Improving Rule Selection from Robot Soccer Strategy with Substrategies

Václav Svatoň[1,2], Jan Martinovič[1,2], Kateřina Slaninová[1,2], and Václav Snášel[1,2]

[1] Department of Computer Science, FEI, VŠB - Technical University of Ostrava, 17. listopadu 15, 708 33, Ostrava-Poruba, Czech Republic
[2] IT4Innovations, VŠB - Technical University of Ostrava, 17. listopadu 15, 708 33, Ostrava-Poruba, Czech Republic
{vaclav.svaton,jan.martinovic,katerina.slaninova,vaclav.snasel}@vsb.cz

Abstract. Robot Soccer is a very attractive platform in terms of research. It contains a number of challenges in the areas of robot control, artificial intelligence and image analysis. This article presents a method to improve the description of the strategy by creating substrategies in strategy and thus ensuring smoother implementation of actions defined by this strategy. In presented method we have extracted sequences of game situations from the log of a game played in our simulator, as they occurred during the game. Afterwards, these sequences were compared by methods for sequence comparison and thus we are able to visualize relations between the sequences of game situations and clusters of similar game situations in a graph. This output seems to be very helpful feedback for further strategy development.

Keywords: Robot Soccer, Strategy, Rule, Sequence.

1 Introduction

A complete set of options which are available to players in any game situation in order to achieve the objective is considered as a strategy in the game theory [1], [2]. The result of this strategy depends not only on the actions of the individual player but also on the actions of other players or elements of the game. The so-called pure strategy contains a list of all possible situations that may arise in the game. Any mapping or description of the space in which we know the geographic positions of the objects location can be considered for the strategy [3]. We have defined a finite set of rules that tell us how these objects can behave in a given situations. This principle can be applied to a number of areas from the real world and is generally called strategy planning [4]. We can use 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 our desired goals. The algorithms and approaches from this article may not serve only for use in the game of robot soccer.

The following sections contain an explanation of our robot soccer architecture, our view of strategies, rules and how we use them for mapping coordinates of the

K. Saeed and V. Snášel (Eds.): CISIM 2014, LNCS 8838, pp. 77–88, 2014.

real world. The article describes the main problem of the current approach to the selection of rules from strategy and in the following section we introduce a new method to improve this approach. Then, our method is practically applied to a robot soccer game created in our robot soccer simulator. The main section of the paper contains the results of experiments focused on the sequence extraction from the robot soccer game strategies and the overall comparison of the old and new method of rule selection from the strategy.

1.1 Robot Soccer Architecture

Our robot soccer library consists of a number of interconnected modules that contain the functionality required for prediction, image analysis and robot control. Such architecture brings many advantages, in particular the possibility to experiment with different methods used to select the best winning strategy, or even create a partially simulated game containing real and simulated robots. These modules falls into three main categories: Game information, Game and Log. Game information is a storage of information about the actual game state. It consists of our and the opponent's robots positions and directions and a ball position. Besides having information about actual game situation on the game field it is possible to fill this storage also with predicted information about the robots and the ball. Game part consists of the all necessary functionality for the calculations over strategies and tactics. Such type of calculations is performed every game step and the results are continuously actualized in the part Game information which is primarily used for the robots control.

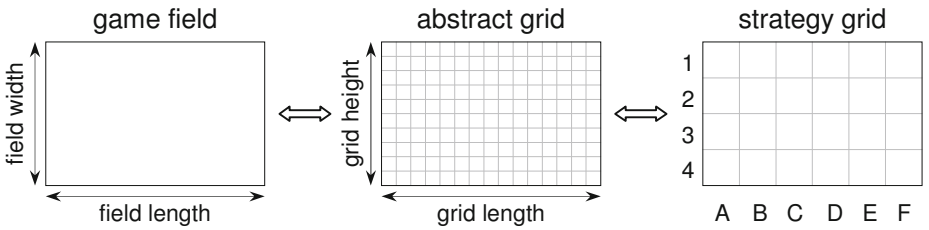

Fig. 1. Inner game field representation

In our work, the game is separated into logical and physical parts [5]. The logical part includes the strategy selection, calculation of robot movement and adaptation of rules to the opponents strategy. The physical part includes robot actual movement on the game field and recognition of the opponent movement. Due to this separation, the logical part is independent on the field size and the resolution of the camera (or physical engine of simulation) used in visual information system. In the logical part, the game is represented as an abstract grid with a very high resolution, which ensures a very precise position of the robots and the ball. However, this detailed representation of the game field is

