12-2008

Scaling up multi-agent reinforcement learning in complex domains

Dan XIAO

Ah-hwee TAN
Singapore Management University, ahtan@smu.edu.sg

Citation

XIAO, Dan and TAN, Ah-hwee. Scaling up multi-agent reinforcement learning in complex domains. (2008). *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'08), Australia, December 9-12*. 326-329.
Available at: https://ink.library.smu.edu.sg/sis_research/6798

Scaling up Multi-Agent Reinforcement Learning in Complex Domains

Dan Xiao and Ah-Hwee Tan
School of Computer Engineering and Intelligent Systems Centre
Nanyang Technological University
Nanyang Avenue, Singapore 639798
{xiao0002,asahtan}@ntu.edu.sg

Abstract

TD-FALCON (Temporal Difference - Fusion Architecture for Learning, COgnition, and Navigation) is a class of self-organizing neural networks that incorporates Temporal Difference (TD) methods for real-time reinforcement learning. In this paper, we present two strategies, i.e. policy sharing and neighboring-agent mechanism, to further improve the learning efficiency of TD-FALCON in complex multi-agent domains. Through experiments on a traffic control problem domain and the herding task, we demonstrate that those strategies enable TD-FALCON to remain functional and adaptable in complex multi-agent domains.

1 Introduction

In a multi-agent system, a key challenge of autonomous agents is to function and adapt by themselves in a complex and dynamic environment. The joint action mechanism is commonly used [11] in multi-agent learning, but the mechanism may cause the combinatorial exploration problem when scaling up.

Addressing the issues, Tan et al. [4, 5] propose the TD-FALCON (Temporal Difference - Fusion Architecture for Learning, COgnition, and Navigation) model, a generalization of Adaptive Resonance Theory (ART) that incorporates temporal difference (TD) methods for reinforcement learning [5, 10]. It has been shown to enable an agent to adapt and function well in both navigation task and pursuit game [10, 7, 9]. However, to apply TD-FALCON for more diverse and complex domains, further improvement in learning efficiency is required.

In this paper, we present two cooperative learning strategies, known as policy sharing and the neighboring-agent mechanism (NAM), so as to make the system more flexible and adaptable in diverse and complex domains. Using policy sharing, the knowledge learned by an agent is shared among multiple agents. With the NAM, only the average states of the neighboring agents are used as the sensory inputs to each individual agent. We study two complex multi-agent domains, namely the traffic control and the herding game. Experiments show that the proposed strategies enable an TD-FALCON agent team to maintain a high level of success rate and learning efficiency.

The rest of this paper is organized as follows. Section 2 provides a summary of the TD-FALCON architecture and mechanism. We then present the two strategies for scaling up the TD-FALCON learning efficiency. Next, the experimental results in the two complex multi-agent domains are presented. The final section concludes and highlights the future work.

2 TD-FALCON

FALCON is an extension of Adaptive Resonance Theory (ART) networks [1, 2] for learning multi-modal pattern mappings across multiple input channels. For reinforcement learning, FALCON uses a 3-channel architecture [6], consisting of a sensory field F_1^{c1} representing the current state, a motor field F_1^{c2} representing the available actions, a feedback field F_1^{c3} representing the values of the feedback received from the environment, and a cognitive field F_2^c learning cognitive nodes, each of which encodes a relation among the patterns in the three input channels.

TD-FALCON incorporates Temporal Difference (TD) methods to estimate and learn value functions [6]. Given the current state s and a set of available actions $\mathcal{A}$, the FALCON predicts the value of performing each available action. The value functions are then processed by an action selection policy to select an action. Upon receiving a feedback (if any) from the environment after performing the action, a TD formula is used to estimate the value of the next state. The value is then used as the teaching signal for FALCON to learn the association from the current state and the chosen action to the estimated value. For a detailed description of the TD-FALCON model, please refer to [6].

3 Strategies for Scaling Up

3.1 Policy Sharing

Using policy sharing [3], multiple agents share a common set of knowledge, including action and value policies. In TD-FALCON, the shared policies are represented by the set of cognitive nodes created in the F_2^c field. Policy sharing can be used to accelerate the learning process among many agents, for the policy updating rate can be multiplied by the number of agents [8]. In addition, policy sharing contributes to reducing the total number of cognitive nodes learned by an agent team, because many cognitive nodes would be duplicated if they were developed individually.

In both the traffic control system and the herding game, the positions and bearings of all agents are different, but they are symmetrical in nature. As such, the cognitive nodes from one agent are reusable by the others.

