

人工智能 LAB1 实验报告

一、实验目标

实现 BFS, A*算法, 完成吃豆人寻找食物的静态搜索。实现 minimax 算法和 alpha-beta 剪枝, 完成吃豆人与对手的动态博弈并减少搜索空间。

二、实验环境和工具

实验环境: Ubuntu 16.04 Python3.6.5

实验工具: PyCharm

三、实验内容和过程

(1) BFS

考虑到 BFS 即广度优先搜索, 区别于 DFS 深度优先搜索, BFS 需要维护一个存储形式, 使得状态先于子状态被遍历, 容易想到是一个队列, 故只需要把 DFS 的栈 Stack 改为队列 Queue 即可。代码如下:

```
def myBreadthFirstSearch(problem):
    visited = {}
    frontier = util.Queue()
    frontier.push((problem.getStartState(), None))

    while not frontier.isEmpty():
        state, prev_state = frontier.pop()

        if problem.isGoalState(state):
            solution = [state]
            while prev_state != None:
                solution.append(prev_state)
                prev_state = visited[prev_state]
            return solution[::-1]

        if state not in visited:
            visited[state] = prev_state

        for next_state, step_cost in problem.getChildren(state):
            frontier.push((next_state, state))

    return []
```

测试结果如下：

```
baelish@baelish-Lenovo-KiaoXin-Chao7000-14(KBR: ~/Desktop/LAB1
=====
*** Question q2
=====
*** PASS: test_cases/q2/graph_backtrack.test
*** solution:
*** expanded_states: ['A', 'B', 'C', 'D']
*** PASS: test_cases/q2/graph_bfs_vs_dfs.test
*** solution:
*** expanded_states: ['A', 'B']
*** PASS: test_cases/q2/graph_infinite.test
*** solution:
*** expanded_states: ['0:A->B', '1:B->C', '1:C->G']
*** PASS: test_cases/q2/graph_manypaths.test
*** solution:
*** expanded_states: ['A', 'B', 'C']
*** PASS: test_cases/q2/pacman_1.test
*** solution length: 48
*** nodes expanded: 269
*** pacman layout: mediumMaze

### Question q2: 4/4 ###

Finished at 17:28:14

Provisional grades
=====
Question q2: 4/4
Total: 4/4

Your grades are NOT yet registered. To register your grades, make sure
to follow your instructor's guidelines to receive credit on your project.

[SearchAgent] using function bfs
[SearchAgent] using problem type PositionSearchProblem
Path found with total cost of 68 in 0.0 seconds
Search nodes expanded: 269
Pacman emerges victorious! Score: 442
Average Score: 442.0
Scores: 442.0
Win Rate: 1/1 (1.00)
Records: Win
autograder.py:17: DeprecationWarning: the imp module is deprecated in favour of importlib; see the module's documentation for alternative uses
import imp
Starting on 5-10 at 17:28:15
```

(2) A*

A*算法需要依照最小的启发值选择搜索子状态，是一种高级的广度优先搜索，容易想到用优先队列存储状态空间，按照启发值选择搜索状态。代码如下：（其中 `f_state` 是开始状态到当前状态的实际值）

```
def myAStarSearch(problem, heuristic):
    visited = {}
    frontier = util.PriorityQueue()
    frontier.push((problem.getStartState(), None, 0),
    heuristic(problem.getStartState()))

    while not frontier.isEmpty():
        state, prev_state, f_state = frontier.pop()

        if problem.isGoalState(state):
            solution = [state]
            while prev_state != None:
                solution.append(prev_state)
                prev_state = visited[prev_state]
            return solution[::-1]

        if state not in visited:
            visited[state] = prev_state

            for next_state, step_cost in problem.getChildren(state):
                frontier.push((next_state, state, f_state+step_cost),
                f_state+step_cost+heuristic(next_state))

    return []
```

