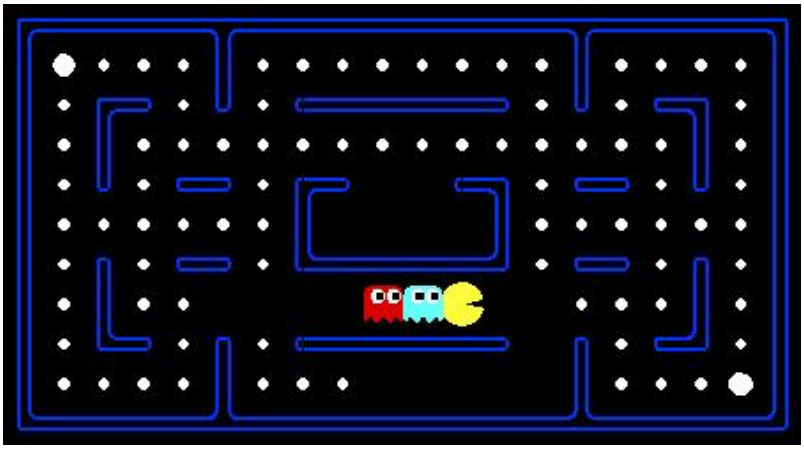
课程作业 1：多智能体搜索



问题描述

在这次课程作业中，你将为吃豆人游戏设计智能体，包括为它实现极小化极大（Minimax）搜索算法、Alpha-Beta 剪枝（Alpha-Beta Pruning）算法和期望最大（Expectimax）搜索算法，并设计评价函数（Evaluation Function）。

作业的程序包位于 QQ 群文件：课程作业\multiagent.zip。需要提前安装 Python 3（可参考 https://inst.eecs.berkeley.edu/~cs188/sp23/projects/proj0/进行安装）。

作业的程序包下载解压后，可通过运行：

```
python pacman.py
```

来游玩吃豆人游戏。也可以运行：

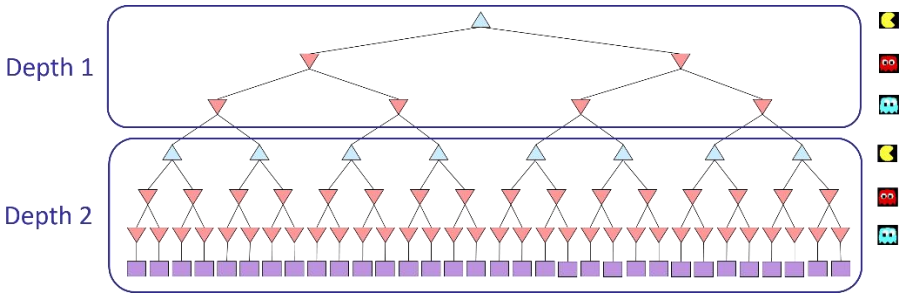
```
python pacman.py -p ReflexAgent
```

来测试项目提供的一个简单的反射型智能体。

所有你自己的算法实现都是位于 `multiAgents.py` 文件中相应名字的类或者函数下面。

作业内容

任务 1：极小化极大搜索（Minimax）



在 `multiAgents.py` 中提供的 `MinimaxAgent` 类中实现一个极小化极大搜索智能体，**所实现的算法应该可以适用于任意数量的幽灵**，所以需要你编写一个比在课堂上看到的稍微更通用的算法。具体来说，你的极小化极大搜索树对于每个 max 层都有多个 min 层（每个幽灵对应一个 min 层）。

所实现的算法还应该能够将博弈树展开到任意深度，并对你的极小化极大搜索树的截断节点使用提供的 `self.evaluationFunction`（默认指向 `scoreEvaluationFunction`）进行评分。注意这里一次搜索包含了一次吃豆人行动和所有幽灵的响应，因此，深度为 2 的搜索树包含了吃豆人和每个幽灵各自行动两次。

可以通过运行：

```
python autograder.py -q q2
```

来测试和调试你的代码实现。

任务 2：Alpha-Beta 剪枝（Alpha-Beta Pruning）

在 `AlphaBetaAgent` 类中实现一个 Alpha-Beta 剪枝算法来更有效地搜索极小化极大树。同样，**所实现的 Alpha-Beta 剪枝算法应当能够扩展到多个 min 层**。

所实现的 Alpha-Beta 剪枝算法应该能够进行加速（也许深度为 3 的 Alpha-Beta 剪枝搜索的运行速度会与深度为 2 的极小化极大搜索一样快）。理想情况下，smallClassic 棋盘上，深度为 3 的剪枝搜索每次行动只需几秒或更快，可通过运行：

```
python pacman.py -p AlphaBetaAgent -a depth=3 -l smallClassic
```

进行测试。

`AlphaBetaAgent` 的极小化极大值应该与 `MinimaxAgent` 的极小化极大值相同。

可以通过运行：

```
python autograder.py -q q3
```

来测试和调试你的代码实现。

任务 3：期望最大搜索（Expectimax）

极小化极大搜索和 Alpha-Beta 剪枝搜索都假设你正在与做出最佳决策的对手进行对抗，但情况并非总是如此。在这个任务中，需要在 `ExpectimaxAgent` 类中实现期望最大搜索算法，它对可能做出次优选择的对手的概率行为建模非常有用。

可以通过运行：

```
python autograder.py -q q4
```

在小型博弈树上来快速测试和调试你的代码实现。

为了观察期望最大搜索和 Alpha-Beta 剪枝搜索之间行为的不同，可以运行：

```
python pacman.py -p AlphaBetaAgent -l trappedClassic -a depth=3 -q -n 10
python pacman.py -p ExpectimaxAgent -l trappedClassic -a depth=3 -q -n 10
```

进行比较。**应该可以观察到 `ExpectimaxAgent` 在大约一半的情况下获胜，而 `AlphaBetaAgent` 总是会输掉游戏。**

任务 4：评价函数（Evaluation Function）

在函数 `betterEvaluationFunction` 中为吃豆人实现一个更好的评价函数。评价函数应该去是估计终止或截断节点状态的效用值。**使用深度为 2 的期望最大搜索，所实现的评价函数应该要在存在一个随机幽灵的 `smallClassic` 棋盘上有 50%的几率吃掉所有豆子，并且仍然保证合理的运行速度**（理想情况下，吃豆人获胜时的平均分应该在 1000 分左右）。

为了在上述条件下测试你的代码实现的得分，可以运行：

```
python autograder.py -q q5
```

各个任务的更详细描述可以参考：
https://inst.eecs.berkeley.edu/~cs188/sp23/projects/proj3/

作业报告

本次作业需要提交报告和代码。对于以上 4 个任务，报告需分别详细介绍代码的实现和实验结果。使用 QQ 群文件中的报告模版（课程作业\报告模版.doc）撰写实验报告。

作业提交

将作业报告存储 PDF 文件，用学号命名，例如 221900001.pdf，并与相应的源码打包为学号命名的.zip 文件，例如 221900001.zip。

上传到百度网盘：
https://pan.baidu.com/disk/main#/transfer/send?surl=ABAAAAAAABFP_Q

提交截止日期：10 月 12 日 23:59:59

学术诚信

允许同学之间的相互讨论，但是**署你名字的工作必须由你自己独立完成**。

如果发现作业之间高度相似将被判定为互相抄袭行为，**抄袭和被抄袭双方的成绩都将被取消**。

应项目开发者的要求，**严禁将作业答案发布在网上**。