MANNY

A multi-agent system for playing Pac-Man

Kubilay Kahveci • Mert Gencturk

1. Introduction

TBC

2. Related Work

TBC

3. Problem Definition and Algorithm

TBC

3.1. Task Definition

TBC

3.2. Algorithm Definition

- In order to solve fundamental navigation problem of the agent in the Pac-Man world, we have initially implemented core search algorithms (DFS, BFS, Uniform Cost Search, A* Search).

4. Experimental Evaluation

Initial experiments for search algorithms have been conducted on simple maze with a single goal square, one agent and no opponents. The aim is to analyze the agent behavior in the Pac-Man world.

4.1. Methodology

TBC

4.2. Results

TBC

4.3. Discussion

- Figure 1 shows the states explored, and the order in which they were visited for BFS and DFS navigation. Brighter red means earlier exploration. In both settings, there is a single goal state which is very deep in the state space. When using BF approach, the agent explores almost all squares before reaching the goal state. However, when it uses DF approach, since it can go deeper towards the goal state quickly, agent finds the goal state without exploring many

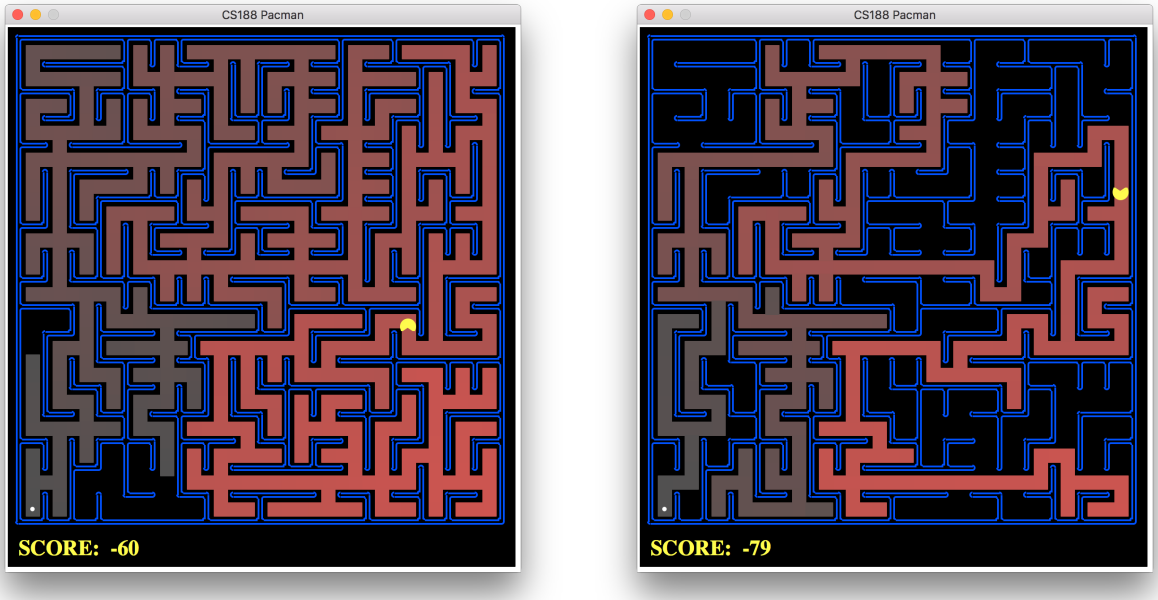

Figure 1: BFS vs. DFS

squares in the maze. This shows the fundamental difference between those two approaches given the solution is in deeper nodes.

- Since the action cost is constant in these setups, Uniform Cost Search behaves exactly like BFS. At each turn, the agent explores the node with the lowest cost, which happens to be the one in breadth first order. Figure 2 shows how A* can be an improvement on UCS even with a very naive heuristic function such as Manhattan Distance. Although agents starts moving very

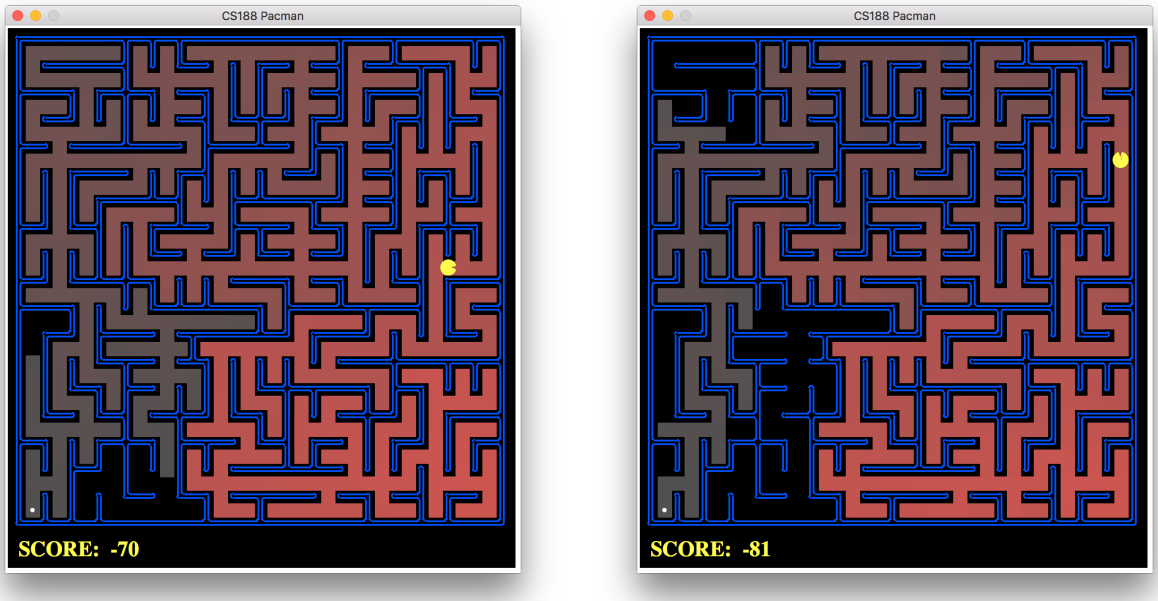

Figure 2: Uniform Cost vs. A*

similarly at the beginning (this is due to selected heuristic function), after some point A* does a good job of preventing non-promising paths.

5. Conclusion

TBC

6. Future Work

TBC

Bibliography

TBC