Sepanta Online Coach 2004 Team Description

Sepanta Robotic Research Foundation
Online Coach Simulation Project Team
Reza Zakery, Reza Hesamifard

coach-sim@srrf.net
{r.zakery, r.hesami}@srrf.net
http://www.srrf.net/projects/robocup/coach-sim

Abstract. This paper describes the main features of the *Sepanta Online Coach 2004* Simulation Team. This team is an extension over *Iranians 2003* which participated for the first time at the RoboCup 2003 competition and ranked 3rd place. Main features will be addressed briefly, including game analysis methods (statistical methods, matcher, formation recognizer, stamina tracker, etc.), strategy builder, rule generator, and role mapping. Finally we will describe our future research directions.

1 Introduction

RoboCup Online Coach Simulation Project subject is to develop online coaches which are able to improve players' performance by giving advice to them [1]. Player agents in soccer simulation environment act on the basis of uncertain knowledge and have single-channel, low-bandwidth, unreliable communication with each other [2]. In such an adversarial environment, the online coach that receives exact and complete information about the field could improve the team's performance. The performance improvement by the online coach also has been studied in other domains [3].

Online coach can use information extracted from previous opponent's game logs and also information gathered while the game proceeds. Although the main research domains are opponent modeling and online adaptation, the necessity of using the standard coaching language (Clang) and the fact that coachable teams are a combination of agents developed by different research groups (so aren't guaranteed to infer similarly from coach advices or even do the basic skills in a similar manner) are other challenges that an online coach must manage.

In this paper, Section 2 describes our online coach game analysis system and explains how the analysis results fed into strategy builder system; Section 3 explains our advice system

and describes how the advices are really generated using our rule generator; finally, Section 4 describes our future research directions.

2 Game Analysis

Sepanta online coach game analysis process can be divided into two phases: opponent modeling and online adaptation. Although these phases aren't fully separated in terms of their operations over row information received by the online coach from log files or soccer server, but they can be separated based on their running sequence.

2.1 Opponent Modeling

Opponent modeling deals with extracting the opponent model by observing opponent log files. This phase is done in several steps. First, online coach reads the log files and saves all the game information in a memory structure. This structure contains global velocity and position of the ball and players, play mode changes, goals scored or conceived, etc. After that, online coach does a second pass on the game data and tries to refine this information into more usable concepts. In this step, first online coach analyzer tries to find important offence and defense regions. Then a lot of statistical work is done to extract information such as number of effective shoots, passes, lost balls, dribbles, etc. Also we have a formation recognizer that extracts the opponent formation using opponent players' average position in a time window. Our online coach uses this module besides game state recognizer that associates a game state (critical, defensive, normal, offensive, and aggressive) to a sequence of cycles. By combining this information online coach tries to model the opponent behavior in different game states.

The results of opponent modeling phase are fed into a strategy builder that takes this information as input and generates some configuration files (for example different positioning configuration files) as output. Currently these files are row formatted but we are working on a XML-based format for defining strategies and hope to replace the row formatted strategy definition files with the new format[1].

2.2 Online Adaptation

The second phase of game analysis is online adaptation. When the game begins, if the coach connects to the server in online mode, soccer server sends the state information each cycle in the form of *global see* messages. These messages contain nearly all the information which online coach can extract from log files. Some exceptions exist, but all the essential information such as mobile objects' global position and velocity are included. Some other

[1] This year some new works has been done in this field by Caspian online coach team that its result is a XML-based strategy building language named *SBLang*. Probably we will use the same language or an extension over it to increase the possibility of future cooperation.

information such as goals scored or conceived, play mode changes, etc. is sent by the server using other types of messages.

In this phase online coach must trace the game and try to adapt the coachable team strategy regarding to its new observations. This adaptation may lead to a very different team strategy by for example turning off the current positioning strategy and turning on another one or just lead to changing some values in a for example positioning formula and redefining the related rules. Of course because of communication restrictions exist, it isn't supposed to define a complete strategy when the game proceeds. Instead, we are using some vectors as parameter holders and just redefine them when it is necessary.

In online adaptation phase, our online coach uses the same methods that have been used in opponent modeling phase besides some new methods. The most important new methods are matcher and stamina tracker. Matcher is a matching algorithm that tries to find a good match between opponent offenders and our defenders. It uses average positions of players and then suggests a matching that will affect the positioning formulas of our defenders and helps them to do marking and tracking with less effort in terms of time, stamina, and recovery.

