Ri-one 2007 Team Description

Shinya Otsuki, Tatsuhiko Katayama, Kouji Ohnaka, Tomohiro Iwasaki, Ryo Ishihara, and Katsuari Kamei

Ritsumeikan University

Abstract. This paper gives a brief description of *Ri-one 2007*, the RoboCup soccer simulation 2D team. *Ri-one* is a club activity team founded in 2005 by *Ritsumeikan University.* We mainly show DDT which is new our decision making architecture followed by some methods used in DDT.

1 Destination based Decision Tree

In *Ri-one2006*, to determine the agents' behavior, they use a model which evaluates World Model with IF-THEN rule. In this model, they decide an action after evaluation of their environment. For example, when an agent make a pass, the agent determine the most appropriate receiver, then perform a pass behavior. Many teams including Ri-one2006 adopt the IF-THEN model. However, the model has a fault that the evaluation is not always done properly, because there are many cases that the agents has two or more solution in evaluation. In addition, evaluation based on the IF-THEN model are performed in order determined before so that agents can have less flexibility.

Therefore, we propose a new decision making model named DDT (Destination based Decision Tree). DDT evaluates the destination of agents and determine their best behavior with decision tree. DDT has destinations of agents as node and their behaviors as leaves. Agents can change the order of evaluation flexibly with recombine the tree structure.

In DDT, structure of the tree is based on following two rules.

1. a lower level node has a more concrete destination than its ancestor
2. a left node contributes better to its ancestor if it is achived than its right node.

To make a decision, agents search DDT with DFS (Depth-First Serch). The agents search the tree until they reach a leaf. After the aegnts reach any leave, they judge whether they can perform a behavior in the leaf. If the agents can perform the behavior, they finish searching there. If not, the agent continue to search. For example, when an agent has DDT shown Fig.1. as a model of making decision, the agent starts judgement of the leaves from the most left leaf **Make a shot**. If the agent can make a shot, perform it and finish searching. If not, the agent judges **Dribble ahead** leaf in next. Note that the order of evaluation is not changed in the above process.

