Phoenix Soccer 2D Simulation
Team Description Paper 2015

Aryan Akbar Poor[1], Mohammad Poor Taheri[1], Alireza Mahmoodi[1], Ali Charkhestani[1]

[1] Iran Atomic Energy High School

Phonix.robocup@gmail.com

Abstract. In this paper we will discuss goals, ideas, and algorithms used in Phoenix Soccer 2D simulation team and try to present a quick overview of team's performance and players' capabilities. In this paper Offense strategy, Defense strategy and behaviors like shoot and block will be explained. It is noteworthy to mention that the base used in Phoenix is Agent2D.

Keywords: Offense strategy, Defense strategy, shoot, block, technical paper, Phoenix, Soccer2D simulation

1 Introduction

Phoenix team is formed from Iran Atomic Energy High School Students with the goal of improving scientific progress and efforts to advance the objectives of RoboCup activities. After Iran Open 2014, Phoenix members were chosen from Atomic Energy High School students interested in this field and started activities from September2014. Team members try to improve offense strategy that is based on chain action and rewrite behaviors like chain action shoot and defense strategy that will be explained in this report.

2 Offense Strategy

First chain action will be explained. There is an object in chain action that is named "State". State has attributes like player positions, ball position and ball owner position. In other words State includes a summary of World Model's information.

In order to make decision, the ball owner player first creates a new State according to world present situation. Then it checks all "on the ball behaviors" including shoot, dribble and cross. It considers possible behaviors among them and creates new state based on ball and ball owner player positions in the Parent State and amount of changes according to the behavior. It saves the new State and executes this algorithm for every new States to form a decision tree. The best state is the one which is closest to the opponent goal. Finally it chooses the chain which leads to the best state and executes the first behavior of that chain.

Fig. 1 shows an example of chain action tree and Fig. 2 shows the process of choosing the best behavior based on the best chain.