not suitable for the strategy description. Too many rules are required to describe a robot behavior. Therefore, a strategy grid is used which has a much lower resolution than an abstract grid. This simplification is sufficient, because it is unnecessary to know the robot's exact position in the scope of a strategy (see Fig. 1). Using the physical part based on the size of the game field and camera/engine resolution, we only need to transform the abstract grid into physical coordinates. The strategy, as we understand, is the quaternion $< X, Y, p, m >$ where

- $X \subset X^U$ where X is a selected set of game situations and X^U is the universe of all situations that may occur during the game
- $Y \subset Y^U$ where Y is a selected set of instructions to control the robots and Y^U is the universe of all possible instructions
- p is mapping $X \rightarrow Y$, each game situation is mapped to an instruction describing how to control our robots in this situation
- m is mapping $X^U \rightarrow X$, all possible situations that may occur during the game are mapped to a selected set of game situations
- $p(m(\overline{x}_u)) \in X^U$ where $\overline{x}_u \in X^U$: every real situation on the game field is assigned to a game situation from the strategy which also contains the instructions on where to move our robots

Strategy is a finite set of the rules that describes the current situation on the game field. Each rule can be easily expressed as the quaternion $< M, O, B, D >$, where M are the grid coordinates of our robots, O are the grid coordinates of opponent's robots, B are grid coordinates of the ball and D are grid coordinates of where our robots should move in the next step. The real situation on the game field is compared with the situations described by the strategy rules during the each game step. On the basis of a priori defined metrics, it is selected the most similar rule from the strategy, according to which are set the positions of the players on the game field in the following step. A detailed description of this metrics and the algorithm for the optimal rule selection can be found in [6].

The robot's behavior in the grid coordinates is then controlled by so called tactics [7]. The tactics contain functions for robot control such as turning the robot, shooting at goal or passing the ball. Therefore, in the terms of hierarchy, the strategy takes care of the placement of the robots in the grid coordinates while the tactics work with the physical coordinates and controls the robot inside the grid coordinate.

A 3D robot soccer simulator has been created using the above mentioned robot soccer architecture, see Figure 2. This simulator has been developed with Unity engine [8]. This engine has been selected for its support of physical engine PhysX [9]. Using already created physical engine eases simulator design very much. Especially, it allows us to avoid the necessity to create our own physics and all the problems connected with own solution of object collision on the game field. We can easily set material and weight of the robots, ball, and the game field using the Unity engine. The object collision is computed by the physical engine itself. The physical engine in Unity is non-deterministic, which means that the same simulation, launched repeatedly can return different results.

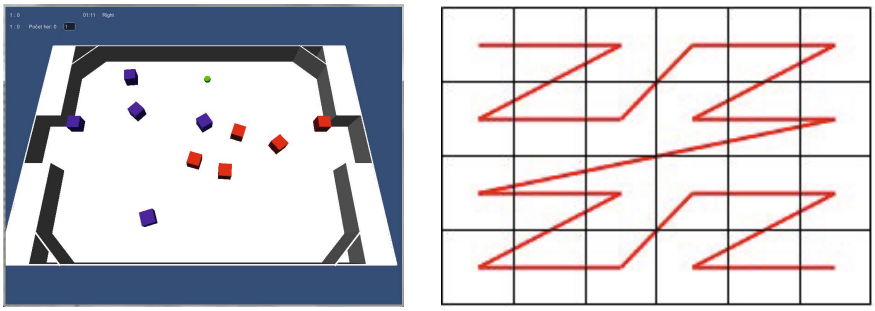

Fig. 2. (a) Robot Soccer Simulator ; (b) Grid 6x4 with Z-order curve

This non-deterministic behavior is a price for the fast physical engine, which is able to perform fast computations. Of course, there exist deterministic engines, but their main disadvantage is their slowness. However, we do not mind the non-determinism in our architecture. Just as in the real world, the robot soccer is a quickly changing dynamic system, and our proposed architecture must count with this non-determinism, and must be able to react to this. In other words, the team with better strategy should be able to win, regardless to minor differences in physical engine computation during the repeatedly launched simulation.

