HELIOS2014 Team Description Paper

Hidehisa Akiyama[1], Tomoharu Nakashima[2], Katsuhiro Yamashita[2], and Satoshi Mifune[2]

Faculty of Engineering, Fukuoka University, Japan[1]
akym@fukuoka-u.ac.jp
Department of Computer Science and Intelligent Systems, Osaka Prefecture University, Japan[2]
tomoharu.nakashima@kis.osakafu-u.ac.jp,
katsuhiro.yamashita@cs.osakafu-u.ac.jp, satoshi.mifune@cs.osakafu-u.ac.jp

Abstract. HELIOS2014 is a 2D soccer simulation team which has been participating in the RoboCup competition since 2000. We recently focus on an online multiagent planning using tree search methodology. This paper describes the overview of our search framework and an evaluation method to select the best action sequence.

1 Introduction

HELIOS2014 is a simulated soccer team for the RoboCup soccer 2D simulation league. The team has been participating in the RoboCup competition since 2000, and won 2 championships and 3 runner-ups in the past 5 years of RoboCup competitions. The team has released a part of their source codes in order to help new teams to participate in the competitions and to start the research of multiagent systems.

We recently focus on an online multiagent planning using a tree search methodology. In this paper, we briefly introduce our search framework and an approach for evaluating search results in order to select the best action sequence.

2 Related Works

We have released several open source software packages that help us to develop a simulated soccer team[1]. Now, we are mainly maintaining the following software packages:

- librcsc: a base library for a simulated soccer team.
- agent2d: a sample team program using librcsc. Newbies can use agent2d as a start point for developing their own team.
- soccerwindow2: a high functional viewer, which can be used as a monitor client, a log player and a visual debugger.

[1] Available at: http://sourceforge.jp/projects/rctools/

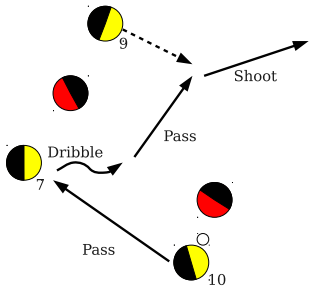

Fig. 1. An example of action sequence. This image shows the chain of four actions: 1) pass from Player 10 to Player 7, 2) dribbling by Player 7, 3) pass from Player 7 to Player 9, and 4) Player 9 shoots to the goal.

– fedit2: a formation editor for agent2d. fedit2 enables us to design a team formation using human's intuitive operations.

They are implemented from scratch without any source codes of other simulated soccer teams. However, several idea were inspired from other released codes, such as CMUnited [1], FC Portugal [2], YowAI [3], TsinguAeolus [4], UvA Trilearn [5, 6] and Brainstormers [7, 8]. We would like to thank those teams for their effort and achievements.

In previous years, we proposed a team formation model that uses Delaunay triangulation [9] and a multiagent planning framework [10] described in the next section. They have been already available in the released software.

3 Online Multiagent Planning using Tree Search

We developed a framework to search for the suitable action sequence in a continuous state-action space using a game tree search methodology. The framework enables an agent to plan cooperative behavior which involves other agents. The first version of search framework was developed in 2009, and we have been continuously improving it in various way.

The framework generates and evaluates a number of action sequences performed by multiple agents. Generated actions are stored as a node of a search tree. A path from the root node to a leaf node represents an action sequence that defines an offense plan taken by multiple agents. Figure 1 shows an example of an action sequence.

In the current implementation, we employed the best first search algorithm [11] as a tree search algorithm. Each node has a value calculated by an evaluation function based on the corresponding action-state pair instance. Since we use the best first search algorithm, the convergence of search is not guaranteed. For more details of this framework, please refer [10, 12].

