other agents have been previously performed. The problem of global coordination is now replaced by problems of coordination sites with fewer agents.

This decomposition can be represented by graphs in which each node is an agent and each edge indicates that two agents must coordinate their actions. This technique is referred to as a coordination graph (CG) [12]. Each dependency corresponds to a local function, which assigns a specific value to each of the different combinations of actions of the agents involved. Figure 4 shows a CG structure for eight agents (numbered circles). For each connection, a dependence function between the agents is specified. For example, the edge between agents 1 and 2, represented by f_{12}, is a function that contains all possible actions of agent 1 with agent 2.

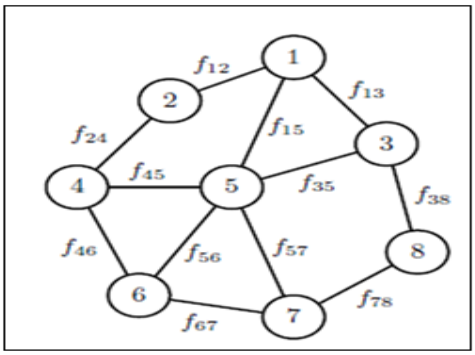

Figure 4 - CG for eight agents.

The overall function is the sum of all local functions. To calculate the action that maximizes the overall function, the variable elimination algorithm can be used [12]. This algorithm is briefly presented below.

The graphical representation of the dependencies of agent actions is generated through CGs, but to find individual stocks that produce optimal system performance, the coordination is measured using the method of variable elimination [12]. This method can be described mathematically by a function that coordinates the relations shown by the graphs.

The idea of variable elimination is to calculate a coordinated action involving a group of n agents with the purpose of maximizing a revenue function. The problem can be described as follows: each agent i selects an individual action a_i from a set of actions A_i, and the resulting joint action $a = (a_1,\ldots a_n)$ generates a callback function R(a). The problem is to find the optimal action that maximizes the callback function $a^* = \text{argmax}_a R(a)$.

The deceptively simple approach to solve the coordination problem is to enumerate all possible joint actions and choose the one that maximizes R(a). In most real applications, this enumeration process becomes computationally impractical. However, several problems in the action of an agent i depend solely on a small group of agents $\Upsilon(i)$ rather than on all agents. Thus, the overall function returns R(a) as a linear combination of local functions, as shown in the following equation:

$$R(a) = \sum_{i=1}^{n} f_{ij}(a_i, a_j) \quad (1)$$

The graph representation was chosen to facilitate the modeling of the problem by making it possible to represent any dependency [12]. To solve the coordination problem, the graph representation finds the best joint action $a^* = \text{argmax}_a$ R(a) using variable elimination [12].

When an agent is selected for elimination, all functions that have a return or dependence associated with their vertices are removed. Then, a callback function conditional is calculated ($\emptyset(a)$), which returns the maximum value that the agent is able to contribute to the system as a function of each combination of actions with its neighbors. The callback function conditional also returns a better response function (B(a)), which returns the corresponding action for this maximized value. The agent communicates this conditional reward function to one of its neighbors and is then eliminated.

The neighbor agent creates a new dependency (connection) between the agent and those involved in the conditional function, and the next agent in the ordering is then selected for elimination. This process is repeated until a single agent remains. This agent sets its action to maximize the final function returns. This individual action is part of the joint action good, and the associated value function's conditional return is equal to the desired value $a^* = \text{argmax}_a$ R(a). A second pass in reverse order is then performed in which each agent calculates its optimal action based on its conditional strategy and the actions of its neighbors are fixed [12]. An example of this method is shown in Figure 5, where four agents are represented with their dependencies. Each node in the graph is an agent, and the actions of agent 1 depend on the actions of agents 2 and 3. The actions of agent 2 depend solely on the actions of agent 1. The actions of agent 3 depend on the actions of agents 1 and 4.

