Project 2 – Pacman Path-based Location Search

CSC 412 –Intelligent Systems

California Baptist University

Instructor: Dr. Dan Grissom

Due: 02/27/2022

Johana Chazaro Cortes

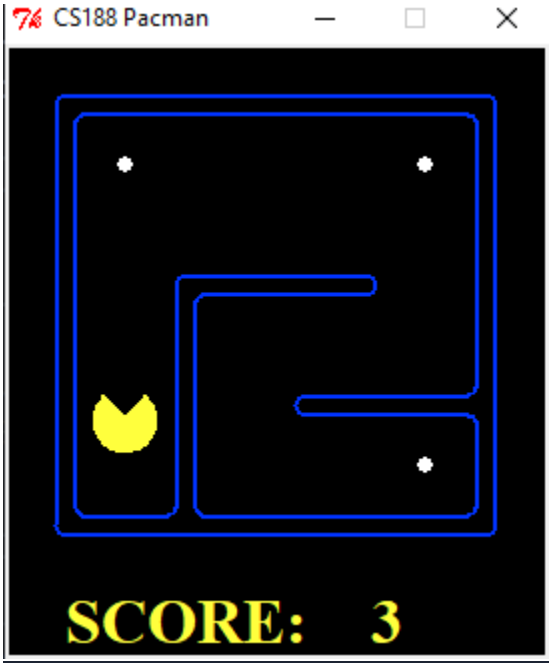
Roberto Rodriguez

Individual Results

Question 1 – Corners Problem – tinyCorners

Command: python2 pacman.py -l tinyCorners -p SearchAgent -a fn=bfs,prob=CornersProblem

Screenshot




```
PS C:\Users\roberto\Documents\vs code projects\CSC 412 Intellegence Systems\searchPacman\PacmanSearch1> python2 pacman.py -l tinyCorners -p SearchAgent -a fn=bfs
,prob=CornersProblem
[SearchAgent] using function bfs
[SearchAgent] using problem type CornersProblem
Path found with total cost of 28 in 0.0 seconds
Search nodes expanded: 252
Pacman emerges victorious! Score: 512
Average Score: 512.0
Scores:        512.0
Win Rate:      1/1 (1.00)
Record:        Win
```

Numbers

Cost	Execution Time (sec)	# Nodes Expanded	Pacman Score
28	0.0	252	512.0

Question 1 – Corners Problem – mediumCorners

Command: python2 pacman.py -l mediumCorners -p SearchAgent -a fn=bfs,prob=CornersProblem

