Report

林逸晗 2016010766

Basic Tasks (1)

1、ReflexAgent： Improve the ReflexAgent in multiAgents.py

算法实现：

核心在于修改 evaluationFunction

主要的考虑的点：

1、以当前的分数为基础分进行考虑
2、需要考虑多个ghost的情况
3、需要考虑考虑ghost的安全时间
4、需要考虑ghost十分近的时候的保命操作
5、需要考虑食物的分布
在函数中，我引入了两个参数alpha（关于最近食物的）和beta（关于ghost的），便于将其归一化考虑 / 调参，总分数=当前分数+ghost相关分数+food相关分数

对于ghost的分数，我做了如下考虑：
1、循环，使用曼哈顿距离判断最近的ghost是哪一个
2、判断当前ghost的剩余安全时间，如果大于十，取当前ghost的距离为关于ghost的分数g_score，否则取其负值为g_score
3、当最近ghost的距离小于2，将食物相关的分数调为0，意思即把所有的权重都放在ghost的距离上。

对于food的分数，我做了如下考虑：
1、计算出当前pacman的位置到最近食物的曼哈顿距离min，到所有事物的总距离sum，祖远距离max
2、尝试（min，max，sum）*alpha，带入最终的 evaluationFunction，选取表现最好的使用

最终，alpha=0.5，beta=0.2，有较好的表现如下：

Average Score: 1240.8

Scores: 1242.0, 1233.0, 1238.0, 1251.0, 1239.0, 1250.0, 1239.0, 1239.0, 1241.0, 1236.0

Win Rate: 10/10 (1.00)

2、MinimaxAgent： Implement minimax algorithm

算法实现：

定义函数：

1、Min(self,num, index, gameState, depth=0)

其中，index表示当前的ghost的序号，gameState为传下来的simulation生成的状态，depth为搜索的深度。

index用于适应镀铬ghost的搜索，相当于每一个ghost产生一个Min层，直到index==num的时候进行depth-1的操作，并且让pacman进行下一步探索，直到depth==0的时候或者局面决出胜负返回最终值。

2、def Max(self,num ,gameState, depth=0)

用于探索Pacman的最大收益的action

其他部分和课件中的伪代码是一致的。

运行结果如下：

Won 0 out of 1 games. Average score: 84.000000 ***

*** PASS: test_cases/q2/8-pacman-game.test

原因解释：

对于对手理性操作必输的局面，Pacman会选择速死，这是考虑考虑到时间的消耗，在逼死的情况下死得越早分数越高。

Basic Tasks (2)

3、AlphaBetaAgent　Implement alpha-beta pruning algorithm

算法实现；在MinMax的基础上进行修改

对于每个Min，Max层都引入一个 (alpha，beta) 区间，相当于是提前确定当前能够判断决定的区间，用于剪枝

对于Min层，为上层提供的信息是：<= beta，其中beta是已经探索过的值中的最小值，意思即，本层以上的搜索要求的是最小值不小于beta，否则对max没有意义，对本层以下的搜索，要求是更新beta值，意图给上层返回最小值的区间。

对于Max层，和Min层是对偶的，很容易理解。

对于多Ghost的情况，同样更新alpha，beta，只不过调用的下一步决策是min，也就是让另外一个ghost决策。

执行结果如下：

Finished running AlphaBetaAgent on smallClassic after 1 seconds.

*** Won 0 out of 1 games. Average score: 84.000000 ***

4、ExpectimaxAgent：　Implement the ExpectimaxAgent

算法实现：

修改很简单，对Minimax的方法进行直接修改，将对下一步的所有返回值求min的策略变成求平均值（即认为ghost的所有的决策均匀出现），这样就考虑了ghost的随机性（这里是完全随机）

运行结果：

*** Finished running ExpectimaxAgent on smallClassic after 1 seconds.

*** Won 0 out of 1 games. Average score: 84.000000 ***

*** PASS: test_cases/q4/7-pacman-game.test

Bonus： Better Evaluation Function

Write a better evaluation function for pacman

直接使用q1的 evaluation function 可以拿4分

为了得到更高的分数，考虑了对安全的ghost进行追逐，对ghost较远的时候不浪费时间去追逐活着躲避，即修改了一下部分的代码，其中scoreg代表队ghost距离的评价：

```
if abs(scoreg) >10:
    alpha = 10
    beta = 0
if abs(scoreg) < 2:
    flag=1
    score = scoreg*100
```

效果：

Average Score: 1097.4

Scores: 1165.0, 1157.0, 1160.0, 1149.0, 1158.0, 971.0, 1178.0, 1174.0, 933.0, 929.0

Win Rate: 10/10 (1.00)

最终运行结果；

```
Finished at 20:53:06

Provisional grades
==================
Question q1: 4/4
Question q2: 5/5
Question q3: 5/5
Question q4: 5/5
Question q5: 6/6
------------------
Total: 25/25
```