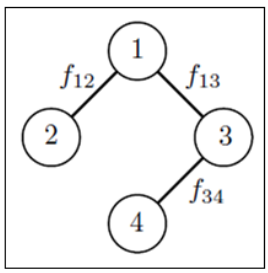

Figure 5 Example of the CG pair deletion variable.

V. MULTIAGENT REINFORCEMENT LEARNING NEURO-FUZZY MODEL WITH MARKET-DRIVEN COORDINATION

The agents within a community group of the application have a role, and they perform some actions related to this role. Their roles define the behavior of each individual agent. For example, in football, in an attack situation, the role of the attacker is given to the player best positioned to take advantage of the situation.

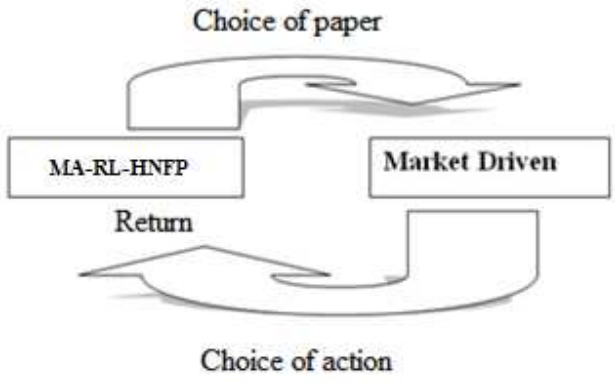

Figure 6 MA-RL-HNFP- MD model.

The model is divided into two stages. The first step uses the MA-HNFP model to learn the best role for each agent. The second step uses the MD concept to choose the best action to be performed by the agent, as shown in Figure 7. This action is chosen from the stocks that belong to the set of actions associated with the role.

The idea of the proposed approach is to use the MA-HNFP model to identify the best role to be played by each agent in a specific environment. Figure 8 shows the block diagram of the MA-RL-HNFP-MD model, for which the RL-HNFP structure decides the role that each agent will have in a given state of the environment. The Market-Driven model decides what action each agent must perform. This model was chosen by learning a single structure (knowledge repository) for the agents of the system because any agent can assume one of the roles associated with a situation of the environment.

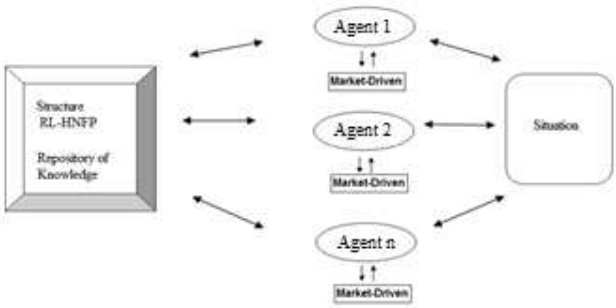

Figure 7 MA-RL-HNFP-MD model.

VI. MULTIAGENT HIERARCHICAL NEURO-FUZZY MODEL WITH COORDINATION BY GRAPHS

This model proposes the integration of the MA-RL-HNFP model with CG. The model involves a central methodology whose objective is to coordinate the agents. The actions of any particular agent depend on the actions of another agent. The model is divided into two stages. In the first stage, the RL-HNFP-MA model, which was used in the previous learning model, undertakes the role specification of an agent in every situation, as shown in Figure 9. After specifying the role of each agent, the second stage begins, where the action is chosen for each individual agent by variable elimination.

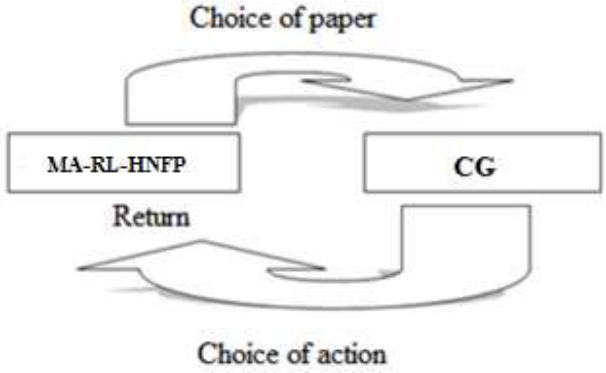

Figure 8 MA-RL-HNFP-CG model.

