ROBOT SOCCER STRATEGY ADAPTATION

Václav Svatoň[(a)], Jan Martinovič[(b)], Kateřina Slaninová[(c)], Václav Snášel[(d)]

[(a),(b),(c),(d)]IT4Innovations, VŠB - Technical University of Ostrava,
17. listopadu 15/2172, 708 33 Ostrava, Czech Republic
[(a),(d)]Faculty of Electrical Engineering and Computer Science, VŠB - Technical University of Ostrava,
17. listopadu 15/2172, 708 33 Ostrava, Czech Republic

[(a)]vaclav.svaton@vsb.cz, [(b)]jan.martinovic@vsb.cz, [(c)]katerina.slaninova@vsb.cz, [(d)]vaclav.snasel@vsb.cz

ABSTRACT
The robot soccer game presents an uncertain and dynamic environment for cooperating agents. Robot soccer is interesting for its multi-agent research, real-time image processing, robot control, path planning and machine learning. In our work, we present an approach to describe the strategies of the robot soccer game and propose a method of strategy adaptation based on the information from the previously played games and therefore acquiring better results than with the original strategy. Because this robot soccer strategy describes a real space and stores physical coordinates of real objects, proposed methods can be used in strategic planning in different areas where we know geographic positions of the objects.

Keywords: strategy adaptation, strategy planning, robot soccer, time-series

1. INTRODUCTION
The game situation on the playground in robot soccer games is typically read in terms of the robot's postures and the ball's position. Using near real-time information of this dynamically changing game situation, the system of robot soccer game would need to continually decide the action of each team robot, and to direct each robot to perform a selected action.
A strategy, as understood by a game theory (Osborne 2004; Kim, Kim, Kim, and Seow 2010), is a complete set of options which are available to players in any given situation in order to achieve the desired objective. This principle can be applied to a number of areas from the real world and is generally called strategy planning (Ontanón, Mishra, Sugandh, and Ram 2007). Games can in general represent any situation in the nature. Game theory could be applied for adversarial reasoning in security resource allocation and scheduling problems. Our approach is to use the strategies to describe a space and objects in it, and to use the subsequent search for the optimal path or relocation of these objects in order to achieve the desired goals.
The very objective of the robot soccer game is as simple as in real soccer. Win the game over an opponent by a higher number of scored goals. To achieve this goal, the best possible cooperation of team players and the adaptability to the opponent's actual strategy is necessary.
This paper is organized as follows: First, basic overview of robot soccer strategy and rule selection will be introduced in Section 2. Then, the proposed approach for the strategy adaptation based on the information from played games will be described in detail. In the Experiments and Results section, results of proposed approach will be demonstrated and evaluated. At the end, both advantages and disadvantages of the approach will be summarized and the future work will be outlined.

2. ROBOT SOCCER STRATEGY AND RULES
Our robot soccer architecture uses two types of representation of the game field (Martinovič, Snášel, Ochodková, Zoltá, and Wu 2010). Very accurate abstract coordinate system representation used to control the robots with the high precision and the grid coordinate system used for the strategy definition and the underlying rules (see Figure 1).

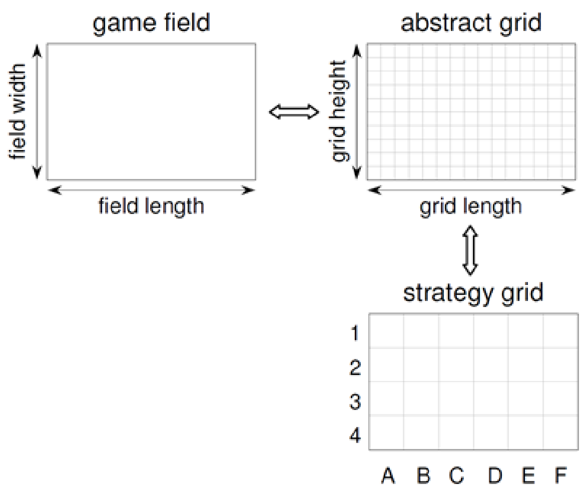

Figure 1: Game Field Representation

By using grid coordinates we reduce the accuracy of the mapping of physical coordinates of the robots into the logical ones, but on the other hand, we dramatically

reduce the number of the rules needed to create a strategy. If necessary, it is possible to convert the grid coordinates back to the physical and to use them for the aforementioned robot control.
Strategy is a finite set of rules that describes the current situation on the game field. Each rule can be easily expressed as the quaternion containing the grid coordinates of our robots, the grid coordinates of opponent's robots, grid coordinates of the ball and the grid coordinates of where our robots should move in the next game step.

2.1. Rule Selection

The selection of the appropriate rule from strategy depends on the current game situation. This situation is represented by the current state of the game that contains information about the position of the robots and the ball on the game field and also their rotation and speed. On the basis of a priori defined metrics the most similar rule from the strategy is selected. This rule also contains the destination coordinates of the players for the following game step.