1.2 Rule Selection

Because the above mentioned robot soccer architecture is based on centralized control in every step of the game we have the access to the grid coordinates of each object on the game field. Therefore, in every step of the game we compare the current situation on the field with the situations described in the rules of the strategy. By comparing, it is meant computing the Euclidean distance between the real situation on the field and the situation described in the selected rule.

We have proposed a method to improve the rule selection from the strategy by using graph and space filling curves [11] in the article [6]. We have chosen a method named Z-order [12] for practical purposes. Z-order or Morton 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 its suited for converting two-dimensional matrix representing the playing field into one-dimensional array of the coordinates of the individual robots. Figure 2 shows the final relocation of the robots located on the game field.

In the next step, an undirected connected graph with the edge evaluation is used. Let the graph be defined as a pair $G =< V, E >$ where V is a non-empty set of vertices and E is a set of two-element sets of vertices also called undirected edges. The set of vertices consists of the individual rules from the strategy in our case. 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 neighbouring vertices which contain the robots grid coordinates.

To select the rule for the next step we just need to compare the real situation on the game field with the situations described by the selected neighbour rules and also with the currently selected rule (rule does not have to be necessarily changed in the each game step).

2 Problems with Current Approach

The method for the selection of the rules from the strategy, which was described in Section 1.2, has one fundamental disadvantage. All the rules which create one strategy are independent of each other. It means that every game step, each current situation on the field is compared with the all rules described in the strategy. Due to the optimisation of this method, space filling curves were used for robots ordering on the field and for the graph precomputing which is necessary for finding the similarity between these rules. This led to the effective optimisation of the rule selection, which achieve much better results than brute force approach for the rule selection. However, the examination of the log of played games showed that during the game, the most similar rule is always selected but without the connection to so-called game situation as was for example intended by the author of the strategy during its design.

Game situation is a subset of rules from the strategy, which should represent a specific intended set of subsequent actions. For example the first five rules from the strategy could represent the left wing offensive play, next five rules the right wing play, and after them followed by rules for the defensive play in the middle of the game field. This interpretation of rules can be intended during the design of a strategy, however due to the actual strategy definition, it is possible that the algorithm for finding the optimal rule selects the offensive rule in one step, and the defensive rule in the following step, thus completely neglecting intended game situations. See Figure 3 in section 4.

3 Substrategies

Our proposed solution of the problem described in Section 2 contains so-called substrategies. Substrategy can be understood as a representation of one specific game situation, for example right wing offensive play. Therefore, the whole strategy can include any number of rules. The algorithm for the most similar rule selection from the strategy was modified so that the rules included in the substrategy to which the current rule belongs are scanned first. For example, if the currently executed rule is from the substrategy which represents the right wing offensive play then we assume that in one game step (which lasts 20ms), the game situation does not change enough to be necessary to compare all the rules from the strategy. Therefore, we limit the rule selection to the same substrategy. Due to this approach, the time necessary for the rule selection is decreased and the succession of the rules in the same substrategy is preserved.

As a result of introduction of substrategies a question arose. When to change the substrategy and therefore change the game situation that is defined by this substrategy during the game. It is not sufficient to permanently scan the rules from the current substrategy. The current executed action, for example yet mentioned right wing offensive play, can be interrupted during the game before its end, for example due to the ball loss as a reason of the opponent's defenders activity. Therefore, the proposed method use threshold for determination, when the game situation on the field is so changed that the next game step will not be restricted only to the actual substrategy, but all the rules from the complete strategy will be scanned. Thus, this approach solves the transition from one substrategy to another and for example after the ball loss during the offensive play, the game is changed into the defensive substrategy. Section 3 is focused on description of the sequence extraction from the game log and therefore on the way how to transparently visualise the game progress from the rule selection point of view.