Stamina tracking is another feature of our online coach. We have developed a stamina tracker that is based on a backward calculus algorithm and tries to keep the track of stamina and recovery of players in the field. Currently we use this feature to recognize when and which player is so tired and must be replaced by a *change player* command (that causes player's stamina and recovery refill). Also this method can be used to recognize tired opponent players and use this tiredness to make score. Of course we haven't tested this yet.

3 Advice System

Our online coach advice system is based on generating rules automatically from some row formatted configuration files. These files are coming from two main sources: hand written, and opponent modeling system. We have a subsystem called rule generator that reads these configuration files and generates Clang rules from them. Using row formatted configuration files causes some problems such as requiring new parsers for each new type of configuration file, lacking some useful capabilities such as loops, decision nodes, etc. So as mentioned before, we are going to replace all of this structure with a XML-based format.

Hand written configuration files are required because the coachable players have no default embedded strategy and we have to define some basic strategy for them. This basic strategy includes basic positioning, basic defense strategy, basic offence strategy, and goalie game restart. Besides these, there are some hand coded plans that supposed to be used as default plans when the online coach has no specific idea about how to deal with an opponent.

Strategy builder as a subsystem of opponent modeling system generates the next series of configuration files. In fact these configuration files are some copies of configuration templates which are filled with suitable information (nearly always some numerical data). Currently we have a fixed set of strategy templates that strategy builder is restricted to them. There are some ideas about making this structure more flexible and letting the

strategy builder to use some sub-strategy templates and combine them to build new strategies.

Before calling the rule generator to generate the required rules, online coach must decide which plans to use. Currently there is no restriction in terms of total number or total size of messages that coach can send before the game starts, so the online coach must be able to define all the plans and then turn on which thinks to be better, but because sending a lot of messages to coachable agents may arise some problems our online coach chooses plans that thinks are more suitable and just sends rules related to them.

4 Role Mapping

Another challenge that the online coach must encounter with arises from the fact that as mentioned before coachable agents are developed by different research groups and aren't guaranteed to infer similarly from coach advices or even do the basic skills in a similar manner. This problem sometimes even forces the online coach to use different rules to describe one concept for different agents. Besides this, because different coachable agent developers use different base codes to extend their agent over it, these agents usually have different performances in different roles. For example agent A may outscore agent B in implementing defensive strategies and agent B may outscore agent C in implementing offensive strategies.

The online coach is supposed to be able to arrange the players in a way that helps the coachable team to make the best possible score. To do that, we must use roles instead of uniform numbers to generate the rules. On the other side, Clang rule definition is based on uniform numbers, not roles. To overcome this problem, our rule generator uses a role to uniform number mapping. This is done by using a special configuration file that is read by rule generator.

5 Future Works

We have finalized implementing the general structure of our online coach. This has resulted in a source code release of our last year team (Iranians 2003)[2]. In combination with the extensive documentation throughout the code, we believe this is a good starting point for new teams. From this point forward, we first focus on switching from row configuration files to a XML-based strategy definition language.

We also are encouraged in testing the clustering approach used by UT Austin [4] online coach and compare its performance against our current analysis methods.

The last and the most interesting domain that we like to apply it to our online coach is fuzzy logic. Coaching a soccer team is a qualitative process that deals with concepts which can not be defined exactly. The nature of coaching causes us to guess that using fuzzy logic in opponent modeling and online adaptation systems will increase our online coach performance.

[2] Available from our website: http://www.srrf.net/projects/robocup/coach-sim

References

1. Mao Chen, Ehsan Foroughi, Fredrik Heintz, Spiros Kapetanakis, Kostas Kostiadis, Johan, Kummeneje, Itsuki Noda, Oliver Obst, Patrick Riley, Timo Steffens, Yi Wang, and Xiang Yin. *Users manual: Robocup soccer server manual for soccer server version 7.07 and later*, 2003. Available at http://sourceforge.net/projects /sserver/.
2. Peter Stone and Manuela Veloso. Task Decomposition, Dynamic Role Assignment and Low-Bandwidth Communication for Real-Time Strategic Teamwork. Artificial Intelligence, Vol. 100, June 1999.
3. Patrick Riley and Manuela Veloso. *Coaching Advice and Adaptation*. In Daniel Polani, Andrea Bonarini, Brett Browning, and Kazuo Yoshida, editors, RoboCup-2003: The Sixth RoboCup Competitions and Conferences, Springer Verlag, Berlin, 2004 (to appear).
4. Gregory Kuhlmann, Peter Stone and Justin Lallinger. *The Champion UT Austin Villa 2003 Simulator Online Coach Team*. In Daniel Polani, Brett Browning, Andrea Bonarini, and Kazuo Yoshida, editors, RoboCup-2003: Robot Soccer World Cup VII, Springer Verlag, Berlin, 2004 (to appear).