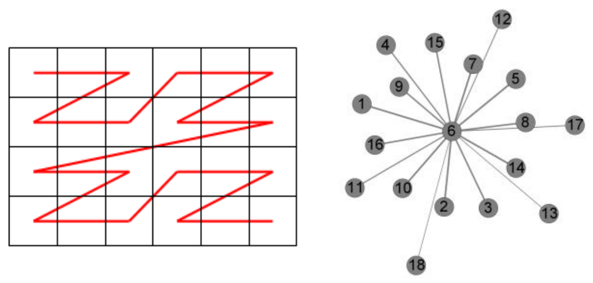

Figure 2: Z-order Mapping and Rule Graph

Because our robot soccer system architecture does not include the robots which are uniquely identifiable by ID or assigned roles, the method based utilizing the graph and Z-order curve (see Figure 2) was devised (Svatoň, Martinovič, Slaninová, and Snášel, 2014). Z-order is a function mapping the multi-dimensional space into the one-dimensional space while preserving the locality of data points. Due to its properties, it is used for converting two-dimensional matrix representing the playing field into one dimensional array of the coordinates of the individual robots. Before the start of the game the graph representing the similarities between the rules is precomputed. The set of vertices consists of the individual rules from the strategy and the edges contain the evaluation which corresponds to a distance between the two neighboring vertices (rules). As a distance is considered a normalized value of Euclidean distance computed from two sorted sequences using the above mentioned Z-order applied to the neighboring vertices which contain the robots grid coordinates.
Using this approach to a rule selection from the strategy, we were also able to utilize the more accurate description of strategies using substrategies. Substrategies allow the author to create a strategy, which will be performed during the game by such way, by which it was intended during its creation. Substrategies represent game situations such as left wing offence or right wing defense and ensures that players will perform the strategy actions more continuously and in a faster way then using the original approach.

3. STRATEGY ADAPTATION

Robot soccer game is dynamic and fast changing environment. In order to properly execute number of different tasks towards the given objective, the system should be able to evolve and have the flexibility to adapt to an opponent´s strategy. Different approach to a similar problem illustrated the use of the fuzzy decision making system (Huang and Liang, 2002) or the use of evolutionary algorithms inspired by biological evolution, such as reproduction, mutation, recombination, and selection to approximate the solution to a problem of strategy selection and adaptation. The work of Nakashima et al. (Nakashima, Takatani, Udo, Ishibuchi, and Nii, 2006) proposes an evolutionary method for acquiring team strategies of RoboCup soccer agents. They define a chromosome as a concatenated string of action rules for all agents. Larik et al. (Larik and Haider, 2016) used evolutionary algorithms for strategy optimization problem, Tominaga et al. (Tominaga, Takemura and Ishii, 2017) proposed an approach using SOM neural networks, Shengbing et al. utilized approach based on swarm intelligence methods (Shengbing, Gang and Xiaofeng, 2016) and for example decision making methods like the work of Akiama et al. (Akiyama, Tsuji and Aramaki, 2016).
There exist many different types of robot soccer games with many different architectures used in these games so it is understandable that the proposed methods for strategy description, rule selection and strategy adaptation are heavily dependent on the used robot soccer game architecture. Also the common problem is that different approaches have different understanding of the term strategy. Some of them see the problem as a low level description of how to move the robot, others as a high level abstraction that is used for the robot control with the higher granularity or from the global point of view.

Using our own architecture, the proposed method for strategy adaptation consist of the following steps:

1. Extract the information about the game progress from a log of a played game.
2. Detect the relevant game situations for strategy adaptation.
3. Analyze the rules preceding the relevant situation.
4. Create aggregated anti-rule/s for the relevant situation.
5. Include the newly created rules to the original strategy.

The basic idea is to reveal weak spots within our own strategy based on the progress of the game played against an opponent and to be able to adapt our strategy to it. In the end having the strategy with the possibility of gaining better results than with the original one. The proposed method can be used for a one-time adaptation from the logs of the played games or for a real-time adaptation during the game.

3.1. Game Information Extraction

The experiments described in Subsection 3.3 were created using our robot soccer architecture and 3D robot soccer simulator (see Figure 3). The game was played between the two teams each consisting of five robots. The standard game lasts for 2 minutes and simulator logs the current state on the game field every 20ms. The final log file consists of 6,000 records containing strategy rule selected for each team, grid coordinates of every robot and the ball, score and game time.

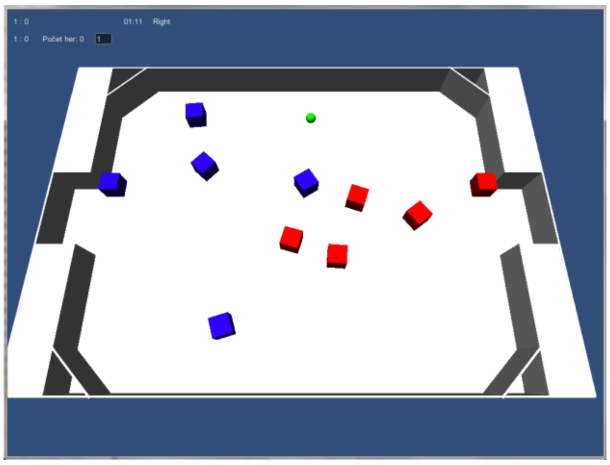

Figure 3: 3D Robot Soccer Simulator

Therefore, we are able to parse a log file for information relevant to a game situation (opponent’s scored goal, etc.) and to extract the strategy rules and grid coordinates of robots preceding this game situation.
The rule selection is invoked every 20ms during the game but during those 20ms the robots do not have enough time to move too far on the game field. Keeping that in mind, it is necessary to extract a wide enough time interval from the log preceding the detected relevant situation. One robot soccer game lasts for 2 minutes therefore it is sufficient to analyze the past 3 seconds which are represented by 150 log entries that will be used for the strategy adaptation. This should be long enough time to actually react to a current situation on the game field.