3.1 Game Profile

The proposed approach for game profile extraction proceeds from the original social network approach with a modification focused on robot soccer game. The modification is based on a definition of the "relationship" between played games. The original approach into the analysis of social networks deals with the assumption that the social network is a set of people (or groups of people) with social interactions among themselves [10]. Social interaction is commonly defined as an interaction between actors, such as communication, personal knowledge of each other, friendship and membership etc.

The modification extends the original approach of social network analysis by the perspective of the complex networks. This type of view differs from the original approach due to the description of the relations between nodes (in the presented context: played games). The relation between the games is defined by their common attributes, characterising by game situations extraxted from game log file.

The game profiles are extracted using the methods from process mining, especially the methods from log mining. Let us assume that an event log from the analysed system contains data related to rules selected from game strategy from played game.

Definition 1. (Base game profile, sequences)
Let $U = \{u_1, u_2, \ldots, u_n\}$*, be a set of games, where* n *is a number of games* u_i*. Then, sequences of strategy rules* $\sigma_{ij} = \langle e_{ij1}, e_{ij2}, \ldots e_{ijm_j} \rangle$*, are sequences of strategy rules executed during a game* u_i *in the simulator, where* $j = 1, 2, \ldots, p_i$ *is number of that sequences, and* m_j *is a length of* j*-th sequence. Thus, a set* $S_i = \{\sigma_{i1}, \sigma_{i2}, \ldots \sigma_{ip_i}\}$ *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* $\sigma_l \in S$ *without this relation to games:* $\sigma_{ij} = \langle e_{ij1}, e_{ij2}, \ldots, e_{ijm_j} \rangle \rightarrow \sigma_l = \langle e_1, e_2, \ldots, e_{ml} \rangle$*, where* $e_{ij1} = e_1, e_{ij2} = e_2, \ldots, e_{ijm_j} = e_{ml}$.

Define matrix $B \in N^{|U| \times |S|}$ *where*

$$B_{ij} = \begin{cases} frequency\ of\ sequence\ \sigma_j \in S\ for\ game\ u_i & \text{if } \sigma_j \in S_i \\ 0 & \text{else} \end{cases}$$

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 such sequence extracted from the game log file was compared with other sequences, while the similar sequences were found. Thus, we are able to visualize them by graph of sequences, where the clusters of similar sequences are showed. The sequence comparison was done by The longest common substring method (LCS), The longest common subsequence method (LCSS) a The time-warped longest common subsequence (T-WLCS). The difference between the used methods and the way of their usage is described in more details in our previous article [13].

4 Experiments

The log of the played game was used for the extraction of the sequences. The log has been generated by the standard game which lasted 2 minutes between two strategies. The following experiments are focused only on the strategy of the left team which was created with the substrategies mentioned in Section 3. Strategy was created with several different game situations which are described in Table 1.

The log file consists of complete information about the game field situation for each game step. Besides the coordinates of the all robots and the ball, it also consists of information about the actual selected rule from the strategy for the left as well as the right side. Thus, the sequence is created by the sequence of the selected rules during the game for the left side team. The whole game lasted 2 minutes; one game step was performed every 20ms. Therefore, the final log file consisted of 6.000 records. It was necessary to decide which game situation will be the basic for the sequence determination. It was selected the situation holding the ball. The robot is holding the ball, when it touched the ball and after that it is inside the set border distance.

We have applied the algorithms for finding the similar sequences under the sequence collection, especially the LCS, LCSS, and T-WLCS method. The visualizations of the found similar sequence clusters are presented in Figures 3, 4, 5 and 6. Each node in the graph represents one sequence. Each sequence is labeled with a sequence number and number determining the possession of the ball (0 - none, 1 - our 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.

