UFAngryBirdsC Agent

Jonathan Ohara de Araujo Fabrício Olivetti de França
Federal University of ABC
jonathan.ohara.araujo@gmail.com

***Abstract*— The UFAngryBirdsC agent is an agent to solve the 21st first levels of Angry Birds for Google Chrome. The agent can be divided in two parts: training and execution. In training mode, the agent calculate all possible shots and test the greater part of them, the agent tests first the most promising shots defined by a heuristic, and then store the results in a file. The results are stored in a graph structure where nodes in odd heights are possible shots and nodes in even heights are possible results of a shot. In execution mode, the agent reads the graph and evaluated the best shot based in ExpectMiniMax algorithm.**

I. Agent

The agent was built assuming that the same shot can generate more than one result (because of communication issues). These results are called states. Each state keeps a lot of information, like the shot that generates him, points, one flag indicating if the game ends, and a number that counts the number of times that the state was reached.

The shots and states are organized in a graph (tree) structure. In graph the nodes in odd heights are shots (or Possible Shots, if they was not tested) and in even heights are states.

The figure 1 shows a simple graph of the fifth level of Angry Birds.

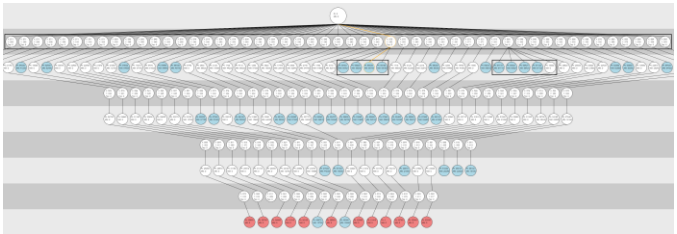

Fig. 1. Level 5 resolution graph.

The agent has two modes: training and execution. The training mode builds the graph, calculating possible shots, testing them and writes the results in some files. The execution mode chooses the best shot according of the level.

II. Training mode

The training mode can be separated in three parts: calculate the possible targets for shots, test the shot and store the result.

To calculate the possible targets, the agent gets all objects of the level (pigs, woods, stone etc), each object can have more than one target point if exists points with euclidean distance greater than 10 pixels (10 pixels is half width of a bird) in object. Then the agent discards unreachable shots by the method “isReachable” from ABUtil.

After get all possible shots, the agent calculates the “fitness” of the shots by the equation (the lowest fitness is chosen first):

V = d * m * x

Where:

V is the fitness;

D is the euclidean distance of the target point from the closest pig;

M is a factor that depends of object (TNT = 0.5, Pig = 1, Ice = 1.5, Wood = 2, Stone = 3)

X is a factor that if the target.x is less than the closestPig.x the value is 1 otherwise the value is 2.

The test generates a state new state. If the shot was already tested, the new state is compared with the old states, if one of these states has similar points, the number of times reached of "old state" is added in one.

The final operation is store the graph. The graph is stored in two JSON files, one for shots and one for states. Each state has an identification of the parent shot, and each shot has an identification of parent state (The root state is indicated by the state without parent shot).

III. Execution mode

In execution mode the graph with shots and states is loaded then the best shot is chosen.

To find the best shot the agent uses a modification of expectMiniMax algorithm.

Simplified code of expectMiniMax used in the agent:

```
function expectiminimax(node)
    if node is a terminal state
        return points;
    else if node is a state
        let α := -∞
        foreach shot of state
            α := max(α, expectiminimax(shot))
    else if node is a shot
        let α := 0
        foreach state of shot
            α := α + (Probability[state] *
expectiminimax(state)
    return α
```

After execute each shot, the agent compares if the state was previously tested. If is true, the agent recalculates the expectminimax based on the actual state. If is a new state, the agent calculates the possible shots and chooses the shot with best fitness (like the training mode).

The follow picture shows more detailed graph of level 9.

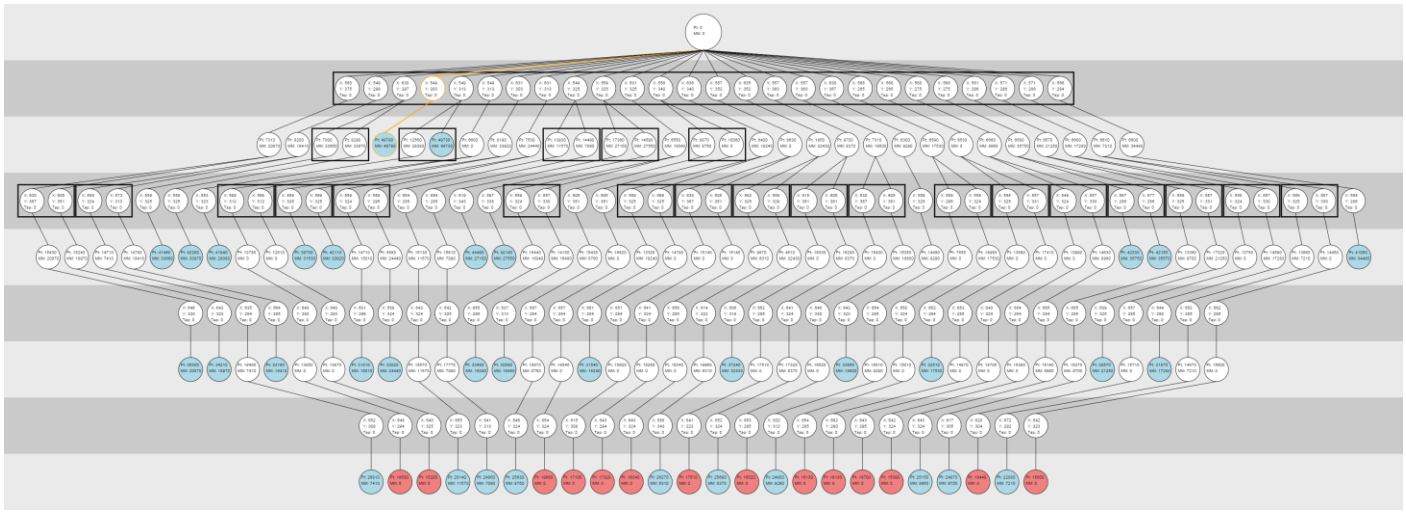

Fig. 2. Level 9 resolution graph.

Image observations:

- Circles: nodes. Inside the circle are information like points and target;
- Blue/Red Circles: Terminating nodes (blue = win, red = loss);
- Orange line: best shot;
- Rectangle around nodes: indicates that the nodes inside the rectangle are brothers;

The images of graph were generated by a tool developed to debug the graphs and results of the agent.