3.2. Strategy Adaptation

Game information extracted from the log is used to create anti-rules. There are 150 log entries for each detected game situation (strategy weakness) that need to be transformed to new rules that should prevent this kind of situation in the future.
Log entries are therefore divided by the offset of 50 entries (1 second of a game) and aggregated to a single rule resulting in 3 new rules for a given game situation.
The aggregation is performed by averaging the grid coordinates of the selected 50 log entries by all the robots and the ball therefore creating one aggregated rule describing 1 second of the given game situation.
In the final phase of the rule adaptation, the modification of the Move coordinates within the rule is applied. These coordinates are overwritten for the two robots to move to a position of the ball. By this way, we are able to create a new anti-rule covering the weak spot of our strategy (missing game situation within the strategy) that is trying to prevent this kind of situation in the future.

Overall process of the rule extraction and strategy adaptation is as follows:

1. Split the 150 log entries by 50 entries for every relevant game situation.
2. Compute average rule from the 50 log entries.
3. Change the grid Move coordinates for 2 of the robots to move towards the ball.
4. If the created anti-rule is not already in the strategy, insert it as a new rule.

This process was used not only for the detection of relevant game situations in the defense but also for the situations occurring during the offensive play. In terms of the defense play, opponent’s scored goal is considered as a relevant game situation. This action can be detected from the log file of the played game or during the current game.
The way to improve our team’s offensive play is to detect a game situation where our team was attacking but was unable to score the goal; therefore the ball was in front of the opponent’s gate but the play did not ended with the scored goal and the ball was afterwards kicked off to some other part of the game field. The idea for the offensive game adaptation is to find these unsuccessful offensive situations and to change the destination coordinates of selected robots to move closer to the opponent’s gate to put more pressure on the goalkeeper.
The overall process of the strategy adaptation can be used in post-game analysis as a way to train the current strategy on a number of already played games or as a fully automatic process that is performed during the ongoing game. The static adaptation method has an advantage in a potentially big initial training set whereas the automatic adaptation process is much more dynamic and therefore able to quickly react to the opponent’s current behavior during the game. Both of these approaches were tested and the results summarized in the following chapter.

3.3. Experiments and Results

Experiments were performed using our robot soccer architecture and within the 3D robot soccer simulator. The game was played between the left team strategy showed in Table 1 representing the strategy with the weak spots (strategy does not contain rules covering the left wing defense and center defense and is mostly focused on offense) and reference strategy of the right team (complete strategy containing basic offense and defense game situations).

Table 1: Left Team Strategy

Substrategy	#	Coordinate	Rule Desc.
Offensive Middle	1	Mine	4,2 4,3 2,2 2,3
		Oppnt	4,2 4,3 5,1 5,4
		Ball	4,2
		Move	5,2 5,3 2,2 3,3
	2	Mine	5,2 5,3 2,2 3,3
		Oppnt	5,2 5,3 5,1 5,4
		Ball	5,2
		Move	6,2 5,3 2,2 3,3
	3	Mine	6,2 5,3 2,2 3,3
		Oppnt	5,2 5,3 5,1 5,4
		Ball	6,2
		Move	6,2 5,3 2,2 3,3
Offensive Left	4	Mine	3,2 3,3 3,1 2,3
		Oppnt	4,1 4,2 5,1 5,3
		Ball	3,1
		Move	4,2 3,2 4,1 3,3
	5	Mine	4,2 3,2 4,1 3,3
		Oppnt	4,1 4,2 5,1 5,3
		Ball	4,1
		Move	5,2 4,2 5,1 3,3
	6	Mine	5,2 4,2 5,1 3,3
		Oppnt	4,1 5,2 5,1 5,3
		Ball	5,1
		Move	5,2 4,3 5,1 3,2
	7	Mine	5,2 4,3 5,1 3,2
		Oppnt	4,1 5,2 5,2 5,3
		Ball	5,2
		Move	6,2 5,3 5,1 3,2
	8	Mine	6,2 5,3 5,1 3,2
		Oppnt	4,2 5,2 6,2 5,3
		Ball	6,2
		Move	6,2 5,3 5,1 3,2
Defensive Right	9	Mine	3,1 3,3 2,1 2,2
		Oppnt	3,1 3,2 5,2 4,3
		Ball	3,1
		Move	2,1 2,3 1,1 2,2
	10	Mine	2,1 2,3 1,1 2,2
		Oppnt	2,1 2,2 4,2 3,3
		Ball	2,1
		Move	2,2 1,3 1,2 2,3
	11	Mine	2,2 1,3 1,2 2,3
		Oppnt	1,2 2,2 4,2 2,3
		Ball	1,2
		Move	2,2 1,3 1,2 2,3

Ten robot soccer games were played between the original left team strategy and the referenced one for the right team. Result scores from these games are visible in Table 2. Resulting score sum is 5:9 in favor of the right team strategy. Left team was able to win 3 times, lost in 5 games and achieved 2 draws. The lack of good defense in the strategy was obvious during the games as most of the opponent's score goals were scored via left side of the game field.

Table 2: Score Table – Original vs. Reference

Game #	Left Score	Right Score	Result
1	0	1	Loss
2	1	0	Win
3	0	2	Loss
4	0	2	Loss
5	1	0	Win
6	1	1	Draw
7	1	0	Win
8	0	1	Loss
9	1	1	Draw
10	0	1	Loss
SUM	**5**	**9**	

Adaptation from the log files

First experiments were performed using the static strategy adaptation method. Log files generated from the games of a Left team strategy and the Right team reference strategy were used for the static strategy adaptation mechanism, described in Subsection 3.2. Each scored goal for the opponent's team was detected as relevant game situation and was used to create 3 anti-rules resulting in sum of 27 new and different rules (see Table 3).