This model aims to reduce the number of situations in which the agents coordinate actions while minimizing the consumption of system resources. When agents have to coordinate actions, the agents learn individual actions that maximize the return to the system.

VII. CASE STUDY

The prey-predator game, also known as the "pursuit game" [21], is used as a case study. In its most traditional form, agents participate in a game with four predators and one prey, which are arranged in an orthogonal grid that is divided into rows and columns such that each grid cell cannot be occupied by more than one participant at a time. The goal is to make the predators capture the prey as quickly as possible.

The application can be considered collaborative because there is interaction between the predators for the sole purpose of capturing the prey. In this case study, MA-RL-HNFP systems were built by following the specifications, assumptions, and restrictions outlined below:

- The space consists of a 9×9 grid of orthogonal positions;
- The four agents (predators) learn a role using MA-RL-HNFP, with the goal of capturing the prey:

  Role 1: Agent captures the prey to the left;

  Role 2: Agent captures the prey to the right;

  Role 3: Agent captures the prey above him;

  Role 4: Agent captures the prey below him;

- Each role has the same four associated actions, which are chosen by market-driven coordination or graph coordination;
- The prey and predators can only move in four directions; diagonal moves are not allowed;
- Each game round consists of a move by each participant;
- The participants do not have the concept of acceleration. Each movement is always performed on a position immediately above, below, left, or right of the agent;
- Participants move alternately, not simultaneously;
- Each predator agent can see the positions of the other agents and prey;
- Predator agents do not know the goals of the other agents;
- Agents can share the 'knowledge' obtained during the learning;
- In a collision after a drive, the agent or prey returns to its original position and the round ends;
- If a participant tries to move off the grid, it returns to the previous position.

1) Training with the MA-RL-HNFP-MD model

This training covers the following stages of training: the mapping of the positions (states) in roles (shares) and the mapping of roles (states) in movements (actions), culminating in integral learning, including both collective learning (coordination of agents) and single-agent learning.

The first stage of training in the predator-prey game is to learn the preferred role (share) of each predator depending on its position with respect to the prey and other predators.

The reinforcement received by the model after each action was constructed in a manner inversely proportional to the distance between the agent and prey, as shown in Equations (11) and (12).

Distance between the agent and prey:

$$d = |(A_x - P_x)| + |(A_y - P_y)| \quad (11)$$

Reward:

$$r = 1 - d_{norm} \quad (12)$$

A_x and P_x are the positions of the agent and the prey and are attached to the x-axis. P_y and A_y are the positions of the agent and the prey and are associated to the y-axis, respectively, and d_{norm} is the distance between the agent and prey, normalized linearly between zero and one. The training process was conducted with the prey always fixed in the position (4, 4). The position of each agent at the beginning of the game, or after each capture, was also initialized in a fixed manner. The corners of the grid were positions (0, 0), (8, 8), (8, 0), and (0 8). The agents needed to get out of the corners and capture the prey, which was fixed exactly the center of the grid.

In the second stage of training, once the roles of the predators are chosen, the predators choose their preferred actions according to market-driven coordination.

2) Training with the MA-RL-HNFP-CG model

Training for the second model is similar to that for the first model and is also divided into two steps. The first step is the same as that of the first model: we learn the best role of each predator, depending on the position (state) with respect to its prey and the other predators. In the second phase of training, once the predator roles are chosen, the preferred action in the model is chosen, and this action is accomplished via graph coordination.

VIII RESULTS

In the first set of tests, 1,000 "persecution" tests were conducted, with the predators always dropping off the corners of the grid and with the prey fixed. It should be noted that 7,000 steps is ideal for catching prey because the optimal path is seven steps. In the second set of tests, 1,000 "persecution" tests were performed, with the predators starting in a random places and prey always fixed. Finally, in a third test 1,000 “persecutions” were performed with random initial positions for all predators and prey. The results are shown in the following tables, in which the results are compared with the original model, MA-RL-HNFP.