Screenshot

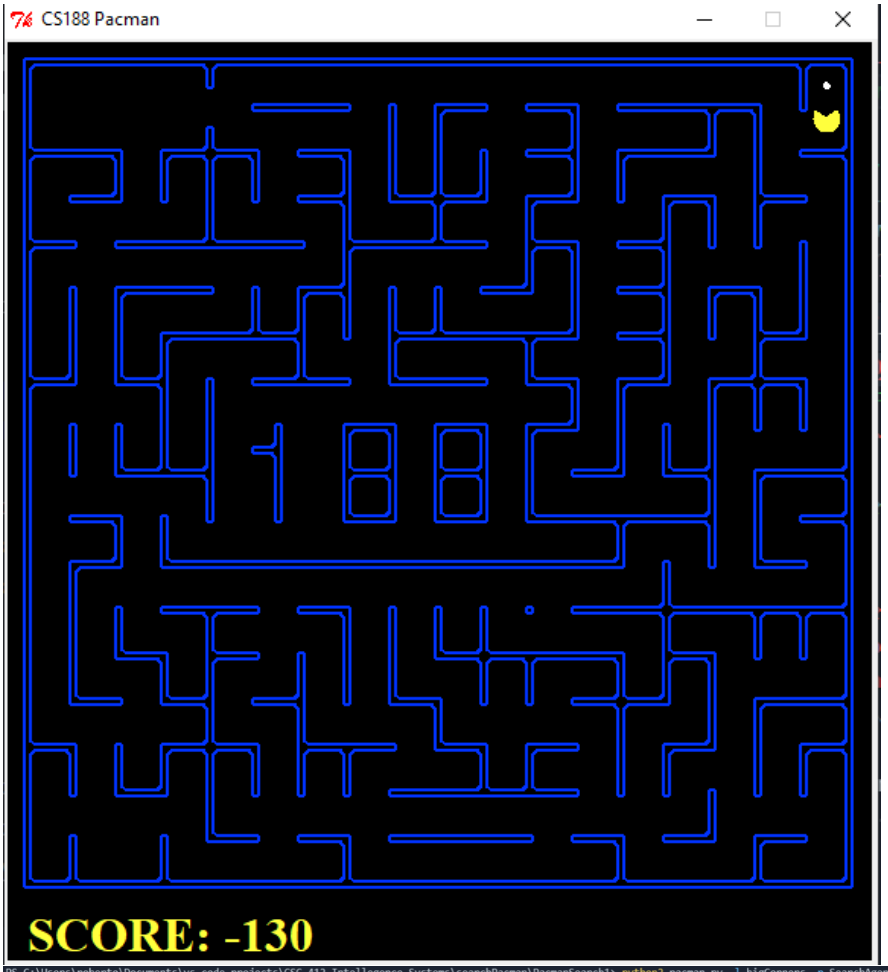
Numbers

Cost	Execution Time (sec)	# Nodes Expanded	Pacman Score
106	0.0	1966	434

Question 1 – Corners Problem – bigCorners

Command: python2 pacman.py -l bigCorners -p SearchAgent -a fn=bfs,prob=CornersProblem -z 0.5

Screenshot



Numbers

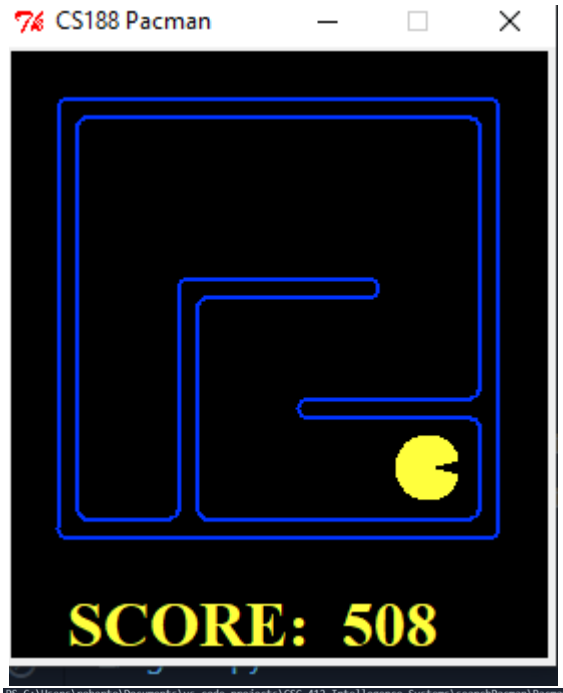
Cost	Execution Time (sec)	# Nodes Expanded	Pacman Score
162	0.1	7949	378.0

Question 2 – CornersProblem: A* Heuristic – tinyCorners

Command: python2 pacman.py -l tinyCorners -p SearchAgent -a

fn=aStarSearch,prob=CornersProblem,heuristic=cornersHeuristic

Screenshot




```
PS C:\Users\roberto\Documents\vs code projects\CSC 412 Intellegence Systems\searchPacman\PacmanSearch1> python2 pacman.py -l tinyCorners -p SearchAgent -a fn=aSt
arSearch,prob=CornersProblem,heuristic=cornersHeuristic
[SearchAgent] using function aStarSearch and heuristic cornersHeuristic
[SearchAgent] using problem type CornersProblem
Path found with total cost of 32 in 0.0 seconds
Search nodes expanded: 193
Pacman emerges victorious! Score: 508
Average Score: 508.0
Scores:        508.0
Win Rate:      1/1 (1.00)
Record:        Win
```

