Cognitive techniques for Control of Dynamic Object Behavior in Group

Kupreenkov Ivan

Saint Petersburg State Polytechnic University
kupreenkov@gmail.com

Abstract. This paper describes a possibility of applying the data mining technique to train agents for teamwork. Data mining modules based on immunological networks capable to be learned on-line in dynamically changing environments are proposed. These modules provide adaptive agent behavior for teamwork, because they are able to self-tuning in non-deterministic environments. Reinforcement learning is considered to be the main method for training the agents during a game/ Examples of immunological networks for agent teamwork implementation are considered and results of experiments are described.

1 Introduction

Currently multi-agent systems are widely applied for distributed control and implemented by the set of the cooperating agents. The agents implement various functions of individual and collective behaviors. The individual behaviors depend on solving problem and role of the agent in system. The collective behavior defines methods of interacting among the agents and can be as cooperative, competitive, or teamwork ones.

The teamwork is special kind of the collective behavior based on common intentions or plans of the agents to perform a common goal. A control for autonomous vehicles performing teamwork with common goal is one of important problems in this field. They can be mobile robots, participating in rescue and military operations, or unmanned vehicles performing various tasks in air, water and space, software robots that are agents solving industrial, economical, social tasks cooperatively, or agents playing in soccer or basketball and participating in rescue team competitions. Teamwork theory and examples of teamwork systems are discussed in detail by Cohen and M. Tambe [1, 2].

During previous years the various architectures for building the teamwork systems have been developed. Logical, reactive, layered, and BDI (Belief-Desire-Intention) architectures [3] are well known. However a problem of on-line learning the agents for teamwork in condition of dynamical changing environment was developed not enough.

Using data mining techniques based on on-line machine learning allows developing information and control systems able to learn from and adapt to their environments [4]. The data mining techniques can be used for creating intelligent systems and agents, which can be able to solve the complex control problems in adaptive manner.

The data mining approach in multi-agent systems has been discussed recently, for example, in [5]. It has resulted to creating the powerful intelligent systems that can be defined as distributed systems able to learn to implementation of mental reasoning for decision making in non-deterministic, dynamically changing environments. At the teamwork main problem is to provide dynamical interactions of the agent with others. The special interacting procedures that realize the teamwork coordination mechanisms must be produced through training. It provides to agents ability to on-line learn how to cooperate and/or compete in teamwork conditions. At the training reinforcement learning is considered to be the main method for on-line learning collective behavior of the agent for teamwork.

In this paper we show how the cognitive agents and uniform learning components for implementing individual and collective behaviors of their agents, can be built. The developed uniform data mining components can be specialized and adapted by tuning. Using examples from RoboFIBA framework we also show, that the on-line adaptive cooperation of the basketball agents, built on basis of layered reactive architecture, as well as using the data mining components, can provide high efficiency in the distributed control.

The work is based on author private experience of multi-agent system design for industrial and socio-technical object control [6], special cognitive soccer agents for participation in RoboCup Simulation League competitions [7], basketball agents for RoboFIBA environment [8] and control systems for intellectual robots [9], [10].

2 The architecture of the agents for teamwork

To provide a complex adaptive individual and collective behavior, the agents in the teamwork systems must be built using combination of multilevel and layered architectures. Based on our experience in the field agent for team competitions (e.g. RoboCup and RoboFIBA), most appropriate architecture must consist of at least three levels.

The low level of the agent has layered reactive architecture. It includes several executive layers. Each executive layer has its priority. It reacts to given situation and forms corresponding actions in response. Set of such reactions defines some primitive executive behavior or agent’s skills. Sequence of the reactions corresponds to the current intentions of the agent.

The middle level of the agent has elements of BDI architecture [3]. It consists of modules built according to concepts of beliefs, desires and intentions which define individual agent’s behavior. The agent’s beliefs correspond to data, facts, and situations that are known to the agent. The desires are defined as goals or plans. They are also known to the agent. The intentions include sequences of actions that must be realized by the agent according to its plan.

The upper level of the agent also has elements of BDI architecture that forms corresponding collective agent's behavior for the teamwork. It uses potential intentions of the agent worked out by the middle level and makes agree with intentions of other agents. In case of conflicts the common beliefs of agents are formed. The conflict is solved after conforming agent's intentions and beliefs. After it, each agent corrects its individual intentions according to the common intentions of the agents.

3 Data mining techniques for training agents

Data mining techniques that can provide training agents for teamwork can be based on use adaptive modules, capable to on-line learning. Well-known artificial neural networks can be used for implementing the adaptive the adaptive modules. However, these networks have significant limitations related to kind of the mapping functions and learning rate. Special data mining modules based on fast adaptive functional approximation and reinforcement learning were developed and examined. Among them cluster module based on neuro- and fuzzy-logic was selected for practical usage and provided good results at training simple operations of soccer agents for RoboCup competitions [10]. However such module is difficultly used for training more complex scenario operations of teamwork.