Model	Prey start	Predator start	Coordination	Pursuit time
MA-RL-HNFP	Fixed	Fixed	13,161	0%
MA-RL-HNFP-MD	Fixed	Fixed	9,200	30%
MA-RL-HNFP-CG	Fixed	Fixed	7,620	43%

Table 1 – First test

Model	Prey start	Predator start	Coordination	Pursuit time
MA-RL-HNFP	Fixed	Random	9,548	0%
MA-RL-HNFP-MD	Fixed	Random	7,355	23%
MA-RL-HNFP-CG	Fixed	Random	6,750	30%

Table 2 – Second test

Model	Prey start	Predator start	Coordination	Pursuit time
MA-RL-HNFP	Fixed	Random	9,476	0%
MA-RL-HNFP-MD	Fixed	Random	7,210	23%
MA-RL-HNFP-CG	Fixed	Random	5,940	38%

Table 3 – Third test

As can be seen from the tables above, the use of coordination methods in the MA environment greatly improved the performance in capturing the prey sooner. Depending on the initial positions of the agents, the improvement range from 23% to 43%, when compared to the original MA-HNFP model.

IX CONCLUSIONS

In this work, two separate systems were developed to evaluate the two proposed models, MA-RL-HNFP-MD and MA-RL-HNFP-CG. In both systems, agents use a shared knowledge structure. The proposed models were tested in a case study involving the pursuit game. The goal was to show how a coordination mechanism improves the performance of MASs and increases the speed of convergence and learning. The results of the proposed models were compared to the results obtained with the original MA-RL-HNFP models [10].

Tests were performed to evaluate the performances of models that include a mechanism for multiagent coordination and models with simple coordination. The tests demonstrated that the former are higher although the learning is very expensive in computational terms. The pursuit game is a simple application that has a single goal, which is to capture the prey. Even with this simple game, we can observe a difference in the performance of the MA-RL-HNFP model when a coordination model and hierarchical structure are used.

In the first test, in which the predators and prey were both fixed, both models yielded better results than in the other experiments. In all experiments, market driven coordination yielded better results than the graph coordination. In all tests, the results obtained using market-driven coordination were much better than those obtained using the other systems.

The two coordination methods discussed here are very expensive in terms of computational effort. The market-driven coordination method runs approximately four times faster than the graph coordination method on average. The market-driven learning method also converges quickly, which indicates that the method is more promising for future case studies.

REFERENCES

[1] J.R KOK, J. T HOEN, P. BAKKER, “Utile coordination: Learning interdependencies among cooperative agents”. In: Proc. IEEE Symp. Comput. Intell. Games, Colchester, U.K., p.29-36, Abr. 2005.

[2] R. FITCH, B. HENGST, D. SUC, G. CALBERT, “Structural abstraction experiments in reinforcement learning”. In: Proc. 18th Aust. Joint Conf. Artif. Intell. (AI-05), Lecture Notes in Computer Science, vol. 3809, Sydney, Australia, p.164-175, Dez. 2005.

[3] M. TAN, “Multi-agent reinforcement learning: Independent vs. cooperative agents”. In: Proc. 10th Int. Conf. Mach. Learn. (ICML-93), Amherst, OH, p.330–337, Jun. 1993.

[4] B. PRICE, and C. BOUTILIER, “Accelerating reinforcement learning through implicit imitation”. J. Artif. Intell. Res., vol. 19, p.569-629, 2003.

[5] J. CLOUSE, “Learning from an automated training agent”. Presented at the Workshop Agents that Learn from Other Agents, 12th Int. Conf. Mach. Learn. (ICML-95), Tahoe City, CA, Jul. 1995.