测试结果如下：

```
baelish@baelish-Lenovo-XiaoXin-Chao7000-14IKBR: ~/Desktop/LAB1
=====
Question q3
*** PASS: test_cases/q3/astar_0.test
*** solution: ['Right', 'Down', 'Down']
*** expanded states: ['A', 'B', 'D', 'C', 'G']
*** PASS: test_cases/q3/astar_1_graph_heuristic.test
*** solution: ['D', 'D', 'D']
*** expanded states: ['S', 'A', 'D', 'C']
*** PASS: test_cases/q3/astar_2_manhattan.test
*** solution length: 68
*** pacman layout:
medLunMaze
*** nodes expanded: 221
*** PASS: test_cases/q3/astar_3_goalAtDequeue.test
*** solution: ['1:A->B', '0:B->C', '0:C->G']
*** expanded states: ['A', 'B', 'C']
*** PASS: test_cases/q3/graph_backtrack.test
*** solution: ['1:A->C', '0:C->G']
*** expanded states: ['A', 'B', 'C', 'D']
*** PASS: test_cases/q3/graph_manypaths.test
*** solution: ['1:A->C', '0:C->D', '1:D->F', '0:F->G']
*** expanded states: ['A', 'B1', 'C', 'B2', 'D', 'E1', 'F', 'E2']
## Question q3: 4/4 ##
Finished at 17:28:15
Provisional grades
=====
Question q3: 4/4
Total: 4/4
Your grades are NOT yet registered. To register your grades, make sure
to follow your instructor's guidelines to receive credit on your project.
[SearchAgent] using function astar and heuristic manhattanHeuristic
[SearchAgent] using problem type PositionSearchProblem
Path found with total cost of 68 in 0.0 seconds
Search nodes expanded: 221
Pacman emerges victorious! Score: 442
Average Score: 442.0
Scores: 442.0
Win Rate: 1/1 (1.00)
Record: Win
autograder.py:17: DeprecationWarning: the inp module is deprecated in favour of importlib; see the module's documentation for alternative uses
import inp
Starting on 5-10 at 17:28:17
```

(3) Minimax

考虑预估步数为 `depth`，写一个递归函数 `minimax(state, depth)`，每次从对手转移执行权到吃豆人时，把 `depth-1` 送入下次递归，如果是吃豆人执行操作，就求相对于吃豆人的状态最大值，反之求相对于吃豆人的状态最小值，把对应状态作为本次操作的子状态。代码如下：

```
def minimax(self, state, depth):
    if state.isTerminated() or depth == 0:
        return None, state.evaluateScore()

    best_state, best_score = None, -float('inf') if state.isMe() else float('inf')

    for child in state.getChildren():
        if child.isMe():
            _, score = self.minimax(child, depth - 1)
        else:
            _, score = self.minimax(child, depth)

        if state.isMe() and score > best_score:
            best_score = score
            best_state = child
        elif not state.isMe() and score < best_score:
            best_score = score
            best_state = child

    return best_state, best_score
```

测试结果如下：

```
baelish@baelish-Lenovo-XiaoXin-Chao7000-14IKBR: ~/Desktop/LAB1
Question q2
=====
*** PASS: test_cases/q2/0_eval-function-lose-states-1.test
*** PASS: test_cases/q2/0_eval-function-lose-states-2.test
*** PASS: test_cases/q2/0_eval-function-win-states-1.test
*** PASS: test_cases/q2/0_eval-function-win-states-2.test
*** PASS: test_cases/q2/0_lecture-6-tree.test
*** PASS: test_cases/q2/0_small-tree.test
*** PASS: test_cases/q2/1-1-minimax.test
*** PASS: test_cases/q2/1-2-minimax.test
*** PASS: test_cases/q2/1-3-minimax.test
*** PASS: test_cases/q2/1-4-minimax.test
*** PASS: test_cases/q2/1-5-minimax.test
*** PASS: test_cases/q2/1-6-minimax.test
*** PASS: test_cases/q2/1-7-minimax.test
*** PASS: test_cases/q2/1-8-minimax.test
*** PASS: test_cases/q2/2-1a-vary-depth.test
*** PASS: test_cases/q2/2-1b-vary-depth.test
*** PASS: test_cases/q2/2-2a-vary-depth.test
*** PASS: test_cases/q2/2-2b-vary-depth.test
*** PASS: test_cases/q2/2-3a-vary-depth.test
*** PASS: test_cases/q2/2-3b-vary-depth.test
*** PASS: test_cases/q2/2-4a-vary-depth.test
*** PASS: test_cases/q2/2-4b-vary-depth.test
*** PASS: test_cases/q2/2-one-ghost-3level.test
*** PASS: test_cases/q2/3-one-ghost-4level.test
*** PASS: test_cases/q2/4-two-ghosts-3level.test
*** PASS: test_cases/q2/5-two-ghosts-4level.test
*** PASS: test_cases/q2/6-led-rope.test
*** PASS: test_cases/q2/7-1a-check-depth-one-ghost.test
*** PASS: test_cases/q2/7-1b-check-depth-one-ghost.test
*** PASS: test_cases/q2/7-1c-check-depth-one-ghost.test
*** PASS: test_cases/q2/7-2a-check-depth-two-ghosts.test
*** PASS: test_cases/q2/7-2b-check-depth-two-ghosts.test
*** PASS: test_cases/q2/7-2c-check-depth-two-ghosts.test
*** Running MinimaxAgent on smallClassic 1 time(s).
Pacman (left) Score: 84
Average Score: 84.0
Scores: 84.0
Win Rate: 0/1 (0.00)
Record: Loss
*** Finished running MinimaxAgent on smallClassic after 0 seconds.
*** Won 0 out of 1 games. Average score: 84.000000 ***
*** PASS: test_cases/q2/8-pacman-game.test

## Question q2: 5/5 ##

Finished at 17:28:18

Provisional grades
=====
Question q2: 5/5
-----
Total: 5/5

Your grades are NOT yet registered. To register your grades, make sure
to follow your instructor's guidelines to receive credit on your project.

autograder.py:17: DeprecationWarning: the inp module is deprecated in favour of importlib; see the module's documentation for alternative uses
import inp
```