3.1 Immunological networks as adaptive modules data mining

In this work, the adaptive modules, capable to on-line learning are proposed to implement on base of immunological networks. Recently studying biological immune systems inspirited to development of artificial immune systems that consist of immunological networks [11]. In our opinion, these networks can be effectively used for implementing the adaptive agent behavior. They are dedicated to self-preservation under hostile environment and have various interesting features such as immunological memory, immunological tolerance, pattern recognition, and so on.

There are several models of artificial immune systems such as models based on idiotypic network hypothesis [12], clonal-selection theory [13], and spatial immune network model [14]. In this work, the idiotypic network model is proposed to be used because it is most appropriate selecting agent behavior under changing conditions of the environment.

3.2 The idiotypic network model

Jerne's idiotypic network hypothesis is based on the fact that each type of antibody also has its specific antigen determinant called an idiotope. This fact allowed Jerne to introduce concept of idiotypic network. In the network antibodies/lymphocytes are not just isolated, namely they are communicating to each other among different species of antibodies/lymphocytes. Each antibody has also paratope that able to recognize corresponding antigen. Idea of Jerne can be shortly described as following. Let’s the idiotope 1 stimulates the B-lymphocyte 2, which atteches the antibody 2 to its surface, through the paratope of antibody 2. Here, the idiotope of antibody 2 works as an antigen. As a result, antibody 2 suppresses the B-lymphocyte 1 with antibody 1. On the other hand, the idiotope of antibody 3 stimulates antibody 1 since it works as an antigen in view of antibody 1. In this way, the stimulation and suppression chains among antibodies form a large-scaled network and work as a self and non-self recognizer. This regulation mechanism provides a new parallel decentralized processing mechanism.

The idiotypic network can be used to forming behavior selection mechanism of an agent. As it is described by Watanabe et al. [15], preliminary behavior primitive (competence modules) must be prepared. For example, if the agents (e.g. robots) work in the environment with obstacles, current situations (e.g. distance and directional to the obstacle, etc.) detected by installed sensors, work as multiple antigens. Each prepared competence module (e.g. simple behavior) is regarded as an antibody, while the interaction between modules is replaced by the stimulation and suppression between antibodies. The basis concept of the method is that idiotypic network selects a competence module (antibody) suitable for the detected current situation (antigens) in a bottom-up manner.

Dynamics of the idiotypic network can be described using main parameter of concentration of *i*-th antibody, which is denoted by A_i , across following equations:

$$\frac{dA_i(t)}{dt} = \{\alpha\sum_{j=1}^{N} m_{ji} a_j(t) - \alpha\sum m_{ik} a_k(t) + \beta m_i - k_i\} a_i(t) \ , \qquad (1)$$

$$a_i(t+1) = \frac{1}{1+\exp(0.5 - A_i(t))},$$

Where in first equation, *N* – the number of antibodies; m_{ji} and m_i – denote affinities between antibody *j* and antibody *i* (i.e. the degree of disallowance), and between antibody I and the detected antigen, respectively. The first and second terms of the right hand of the equation denote the stimulation and suppression from other antibodies, respectively. The third term represents the stimulation from the antigen? And forth term the dissipation factor (i.e. natural death). Second equation is squashing function used to ensure the stability of the concentration. Selection of antibodies can be simply carried out on roulette-wheel manner according to the magnitude of concentrations of the antibodies. Note that only one antibody is allowed to activate and act its corresponding behavior to the world.

3.3 Adjustment mechanism for training

The immunological network needs in adjustment mechanism that can be considered as the adaptation by changing parameters for prepared network. Such mechanism can be realized by the use special procedure of calculation of degrees of stimuli m_{ij} , which are described in each idiotope [15]. The mechanism starts from the situation where idiotopes of the prepared antibodies are undefined, and then obtains idiotopes using reinforcement learning [16].

Reinforcement learning problem relates to learning from interaction of agent with the environment to achieve a goal. The agent interacts with the environment at each of a sequence of discrete time steps $t_k, k = 0,1,2,3,...$. At each time step, t, the agent receives some representation of the environment's *state*, $s_t \in S$, where S is the set of all possible states, and on that basis selects an *action*, $a_t \in A(s_t)$, where $A(s_t)$ is the set of actions available in state s_t. One time step later, in part as a consequence of its action, the agent receives a numerical *reward*, $r_{t+1} \in R$, and finds itself in a new state, s_{t+1}. At each time step, the agent implements a mapping from state representations to probabilities of selecting each possible action. This mapping is called the agent's *policy* and denoted π_t, where $\pi_t(s,a)$ is the probability that $a_t = a$ if $s_t = s$. Reinforcement learning methods specify how the agent changes its policy as a result of its experience. The agent's goal, roughly speaking, is to maximize the total amount of reward it receives over the long run.