These rules were included to the original strategy for the left team and played once again against the reference strategy. The results are shown in Table 4. The adapted strategy scored the same amount of goals as the reference strategy with two wins, one loss and seven draws. The adapted strategy is now able to compete with the reference one thanks to the new rules that are able to react to the previously unsuccessful game situations.

Table 3: Adapted Defense Rules

#	Mine	Oppnt	Ball	Move
1	32 51 62 33	62 13 52 62	13	32 51 13 13
2	32 51 62 43	62 13 52 62	13	32 51 13 13
3	32 51 62 53	62 13 52 62	13	32 51 13 13
4	21 32 52 33	52 32 42 53	21	21 32 21 21
5	21 42 52 33	52 32 42 43	11	21 42 11 11
6	21 42 52 33	52 22 42 43	12	21 42 12 12
7	51 53 32 52	63 62 52 44	44	51 53 44 44
8	41 53 33 62	63 62 52 33	33	41 53 33 33
9	42 53 23 62	63 62 52 23	13	42 53 13 13
10	32 62 51 24	33 14 22 53	14	32 62 14 14
11	32 62 51 14	33 13 22 53	13	32 62 13 13
12	32 62 51 24	33 13 22 53	13	32 62 13 13
13	22 33 32 24	52 43 42 53	32	22 33 32 32
14	22 33 32 34	52 32 32 43	23	22 33 23 23
15	22 33 33 43	53 32 32 33	13	22 33 13 13

16	22 33 52 43	52 23 22 43	23	22 33 23 23
17	22 23 52 43	52 33 22 53	23	22 23 23 23
18	22 23 52 43	52 22 32 53	12	22 23 12 12
19	22 44 62 33	62 53 52 54	34	22 44 34 34
20	22 44 62 34	62 53 52 44	23	22 44 23 23
21	22 54 62 23	62 53 52 44	13	22 54 13 13
22	22 53 33 52	62 53 52 63	52	22 53 52 52
23	22 52 33 52	62 53 42 63	42	22 52 42 42
24	22 53 33 62	62 53 42 63	23	22 53 23 23
25	62 32 53 24	24 52 62 53	14	62 32 14 14
25	62 22 53 14	13 52 62 53	13	62 22 13 13
26	62 22 53 23	13 52 62 43	13	62 22 13 13
27	32 51 62 33	62 13 52 62	13	32 51 13 13

Table 4: Score Table – Adapted Defense vs. Reference

Game #	Left Score	Right Score	Result
1	0	0	Draw
2	0	0	Draw
3	0	2	Loss
4	0	0	Draw
5	0	0	Draw
6	0	0	Draw
7	1	0	Win
8	1	0	Win
9	1	1	Draw
10	1	1	Draw
SUM	**4**	**4**	

Using the same training set of 10 previously generated log files, the strategy was further updated using the mechanism described in Subsection 3.2 with the intent to improve the offensive play of the Left team strategy. During the current robot soccer game, there is a number of unsuccessful goal attempts on the side of each competing team therefore also bigger number of detected relevant situations can be expected resulting in bigger number of generated offensive rules.

Proposed adapting mechanism generated 138 new and unique rules and because 3 rules are created for each detected relevant situation (3 seconds of game play preceding this situation) the number of detected situations can be roughly estimated to 46. This number is probably higher as the same or very similar rules are considered as duplicates and therefore they are discarded from the strategy addition. The new rules were added to the Left team strategy and again played against the reference one from the right team. The results are shown in Table 5.

Table 5: Score Table – Adapted Defense + Offense vs. Reference

Game #	Left Score	Right Score	Result
1	1	0	Win
2	2	0	Win
3	1	1	Draw
4	3	1	Win
5	0	0	Draw
6	2	0	Win
7	1	0	Win
8	1	1	Draw
9	0	1	Loss
10	2	0	Win
SUM	**13**	**4**	

The results show that the Left team strategy was able to win the match over the opponent's strategy in 6 cases, achieved 3 draws and lost only in one played game. The difference in the resulting strategy is most visible on the number of scored goals. The original Left team strategy lost the game over an opponent by a higher number of scored goals. The same strategy adapted by 27 defensive rules was able to achive more balanced gameplay but the strategy consisting of both deffense and offense adaptation rules, comprising of 176 rules, was able to score 9 more goals than the opponent‘s strategy over the course of 10 matches.

Adaptation during the game

The same mechanism of strategy adaptation from the defensive and also offensive point of view was used directly in our robot soccer simulator (see Figure 3) during the actual game between the original Left team strategy (Table 1) and the reference strategy used for the right team.