Numbers

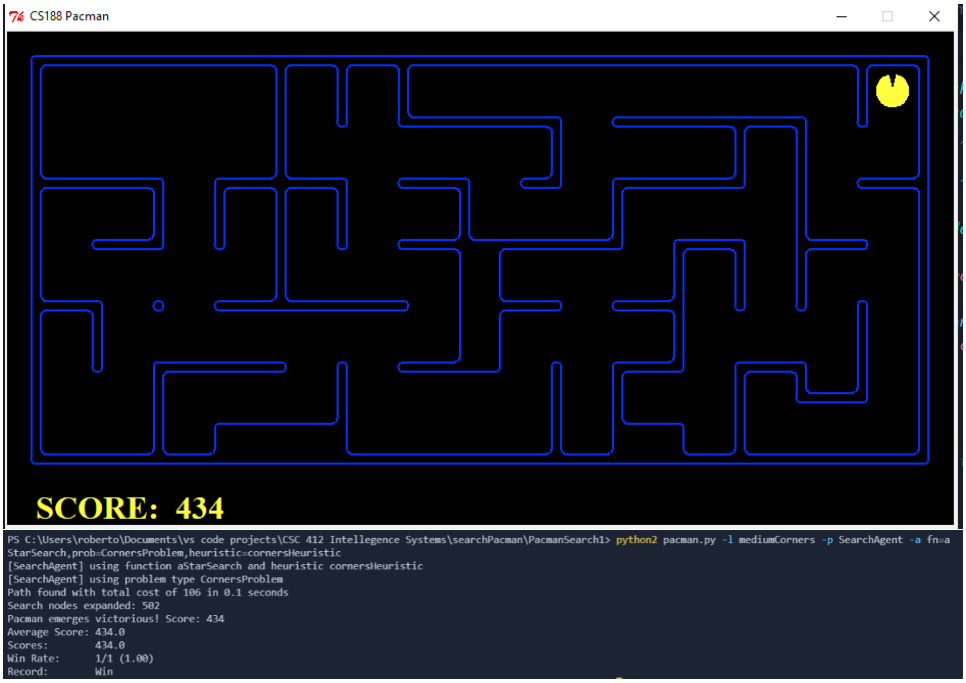
Cost	Execution Time (sec)	# Nodes Expanded	Pacman Score
32	0.0	193	508

Question 2 – CornersProblem: A* Heuristic – mediumCorners

Command: python2 pacman.py -l mediumCorners -p SearchAgent -a

fn=aStarSearch,prob=CornersProblem,heuristic=cornersHeuristic

Screenshot



Numbers

Cost	Execution Time (sec)	# Nodes Expanded	Pacman Score
106	0.1	502	434

Question 2 – CornersProblem: A* Heuristic – bigCorners

Command: python2 pacman.py -l bigCorners -p SearchAgent -a

fn=aStarSearch,prob=CornersProblem,heuristic=cornersHeuristic -z 0.5

Screenshot

Numbers

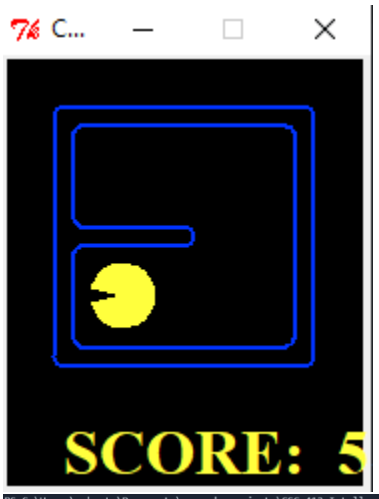
Cost	Execution Time (sec)	# Nodes Expanded	Pacman Score
162	0.2	1211	378.0

Question 3 – Eating All The Dots: A* Heauristic – testSearch

Command: python2 pacman.py -l testSearch -p SearchAgent -a

fn=aStarSearch,prob=FoodSearchProblem,heuristic=foodHeuristic

Screenshot




```
PS C:\Users\roberto\Documents\vs code projects\CSC 412 Intellegence Systems\searchPacman\PacmanSearch1> python2 pacman.py -l testSearch -p SearchAgent -a fn=aSta
rSearch,prob=FoodSearchProblem,heuristic=foodHeuristic
[SearchAgent] using function aStarSearch and heuristic foodHeuristic
[SearchAgent] using problem type FoodSearchProblem
Path found with total cost of 7 in 0.0 seconds
Search nodes expanded: 8
Pacman emerges victorious! Score: 513
Average Score: 513.0
Scores:        513.0
Win Rate:      1/1 (1.00)
Record:        Win
```