Reinforcement signals are used in order to calculate parameter m_{ij} of each antibody. Let's assume that antigens 1 and 2 invade in immune network interior and each antigen simultaneously stimulates antibody 1 and 2. Consequently, the concentration of each antibody increases. However, since the priority between antibodies is unknown (because idiotopes are initially undefined), in this case either of them can be selected randomly. Now, let's assume that the network randomly selects antibody 1 and then receives a positive reinforcement signal as reward. To make the network tend to select antibody 1 under the same or similar antigens (situation), we record the number of antibody 1 (i.e. 1) in idiotope of antibody 2 and increase a degree of stimuli m_{21}. In generalized case, modify the degree of stimuli can be used such equations:

$$m_{12} = \frac{T_p^{Ab1} + T_r^{Ab2}}{T_{Ab2}^{Ab1}}, \tag{2}$$

$$m_{12} = \frac{T_r^{Ab1} + T_p^{Ab2}}{T_{Ab2}^{Ab1}}, \tag{3}$$

where T_p^{Ab1} and T_r^{Ab1} are number of times of obtaining penalty and reward when antibody 1 is selected; T_{Ab2}^{Ab1} is number of times when both antibodies 1 and 2 are activated by their antigens.

4 Investigation of the RoboFIBA basketball agents

4.1 RoboFIBA virtual system

RoboFIBA virtual system is client-server one that consists of server and two competitive teams of basketball agents. The server was developed to be the heart of the client-server system for competition of agent teams. The server was implemented on Delphi 7 language. It is compatible with OS Windows XP, and also with other OS of Windows family and requires Hardware with not less than 64MB RAM, 1 GHz CPU speed, 20 Mb free disk space.

The **server** includes Communication, Logical, and Graphic modules.

The Communication module provides connection with clients, data transfer over TCP/IP protocol, and interaction with the logic module.

The Logical module realizes mathematical model of the environment. The basic stages of functioning of the logical module:

1) Change of a status of agents in the environment;
2) Processing of a simulation step of the environment according to a new status of agents;
3) Preparation of the sensor information for agents;

The Graphical module visualizes objects in the environment.

The schema of interaction of modules is presented in fig. 1.

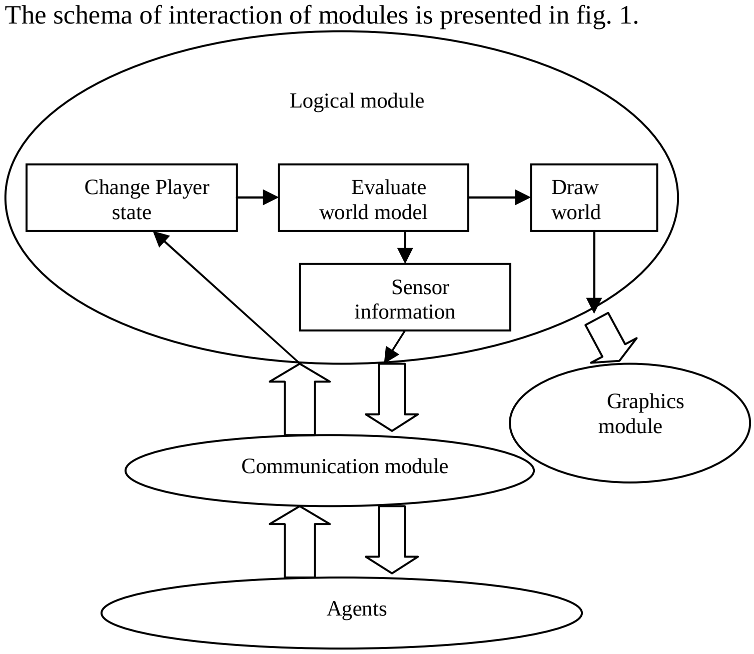
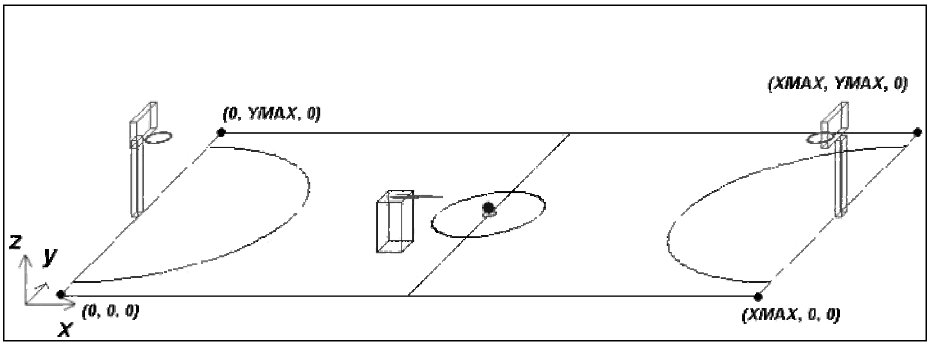


Fig. 1. The scheme of interaction of RoboFIBA Server modules

The **RoboFIBA environment** consists of the court, two basket rings with backboard, ball and 10-players. The size of the court is *xmax* by *ymax* (fig. 2).