4 Action Evaluation by an SIRMs Fuzzy Model

There are several ways to improve the performance of our search framework. One way is to improve the evaluation function for each action. Currently, the value of an action is evaluated based on its target position. For example, the value of a pass is evaluated from its surrounding conditions such as the position of the target and the distance to the nearest opponent player. The evaluation can be done by using a non-linear black-box system such as a neural network instead of a heuristic function that is used in the current framework. An SIRMs fuzzy model is used for this purpose. In the training process, training patterns are generated from strong teams' successful action sequences. The following subsections describe this new training process.

4.1 SIRMs Fuzzy Model

Single-Input Rule-Modules (SIRMs) [13] fuzzy model is one of common fuzzy models. The characteristic feature of SIRMs fuzzy model is that a model consists of a set of rule modules, each of which corresponds to only one input attribute. Figure 2 shows the overview of the SIRMs fuzzy model.

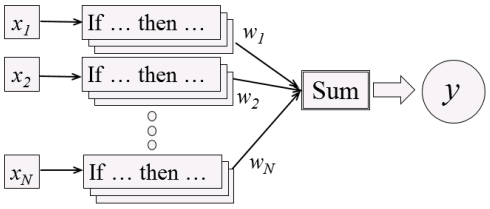

Fig. 2. Overview of an SIRMs fuzzy model

Suppose that a problem at hand has n input attributes. Then the i-th rule-module consists of the following fuzzy if-then rules:

$$R_{ij}\text{: If } x_i \text{ is } A_{ij} \text{ then } y \text{ is } y_{ij}, i = 1, \ldots, n, j = 1, \ldots, m. \tag{1}$$

where R_{ij} is the label of the j-th fuzzy if-then rule in the i-th rule module, A_{ij} is the antecedent fuzzy set, y_{ij} is the consequent real value, and m is the pre-specified number of fuzzy sets for each attribute. It is assumed that an input vector is represented as $\mathbf{x} = (x_1, x_2, \ldots, x_n)$. The i-th rule module has a weight value w_i that is used to calculate the final output y of the overall SIRMs fuzzy model as follows:

$$y = w_1 \cdot y_1 + w_2 \cdot y_2 + \ldots + w_n \cdot y_n, \tag{2}$$

where y_i is the output from the i-th rule module that is calculated by the following equation:

$$y_i = \frac{\sum_{j=1}^{m} \mu_{ij} \cdot y_{ij}}{\sum_{j=1}^{m} \mu_{ij}}, \tag{3}$$

where μ_{ij} represents the membership value of A_{ij} with x_j.

Since the SIRMs fuzzy model falls into supervised learning method, it should be assumed that there are training patterns with a target value. The parameters in an SIRMs fuzzy model are w_i, y_{ij}, and the parameters that specify the shape of membership function for A_{ij}. The learning update is based on the delta rule that minimizes the squared error function between the target output value and the real output value from the SIRMs fuzzy model.

The aim of using SIRMs fuzzy model is to learn good strategies from existing strong teams. Thus training patterns for the SIRMs fuzzy model are generated from game logs that are produced after the games of the target team finish. Our implementation of action evaluation considers the location of the point (x and y), the location of the acting player (x and y), and the distance between the point and the nearest opponent player. Thus the SIRMs fuzzy model that is used in our team has five inputs. Therefore, there are five rule modules in the SIRMs fuzzy model. The next subsection explains how to generate training patterns for the SIRMs fuzzy model.

4.2 Extraction of Training Patterns from Game Logs

From the game logs, only useful action sequences are used to train the SIRMs fuzzy model. Here the sequence that leads the ball to the opponent's penalty are referred to as "successful episodes". Successful episodes are extracted from game logs of the target teams. For each successful episodes, attacking actions such as pass, dribble, and shoot are recorded along with the information necessary for training the SIRMs fuzzy model (i.e., the position of the action point and the attacking player, and the distance between the action point and its nearest opponent). The target output value for the attacking points are set to +1.0. Other non-successful episodes are also extracted to generate the negative training patterns with the target output value -1.0.

The SIRMs fuzzy model is first initialized randomly and trained so that the output value for successful episodes becomes +1.0 and for non-successful episodes the output value becomes -1.0.