The whole adaption process was implemented as a part of our robot soccer library. This library is an integral part of our 3D robot soccer simulator and contains all functionality necessary for robot control, rule selection from strategy, path planning, tactics execution and much more. The pseudocode for the main part of the process is shown in Algorithm 1.

```
//called every game step - 20ms
AdaptStrategy(LogHolder log, Strategy str)
{
    //get defense rules
    List<Rules> defRules = AdaptDefense(log, str);
    //get offense rules
    List<Rules> offRules = AdaptOfense(log, str);

    foreach defRules
    {
        //if the rule is unique, insert into strategy
        if(SimilarRuleCheck(defRules, str))
        {
            AddRulesToStrategy(defRules);
        }
    }
    foreach offRules
    {
        //if the rule is unique, insert into strategy
        if(SimilarRuleCheck(offRules, str))
        {
            AddRulesToStrategy(offRules);
        }
    }
}
AdaptDefense(LogHolder log, Strategy str)
{
    //check current game situation
    LogInfo lastEntry = GetLastLogEntry(log);

    //relevant situation detection, did opponent score a goal?
    if(GoalCheck(lastEntry))
    {
        //extract previous 3s of gameplay, 150 log entries
        ExtractPreviousEntries();

        //generate new rule, move selected players closer to
        //left team gate
        foreach 50 entries
        {
            List<Rules> defRules =
                GenerateDefensiveRule();
        }
    }
    return defRules;
}
AdaptOffense(LogHolder log, Strategy str)
{
    //check current game situation
    LogInfo lastEntry = GetLastLogEntry(log);

    //relevant situation detection, was the ball before the
    //opponent's gate but the goal was not scored?
    if(UnsuccessfulGoalCheck(lastEntry))
    {
        //extract previous 3s of gameplay, 150 log entries
        ExtractPreviousEntries();

        //generate new rule, move selected players closer to
        //right team gate
        foreach 50 entries
        {
            List<Rules> offRules =
                GenerateDefensiveRule();
        }
    }
    return offRules;
}
```
Algorithm 1: Real-time Adaptation Pseudocode

The offensive adaptation mechanism is able to generate a large number of rules therefore it was necessary to use the effective algorithm for rule comparison part of the implementation that is checking whether the generated rule (or a very similar one) is already part of the current strategy. This effective rule comparison algorithm was already developed as a part of a rule selection mechanism described in Subsection 2.1 and thus simply put to use also in a proposed strategy adaptation process. Each generated rule is simply compared to every rule from the current strategy based on their Z-order coordinates. This implementation uses space filling curves to order the robots on the game field thus saving time during rule selection and rule generation in the strategy evaluation and adaptation phase of the robot soccer game.

The real-time adaptation experiments were performed for 5 sets of played games. Each set consists of 10 matches between the original Left team strategy (see Table 1) and the right team reference strategy. During this matches the left team was able to perform automatic defensive and offensive strategy adaptation therefore able to detect the relevant game situations and to add newly generated rules to its own strategy set. The adapted strategy remained stored over the course of these 10 games thus able to improve in each game iteration.

The results from these game sets are shown in Tables 6 to 10. Overall results represented by a number of scored goals show that the adapted Left team strategy is a balanced opponent (3 wins and 2 loses) for the reference strategy and is able to score a sufficient number of goals against it. The tables also contain the number of automatically added defensive and offensive rules for each game iteration.

Table 6: Score Table – Real-Time Adaptation, Set 1

Game #	Left Score	Right Score	Defense Adapt Rules	Offense Adapt Rules
1	0	0	0	10
2	1	2	6	12
3	0	1	8	20
4	0	1	11	24
5	2	0	11	27
6	4	0	11	31
7	0	1	13	32
8	2	2	17	37
9	0	2	20	40
10	1	0	20	41
SUM	**10**	**9**	**20**	**41**

Table 7: Score Table – Real-Time Adaptation, Set 2

Game #	Left Score	Right Score	Defense Adapt Rules	Offense Adapt Rules
1	0	3	5	3
2	0	1	5	12
3	1	1	5	19
4	2	1	5	20
5	0	1	5	23
6	1	0	5	25
7	1	0	5	34
8	0	1	5	39
9	0	0	6	41
10	1	0	6	42
SUM	**6**	**8**	**6**	**42**

Table 8: Score Table – Real-Time Adaptation, Set 3

Game #	Left Score	Right Score	Defense Adapt Rules	Offense Adapt Rules
1	1	2	3	8
2	1	0	3	11
3	4	1	6	12
4	2	1	8	12
5	1	3	14	18
6	0	1	14	28
7	0	1	16	31
8	1	0	16	32
9	1	0	16	40
10	1	0	16	44
SUM	**12**	**9**	**16**	**44**

Table 9: Score Table – Real-Time Adaptation, Set 4

Game #	Left Score	Right Score	Defense Adapt Rules	Offense Adapt Rules
1	2	2	1	11
2	0	1	3	14
3	1	2	4	17
4	2	0	4	22
5	1	1	6	22
6	0	0	6	24
7	0	2	8	30
8	2	1	10	35
9	0	0	10	35
10	0	1	12	35
SUM	**8**	**10**	**12**	**35**

Table 10: Score Table – Real-Time Adaptation, Set 5

Game #	Left Score	Right Score	Defense Adapt Rules	Offense Adapt Rules
1	1	1	3	6
2	1	0	3	13
3	0	0	3	19
4	0	1	4	27
5	0	1	7	30
6	1	0	7	32
7	0	1	10	38
8	0	0	10	40
9	2	0	10	42
10	1	1	11	44
SUM	**6**	**5**	**11**	**44**

Following the proposed adaptation mechanism, the results show that the new defense adaptation rules are added only in the case of opponent's goal thus enabling the left team to learn the opponent's attack pattern and devise a rules to block it.
The largest number of rules is also added in the first game iterations but that is understandable because the smaller the rule set of a strategy the smaller chance that the newly generated rule will be the same or similar to some other rule from the strategy. As the game progresses the number of rules in the strategy also grows, more generated rules are discarded because of it during the game.

3.4. Validation and Visualization

The proposed approach could be validated using the sequence extraction and comparison which is also very useful for the visualization. This method proceeds from the original social network approach with the adaptation to robot soccer games. The sequences or game profiles can be extracted from the log that contains data related to rules selected from the game strategy during the game.