Fig, 2. Model of environment

The position of each player is defined in 3D space:

$$P = \{p_i\}\text{, где } p_i \text{ - } (x_i, y_i, z_i)\,. \tag{4}$$

Positions of players are subject to the constraint:

$$(R_i \cap R_j) = 0\,, \tag{5}$$

where R_i, R_j - circles of radius *r*, placed in points p_i and p_j .

The position of the ball is also defined in 3D space:

$$Ball = \{x, y, z\}\,. \tag{6}$$

The ball may have status FREE (the ball is free) and BUSY (the ball is taken by player). The player is identified by the number of its team as $TeamID = \{0,1\}$ and by player number in the team - $PlayerID = \{1,2,3,4,5\}$.

In the server uses **action model** that the time updates in discrete steps. A simulation step is 100 ms.

The server can process the limited number of actions that defined as commands sent by a player (one command of each player is executed for one step of time):

1) SHOOT (power *Pow*, direction *DirXY* and *DirZ*). The player shoots the ball with the power *Pow*, in direction of horizontal plane *DirXY* and in direction of vertical plane *DirZ*.

2) PASS (power *Pow*, direction *DirXY* and *DirZ*). The player passes the ball with power *Pow*, in direction of horizontal planes *DirXY* and in direction of vertical plane *DirZ*. The ball, moving with the power *Pow* in direction *DirXY* and *DirZ,* is switched in state «FREE».

3) RUN (power *Pow*). The player runs with power *Pow* in current direction..

4) TURN_DIRECTION (direction *DirXY*). The player changes its body direction to *Dir XY*.

5) CATCH. The player captures the ball. If distance between the ball and the player is less than *CatchableDist*, the ball belongs to the player. If more then one player is within distance *CatchableDist* to the ball, the ball will go to the nearest player. Catch action is executed only when the ball is free.

Model of movement defines a position of the player with coordinates $\{x_1, y_1, z_1\}$, power pow, and a direction $DirXY$ in the next simulation step is calculated as follows:

$$\begin{aligned} x_2 &= x_1 + pow * \cos(dirXY * Pi/180) \\ y_2 &= y_1 + pow * \sin(dirXY * Pi/180), \\ z_2 &= z_1 \end{aligned} \tag{7}$$

where $\{x_2, y_2, z_2\}$ is the new coordinates of the player.

Position of the ball with coordinates $\{x_1, y_1, z_1\}$, power pow, direction $DirXY$ and $DirZ$ in the next simulation step is calculated in the following way:

$$\begin{aligned} x_2 &= x_1 + pow * \cos(dirZ * Pi/180) * \cos(dirXY * Pi/180) \\ y_2 &= y_1 + pow * \cos(dirZ * Pi/180) * \sin(dirXY * Pi/180) \\ zspeed &= zspeed - GRAVITY \\ z_2 &= z_1 + zspeed \end{aligned}, \tag{8}$$

where $\{x_2, y_2, z_2\}$ is the new coordinates of the ball, $GRAVITY$ is the acceleration of free falling, *zspeed* is the vertical speed of the ball, calculated in the moment of shoot or pass by the formula (9).

$$zspeed = pow * \sin(dirZ * Pi/180). \tag{9}$$

Sensor model implemented in the server allows to send the following information to players:

1) Own coordinate;
2) Coordinates, *TeamID*, *PlayerID*, *DirXY* parameters of all partners and opponents;
3) Coordinates and status of the ball, *TeamID* and *PlayerID* of player, who controls the ball, if status of the ball is BUSY.

In each moment of time the server defines the current **status of game**, what limits the actions of players according to rules of the basketball [8]. Table 1 describes possible statuses of a game.

Table 1. Status of game.

Status	Value	Team 0 access actions	Team 1 access actions	Description
PLAYIN	0	RUN, SHOOT, PASS, CATCH, TURN_DIRECTION	RUN, SHOOT, PASS, CATCH, TURN_DIRECTION	Game
TEAM_0_GOAL_2P	1	RUN	RUN, CATCH	Team 0 get 2 points
TEAM_1_GOAL_2P	2	RUN, CATCH	RUN	Team 1 get 2 points
TEAM_0_GOAL_3P	3	RUN	RUN, CATCH	Team 0 get 3 points
TEAM_1_GOAL_3P	4	RUN, CATCH	RUN	Team 1 gets 3 points
TEAM_0_IN	5	RUN, CATCH	RUN	Team 0 enters the ball in game. Any player of team 0 should catch

				the ball.
TEAM_1_IN	6	RUN	RUN, CATCH	Team 1 enters the ball in game. Any player of team 1 should catch the ball.
SPOR_BALL	7	RUN, CATCH	RUN, CATCH	Disputable ball. The team receives ball, if its player will catch the ball faster.
ONLY_PASS	8	PASS	PASS	The player, who controls the ball, should passes to the partner.

