1C5-2

Is XCS approach good for Organizational-Learning Oriented Classifier Systems?

Analysis through Pac-Man World

Mhd Irvan[*1] Takashi Yamada[*2] Takao Terano[*3]

[*1] Tokyo Institute of Technology irvan@trn.dis.titech.ac.jp

[*2] Tokyo Institute of Technology tyamada@trn.dis.titech.ac.jp

[*3] Tokyo Institute of Technology terano@dis.titech.ac.jp

Abstract: Organizational-learning oriented classifier system (OCS) is a learning classifier system approach to multi-agent learning. It introduces the concept of organizational learning between multiple agents that maintain their own individual classifier systems. We investigate the effectiveness of XCS classifier system's principles as parts of OCS decision-making process. The simulation is done through simulation of the proposed method inside a simplified Pac-Man world.

1. Introduction

Multi-agent system is useful to find solutions that are difficult to find by individual intelligent agent. However, there is a foundational issue for multi-agent systems: how to provide rapid performance and high accuracy in the face of new challenges emerged in dynamic environment. To address this issue we propose a learning classifier systems (LCS) approach, namely XCS, to multi-agent organizational learning.

Organizational-learning oriented Classifier System (OCS) [Takadama 1999] is a multi-agent version of LCS, utilizing the idea of organizational learning (OL) in organization and management science. It has been proven to find good solutions for multi-agent problems at small computational costs compared to traditional LCSs, such as Michigan and Pittsburgh approaches [Takadama 2000].

XCS [Wilson 1995] is an LCS that differs from more traditional LCSs. In XCS, classifier fitness is based on the accuracy of a classifier's payoff prediction instead of the prediction itself. It has been found to be successfully addressing major problems identified in other LCS implementations.

The purpose of this research is to seek the answer whether XCS properties help OCS algorithm to converge faster. To test this, we simulate the algorithms through Pac-Man-like world. In Pac-Man world situation, the perspective from the ghosts (enemies of Pac-Man) can be seen as an organizational problem, since all of them pursue the same goal (to capture Pac-Man) and may need to share knowledge amongst themselves. This kind of situation is ideal for multi-agent simulation. Details about Pac-Man world are discussed in section 4.1.

2. Related Work

Online evolutionary learning [Yannakakis 2005] has been applied to generate adaptive ghost behavior in Pac-Man game, while OCS has been implemented in multi-robot formation control for area coverage problem in a space exploration scenario [Leitner 2009] and task scheduling for space station crew [Takadama 2002]. There has been no work done in regard to OCS and Pac-Man, however.

Contact: Mhd Irvan, Tokyo Institute of Technology, irvan@trn.dis.titech.ac.jp

3. Proposed Approach

Recent successes of XCS have drawn our interest in investigating its principle for the extension of OCS. The dynamic properties of XCS for maintaining complex population of solutions should boost the decision-making performance of OCS by providing constant accurate decision compared to the original OCS.

Figure 1 describes the flow of our proposed algorithm. For detailed explanation of OCS learning mechanisms (Collective Knowledge Reuse, Rule Generation, Rule Exchange, Reinforcement Learning), readers are referred to [Takadama 1999]. The main difference between the original OCS with our

```
run OCS {
  do {
    iteration = 0;
    Collective Knowledge Reuse;
    do {
      Rule Generation;
      run xcs {
        get Input;
        generate Match Set [M] out of Population [P];
        generate Prediction Array PA out of [M];
        select Action according to PA;
        generate Action Set [A] out of [M] according to Action;
        execute Action;
        get Reward;
        if (Previous [A] is not empty) {
          calculate Payoff using Reinforcement Learning;
          update previous [A] using Payoff
          run Genetic Algorithm in previous [A];
        }
        if (End of Problem) {
          Payoff = Reward;
          update current [A] using Payoff;
          run Genetic Algorithm in current [A];
          empty previous [A];
        }
        else {
          previous [A] = current [A];
          previous Reward = current Reward;
          previous Input = current Input;
        }
      }
      Rule Exchange;
    } while (termination criteria are not met)
    iteration++;
    Collective Knowledge Reuse;
  } while (iteration < max iteration);
}
```

Figure 1: The flow of the proposed algorithm

implementation is the insertion of XCS procedures inside the solution search process, and the execution of Reinforcement Learning takes place in the XCS.

4. Experiment

4.1 Pac-Man World Problem

The simulation test-bed used in this research is a simplified version of the original Pac-Man computer game released by Namco. In our simulator, the agents act as the enemies of Pac-Man, while Pac-man is controlled through predetermined and random movements. The performance is measured by number of steps required to capture Pac-Man.