Numbers

Cost	Execution Time (sec)	# Nodes Expanded	Pacman Score
7	0.0	8	513.0

Question 3 – Eating All The Dots: A* Heauristic – trickySearch

Command: python2 pacman.py -l trickySearch -p SearchAgent –a
fn=aStarSearch,prob=FoodSearchProblem,heuristic=foodHeuristic

Screenshot


```
PS C:\Users\roberto\Documents\vs code projects\CSC 412 Intellegence Systems\searchPacman\PacmanSearch1> python2 pacman.py -l trickySearch -p SearchAgent -a fn=aS
tarSearch,prob=FoodSearchProblem,heuristic=foodHeuristic
[SearchAgent] using function aStarSearch and heuristic foodHeuristic
[SearchAgent] using problem type FoodSearchProblem
Path found with total cost of 60 in 2.1 seconds
Search nodes expanded: 5423
Pacman emerges victorious! Score: 570
Average Score: 570.0
Scores:        570.0
Win Rate:      1/1 (1.00)
Record:        Win
```

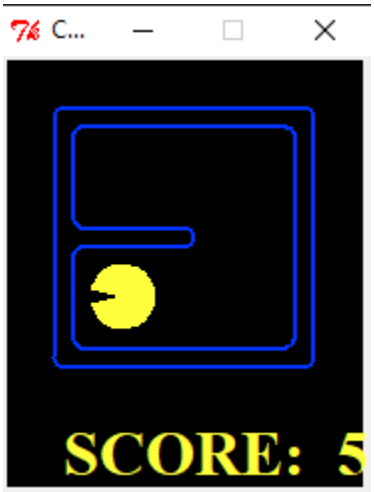
Numbers

Cost	Execution Time (sec)	# Nodes Expanded	Pacman Score
60	2.1	5423	570.0

Question 4 – ClosestDotSearchAgent – testSearch

Command: python2 pacman.py -l testSearch -p ClosestDotSearchAgent

Screenshot




```
PS C:\Users\roberto\Documents\vs code projects\CSC 412 Intellegence Systems\searchPacman\PacmanSearch1> python2 pacman.py -l testSearch -p ClosestDotSearchAgent
[SearchAgent] using function depthFirstSearch
[SearchAgent] using problem type PositionSearchProblem
Path found with cost 7.
Pacman emerges victorious! Score: 513
Average Score: 513.0
Scores:        513.0
Win Rate:      1/1 (1.00)
Record:        Win
```

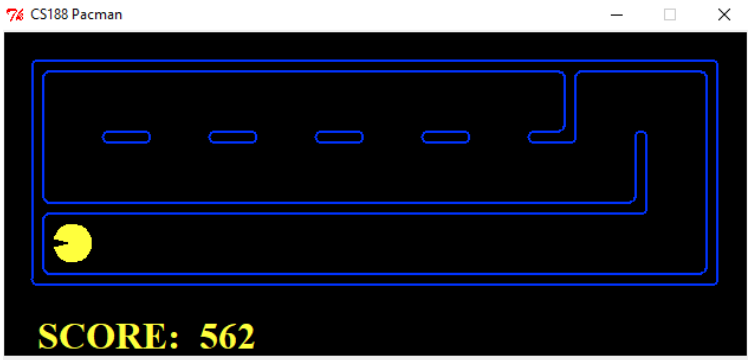
Numbers

Cost	Execution Time (sec)	# Nodes Expanded	Pacman Score
7			513.0

Question 4 – ClosestDotSearchAgent – trickySearch

Command: python2 pacman.py -l trickySearch -p ClosestDotSearchAgent

Screenshot




```
[SearchAgent] using function depthFirstSearch
[SearchAgent] using problem type PositionSearchProblem
Path found with cost 68.
Pacman emerges victorious! Score: 562
Average Score: 562.0
Scores:        562.0
Win Rate:      1/1 (1.00)
Record:        Win
```