4.2 Basketball agent

To provide complex individual and collective behaviors at the teamwork in the RoboFIBA environment, basketball agents must be built using multi layered architectures. The basketball agent has three-level architecture that is similar to the soccer agent architecture used in [6]. Note, that one of the variants of the agent with described architecture was used for creation of the soccer agent of team STEP (Soccer Team of ElectroPult) that has became by winner of World Championship RoboCup-2004 in Simulation 2D Soccer League.

The low level of the basketball agent has several executive reactive layers. Number of the layers can be changed at agent's behavior tuning. Each executive layer has its priority and reacts to given situation by forming the corresponding actions in response on input information. Set of such reactions defines some primitive executive behaviors (agent's skills). Sequence of the selected reactions corresponds to current intention of the agent.

The middle level of the agent has set of production rules defining individual behavior of the agent. These rules use the conditions in form of data, facts, and situations that are known for the agent. They carry out decisions for selection of primitive behaviors that must be realized at low level of the agent.

The upper level of the agent also has set of production rules that form corresponding collective agent's behavior. At this level the agent first makes decision on selection of whether individual or collective behavior. If the agent selects a collective behavior, then it must take into account positions of partners and opponents. In case of arising conflicts the current collective behaviors of agents are formed. The conflict is solved using special rules for conforming agent's actions.

The **structure of the agent** is presented in fig. 3.

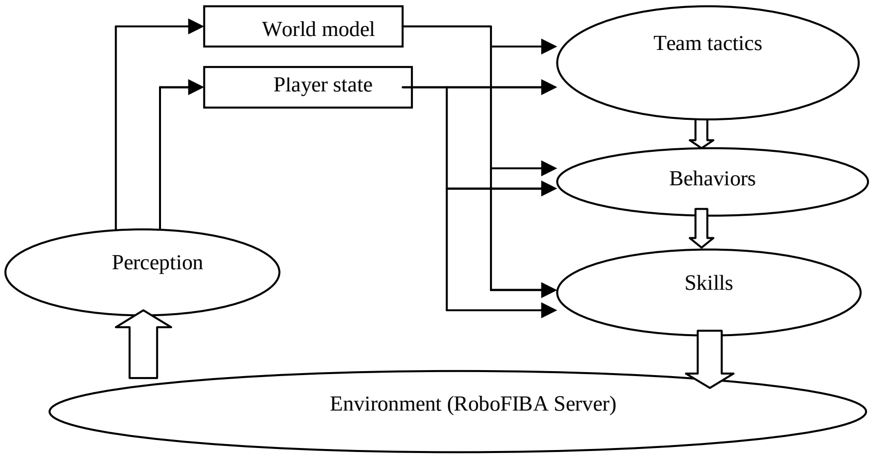


Fig.3. Stricture of the basketball agent

Base of teamwork is **tactics of agent** defined by its second level. At the second level a player could make decision on attack, defense, or catch the ball. The goal of the team in the attack is to score a ball in a basket ring. The goal of the team in the defense is to not allow the opponents to finish the attack. There are some variants of the organization of the defense. In the given example a personal marking of players is used. During the defense, players have the following tasks:

- To block the free moving opponent to the basket;
- To intercept the ball while opponent passes;

In the attack, it is necessary to solve the following tasks:

- To deliver the ball to opponent basket;
- To perform an accuracy shoot;

Catching ball actions should take into account first of all a status of the ball and position of the player related to the ball.

At the team tactic level the player makes a decision on current action in the team. Formally algorithm of decision making is described as (10):

$$\begin{array}{l} ((BALL = FREE) \wedge (t_0 \le t_1) \wedge (t_0 \le t_1)) \Rightarrow (Go\ to\ get\ the\ ball) \\ else\ (t_1 \le t_2) \Rightarrow (Go\ to\ the\ attackhalf) \\ else\,(Go\ back\ to\ the\ defense\ half) \end{array} , \qquad (10)$$

where t_0 is distance between the player and the ball, t_1 is distance between the ball and partner nearest to the ball, t_2 is distance between the ball and opponent nearest to ball.

The given algorithm is the same as described in [17].

If the ball is busy, the player has two options (11):

$$\begin{array}{l} ((BALL = BUSY) \wedge (BALLTeamId = PlayerTeamId)) \Rightarrow (Go\ to\ get\ the\ ball) \\ ((BALL = BUSY) \wedge (BALLTeamId \ne PlayerTeamId)) \Rightarrow (Go\ to\ the\ attackhalf) \end{array} , \qquad (11)$$

where $BALLTeamId$ is number of team, who controls the ball, $PlayerTeamId$ is number of player in the team.