The Figure 3 shows clusters of sequences extracted from the log of the played game with our test strategy but without the implementation of subtrategies. Method T-WLCS achieved the best results of all three proposed methods used

to find similar sequences. Upon a closer examination of extracted sequences, it is evident that during the game there was frequent switching of rules also independent of the intended game situation. Graph in Figure 3 shows number of sequences that are not part of any main cluster (137-1, 50-2, 12-2, 60-1, ...). These sequences contain rules from several different game situations. See Table 2 for sequence 137-1 and it's rules.

Figures 4, 5 and 6 show the clusters of extracted sequences from the log of the game which was created with rules divided into subcategories. For a real game, it was necessary to set the threshold determining when to scan all the rules from the strategy and therefore allow the transition between substrategies. This threshold is represented by the Euclidean distance (similarity) between the real situation on the game field and the rule from the strategy. Thresholds were chosen in values of 300, 400 and 500, because the average distance during the game varies from 100 to 700.

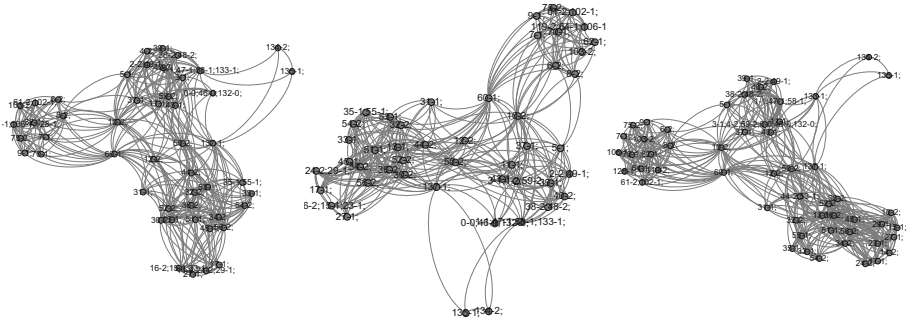

Fig. 3. Strategy without substrategies (a) LCS; (b) LCSS; (c) T-WLCS

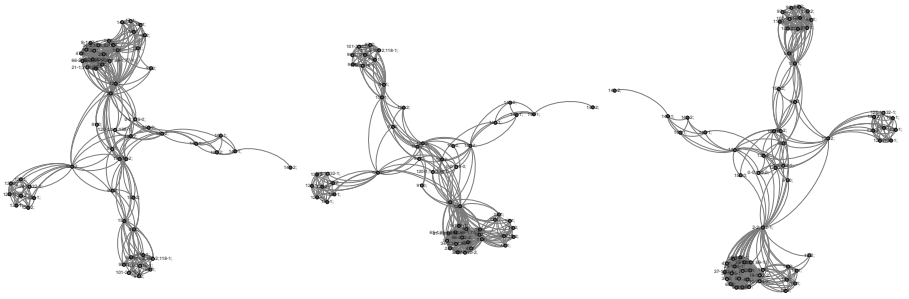

Fig. 4. Strategy with substrategies and threshold 300 (a) LCS; (b) LCSS; (c) T-WLCS

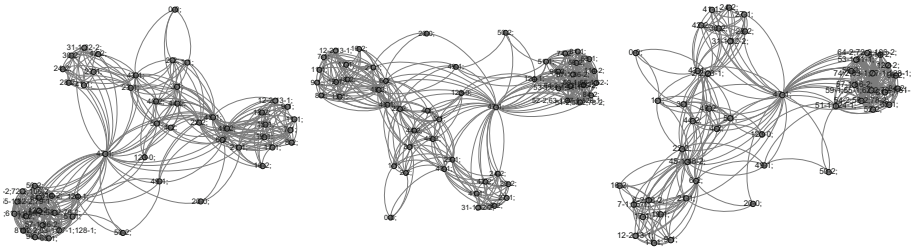

Fig. 5. Strategy with substrategies and threshold 400 (a) LCS; (b) LCSS; (c) T-WLCS