3.2 Neighboring-Agent Mechanism

By extending the Center of Agent Team (CAT) algorithm [10], we propose the Neighboring-Agent Mechanism (NAM), where only the signals received from neighboring (detectable) agents are processed by each individual agent. By doing so, the state space of each agent is reduced significantly.

Assuming that in a multi-agent domain, there are n moving agents, marked as $A_1, A_2, \cdots, A_n$, and a moving target, marked as T. Let $d^{ij} (1 \le i, j \le n)$ be the distance between A_i and A_j. Let $\overline{d}^i$ be the distance between A_i and T. Let b^{ij} be the bearing from A_i to A_j. Let $\overline{b}^i$ be the bearing from A_i to T. Let σ be the Critical-Point Distance (CPD).

Definition: If $d^{ij} \ge \sigma$ or $\overline{d}^i \ge \sigma$, we say that A_j or T is **undetectable** from A_i. Otherwise, if $d^{ij} < \sigma$ or $\overline{d}^i < \sigma$, we say that A_j or T is **detectable** from A_i.

The principles of the NAM algorithm are: (1) A_i can not receive any signal from A_j until $d^{ij} < \sigma$, or any signal (except the bearing signal) from T until $\overline{d}^i < \sigma$. (2) The signals that A_i receives from multiple agents are summarized as if from the center of those agents, marked as A_I.

In general, the sensory inputs of an agent $A_i (1 \le i \le n)$ can be denoted as $S^i = (S^{it}, S^{io})$, where S^{it} and S^{io} denote sensory inputs from T and A_I respectively.

Furthermore, $S^{it} = (D^{it}, B^{it})$, where D^{it} and B^{it} stand for the distance signal and the bearing vector that A_i receives from T. We have

$$D^{it} = \begin{cases} \frac{1}{1+\overline{d}^i} & \text{if } \overline{d}^i < \sigma \\ \varepsilon & \text{if } \overline{d}^i \ge \sigma \end{cases} \tag{1}$$

where ε is a dummy value, for fixing the dimension of sensory inputs. To compute the bearing vector B^{it}, we consider eight directions as $0 \le \overline{b}^i \le 7$. Therefore

$$B_k^{it} = \begin{cases} 1 & \text{if } k = \overline{b}^i \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

where $0 \le k \le 7$.

The sensory inputs that A_i receives from A_I are denoted as $S^{io} = (D^{io}, B^{io})$, where D^{io} and B^{io} are the distance signal and the bearing vector respectively. We have

$$D^{io} = \begin{cases} \frac{1}{1+d^{io}} & \text{if other agents are detectable} \\ \varepsilon & \text{otherwise} \end{cases} \tag{3}$$

where d^{io} is the distance between A_i and A_I. If there are other agents detectable to the agent A_i, we use

$$B_k^{io} = \begin{cases} 1 & \text{if } k = b^{io} \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

where $0 \le k \le 8$, and b^{io} $(0 \le b^{io} \le 8)$ is the bearing from A_I to A_i. This bearing vector has an extra value for the coincidence of A_I and A_i. If no other agents are detectable, $B_k^{io} = 0$.

To obtain the bearing and the distance from A_I to A_i, we should know the coordinates (relative to A_i) of A_I. Assume that among the $n-1$ agents, there are p $(1 \le p \le n-1)$ agents, marked as $A_1^O, A_2^O, \cdots, A_p^O$, detectable to the agent A_i. The coordinates (relative to A_i) of an agent A_m^O $(1 \le m \le p)$ can be obtained as (x^m, y^m) $(-\sigma < x^m, y^m < \sigma)$. It follows that the coordinates (x^o, y^o) (relative to A_i) of A_I are

$$x^o = \sum_{m=1}^{p} \frac{x^m}{p}, y^o = \sum_{m=1}^{p} \frac{y^m}{p}, \tag{5}$$

and the distance d^{io} is $d^{io} = max(|x^o|, |y^o|)$.

In terms of space complexity, the state space of A_i can be computed as

$$|S^i| = |D^{it}| \cdot |B^{it}| \cdot |D^{io}| \cdot |B^{io}|. \tag{6}$$

Furthermore, $|D^{it}| = \sigma$, $|B^{it}| = 8$, $|D^{io}| = \sigma$ and $|B^{io}| = 9$. The overall state space of A_i is

$$|S^i| = 72\sigma^2. \tag{7}$$

Since σ is a constant, the state space is only $O(1)$.

