AI LAB2 实验报告

一、算法实现思路

1. BFS

和助教实现的DFS思路相同，不同之处是将存放待访问节点的数据结构改为从队列改成栈。具体实现逻辑：

1. 根节点入栈
2. 栈顶元素（第一步就是根节点）出栈，将它的children都入栈
3. 不断执行(2)，直到栈为空（本实验中在栈空之前一定能够到达终点，故用 $problem.isGoalState()$ 来判断终止条件）
4. 将出栈的元素按顺序输出即为solution

2. Astar

和BFS的思路相同，不同之处是将栈改为优先队列。优先队列的优先级别由 $newcost + heuristic(nextstate)$ 来决定。具体实现逻辑：

1. 根节点入优先队列
2. 优先队列优先级别最高者 S_n （第一步就是根节点）出队，将它的children S_{n+1} 入队，children的优先级别为 $cost(S_n, S_{n+1}) + heuristic(S_{n+1})$
3. 不断执行(2)，直到优先队列为空（本实验中在优先队列空之前一定能够到达终点，故用 $problem.isGoalState()$ 来判断终止条件）
4. 将出队元素按顺序输出即为solution

3. MinMax

书上的朴素版思路：

```
function MINIMAX-DECISION(state) returns an action
  return arg max_{a ∈ ACTIONS(s)} MIN-VALUE(RESULT(state, a))

function MAX-VALUE(state) returns a utility value
  if TERMINAL-TEST(state) then return UTILITY(state)
  v ← −∞
  for each a in ACTIONS(state) do
    v ← MAX(v, MIN-VALUE(RESULT(s, a)))
  return v

function MIN-VALUE(state) returns a utility value
  if TERMINAL-TEST(state) then return UTILITY(state)
  v ← ∞
  for each a in ACTIONS(state) do
    v ← MIN(v, MAX-VALUE(RESULT(s, a)))
  return v
```

图 5.3　极小极大值决策算法

在此基础上，本实验需要做2个改进：

1. 除了best_score，我们还需要输出best_state

实现思路：

在更新v(best_score)时一并更新best_state，也就是将child赋给best_state

2. 我们用depth来控制决策树的生成深度

实现思路：

根据实验文档，depth是对于每个agent而言的。由于第一个行动的是人，因此，状态从人到鬼depth不变，状态从鬼到人depth-1。终止条件为$state.isTerminated() == True$ 或者 depth == 0

4. AlphaBeta

书上的朴素版思路：

```
function ALPHA-BETA-SEARCH(state) returns an action
  v←MAX-VALUE(state, −∞, +∞)
  return the action in ACTIONS(state) with value v

function MAX-VALUE(state, α, β) returns a utility value
  if TERMINAL-TEST(state) then return UTILITY(state)
  v←−∞
  for each a in ACTIONS(state) do
    v←MAX(v, MIN-VALUE(RESULT(s,a), α, β))
    if v ≥ β then return v
    α←MAX(α, v)
  return v

function MIN-VALUE(state, α, β) returns a utility value
  if TERMINAL-TEST(state) then return UTILITY(state)
  v←+∞
  for each a in ACTIONS(state) do
    v←MIN(v, MAX-VALUE(RESULT(s,a), α, β))
    if v ≤ α then return v
    β←MIN(β, v)
  return v
```

图 5.7　α-β搜索算法

在此基础上，本实验需要做3个改进：

1. 除了best_score，我们还需要输出best_state

实现思路：

在更新v(best_score)时一并更新best_state，也就是将child赋给best_state

2. 我们用depth来控制决策树的生成深度

实现思路：

根据实验文档，depth是对于每个agent而言的。由于第一个行动的是人，因此，状态从人到鬼depth不变，状态从鬼到人depth-1。终止条件为$state.isTerminated() == True$ 或者 depth == 0