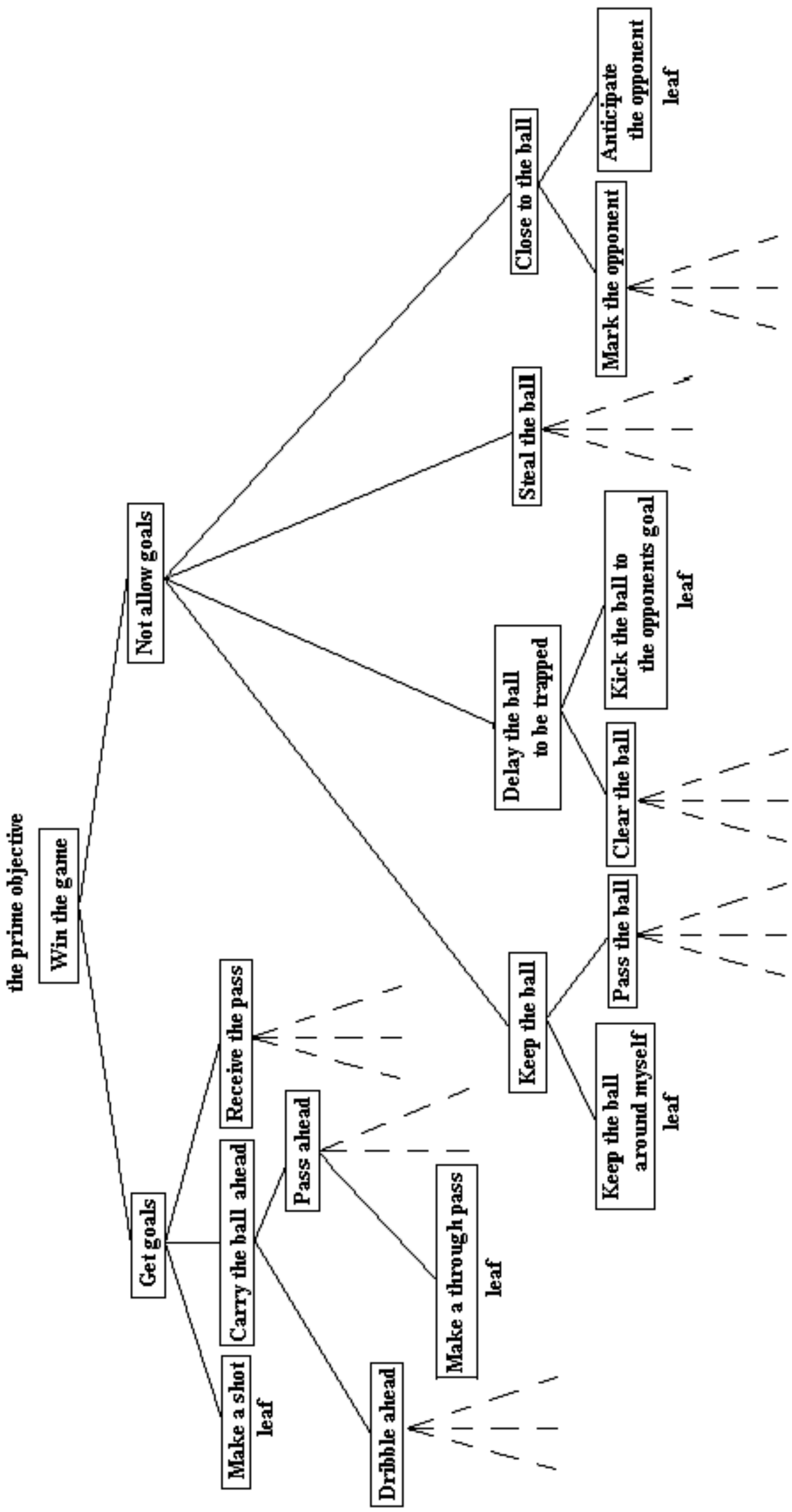

Fig. 1. Example of DDT

Processes to change the evaluation order is done at the same time as seaching. When an agent reach any node, the agents evaluate its descendants' achievement of the destination. Then, the agent recombine the branch in the order of ascending degree of the achievements. For instance, we think the situation that an agent reach the node **Carry the ball ahead** when the agent have the ball. The agent judges **Pass ahead** is not achived yet because the agent dose not have the ball if the pass behavior is performed. Moreover, the agent judged **Dribble ahead** destination is achieved a little, because the agent is nearby the ball if the dribble behavior is performed. **Pass ahead**'s degree of achievement is less than **Dribble ahead**'s, so that the agent recombine the branch as shown in Fig.2.

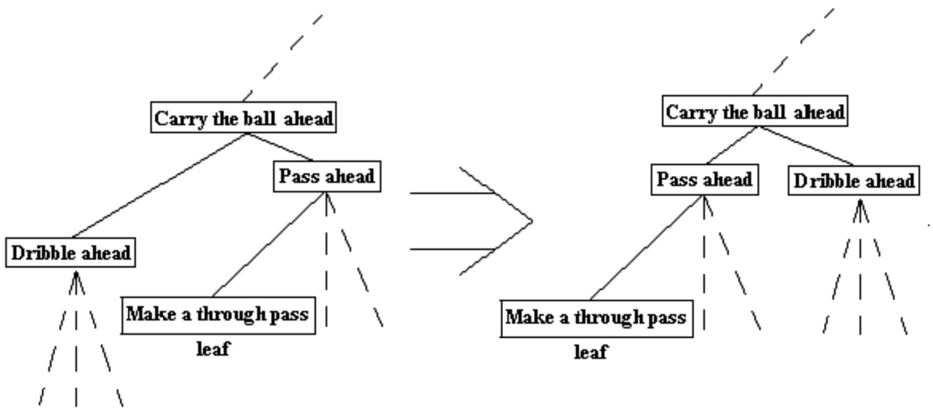

Fig. 2. Recombine of Branch

DDT enable the agents to make a decision more flexibly than IF-THEN rules. However, there are many heulistic parts in DDT like the division of the destinations, because the destination is too complex idea. We have been trying to find how to put the destinations well.