As before [10], we use a hybrid reward function incorporating individual and team payoffs. The reward function of A_i is $r_i = \frac{1}{(1+d^i)(1+d^c)}$, where d^i and d^c are the distances from A_i and the CAT to T respectively.

4 Experiments

We conduct tests for the above-mentioned approaches in the traffic control system and the herding task. With the growth of the complexity, the TD-FALCON system may have to develop more cognitive nodes for more situations, causing a scalability problem.

4.1 Traffic Control System

We simulate a traffic control system by setting up cities and paths in a game field (Figure 1). The game rules are: (1) Six cities numbered from 1 to 6 are connected via paths. (2) Initially, three autonomous vehicles (AV) named A, B and C are located at the City 1, 2 and 3 respectively. The targets of AV A are City 2 and 3, the targets of AV B are City 3 and 1, and the targets of AV C are City 1 and 2. (3) In each time step, an AV can move 0, 1 or 2 squares. (4) If an AV manages to arrive at its two targets within a specified period, it is deemed a success case. Otherwise, it is a failure case. We conduct comparative experiments between policy-sharing and non-policy-sharing mechanisms on the domain.

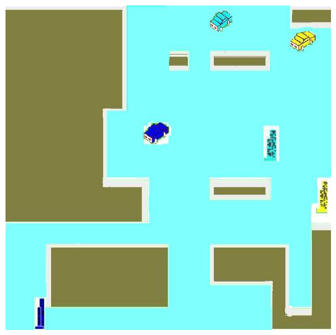


Figure 1. The traffic control system.

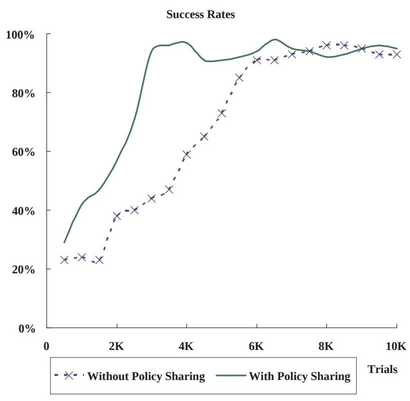


Figure 2. Success rates in traffic control.

Figure 2 shows the success rates of 4-agent TD-FALCON teams with and without the policy sharing mechanism. It can be noticed that the agent team with policy sharing converges much faster than the counterpart without policy sharing. The policy-sharing team obviously doubles the convergence speed of the non-policy-sharing team.

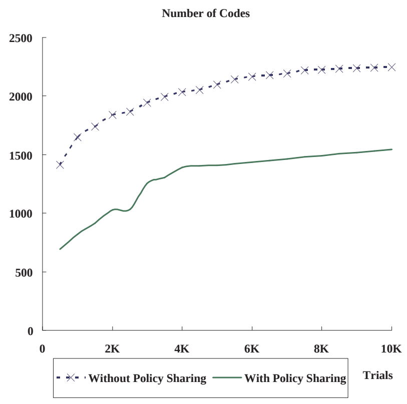
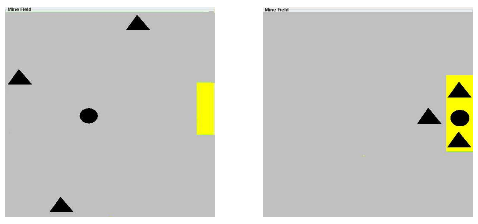


Figure 3. Number of nodes in traffic control.

Figure 3 depicts the number of nodes of 4-agent TD-FALCON teams with and without policy sharing. It can be seen that both agent teams have nearly linear rate of node increase. The non-policy-sharing team in total produces 50% more cognitive nodes than its counterpart, but the policy-sharing team produces more nodes than an individual agent from the non-policy-sharing team.

4.2 Herding Game

In the herding game, three shepherds encircle and force a sheep to enter the corral at the border of a grassland. Initially, the shepherds are located at three borders respectively and the sheep is located at the center of the grassland (Figure 4(a)). The game rules of the herding task are: (1) At each time step, a shepherd can either stay stationary or move with the pace of one square. (2) The sheep moves one square every two time steps. (3) The sheep will move towards a border if there is an open path. (4) If the sheep is forced into the corral, the game is successful (Figure 4(b)). (5) If the sheep manages to escape to a border, or the game times out, it is a failure.