(4) minimax 的 alpha-beta 剪枝

在 minimax 算法上做剪枝，记录对手执行不同操作时的最大状态值 **alpha**，如果在某次轮到吃豆人执行操作时，不管执行何种操作，获得状态值永远小于 **alpha**，那么这个状态下的搜索子空间的状态值不可能比 **alpha** 大，对最终结果没有影响，故可以剪枝。**beta** 同理，代码如下：

```
def alphabeta(self, state, depth, alpha, beta):
    if state.isTerminated() or depth == 0:
        return None, state.evaluateScore()

    best_state, best_score = None, -float('inf') if state.isMe() else float('inf')

    for child in state.getChildren():
        if child.isMe():
            _, score = self.alphabeta(child, depth - 1, alpha, beta)
        else:
            _, score = self.alphabeta(child, depth, alpha, beta)

        if state.isMe():
            if score > beta:
                return child, score
            if score > alpha:
                alpha = score
            if score > best_score:
```

```

        best_score = score
        best_state = child
    else:
        if score < alpha:
            return child, score
        if score < beta:
            beta = score
        if score < best_score:
            best_score = score
            best_state = child

    return best_state, best_score

def getNextState(self, state):
    best_state, _ = self.alphabeta(state, self.depth, -float('inf'), float('inf'))
    return best_state

```

测试结果如下：

```

baelish@baelish-Lenovo-XiaoXin-Chao7000-14IKBR: ~/Desktop/LAB1
Question q3
*****
*** PASS: test_cases/q3/0-eval-function-lose-states-1.test
*** PASS: test_cases/q3/0-eval-function-win-states-1.test
*** PASS: test_cases/q3/0-eval-function-win-states-2.test
*** PASS: test_cases/q3/0-lecture-6-tree.test
*** PASS: test_cases/q3/0-small-tree.test
*** PASS: test_cases/q3/1-1-minimax.test
*** PASS: test_cases/q3/1-2-minimax.test
*** PASS: test_cases/q3/1-3-minimax.test
*** PASS: test_cases/q3/1-4-minimax.test
*** PASS: test_cases/q3/1-5-minimax.test
*** PASS: test_cases/q3/1-6-minimax.test
*** PASS: test_cases/q3/1-7-minimax.test
*** PASS: test_cases/q3/1-8-minimax.test
*** PASS: test_cases/q3/2-1a-vary-depth.test
*** PASS: test_cases/q3/2-1b-vary-depth.test
*** PASS: test_cases/q3/2-2a-vary-depth.test
*** PASS: test_cases/q3/2-2b-vary-depth.test
*** PASS: test_cases/q3/2-3a-vary-depth.test
*** PASS: test_cases/q3/2-3b-vary-depth.test
*** PASS: test_cases/q3/2-4a-vary-depth.test
*** PASS: test_cases/q3/2-4b-vary-depth.test
*** PASS: test_cases/q3/2-one-ghost-1level.test
*** PASS: test_cases/q3/3-one-ghost-4level.test
*** PASS: test_cases/q3/4-two-ghosts-1level.test
*** PASS: test_cases/q3/5-two-ghosts-4level.test
*** PASS: test_cases/q3/6-tied-root.test
*** PASS: test_cases/q3/7-1a-check-depth-one-ghost.test
*** PASS: test_cases/q3/7-1b-check-depth-one-ghost.test
*** PASS: test_cases/q3/7-1c-check-depth-one-ghost.test
*** PASS: test_cases/q3/7-2a-check-depth-two-ghosts.test
*** PASS: test_cases/q3/7-2b-check-depth-two-ghosts.test
*** PASS: test_cases/q3/7-2c-check-depth-two-ghosts.test
*** Running AlphaBetaAgent on smallClassic 1 time(s).
Pacman died! Score: 84
Average Score: 84.0
Scores: 84.0
Min Rate: 0/1 (0.00)
Record: Loss
*** Finished running AlphaBetaAgent on smallClassic after 0 seconds.
*** Non 0 out of 1 games. Average score: 84.000000 ***
*** PASS: test_cases/q3/8-pacman-game.test

### Question q3: 5/5 ###

Finished at 17:28:19

Provisional grades
=====
Question q3: 5/5
Total: 5/5

Your grades are NOT yet registered. To register your grades, make sure
to follow your instructor's guidelines to receive credit on your project.

Pacman died! Score: 390
Average Score: 390.0

```