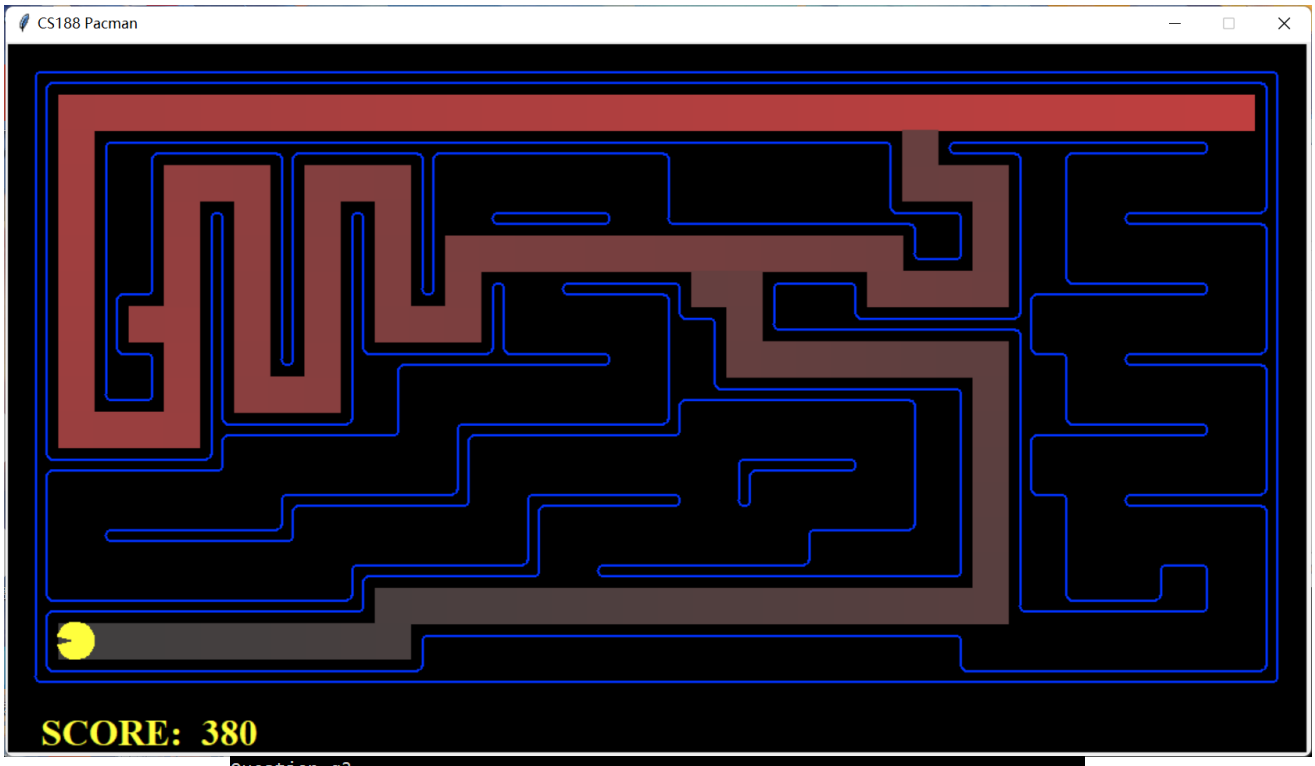
3. 剪枝条件的更改

实现思路：

当按照原条件，也就是 $v >= \beta$ 时进行剪枝，会发现测试样例 6-tied-root 无法通过。分析得出应该将剪枝条件修改为$v > \beta$。 同理将 $v <= \alpha$ 改为 $v < \alpha$