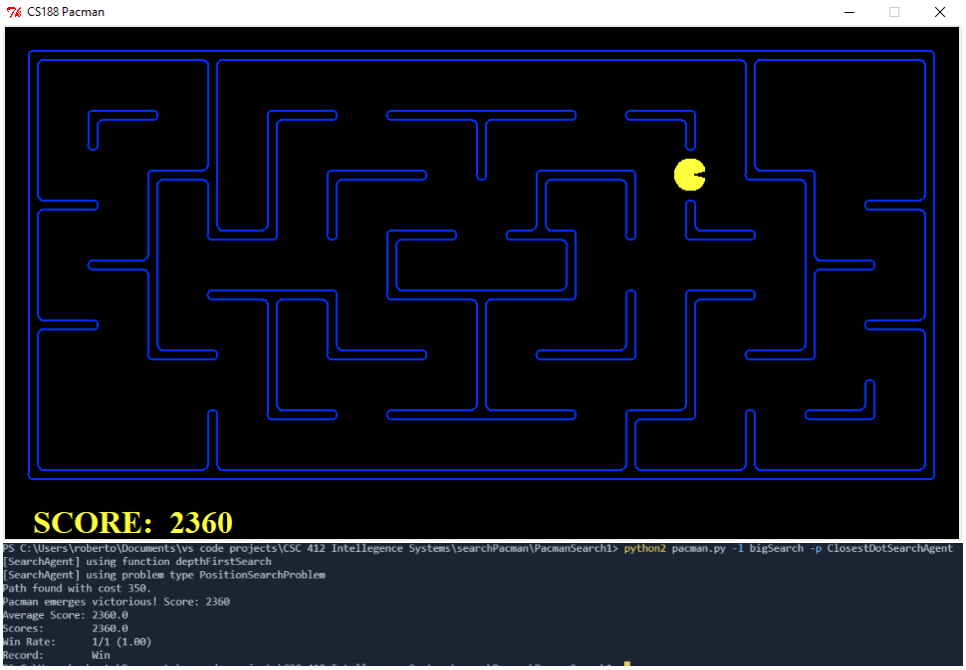
Numbers

Cost	Execution Time (sec)	# Nodes Expanded	Pacman Score
68			562

Question 4 – ClosestDotSearchAgent – bigSearch

Command: python2 pacman.py -l bigSearch -p ClosestDotSearchAgent

Screenshot



Numbers

Cost	Execution Time (sec)	# Nodes Expanded	Pacman Score
350			2360.0

Summary

Summary Chart

The following chart summarizes the individual results provided in the previous sections 1 & 2:

Questions 1 - 2

Alg.	Maze	Cost (Solution Quality)	Execution Time (s)	# Nodes (Time Complexity)
BFS Corners	Tiny	28	0.0	252
	Med	106	0.0	1966
	Big	162	0.1	7949
A* Corners	Tiny	32	0.0	193
	Med	106	0.1	502
	Big	162	0.2	1211

The following chart summarizes the individual results provided in the previous sections 3 & 4

Alg.	Maze	Cost (Solution Quality)	Execution Time (s)	# Nodes (Time Complexity)
A* All Food	Test	7	0.0	8
	Tricky	60	2.1	5423
Greedy All Food	Tiny	7	-	-
	Med	68	-	-
	Big	350	-	-

Summary Explanation

Our approach to identifying if a corner has been reached began by assuming the use of Boolean values and identifying the successors where there is no wall. The representation of each state changed by implementing a tuple which contained true false values to determine whether a corner has been visited by the pacman. This helped decide when the pacman has reached the goal state. Once visited, the tuple updated the state.

In order to achieve the goal state, we learned how to calculate the Manhattan distance of the closest food. This calculation was executed until the goal state was achieved in which used to determine the heuristic. We realize that with the optimal heuristic in A*, the pacman will look for the most optimal path to take even if that means avoiding the closest foods to grab ones further back as seen in trickysearch. While the greedy algorithm will only look at which food is closest. In bigsearch this is most exemplified when the pacman avoids a food dot in favor of closer ones and is required to do massive backtracking to get the avoided food.

Repo: https://github.com/joChazaro/PacmanSearch1