Attack actions. Goal of the team in attack is to score the ball in the basket ring. First it is necessary to deliver the ball up to the basket of opponent. The ball can move on the court using two of the ways:

1) Player runs with ball;

2) Player passes the ball to partner;

Two selection strategies can be used such as selection of partner nearest to opponent basket ring and selection of partner, whose position is optimal. The algorithms implementing these strategies are described in detail in [2]. Optimality of the position of each of partners on the court is described by some value. This value is named an evaluation of the player. When a player needs to pass the ball, he selects teammate with the highest evaluation.

Evaluations of all partners are defined in following way:

$$t_1^n = f(d_1^n) + \sum_{\forall m, d_{2m}^n < \varepsilon} g(d_{2m}^n), \tag{12}$$

where d_1^n is the distance between nth teammate and basket; d_{2m}^n is the distance between nth teammate and m-th opponent; $f(x)$ is a function that evaluates the goodness to shoot for the teammate; $g(x)$ is a function that evaluates the threat from opponents.

When partner is defined, ball's trajectory is calculated. Initial parameters of a pass are calculated such a way that the ball has not been intercepted by opponent and the given partner will be the first player who can intercept the ball.

1) The horizontal direction of the ball is defined as:

$$\begin{array}{l} DirXY = \left| ArcCos((Partner.x - Agent.x) / Dist(Partner, Agent)) * 180 / Pi \right| \\ if\ (Partner.y < Agent.y) \\ then\ \ (DirXY = -DirXY) \end{array}, \tag{13}$$

where $DirXY$ is horizontal direction of pass; $Partner$ is the coordinate of the partner who receives the pass; $Agent$ is the coordinate of the partner which passes; $Dist$ is function defining distance between two points on a court.

2) Definition of ranges of change of initial parameters of a pass is made as follows:

$$\begin{array}{l} Pow = 1..30 \\ DiZ = 40..75, \\ Time = 0..50 \end{array} \tag{14}$$

where Pow is the power of pass, DiZ is the vertical speed of pass, $Time$ is count of time steps.

4) Initial ball's parameters are calculated as:

$$\begin{aligned} &Ball.x = Agent.x \\ &Ball.y = Agent.y \\ &Ball.z = PlayerHeight \\ &Vz = Pow * \sin(DirZ * Pi/180) \end{aligned}, \tag{15}$$

where *Ball* is the coordinate of the ball; *PlayerHeight* is the constant of the server defining height of the player; *Vz* is vertical speed of the ball.

4) The ball's parameters are calculated as follows with time:

$$\begin{aligned} &Vz = Vz - G \\ &Ball.x = Ball.x + Pow * \cos(DirZ * Pi/180) * \cos(DirXY * Pi/180) \\ &Ball.y = Ball.y + Pow * \cos(DirZ * Pi/180) * \sin(DirXY * Pi/180) \\ &Ball.z = Ball.z + Vz \end{aligned}, \tag{16}$$

where *G* is acceleration of free falling;

5) A hit of the ball in the given zone is defined using the following condition:

$$Dist(Ball, Partner) < \text{CatchableDist} \tag{17}$$

Coordinates of a ball and initial parameters of a pass are saved as:

$$\begin{aligned} &BallFinal = Ball \\ &MinDirZ = DirZ \\ &MinPow = Pow \end{aligned}. \tag{18}$$

6) Search of the nearest *BallFinal* to the partner is realized by the following way:

$$\begin{aligned} &\underset{\Delta \to 0}{MinDirZ, MinPow} \\ &where\ \ \Delta = Min(BallFinal - Partner) \end{aligned}. \tag{19}$$

Parameters $MinDirZ, MinPow$ are selected, at which distance between the ball and partner is the least one.

Graphic interpretation of the given algorithm is presented in fig. 4. The given algorithm builds several trajectories of a throw (curves 1, 2, 3 in fig. 4) and chooses one of them, which is closer to the partner. In case shown in fig.4, such curve is the trajectory 3. This algorithm also calculates parameters of shoot to the basket ring.

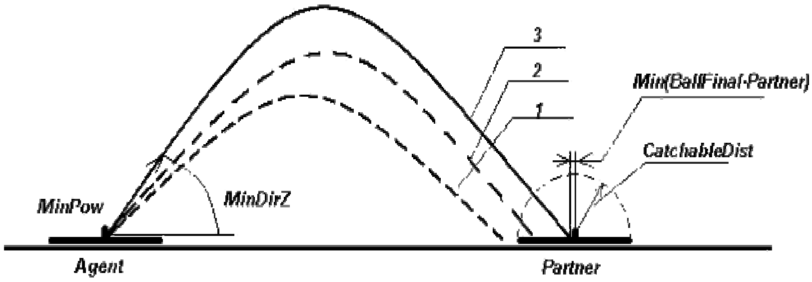


Fig. 4. Minimization of pass parameters

Defense actions. As defensive strategy, the personal marking of players is used. The player *i* chooses of the opponent *j* for marking if the condition (20) is satisfied.

$$PlayerID_i = PlayerID_j. \qquad (20)$$

Further it is necessary to choose a position on a field, to which the player should move to mark the opponent. Coordinates of this position are calculated by the formula (21):

$$P_i = (P_j + (P_j + CircleOwn)/2))/2, \qquad (21)$$

where P_j is coordinate of the *j*th opponent; *CircleOwn* is coordinate of the own basket ring; P_i is required position.