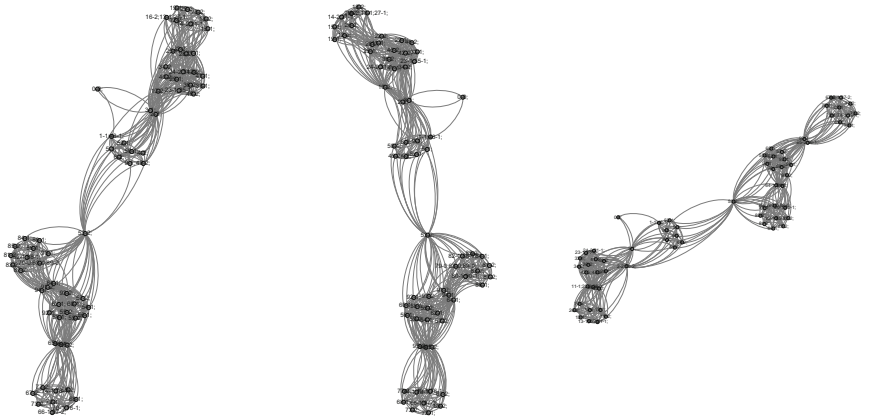

Fig. 6. Strategy with substrategies and threshold 500 (a) LCS; (b) LCSS; (c) T-WLCS

All the graphs on Figure 4, 5, and 6 show that lower thresholds are still causing frequent transitions between substrategies and therefore causing frequent switching between the game situations which are represented by these substrategies. With the increasing threshold value, the players remain in the selected game situation thus ensuring smoother progress of individual actions and therefore the overall smoother game. This is most noticeable from the graph in Figure 6. Extracted sequences are clearly divided into six main clusters which altogether represent all six game situations defined in the test strategy (see Table 1).

Sequence 53-2 (see Table 3) from the graph in Figure 6 is worth mentioning. This sequence is not part of the main sequence clusters because it contains the rules from several different substrategies. These rules are very similar to one another. These rules represent the starting position of every player at the start of the game because almost every game situation defined in the strategy starts