二、算法测试截图

1. BFS



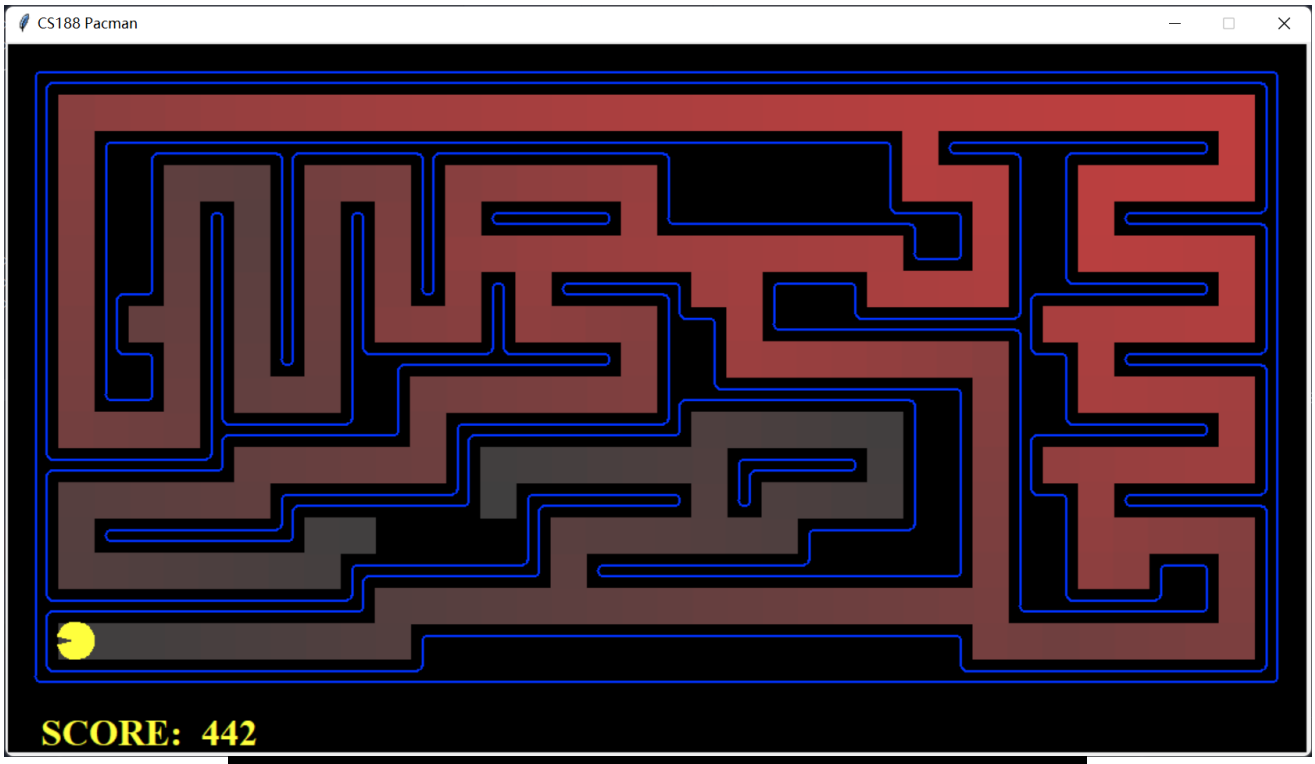

```
Question q2
===========
*** PASS: test_cases\q2\graph_backtrack.test
***     solution:               ['1:A->C', '0:C->G']
***     expanded_states:        ['A', 'B', 'C', 'D']
*** PASS: test_cases\q2\graph_bfs_vs_dfs.test
***     solution:               ['1:A->G']
***     expanded_states:        ['A', 'B']
*** PASS: test_cases\q2\graph_infinite.test
***     solution:               ['0:A->B', '1:B->C', '1:C->G']
***     expanded_states:        ['A', 'B', 'C']
*** PASS: test_cases\q2\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']
*** PASS: test_cases\q2\pacman_1.test
***     pacman layout:          mediumMaze
***     solution length: 68
***     nodes expanded:         269

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


Finished at 12:29:27

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

2. Astar



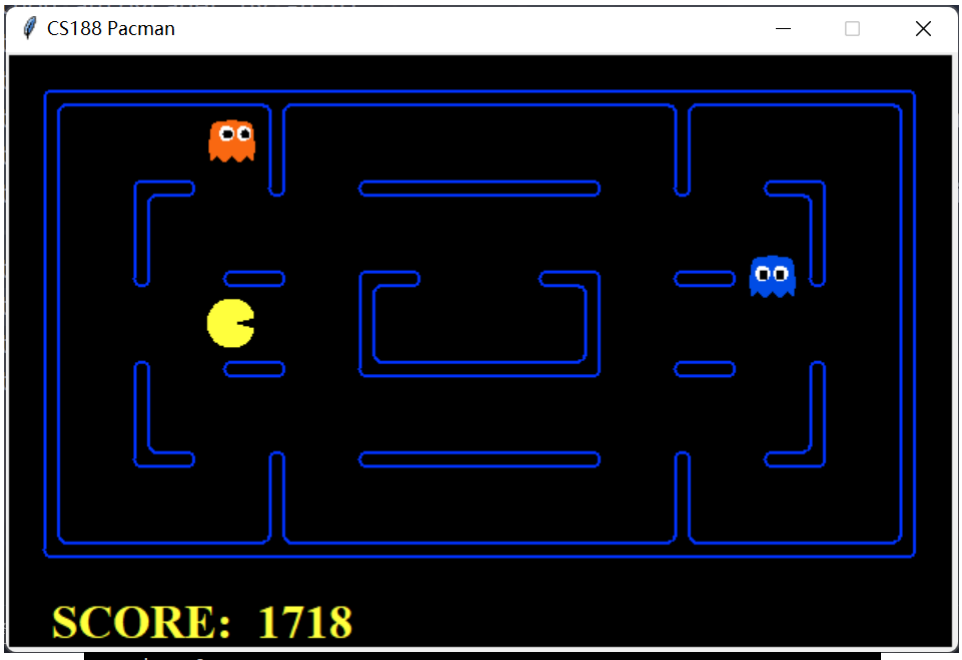