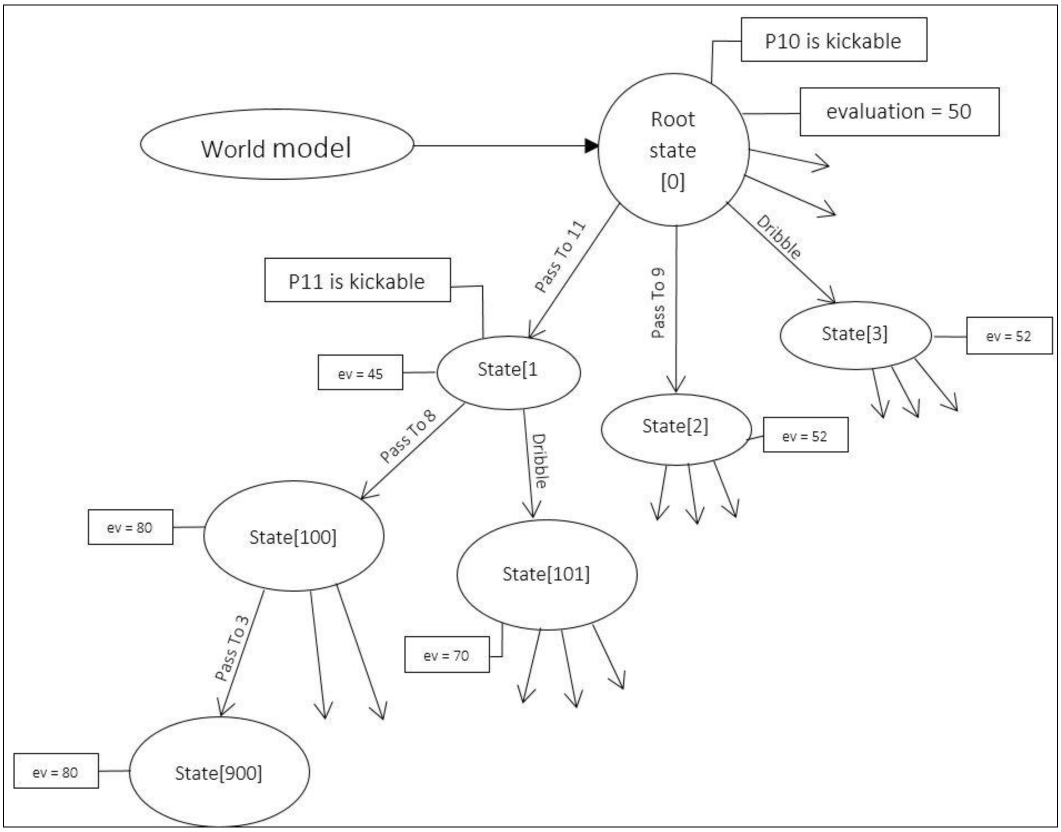

Fig. 1.an example of chain action tree

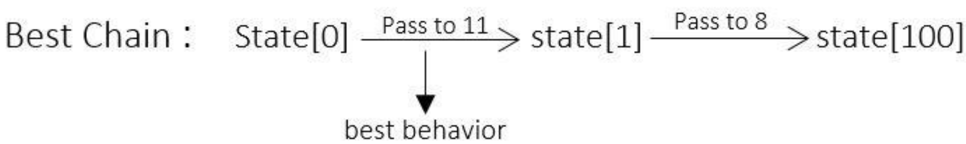

Fig. 2.executing the best behavior based on the best chain

According to the above mentioned explanations, one of the biggest problems in this structure is evaluation of each state and it will be better to add another attribute like safety in state evaluation. For this purpose in each state the most dangerous opponent player which is the nearest opponent to the ball will be chosen and according to this player's distance, we reduce state value. This will cause Phoenix players to own the balls for longer period and losing the ball less often. This have two direct results; firstly, the opponent will own the ball less often, so it will decrease offense power of the opponent; secondly, by increasing players passes, opponents' players' energy will decrease so fast.

The amount of opponent players' danger counted according to the time that it will take to reach the ball. This amount will be 1 at least; so if the danger amount was 1, it means that this player was in the most dangerous situation and the amount of reduction in state value would be the most. In order to do this we considered an array and filled it based on our experience. For example the value in cell 5 was 20, means that if in one state an opponent player's distance from ball was 5 then 120 units should be subtracted from the value of the state. This evaluation caused some problems that could be solved by an idea like Genetic algorithm. The better the opponents block, the value in this array should increase. So these values were counted for two branches.

First this 10 cell array was filled based on experience and then 10 games between Phoenix with Wright-Eagle were executed with "Auto play" software. According to the average of the outcome results of each array was analyzed by using Cyrus analyzer software, so percentage of owning the ball, number of correct passes and number of correct dribbles were counted.

$$\text{Ev of each array} = \alpha + \beta + \gamma + \lambda$$

$$\alpha = \text{Phonix score} - \text{WE score}$$

$$\beta = (\text{Phonox ball} \dots\ - \ \text{WE ball}\dots)/100$$

$$\gamma = \text{correct pass} / \text{all pass}$$

$$\lambda = \text{correct drrible} / \text{all drrible}$$

Then five couples from arrays were chosen based on results of each array and split the array into two parts and combined them with each other. In order to get better results we added each number with a number between (-5, 5) and then did the above mentioned actions. The condition of descending array was checked in each new array. Figure 3 shows an example of it.

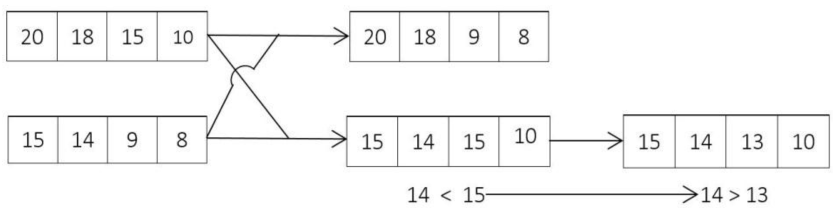

Fig.3 .arraye combination

In table 1the average of the results with the team WrightEagle which has a strong block behavior and Axiom which has normal block is shown.

Table 1. Comparing block with WE and Axiom

	WE	Axiom
Agent 2D	0-10	0-5
Phoenix	0-6	2-3

Agent 2D with this idea ← **Phoenix**

In order to increase offense speed and counter attack power, the time interval between "present state" and "the state that is being evaluated" was counted and applied in evaluation.

3 Shoot

Shoot behavior was implemented from the beginning completely. An algorithm was used to omit parts of the goal according to the "out" possibility and another algorithm was used to choose best shoot according to the goal possibility that is explained in the following.

Most teams assumed goal smaller than the real size to reduce the effect of noise, but this was fix value. For example they assume the goal size [-5, 5] instead of [-7, 7]. In Phoenix with too much tests and observation we got to the conclusion that shoot noise is angular. The more the distance from goal, the more the difference between shoot point and result point.