4.3 Computational Experiments

We examine the effectiveness of the SIRMs fuzzy model using simple experiments. Although the ultimate aim of this method is to make our team stronger, in the experiments in this subsection the aim is to examine whether the team performance can be enhanced to successfully bring the ball to the opponent's penalty

area. Table 1 shows the experimental results. The table shows the number of through passes during 100 games against three Japanese teams (A_TSU_BI-, HillStone, and KU_BOST) and agent2d.

Table 1. Experimental results.

Opponent	Without the SIRMs model	With the SIRMs model
A_TSU_BI-	327	378
HillStone	682	580
KU_BOST	310	403
agent2d	172	461

From this table, it can be seen that the SIRMs fuzzy model enables the team to conduct more through passes comparing to the case without the SIRMs fuzzy model. This shows the effectiveness of the method for successfully learn good strategies from strong teams.

5 Conclusion and Future Works

This paper described the research focus and the current effort of HELIOS2014. We have been applying a game tree search methodology for online multiagent planning. Now, we are trying to construct the evaluation function for selecting the best action sequence, using an SIRMs fuzzy model. Experiments shows the SIRMs fuzzy model could improve the performance of teamwork.

References

1. Stone, P., Riley, P., Veloso, M.: The CMUnited-99 champion simulator team. In Veloso, M., Pagello, E., Kitano, H., eds.: RoboCup-99: Robot Soccer World Cup III. Volume 1856 of Lecture Notes in Artificial Intelligence. Springer Verlag, Berlin (2000) 35–48
2. Reis, L.P., Lau, N., Olivéira, E.: Situation based strategic positioning for coordinating a simulated robosoccer team. Balancing Reactivity and Social Deliberation in MAS (2001) 175–197
3. Nakayama, K., Ako, T., Suzuki, T., Takeuchi, I.: Team YowAI-2002. RoboCup 2002: Robot Soccer World Cup VI (2002)
4. Yao, J., Chen, J., Cai, Y., Li, S.: Architecture of tsinghuaeolus. In: RoboCup 2001: Robot Soccer World Cup V, Springer-Verlag (2002) 491–494
5. Kok, J.R., Vlassis, N., Groen, F.: UvA Trilearn 2003 team description. In Polani, D., Browning, B., Bonarini, A., Yoshida, K., eds.: Proceedings CD RoboCup 2003, Padua, Italy, Springer-Verlag (July 2003)
6. UvA Trilearn 2003. http://www.science.uva.nl/~jellekok/robocup/2003/
7. Riedmiller, M., Gabel, T., Knabe, J., , Strasdat, H.: Brainstormers 2d - team description 2005. In Bredenfeld, A., Jacoff, A., Noda, I., Takahashi, Y., eds.: Proceedings CD RoboCup 2005, Springer-Verlag (2005)

8. Brainstormers 2d: Brainstormers Public Source Code Release. http://sourceforge.net/projects/bsrelease/
9. Akiyama, H., Noda, I.: Multi-agent positioning mechanism in the dynamic environment. In: RoboCup 2007: Robot Soccer World Cup XI. (2008)
10. Akiyama, H., Nakashima, T., Aramaki, S.: Online cooperative behavior planning using a tree search method in the robocup soccer simulation. In: Proceedings of 4th IEEE International Conference on Intelligent Networking and Collaborative Systems (INCoS-2012). (2012)
11. Russell, S.J., Norvig, P.: Artificial Intelligence: A Modern Approach (3rd ed.). Prentice Hall (2009)
12. Akiyama, H., Nakashima, T.: HELIOS2012: RoboCup 2012 Soccer Simulation 2D League Champion. RoboCup-2012: Robot Soccer World Cup XVI (2013)
13. Yubazaki, N., Yi, J., Hirota, K.: Sirms (single input rule modules) connected fuzzy inference model. Journal of Advanced Computational Intelligence and Intelligent Informatics **1**(1) (1997) 23–30