Table 1. Strategy

Substrategy	Rule num.	Rule description
Offensive middle	1	M 3,2 3,3 2,1 2,4 O 4,2 4,3 5,1 5,4 B 4,2 D 3,2 3,3 2,1 2,4
	2	M 4,2 4,3 2,2 2,3 O 4,2 4,3 5,1 5,4 B 4,2 D 5,2 5,3 2,2 3,3
	3	M 5,2 5,3 2,2 3,3 O 5,2 5,3 5,1 5,4 B 5,2 D 6,2 5,3 2,2 3,3
	4	M 6,2 5,3 2,2 3,3 O 5,2 5,3 5,1 5,4 B 6,2 D 6,2 5,3 2,2 3,3
Offensive right wing	5	M 3,2 3,3 2,1 2,4 O 4,2 4,3 5,1 5,4 B 4,3 D 3,2 4,3 2,2 2,3
	6	M 3,2 4,3 2,2 2,3 O 4,2 4,3 5,1 5,4 B 4,3 D 4,3 4,4 3,2 2,3
	7	M 4,3 4,4 3,2 2,3 O 4,2 4,4 5,2 5,3 B 4,4 D 4,3 5,4 4,2 2,3
	8	M 4,3 5,4 4,2 2,3 O 4,3 5,4 5,2 5,3 B 5,4 D 4,3 5,3 5,2 2,3
	9	M 4,3 5,3 5,2 2,3 O 4,3 5,3 5,2 5,3 B 5,3 D 4,3 6,3 5,2 2,3
	10	M 4,3 6,3 5,2 2,3 O 4,3 6,3 5,2 5,3 B 6,3 D 4,3 6,3 5,2 2,3
Offensive left wing	11	M 3,2 3,3 2,1 2,4 O 4,2 4,3 5,1 5,4 B 3,2 D 3,2 3,3 3,1 2,3
	12	M 3,2 3,3 3,1 2,3 O 4,1 4,2 5,1 5,3 B 3,1 D 4,2 3,2 4,1 3,3
	13	M 4,2 3,2 4,1 3,3 O 4,1 4,2 5,1 5,3 B 4,1 D 5,2 4,2 5,1 3,3
	14	M 5,2 4,2 5,1 3,3 O 4,1 5,2 5,1 5,3 B 5,1 D 5,2 4,3 5,1 3,2
	15	M 5,2 4,3 5,1 3,2 O 4,1 5,2 5,2 5,3 B 5,2 D 6,2 5,3 5,1 3,2
	16	M 6,2 5,3 5,1 3,2 O 4,2 5,2 6,2 5,3 B 6,2 D 6,2 5,3 5,1 3,2
Defensive middle	17	M 3,2 3,3 2,1 2,4 O 4,2 4,3 5,1 5,4 B 3,2 D 3,2 3,3 2,2 2,3
	18	M 3,2 3,3 2,2 2,3 O 3,2 3,3 4,1 5,3 B 3,2 D 2,2 2,3 1,2 1,3
	19	M 2,2 2,3 1,2 1,3 O 2,2 2,3 4,2 5,3 B 2,2 D 2,2 2,3 1,2 1,3
	20	M 2,2 2,3 1,2 1,3 O 1,2 2,3 3,2 4,3 B 1,2 D 2,2 2,3 1,2 1,3
Defensive right wing	21	M 3,2 3,3 2,1 2,4 O 4,2 4,3 5,1 5,4 B 3,3 D 3,2 3,3 2,2 2,3
	22	M 3,2 3,3 2,2 2,3 O 3,2 3,3 5,2 5,3 B 3,3 D 3,3 3,4 2,2 2,3
	23	M 3,3 3,4 2,2 2,3 O 3,3 3,4 5,2 4,3 B 3,4 D 2,3 2,4 2,2 1,3
	24	M 2,3 2,4 2,2 1,3 O 2,3 2,4 4,2 3,3 B 2,4 D 3,3 2,3 2,2 1,3
	25	M 3,3 2,3 2,2 1,3 O 2,2 2,3 4,2 3,3 B 2,3 D 3,3 2,3 2,2 1,3
	26	M 3,3 2,3 2,2 1,3 O 2,2 1,3 4,2 3,3 B 1,3 D 3,3 2,3 2,2 1,3
Defensive left wing	27	M 3,2 3,3 2,1 2,4 O 4,2 4,3 5,1 5,4 B 4,2 D 3,2 3,3 2,2 2,3
	28	M 3,2 3,3 2,2 2,3 O 4,2 3,3 5,2 5,3 B 3,2 D 3,1 3,3 2,1 2,2
	29	M 3,1 3,3 2,1 2,2 O 3,1 3,2 5,2 4,3 B 3,1 D 2,1 2,3 1,1 2,2
	30	M 2,1 2,3 1,1 2,2 O 2,1 2,2 4,2 3,3 B 2,1 D 2,2 1,3 1,2 2,3
	31	M 2,2 1,3 1,2 2,3 O 1,2 2,2 4,2 2,3 B 1,2 D 2,2 1,3 1,2 2,3

with the default positions of all players. Therefore the sequence was found that contains these seemingly different rules and thus represents the connection point between the clusters of sequences.

5 Conclusion and Future Work

Main part of the article discussed the strategies and the method for rule selection from strategies. Improvement of strategy using substrategies was presented. This method also allows the author to create a strategy, which will be performed during the game by such way, by which it was intended. Substrategies represent game situations such as left wing offence or right wing defence and ensures that

Table 2. Sequence 137-1 from Figure 3

137-1
2,2,2,2,2,2,2,2,2,2,28,28,22,22,22,22,22,22,22,22,22,22,22,22,22,22,2,12,12,12,12,12,12,12, 12,6,6,6,6,6,6,6,6,6,6,6,6,6,6,6,9,9,9,9,9,9,9,9,9,9,6,6,6,2,2,2,2,6,6,6

Table 3. Sequence 53-2 from Figure 6

53-2
29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29, 29,29,29,29,29,29,29,18,18,18,18,18,18,18,18,18,18,18,18,18,18,29,29,29,29,29,29,29,29, 29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,29, 29,29,29,29,29,29,29,29,29,29,29,29,29,29,29,13,13,13,13,13,13,13,13,13,13,13,13,13, 13,13,13,13,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,2,2,2,2,2,2,2,2,2,2,2,2,2, 2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,3,3,3,3,3,3,3,3,3,3,3,3

players will perform the strategy actions more continuously and in faster way then using the original approach.