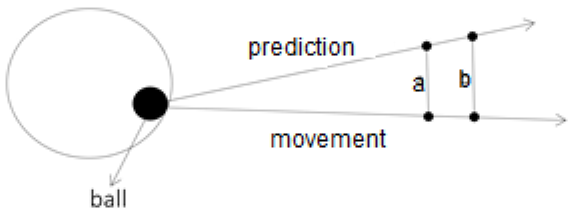

Fig. 4.the relation between shoot noise and angle

As you can see in figure 4 b is more than a, and this continuous value is growing as the distance from the goal increased. So when we are farther from goal this value should be more. We also understood that if ball poscount, ball speed and the ball owner player speed increased this difference would be even more. This value is usually between α and β. The error calculated based on formula 1(Calculation of angle difference) and 2(Calculation of distance error).

$$ErrAngle=\alpha+(\beta-\alpha)\left(\frac{Vplayer}{Vmaxplayer}\times\frac{Vball}{Vmaxball}\times\frac{distTotarget}{40}\times\frac{ballposcount}{ballposcount+1}\right) \quad (1)$$

$$ErrDist = distTotarget \times \sin(ErrAngle) \quad (2)$$

In phoenix algorithm the goal was divided into n different points that player assumes he can shoot to one of those points. Shoots those were likely to result in a goal saved and evaluated. The best shoot is the one that would cause a goal with the most probability. In order to calculate goal probability, the minimum time difference between “the time that it takes for the ball to reach points in the path” and “the time that it took for the opponent to reach the same point” was used. For better understanding take a look at figure 5.

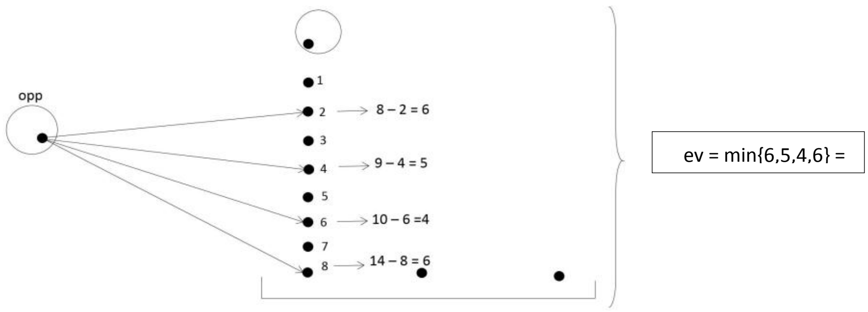

Fig. 5. Shoot evaluation

4 Defense Strategy

In present defense strategy, just block has been implemented. We are trying to add mark behavior for the competition and by using coordination between these two behaviors achieve a harmonic strategy for defense behavior.

In this strategy we attempted to use opponent decision tree with the depth that related to each player, to make the best decision for mark and block.

5 Block

In phoenix block each player simulated the opponent players’ paths and then calculated the minimum time for himself and his teammates to block the opponent. In the case he could block the opponent before his teammates, he performed the block behavior. Most teams use Agent2D base dribble behavior, so in order to simulate opponent’s dribble we reversed the dribble evaluation in Agent2D and achieved a very accurate simulation.

In this method some points around ball owner player considered and evaluated by reversed evaluation and best opponent dribble was predicted. We assumed that the dribble speed is fixed.

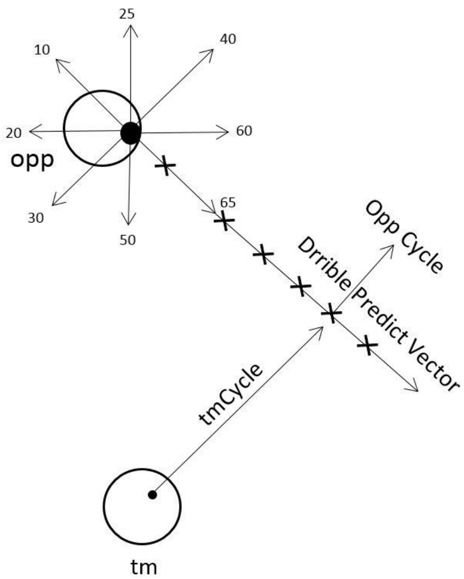

Fig. 6. Dribble simulation

Dribble simulation is showed in figure 6. If tmCycle was less than oppCycle this target is suitable for block. In future we decide to calculate dribble speed more accurately and dynamically.

References

1. " RoboCup Soccer Simulation", http://en.sourceforge.jp/projects/rctools/, 2015
2. R.Khayami, N.Zare, M.karimi, P.Mahor, F.Tekara, E.Asali, A.Keshavarzi, A.Afshar, M.Asadi, M.Najafi, "CYRUS 2D Simulation Team Description Paper 2014", The 18th annual RoboCup International Symposium, Brazil, Joao Pessoa, 2014
3. Gabel, T., Riedmiller, M. Trost, F.: A Case Study on Improving Defense Behavior in Soccer Simulation 2D: The NeuroHassle Approach. In: Springer-Verlag Berlin Heidelberg (2009) 61-72