The definition of a game profile is as follows:

Let $U = \{u_1, u_2, \dots u_n)$, be a set of games, where n is a number of games u_i. Then, sequences of strategy rules $\sigma_{ij} = <e_{ij1}, e_{ij2}, \dots e_{ijm_j}>$, are sequences of strategy rules executed during a game u_i in the simulator, where $j = 1, 2, \dots, p_i$ is a number of that sequences, and m_j is a length of j-th sequence. Thus, a set $S_i = \{\sigma_{i1}, \sigma_{i2}, \dots \sigma_{ip_i}\}$ is a set of all sequences executed during a game u_i in the system, and p_i is a number of that sequences.
Sequences σ_{ij} extracted with relation to certain game u_i are mapped to set of sequences $o_l \in S$ without this relation to games: $\sigma_{ij} = <e_{ij1}, e_{ij2}, \dots, e_{ijm_j}> \longrightarrow \sigma_l = <e_1, e_2, \dots, e_{m_l}>$, where $e_{ij1} = e_1, e_{ij2} = e_2, \dots, e_{ijm_j} = e_{m_l}$.
Define matrix $B \in N^{|U|x|S|}$ where

$$B_{ij} = \left\{ \begin{array}{r} \text{frequency of sequence } \sigma_j \in S \text{ for game } u_i \text{ if} \\ \sigma_j \in S_i, \\ \text{else } 0 \end{array} \right\}$$

A base game profile of the games $u_i \in U$ is a vector $b_i \in N^{|S|}$ represented by row i from matrix B.

Each sequence is labeled with a sequence number and number determining the possession of the ball (0 - none, 1 - left team, 2 - opponent's team). Each sequence contains a list of rules selected for the left team in every game step until the team possession of the ball has changed. These sequences are then compared using the method for sequence comparison – LCS (longest common substring), LCSS (longest common subsequence) and T-WLCS (time-warped longest common subsequence). Results from the sequence comparison can be visualized by clustering of similar sequences.
Visualization of LCSS method for the original Left team strategy and the strategy adapted by defense rules from the log files (Table 3) can be seen in Figures 4 and 5. The clusters of similar sequences often represent the game situations defined within the original strategy. Therefore, the original strategy should contain the original game situations and the adapted strategy should also contain new clusters consisting of newly created anti-rules.
In most cases, the strategies are created manually with some specific game situations in mind. For example, the first X number of rules representing the left wing defense, next Y number of rules representing right wing

offense etc. Therefore, the above-described sequence extraction method and subsequent sequence comparison and visualization could be also used for the strategy validation and visualization of the progress of the robot soccer game in relation to strategy adaptation.

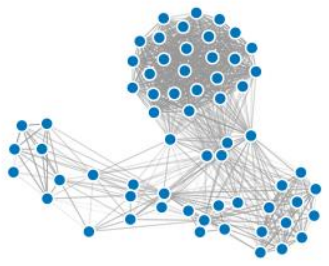
Figure 4: Clusters of Similar Sequences – Original Left Team Strategy LCSS

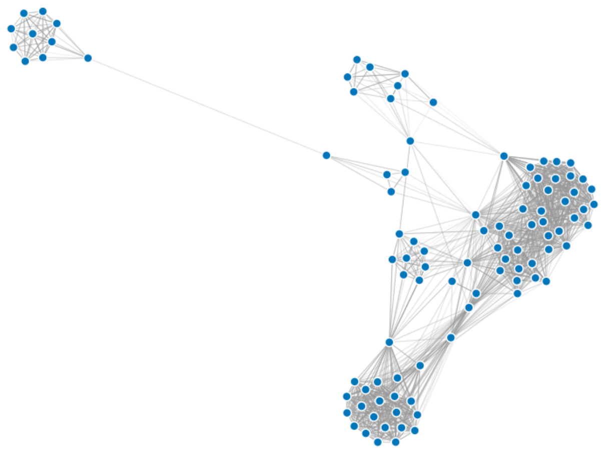
Figure 5: Clusters of Similar Sequences – Adapted Strategy LCSS

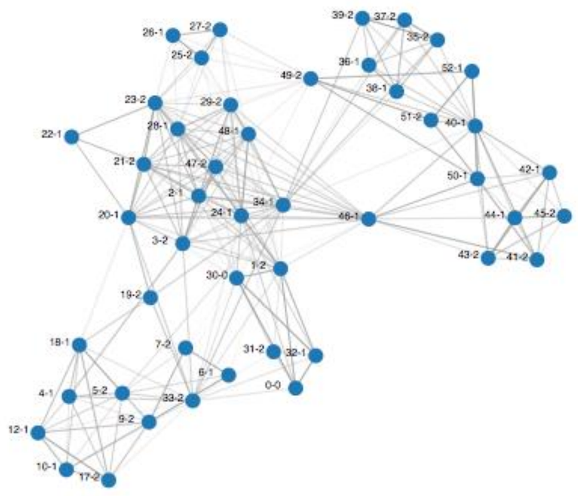
Figure 6: Clusters of Similar Sequences – Set 1, #1 LCSS

At closer examination of the extracted sequences from the played games of Set 1 (Table 6), the correlation between clusters and the rules can be seen. The clusters of similar sequences for selected iteration games in Set 1 are shown in Figure 6 to 10.