(a) Starting scenario (b) Success scenario

Figure 4. Herding task (sheep are marked as circles and shepherds as triangles)

Based on a 16 by 16 game field, we organize experiments to do comparative studies of the herding task using the CAT [10], the NAM and the joint action mechanism. For the NAM team, we take $\sigma = 2$ and $\varepsilon = 0.001$.

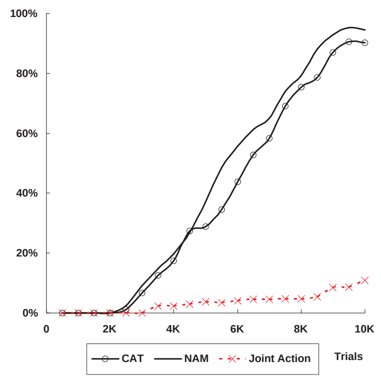


Figure 5. Success rates of the three TD-FALCON teams in the herding game.

Figure 5 highlights the success rates of TD-FALCON teams implemented with the CAT, the NAM and the joint action algorithms. It is witnessed that both of the CAT and NAM teams can function well in the herding game. However, the NAM team can produce better performance than the CAT team. Figure 5 also shows that the TD-FALCON team implemented with the joint action mechanism converges very slowly.

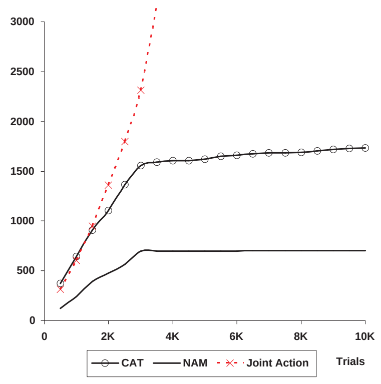


Figure 6. Number of nodes of the three TD-FALCON teams in the herding game.

Figure 6 shows the number of cognitive nodes generated from the three agent teams. We can see that the number of nodes in the joint action team increases very rapidly. It is also obvious that the NAM team produces much fewer cognitive nodes than the CAT team.

5 Conclusion

The main issues in multi-agent domains are the complexity of the task and the increased state space. Many existing works adopt the the joint action mechanism, but it is ineffective in practice. To address the issues, we have proposed the strategies of policy sharing and the neighboring-agent mechanism (NAM), that enable TD-FALCON to remain adaptable and functional in complex domains, specifically the traffic control and the herding tasks. The strategies, in principles, can also be used in other multi-agent learning algorithms/systems.

However, our work so far is limited to those domains with homogeneous agents. Future work will explore the workability of the TD-FALCON architecture in domains with heterogeneous agents.

References

[1] G. A. Carpenter and S. Grossberg. A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing*, 37:54–115, June 1987.

[2] G. A. Carpenter and S. Grossberg. ART 2: Self-organization of stable category recognition codes for analog input patterns. *Applied Optics*, 26:4919–4930, July 1987.

[3] R. Salustowicz, M. Wiering, and J. Schmidhuber. Learning team strategies with multiple policy-sharing agents: A soccer case study, 1997.

[4] A.-H. Tan. FALCON: A fusion architecture for learning, cognition, and navigation. In *Proceedings, IJCNN, Budapest*, pages 3297–3302, 2004.

[5] A.-H. Tan. Self-organizing neural architecture for reinforcement learning. In *Proceedings, J. Wang et al. (Eds.): ISNN'06, LNCS 3971, Chengdu, China*, pages 470–475, 2006.

[6] A.-H. Tan, N. Lu, and D. Xiao. Integrating temporal difference methods and self-organizing neural networks for reinforcement learning with delayed evaluative feedback. *IEEE Transactions on Neural Networks*, 9(2):230–244, 2008.

[7] A.-H. Tan and D. Xiao. Self-organizing cognitive agents and reinforcement learning in a multi-agent environment. In *Proceedings, IAT'05, France*, pages 351–357, 2005.

[8] M. Tan. Multi-agent reinforcement learning: independent vs. cooperative agents. In *Proceedings, 10th International Conference on Machine Learning*, pages 330–337, 1993.

[9] D. Xiao and A.-H. Tan. Cooperative cognitive agents and reinforcement learning in pursuit game. In *CIRAS'05*, Singapore, 2005.

[10] D. Xiao and A.-H. Tan. Self-organizing neural architectures and cooperative learning in multi-agent environment. *IEEE Transactions on Systems, Man, and Cybernetics - Part B*, 37(6):1567–1580, 2007.

[11] E. Yang and D. Gu. Multiagent reinforcement learning for multi-robot systems: A survey, 2004. Technical Report CSM-404.