P_i is the point located as a midpoint linking opponent and midpoint, linking opponent and the basket ring. If player is closer to the opponent, then marking player will be closer to him.

At the moment of a pass or a throw when the ball is switched in a status FREE, the marking player can make interception of the ball. If it was possible, then the player is switched to attack, else player is switched to defense.

Example of immunological network for simple agent behavior.

At first, consider implementation of immunological network which must select behavior of the agent in **simple situation**: basketball agent tries to solve either to throw the ball to opponent ring, to catch ball, or to pass to other agent (partner) such a way, that it will not be intercepted by another agent (opponent).

The idea is calculation of the distances to ring, opponent, and ball positions. Agent calculates these parameters (conditions) that denotes as antigens.

All these conditions have resulted in applying immunological network represented in Fig.7, which can learn given examples, and then work even in that situations, which were not given to system at the learning time.

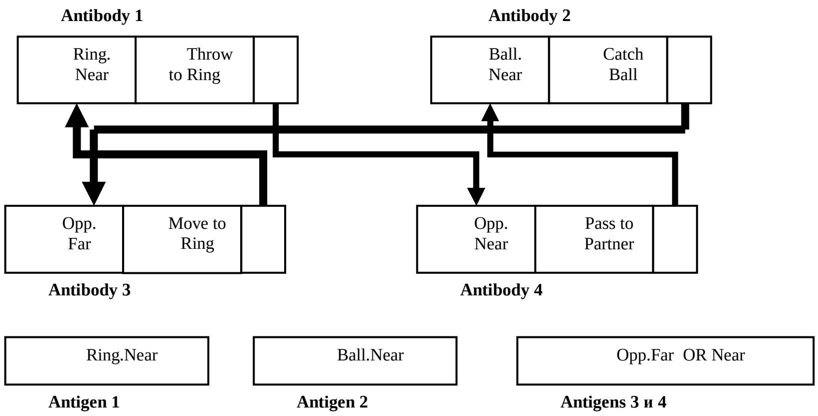


Fig.6. Example of the network 1

The network can be described as following. Here, four antibodies are prepared in advance that respond each to corresponding antigen. If antibody 1 is activated, it means that antigen1 (Ring.Near) is detected and Throw to Ring behavior. However, if opponent is near (Opp.Near) it antibody would give way to other antibodies represented by its idiotope (in this case, antibody 4) to make Pass to Partner behavior. Now assume that opponent is far (Opp.Far), in this case antibodies 1, 2, and 4 are simultaneously stimulated by the antigens. As a result, the concentration of these antibodies increases. However, due to the interactions indicated be arrows among the antibodies through their paratopes and idiotopes, the concentration of each antibody varies. Finally, antibody 2 will have the highest concentration, and then is allowed to be selected. This means that player catches ball (Catch Ball). In the case where opponent is near (Opp.Near), antibody 1 tends to be selected in the same way. This means that player with the ball tries to do Move to Ring behavior. As observer in this example, the interactions among the antibodies work as a priority adjustment mechanism.

Example of immunological network for complex agent behavior.

Second example relates to more complex teamwork behavior. Basketball agent can percept several situations (antigens) and form in respond corresponding behavior (if antibodies recognize these antigens via their paratopes).

Antigens/Preconditions of paratopes:

Time for attack remained is much or few (Time.Much/Time.Few);

Player is free or marked (Iam.Free/Iam.Marked);

Distance to ring is little, middle, or far (Ring.Near/Ring.Middle/Ring.Far);

Partner is free or marked (Part.Free/Part.Marked).

Behaviors of paratopes:

Player can select such **behaviors** as to throw to ring (Shoot), to dribble of ball (Dribble), to pass to partner (Pass), or to explore of situation (Explore).

The immunological network is represented in Fig. 8.

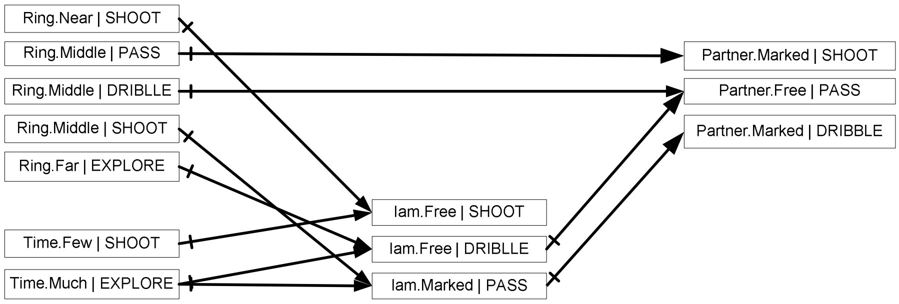


Fig. 7. Example of the network 2

In Fig. 7 the interactions between antibodies with degree of stimuli equal or more than 0.6 are shown.