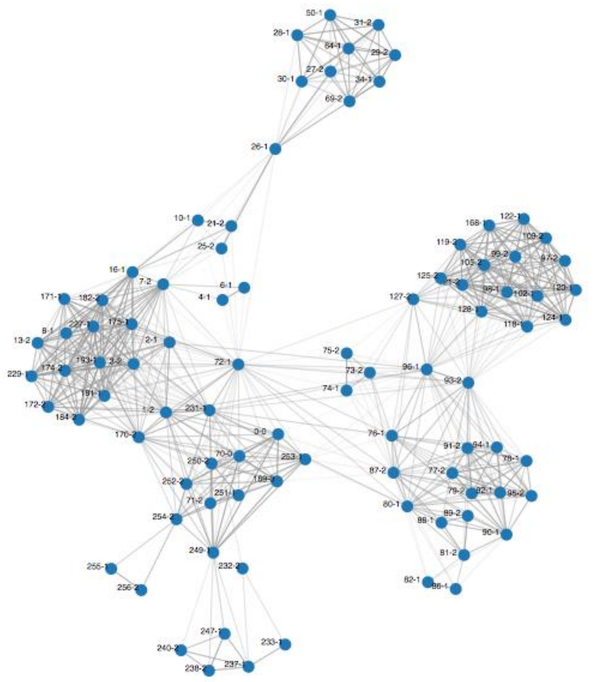
Figure 7: Clusters of Similar Sequences – Set 1, #2 LCSS

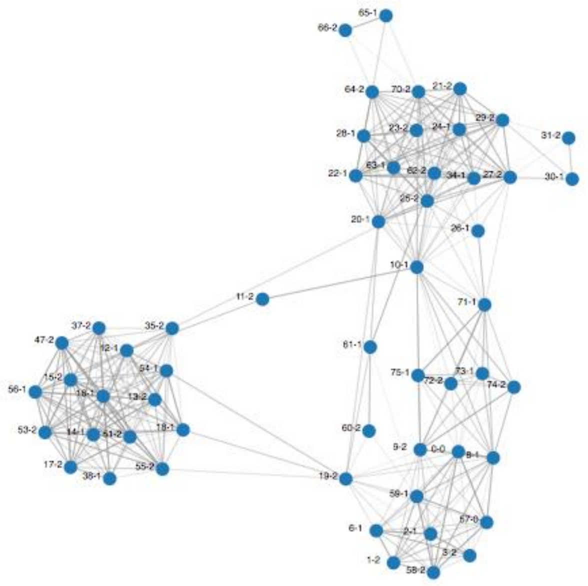
Figure 8: Clusters of Similar Sequences – Set 1, #4 LCSS

With the increasing number of game iterations, there is also increasing number of rules in the strategy. These rules are being selected during the game thus being part of a newly created sequences that represent specific game situations that occurred during the game.

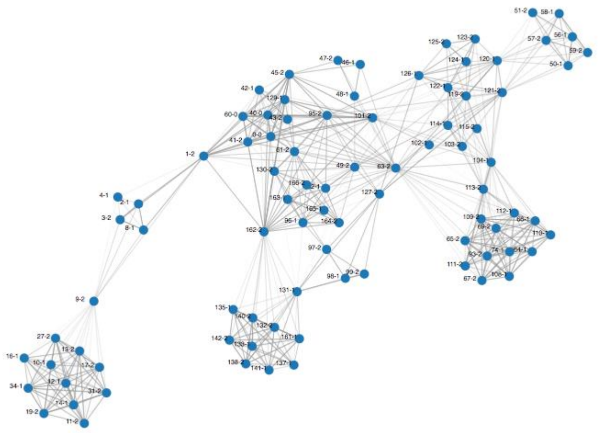
Figure 9: Clusters of Similar Sequences – Set 1, #6 LCSS

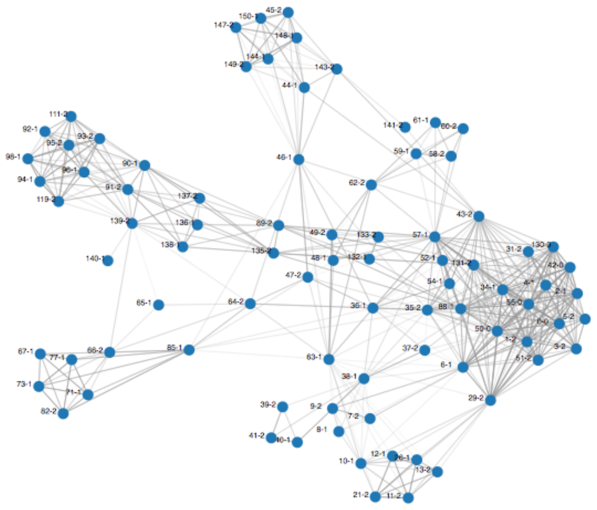
Figure 10: Clusters of Similar Sequences – Set 1, #8 LCSS

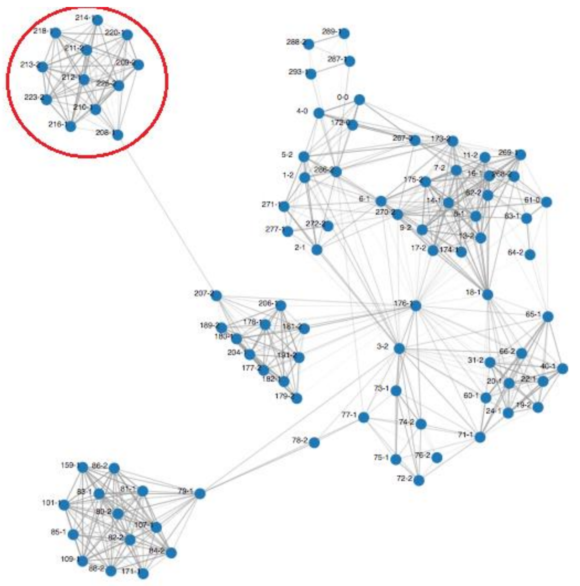
Figure 11: Clusters of Similar Sequences – Set 1, #10 LCSS

Each node represents a sequence comprised of selected strategy rules. Each sequence is labeled with a sequence number and with a number determining the possession of the ball. Therefore, each sequence number could be mapped to a sequence of strategy rules that were performed during some game situation. The selected sequence cluster from the Figure 11 is described in Table 11.