Figure 2 shows an ongoing game in our Pac-Man simulator. Pac-Man must clear all food appeared in a maze-like environment while avoiding four ghosts that chase him. The game is over when either all food is eaten by Pac-Man or the ghosts manage to capture him. In that case, the game restarts from the same initial state, and a new round begins.

For simplicity, several features from the original game are omitted. While some of these are due to lack of time, others are considered unnecessary for the purpose of this research. The main differences are:

- Pac-Man and the ghosts move at an identical speed.
- There is no tunnel connecting the edges of the maze.
- There are no fruits and power pills.
- All ghosts start immediately without waiting time.

4.2 Experimental Settings

The maze is modeled to be a close resemblance to the first stage of the original Pac-Man game. All ghosts start at the center of the stage, while Pac-Man starts at the middle of the bottom part of the stage.

Each ghost maintains his own classifier system. They sense the input from the environment: their location and Pac-Man's location. The classifier consists of input – action pair of rule, and the strength of the rule.

The game is played for 100 cycles. The game moves on to the next cycle after either Pac-Man is caught or a maximum of 1000 time steps is reached. The knowledge that the ghosts learned from the previous cycle is carried on to the next game cycle. This experiment is done two times with different Pac-Man behaviors: predetermined movements, and random movements. Finally, this simulation is conducted twice: comparing OCS with and without XCS implementation.

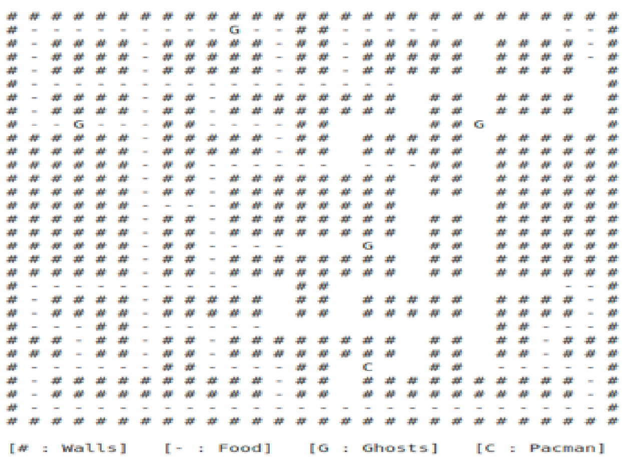

Figure 2: Snapshot of the Pac-Man game

4.3 Result

Our experiments show that the implementation of XCS helps OCS in maintaining a pool of classifiers of accurate rule-based prediction for chasing Pac-Man. It does not appear to produce aggressive movements (such as always taking the shortest path to Pac-Man). Instead, it cleverly tries to predict Pac-Man's next movements and move towards that direction.

Although the shared knowledge and rule exchange between ghosts help a ghost to predict future situation when he encounter new input, this may seem to generate unnecessary similar movements. For example, when a ghost is very near to another ghost, they tend to follow the same movements toward Pac-Man instead of taking a different path to ambush Pac-Man from the other direction.

Nevertheless, with XCS principles in place, OCS does converge to the correct solution faster for most of the time than the traditional OCS model.

5. Conclusion and Future Work

The results from the experiment suggest that XCS properties help OCS perform better than traditional OCS model. The ghosts' behaviors generated by our algorithm are interesting to investigate for future research, such as when it is put against evasive Pac-Man strategies or against human players.

Our model is still in early phase. Further tweaking and testing is necessary to improve and confirm its robustness. However, we believe that XCS has a very promising approach for future extension of OCS. An optimization for our model to improve the cooperation between agents is planned.

References

[Leitner 2009] Leitner, J., Multi-robot formations for area coverage in space applications, *Master Thesis,* Lulea University of Technology, 2009

[Takadama 1999] Takadama, K. et al., Making Organizational Learning Operational: Implications from Learning Classifier Systems, *Computational and Mathematical Organization Theory (CMOT),* Vol. 5, No. 3, pp. 229-252, Kluwer Academic Publishers, 1999

[Takadama 2000] Takadama K., Terano T., and Shimohara K., Learning Classifier Systems Meet Multiagent Environments, *Proceedings of the 3rd International Workshop on Learning Classifier Systems (IWLCS '00),* pp. 192-210, Springer, 2000

[Takadama 2002] Takadama K., Nakasuka S., and Shimohara K., Robustness in organizational-learning oriented classifier system, *Soft Computing 6 (2002),* pp. 229-239, Springer-Verlag, 2002

[Wilson 1995] Wilson, S. W., Classifier Fitness Based on Accuracy, *Evolutionary Computation,* Vol 3. No. 2, pp. 149-175, 1995

[Yannakakis 2005] Yannakakis, G. N., and Hallam, J., A Generice Approach for Generating Interesting Interactive Pac-Man Opponents, *Proceedings of the IEEE Symposium on Computational Intelligence and Games,* pp. 94-101, 2005