The network selects behaviors in according with following explanation. If time for attack is much (Time.Much) and Distance to Ring is big (Ring.Far), then player with ball explores situation (Explore) and priority of behaviors Pass or Dribble is defined from state of player (Free or Marked) and state of partner (Free or Marked). If time for attack is few (Time.Few) or Distance to Ring is little (Ring.Near), then always the highest priority has throw to ring behavior (Shoot).

5 Conclusion

Developed the data mining module based on immunological networks able to on-line learning can increase the effectiveness of the teamwork.

Using the proposed architecture and the developed algorithms of behavior, the basketball agents based on the data mining module may be developed and investigated. They allowed training agents and automatically forming immunological networks that provided increasing effectiveness of attack and defense of the teams. In the future we intend to extend the field of application of the data mining module for agent’s coordination. Also we plan to implement data mining module based upon reinforcement learning into agent’s scenario behavior.

6 References

1. Cohen P. R., Levesque H. J. Teamwork, Nous 25 (1991).
2. Tambe, M. Towards flexible teamwork, Journal of Artificial Intelligence Research (JAIR) 7, (1997).
3. Bratman M.E., Israel D.J., and Pollack M.E. Plans and resource-bounded practical reasoning. In Computational Intelligence, #4: 349-355, 1988.
4. Stankevich L.A., Serebryakov S.V., Ivanov A.V. Data Mining Techniques for RoboCup Soccer Agents. Autonomous Intelligent Systems^ Agents and Data Mining. Proceedings of International Workshop. AIS-ADM 2005 (St. Petersburg, Russia, June 6-8, 2005). Springer 2005.

5. Mitkas, P. Knowledge Discovery for Training Intelligent Agents: Methodology, Tools, and Applications. Proceedings of International Workshop AIS-ADM 2005, St. Petersburg, Springer, 2005.
6. Nurulin, Y.R., Stankevich, L.A. Cognitive modeling and management of socio-technical systems. Proceeding of ICIL'99, S-Petersburg, Russia, 28 June/1 Jule 1999, Printed by Univ. of Southampton, UK, 1999.
7. Stankevich, L.A. A cognitive agent for soccer game. Proceeding of First Workshop of Central and Eastern Europe on Multi-agent Systems (CEEMAC'99), Printed by "Anatolyi", S-Petersburg, 1999.
8. Stankevich, L., Trotsky, D. RoboFIBA, 2005.(in Russian).
9. Stankevich L.A. Multiagent technology in cognitive control systems for autonomous robots. «Extreme robotics. Proceeding of X-th Conference, St. Petersburg», St. Petersburg STUPress, 1999 (in Russian).
10. Stankevich L.A. Cognitive structures and agents in intellectual robot control systems. AI News #1, 2004, pp. 41-55 9 (in Russian).
11. Dasgupta, D., Attoh-Okine, N. Immunity-Based Systems: a Survey. Workshop Notes-Workshop 4: Immunity-Based systems, InternationalConference on MAS, December, Keihanna Plasa, Tokyo, Japan (1996), pp. 13-25.
12. Jerne N.K. Towards a network theory of the immune system // Annales d'Immunologie (Institut Pasteur), 1974, 125C, pp. 373-389.
13. De Castro L.N. and Timmis J. An artificial immune network for multimodal optimization // 2002 Congress on Evolutionary Computation. Part of the 2002 World Congress on Computational Intelligence, Honolulu, Hawaii, USA, 2002, pp. 699-707.
14. Tarakanov A.O., Kvachev S.V., and Sukhorukov A.V. A Formal Immune Network and Its Implementation for On-Line Intrusion Detection // LNCS 3685: MMM-ACNS 2005 / Gorodetsky V., Kotenko I., and Skormin V. (Eds.), Springer-Verlag, Berlin Heidelberg, 2005, pp. 394-405.
15. Watanabe Y., Ishiguro A., and Uchikawa Y. Decentralized Behavior Arbitration Mechanism for Autonomous Mobile Robot Using Immune Network // Artificial Immune Systems and Their Applications / Dasgupta D. (Ed.)", Spring-Verlag, Berlin, 1999.
16. Richard S. Sutton and Andrew G. Barto. Reinforcement Learning: An introduction. MIT Press, Cambridge, MA 1998.
17. Bingchen Hu, Jiming Liu, Xiaolong Jin. From local behaviors to global performance in a multi-agent RoboNBA system. In Proceeding of IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT 2004, Beijing), 2004, pp. 309-314.