Table 11: Set 1, #10, Selected Sequence Cluster

Sequence #	Rules
208 - 1	34,34,34,34,34,34,34,34,34,34,34,34,34, 34,34,34,34,34,34,34,34,34,34,34,34,34, 34,34,34,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8, 8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8, 8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8, 8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8, 8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8, 8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8, 8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8, 8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8, 8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8, 8,8,8,8,8
209 - 2	8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8, 8,8,8,8,8,8,8,8,8,8
210 - 1	34,34,34,34,34,8,8,8,8,8,8,8,8,8,8,8,8,8, 8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8, 8,8,8,8,8,8,8,8,8,8,8,8,8
211 - 2	8,8,8,8
212 – 1	8,8
213 - 2	8,8,8,8,8,8,8,8
214 - 1	8
216 - 1	8,8,8,8,8
218 - 1	8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8, 8,8,8,8,8
220 - 1	8,8,8,8,8,8,8,8,34,34,34,34,34,8,8,8,8,8, 8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8,8, 8,8,8,8,8,8,8,8,8
223 - 2	8,8,8,8,8,8
226 - 2	8,8,8

Table 12: Selected Sequence Cluster - Rules

#	Mine	Oppnt	Ball	Move
8	62 53 51 32	42 52 62 53	62	62 53 51 32
34	51 32 62 64	62 63 52 53	54	51 32 62 63

This cluster consists of 12 sequences containing rules number 8 and 34 (see Table 12). The rule number 8 is the rule originating from the original Left team strategy and the rule number 34 is the offensive rule that was created during the strategy adaptation. Our robot soccer architecture does not require for each robot to be uniquely identifiable during the game. Therefore the robots' roles are mutually interchangeable. In relation to this, it is apparent the similarity of the rule 8 and 34. Further analysis of the extracted sequences will be the focus of our future work and the theme of the follow-up articles.

CONCLUSION AND FUTURE WORK

In this work, the strategies of the robot soccer game were discussed. The description of approach for the strategy adaptation was presented. The main part of the article discussed the static strategy adaptation using the log files of a previously played games and also the real-time strategy adaptation performed directly during the robot soccer game. The adaptation was performed for the defensive and also offensive part of the strategy.

The adaptation method performed after the game using the log files has an advantage in a potentially big initial training set whereas the real-time adaptation process is much more flexible and able to quickly react to the opponent’s current behavior during the game. On the other hand, the real-time adaptation process might also need a sufficient number of iterations to achieve the desired results.

The future work will be mainly focused on improvements in the area of strategy adaptation and evaluation, and of course on the overall improvements of the developed robot soccer game architecture.

ACKNOWLEDGMENTS

This work was supported by The Ministry of Education, Youth and Sports from the National Programme of Sustainability (NPU II) project „IT4Innovations excellence in science - LQ1602“.

REFERENCES

Osborne M. J., 2004. An introduction to game theory. New York Oxford, Oxford University Press.

Kim J-H, Kim D-H, Kim Y-J, Seow K. T., 2010. Soccer Robotics, Springer Tracts in Advanced Robotics.

Ontanón S., Mishra K., Sugandh N. and Ram A., 2007. Case-Based Planning and Execution for Real-Time Strategy Games. Lecture Notes in Computer Science, Volume 4626, 164-178.

Martinovič J., Snášel V., Ochodková, Zoltá L., Wu J., Abraham A., 2010. Robot soccer - strategy description and game analysis. Modelling and Simulation, 24th European Conference ECMS.

Svatoň V., Martinovič J., Slaninová K. and Snášel V., 2014. Improving Rule Selection from Robot Soccer Strategy with Substrategies. In Computer Information Systems and Industrial Management - 13th IFIP TC8 International Conference (CISIM).

Huang HP, and Liang CC, 2002. Strategy-based decision making of a soccer robot system using a real-time self-organizing fuzzy decision tree. Fuzzy Sets Syst. 127, 1.

Nakashima T., Takatani M., Udo M., Ishibuchi H., and Nii M., 2006. Performance evaluation of an evolutionary method for robocup soccer strategies. In RoboCup 2005: Robot Soccer World Cup IX, Springer.

Larik A. S. and Haider S., 2016. On using evolutionary computation approach for strategy optimization in robot soccer. 2nd International Conference on Robotics and Artificial Intelligence (ICRAI)

Tominaga M., Takemura Y., and Ishii K., 2017. Strategy Analysis of RoboCup Soccer Teams Using Self-Organizing Map.

Shengbing Ch., Gang Lv, and Xiaofeng W., 2016. Offensive strategy in the 2D soccer simulation league using multi-group ant colony optimization. International Journal of Advanced Robotic Systems 13.

Akiyama H., Tsuji M., and Aramaki S., 2016. Learning Evaluation Function for Decision Making of Soccer Agents Using Learning to Rank. Soft Computing and Intelligent Systems (SCIS) and 17th International Symposium on Advanced Intelligent Systems, 2016 Joint 8th International Conference on. IEEE