2 DDT in the Dribble behavior

In *Ri-one2007*, the agents use the DDT decision making model. In the DDT model, it is too difficult to see clear defference between the behavior and the destination. For example, the **Pass ahead** destination can be implemented as a behavior. In addition, the destination also can be also divided into the lower destinations (See Fig.3). For this reason, we also use the DDT model in the behaviors level. We introduce the dribble behavior based on the DDT.

In DDT, the dribble behavior is represented as Fig.4. We assume that the search of DDT starts from the node **Dribble ahead**. First, the agent evaluate

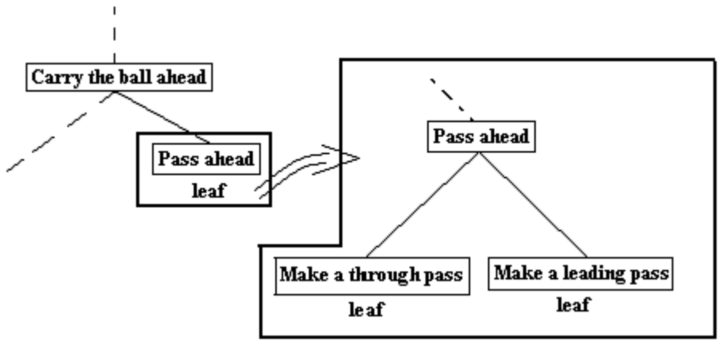

Fig. 3. Border of the behavior and destination

the two descendants **Dribble to close to ball** and **Dribble to body direction**. This evaluation is done with the distance from the agent to the opponent's goal. If the agent is far away the opponent's goal, the destination **Dribble to close to ball** is take precedence If not, the agent take precedence the destination **Dribble to body direction**. Secondly, the agent recombine the tree and continues to search. If the agent reach any leaf, the search is finished. Moreover, the procedure of DDT is also finished. Finally, the agent determine the actual action with calculation. We do not divide the tree into the lower level to build existing methods with IF-THEN rules into the tree.

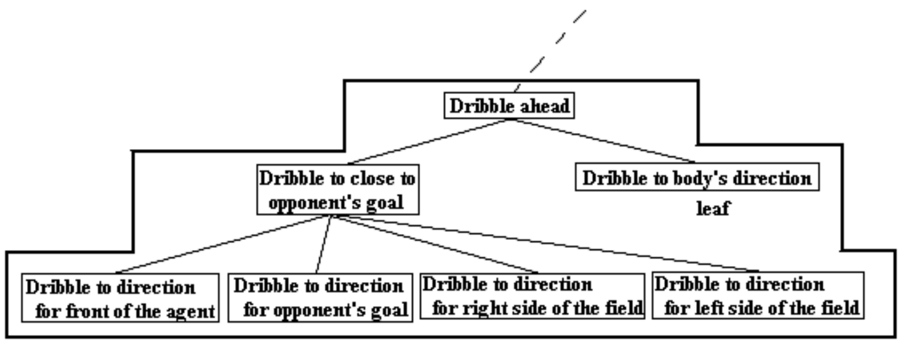

Fig. 4. Dribble behavior with DDT

The process of calculation to determine the actual action consists of following three steps.

1. determine the number of dash action
2. calculate the position to kick the ball to
3. kick the ball or dash to ball

In the first step, the number of dash is determined heulistically. The agent determines the number of dash n with distance from the agent to the closest opponent at the direction *direction* determined by DDT. In the secod step, the agent calculates a maximal distance *dist* which the agent can move with n times dash. Moreover the agent calculates a vector v with *dist* and *direction*. Finally, if the agent can kick the ball, the agent kick the ball with the vector v. If not, the agent dash to the ball.

3 Intercept Behavior

In this section, we show how to determine the optimal interception point In *Ri-one 2007*. Note the interception cycle means the least number of the cycles to intercept the ball.