[6] L. BUSONIU, R. BABUSKA, and B. DE SCHUTTER, “A Comprehensive Survey of Multiagent Reinforcement Learning”. In: Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions, Volume: 38, Issue: 2, Mar. 2008.

[7] G. WEISS, S. Sen, “Adaptation and Learning in Multi-Agent Systems”. In: IJCAI "95 workshop, Montréal, Canada, August 21, 1995. Berlin: Springer, 238 p, 1996.

[8] B. GOODWINE, P. ANTSAKLIS, “Multiagent coordination exploiting system symmetries” American Control Conference (ACC), Topic(s): Robotics & Control Systems, Page(s): 830 – 835, 2010.

[9] J. O. BERNDT, and O. HERZOG, “Efficient Multiagent Coordination in Dynamic Environments” IEEE / WIC / ACM International Conferences on Web Intelligence and Intelligent Agent Technology (WI-IAT) Lyon, France. IEEE Computer Society, Pages 188-195, 2011.

[10] M. C. FRANÇA, M.M.B.R VELLASCO, K. FIGUEIREDO,; “Building Multi-agent systems with reinforcement learning hirearchical neuro-fuzzy modelos”. X Brazilian Congress on Computational Intelligence (CBIC), 2011.

[11] T. EGUCHI, K.HIRASAWA, J. HU, “A study of evolutionary multiagent models based on symbiosis.;IEEE Transactions on Systems, Man, and Cybernetics, Part B 179-193, 2006.

[12] N. VLASIS, “Collaborative multiagent reinforcement learning by payoff propagation”. Journal of Machine Learning Research, 7:1789-1828, 2006.

[13] L. P. REIS, and N. LAU; “FC Portugal - High-level Coordination Methodologies in Soccer Robotics” International Journal of Advanced Robotic Systems: Soccer Robotics, Edited by Pedro Lima, 2007.

[14] B. P. Sellner, F Heger, "Coordinated Multi-Agent Teams and Sliding Autonomy for Large-Scale Assembly," Proceedings of the IEEE - Special Issue on Multi-Robot Systems, Vol. 94, No. 7, July 2006.

[15] F. Souza, M.M.B.R. Vellasco, M.A.C. Pacheco. Hierarchical Neuro-Fuzzy QuadTree Models. Fuzzy Sets & Systems vol 130/2, 2002, 189-205.

[16] M.M.B.R. Vellasco, M.A.C. Pacheco, K. Figueiredo, F.J.de Souza, “Hierarchical Neuro-Fuzzy Systems – Part I”, Encyclopedia of Artificial Intelligence, Information Science Reference, 2008.

[17] K. Figueiredo. M. Santos, M.M.B.R. Vellasco; M.A.C. Pacheco. “Modified Reinforcement Learning Hierarchical Neuro-Fuzzy Politree Model for controlo f autonomous agents”. International Journal of Simulation Systems, Sciehce & Technology, Vol. 6, No. 10/11, pp. 4-13, 2005.

[18] M.M.B.R. Vellasco, M.A.C. Pacheco, K. Figueiredo, F.J.de Souza, “Hierarchical Neuro-Fuzzy Systems – Part II”, Encyclopedia of Artificial Intelligence, Information Science Reference, 2008.

[19] R.S. SUTTON, and A.G. BARTO, “Reinforcement Learning: An Introduction”. Cambridge, MA: MIT Press, 1998

[20] H. KOSE, U. TATLIDEDE, C. MERICLI, K. KAPLAN, “Q-Learning based Market-Driven Multi-Agent Collaboration in Robot Soccer”, Proceedings, Turkish Symposium On Artificial Intelligence and Neural Networks, pp.219-228, Izmir, Turkey, June 10-11, 2005.

[21] M. BENDA, V. JAGANNATHAN, “On optimal cooperation of knowledge sources an Empirical investigation”, Tech. Rep. BCS–G2010–28, Boeing Advanced Technology Center, Boeing Computing Services, Seattle, Washington, Jul. 1986.