Angry birds Phoenix

Brief Code explanation

Analysis.java:

Format: Name of function // It's working

Cluster //A linkedlist of type rectangle of all the blocks which are joined together(threshold 4 blocks)

addWoodenObstacles() //add all the wood blocks

addIceObstacles() //add all ice blocks

addStoneObstacles() // add all stone blocks

addpigs() //add all pigs

getJointObstacles() // returns the list of all obstacles

getClusterBorder() //returns a rectangular border along with points which contains a cluster(group of blocks joined together)

CountPigsInCluster() //counts the no. of pigs enclosed in a given cluster(rectangle)

GetClusters() //returns a linkedlist of all the clusters present in the given screenshot

targetsInCluster() // returns a list of the leftmost (attackable) points at which joints are present in a given cluster can be used to find weakPoints for different birds after choosing a cluster

WeakSpotsForRedBird() // find a cluster with maximum pigs inside it and return the center of that cluster as the target point, and if no cluster contains a pig, return the first pig as the target point

NaiveAgent.java:

FindTapInterval(screenshot,releasepoint,sling,targetpoint)
Comparing the x coordinate of target point and the blocks to find the ones in between the trajectory and accordingly, set the tap interval of the bird.

solve(){
//choosing the target

For YellowBird, we chose the pig having the maximum factor(calculated using an equation involving x-y coordinates of pigs) and the center of that pig as the target

For other birds, as explained in Analysis.java, we used its method, WeakSpotForRedBirds() to choose the target point.

If it is the firstshot and the screenshot contains a round object made up of wood or stone, we directly hit that object with the higher angle returned by the trajectory planner.

//choosing the trajectory

As the trajectory planner returns either 1 or 2 trajectories between release point and the target point, we introduced two cases:

In case 1 trajectory is returned, compute the tap interval and directly shoot.

In case 2 trajectories are returned, initially the agent computes the tap interval, and then checks for the trajectory having a reachable path to target(without any obstacles) and computes the shot for the reachable one.

If it finds both the trajectories as unreachable, it randomly chooses between high and low angle with equal probabilities.

*firstshot always shoots at lower angles
}

Scores

Total: 981740
Level 1: 33610
Level 2: 60480
Level 3: 41290
Level 4: 28980
Level 5: 67110
Level 6: 25980
Level 7: 30110
Level 8: 49570
Level 9: 49900
Level 10: 53370
Level 11: 58390
Levle 12: 58410
Level 13: 51330
Level 14: 65640
Level 15: 31230
Level 16: 51140
Level 17: 42270
Level 18: 51080
Level 19: 32110
Level 20: 36150
Level 21: 63590