A prediction of the interception cycle consists of following three steps. First of all, we predict the future ball position after i cycles b_i. Let v_i be the velocity of the ball after i cycles. b_i is shown as (1). Secondly, we predict C_{turn} cycles required to direct the body to b_i, and C_{dash} to reach b_i with dash. Let the agent's speed after t cycles be $spd(t)$ and the maximal angle the agent can turn with a turn command be $ang\_turn(t)$. Because the angle is affected by the agent's speed, $ang\_turn(t)$ is represented as the equation (2). When $ang\_total(n)$ is the angle which the agent can turn with n turn commands, $ang\_total(n)$ is represented by (3). If the angle needed to turn is ang_diff, we can determine C_{turn} with resolving the equation (4). Moreover C_{dash} can be determined with (5). $dist(i)$ in the equation (5) is the distance from the agent's current position to b_i. Finally, if $C_{turn} + C_{dash} \leq i$, we detemine i to be the interception cycle.

$$b_i = b_0 + v_0(1 - ball\_decay^i)/(1 - ball\_decay) \tag{1}$$

$$ang\_turn(t) = 180.0/(1.0 + inertia\_moment \cdot spd(t)) \tag{2}$$

$$ang\_total(n) = \sum_{k=0}^{n-1} 180.0/(1.0 + inertia\_moment \cdot spd(k)) \tag{3}$$

$$ang\_diff - \sum_{k=0}^{n-1} 180.0/(1.0 + inertia\_moment \cdot spd(k)) = 0 \tag{4}$$

$$dist(i) - kickable\_area\_width - C_{dash} \cdot player\_max\_speed = 0 \quad (5)$$

After the prediction of the interception cycle, we determine the optimal interception point. When the agent's intercept cycle is *i_agent* and the opponents' minimum intercept cycle is *i_opponents* The agent can trap the ball between b_{i_agent} and $b_{i\_opponents-1}$. We evaluate each of them, according to the equation (6). In the equation (6), v_{i_kick} represents the velocity the agent can add to the ball with maximal kick from b_i to the specific position. The specific position is, for example, the opponent's goal, a teammate's position receiving pass We define a point which has the maximal evaluation to be the optimal interception point. The equation (6) is represented graphically in Fig.5

$$\begin{cases} Evaluation(i) = |v_i + v_{kick\_i}|(1 - ball\_decay^q)/(1 - ball\_decay) \\ q = (i\_opponent - 1) - i\_agent \end{cases} \quad (6)$$

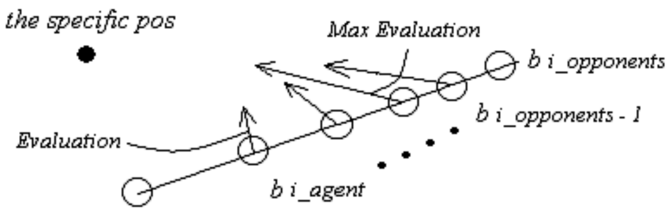

Fig. 5.

4 Positioning

In *Ri-one 2007*, we have made the method to do positioning in specific situation. To improve offensive power of out team, we need another positioning system in attacking, if the through pass which is our main way of attacking, is interfered. Note that the method we have made is for receiving passes efficiently, so that it is for attacking agents without the ball. Moreover it can not be used when the ball is not in our possession or the ball is over the offside line. In those situations, the agents use the *SBSP* (the positioning system with some parameters set to the each agents) [1], or perform the mark.

First of all, we set three parameters, *x_min*, *y_min* and *y_max*, to each attacking agents. These three parameters are the distance of x or y direction from the ball. Usually, we set about $20.0m$ to them. Using these parameters and

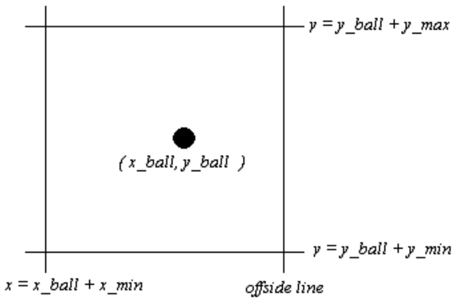

Fig. 6. Example of making box

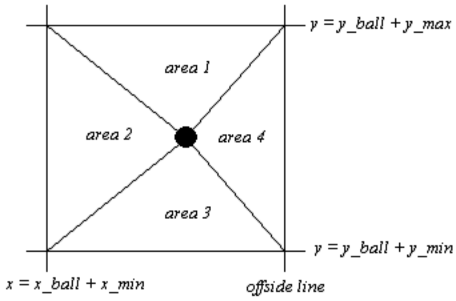

Fig. 7. Example of division into four area

x coordinate of the offside line, we make a box (Fig.6). Moreover, we divide the box into four area. It is shown in Fig.7.