An important task is to devise a way how to find an optimal threshold for substrategy selection. Such threshold does not restrict the players during the game and force them to perform still the same game situation or on the contrary not too lenient one to cause a frequent switching of rules from the various substrategies.

Generally speaking, games represent any situation in nature. Game theory can be applied for example for adversarial reasoning in security resource allocation and scheduling problems. Randomized policies mitigate a key vulnerability of human plans: predictability. We intend to use the presented method in the area of traffic prediction in the future work.

Acknowledgements. This work was supported by the European Regional Development Fund in the IT4Innovations Centre of Excellence project (CZ.1.05/1.1.00/02.0070) and the national budget of the Czech Republic via the Research and Development for Innovations Operational Programme, and supported by the project New creative teams in priorities of scientific research, reg. no. CZ.1.07/2.3.00/30.0055, supported by Operational Programme Education for Competitiveness and co-financed by the European Social Fund and the state budget of the Czech Republic, and co-financed by SGS, VSB Technical University of Ostrava, Czech Republic, under the grant No. SP2014/154 'Complex network analysis and prediction of network object behavior'. At this place we would like to thank the students of bachelors and masters studies at the VSB - Technical University of Ostrava who participated in the development and further improving of the simulator for robot soccer which has been used to create the above mentioned experiments.

References

1. Osborne, M.J.: An introduction to game theory. Oxford University Press, New York (2004)
2. Camerer, C.F.: Behavioral Game Theory: Experiments in Strategic Interaction. Princeton University Press (2003)
3. Ontañón, S., Mishra, K., Sugandh, N., Ram, A.: Case-based planning and execution for real-time strategy games. In: Weber, R.O., Richter, M.M. (eds.) ICCBR 2007. LNCS (LNAI), vol. 4626, pp. 164–178. Springer, Heidelberg (2007)
4. Kim, J.-H., Kim, D.-H., Kim, Y.-J., Seow, K.T.: Soccer Robotics. Springer Tracts in Advanced Robotics. Springer (2010)
5. Martinovič, J., Snášel, V., Ochodková, Z.L., Wu, J., Abraham, A.: Robot soccer - strategy description and game analysis. In: 24th European Conference on Modelling and Simulation, ECMS 2010, Kuala Lumpur, Malaysia (2010)
6. Snášel, V., Svatoň, V., Martinovič, J., Abraham, A.: Optimization of Rules Selection for Robot Soccer Strategies. International Journal of Advanced Robotic Systems (2014)
7. Klancar, G., Lepetic, M., Karba, R., Zupancic, B.: Robot Soccer Collision Modelling and Validation in Multi-Agent Simulator. Mathematical and Computer Modelling of Dynamical Systems: Methods, Tools and Applications in Engineering and Related Sciences 9, 137–150 (2003)
8. Unity Technologies, Unity @ONLINE, `https://unity3d.com/`
9. NVIDIA Corporation, PhysX @ONLINE, `http://www.geforce.com/hardware/technology/physx`
10. Newman, J.E.M.: Networks: An Introduction. Osford University Press (2010)
11. Sagan, H.: Space-filling curves. Springer (1994)
12. Morton, G.M.: A computer Oriented Geodetic Data Base and a New Technique in File Sequencing. Technical Report, IBM Ltd. Ottawa, Canada (1966)
13. Svatoň, V., Martinovič, J., Slaninová, K., Bureš, T.: Improving strategy in robot soccer game by sequence extraction. In: KES-2014, 18th International Conference on Knowledge-Based and Intelligent Information and Engineering Systems, Gdynia, Poland (2014)