```
Question q3
===========
*** PASS: test_cases\q3\astar_0.test
***     solution:               ['Right', 'Down', 'Down']
***     expanded_states:        ['A', 'B', 'D', 'C', 'G']
*** PASS: test_cases\q3\astar_1_graph_heuristic.test
***     solution:               ['0', '0', '2']
***     expanded_states:        ['S', 'A', 'D', 'C']
*** PASS: test_cases\q3\astar_2_manhattan.test
***     pacman layout:          mediumMaze
***     solution length: 68
***     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 12:29:30

Provisional grades
==================
Question q3: 4/4
------------------
Total: 4/4
```

3. MinMax




```
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-minmax.test
*** PASS: test_cases\q2\1-2-minmax.test
*** PASS: test_cases\q2\1-3-minmax.test
*** PASS: test_cases\q2\1-4-minmax.test
*** PASS: test_cases\q2\1-5-minmax.test
*** PASS: test_cases\q2\1-6-minmax.test
*** PASS: test_cases\q2\1-7-minmax.test
*** PASS: test_cases\q2\1-8-minmax.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-tied-root.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 died! 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 12:29:34

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

4. AlphaBeta


```
Question q3
===========

*** PASS: test_cases\q3\0-eval-function-lose-states-1.test
*** PASS: test_cases\q3\0-eval-function-lose-states-2.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-minmax.test
*** PASS: test_cases\q3\1-2-minmax.test
*** PASS: test_cases\q3\1-3-minmax.test
*** PASS: test_cases\q3\1-4-minmax.test
*** PASS: test_cases\q3\1-5-minmax.test
*** PASS: test_cases\q3\1-6-minmax.test
*** PASS: test_cases\q3\1-7-minmax.test
*** PASS: test_cases\q3\1-8-minmax.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-3level.test
*** PASS: test_cases\q3\3-one-ghost-4level.test
*** PASS: test_cases\q3\4-two-ghosts-3level.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
Win Rate:      0/1 (0.00)
Record:        Loss
*** Finished running AlphaBetaAgent on smallClassic after 0 seconds.
*** Won 0 out of 1 games. Average score: 84.000000 ***
*** PASS: test_cases\q3\8-pacman-game.test

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

Somthing want to say

感谢两位助教！

无论是作业问题（我记得那个bp算法找过王佳禾助教两三次），还是考前复习（有几处疑问得到了梁聪助教的立刻回复），抑或是presentation的修改建议和评语（梁聪助教的建议帮助很大），还是本次实验（两位助教都打扰了），助教们的回复都很积极及时而且很有东西（指能够简洁有效地解决我的问题）。

谢谢两位助教！祝两位助教paper ++！这门课程的助教体验非常好！