Secondly, the agent determine the farthest spot from the opponents in each area. To determine the spot, the agent calculate the widest direction and the widest width among the opponents, and determine the intersection of them. The widest direction means the direction having the largest angle between the opponents. Moreover, the widest width means the distance of x or y coordinates which has the farthest between the opponents. In the area, *area*1 or *area*3 in Fig.6, the widest width is decided by y coordinate distance. In the other area, the widest width is decided by y coordinate distance. We define the intersection of them to be the farthest spot (Fig.8. Fig.9, Fig.10). The agent get four spots with this operation.

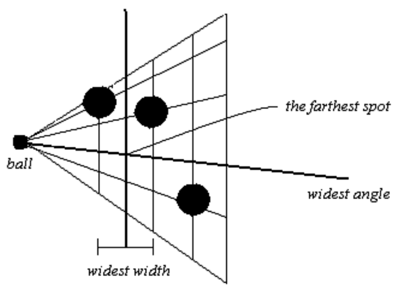

Fig. 8. Example in *area*4

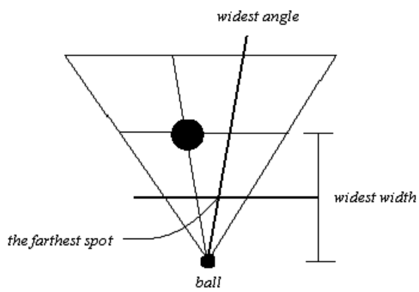

Fig. 9. Example in *area*3

Finally, we define the closest spot to the agent to be the position the agent should move to (Fig.11).

With this method, the agents can find the spot where the agent can receive pass safely. It shows it strength especially in the middle of the opponents.

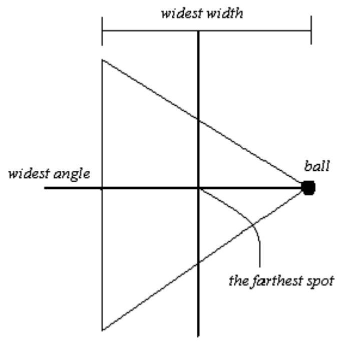

Fig. 10. Example in $area2$ (with no opponents)

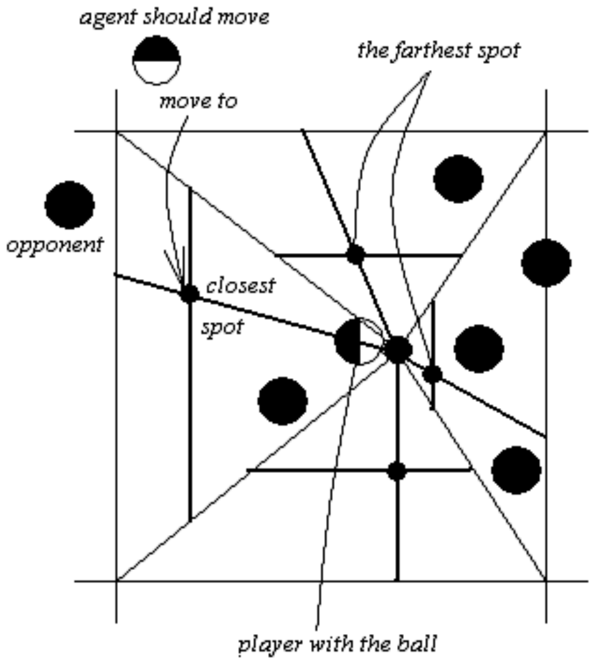

Fig. 11. Determination of the position the agent should move to

References

1. Luis Paulo Reis, Nuno Lau and Eugenio Costa Oliveira. Situation Based Strategic Positioning for Coordinating a Team of Homogeneous Agents