HW-1: Uninformed and Informed Search

Introduction To Artificial Intelligence

April 20, 2018
Spring Term

Prepared for

Oregon State University

Professor Rebecca Hutchinson

Prepared by

Jeremy Fischer

Yipeng Song

CONTENTS

1 METHODOLOGY

In this assignment, we use Python language to find the solution to wolves and chicks puzzle by using the uninformed and informed searching algorithms. For uninformed search, we implement Breadth First Search (BFS), Depth First Search (DFS), and Iterative Deepening Depth First Search (IDDFS). For informed search, we implement A-Star Search (A*).

We tested the four searching algorithms by:

- Print out the *action* and *result* pairs generated from a node sent to the expand function and verify that they are the correct next node for the fringe.
- Verify the number of wolves is less than the number of chickens on a given ban
- Test the solution on the website application shared on the assignment page.

Test Cases

Starting Left Bank	Starting Right Bank	Goal Left Bank	Goal Right Bank
0,0,0	3,3,1	3,3,1	0,0,0
0,0,0	9,8,1	9,8,1	0,0,0
0,0,0	100,95,1	100,95,1	0,0,0

1.1 Breadth First Search

The BFS searching algorithm will expand every single node in the tree level by level until we reach the goal state. Starting at the root, after visiting one node, this algorithm will append all of its neighbor nodes to the end of the list and will not continue to the next level of the tree until every neighbor node has been visited.To implement this, we can use a First In First Out (FIFO) queue. Thus, the first node in the queue will be expanded first.

1.2 Depth First Search

The DFS searching function is implemented very similarly to the BFS, but instead of adding the neighbor nodes to the end of the queue, they are added to the front. Therefore, this algorithm will go as far as possible along each branch before backtracking. To implement this, we can use a Last In First Out (LIFO) queue. Thus, it will continue to explore one path until a terminal.

1.3 Iterative Deepening Depth First Search

The IDDFS will start at the root node and repeatedly applies depth first searching with increasing depth limits until the goal is found. We set the maximum depth to 1500. Meaning, no solution is found once the current depth hits the maximum depth limit. This limit is set to ensure the program will not run too long before it ends up with a final decision.

1.4 A* Search

For the A* Search, we first made a heuristic function, $H(n) = 2*(n-2) + 1 = 2n - 3$, where n is the number of animals on the starting bank. We came to this heuristic function by the following thought process: if we abstract away the restriction of the number of wolves on the bank must be less than or equal to the number of chickens, then the fastest we could move all animals from the starting bank to the other bank is by moving the maximum amount of animals the boat can hold from the starting bank to the other bank, bringing one back, and repeating. This led us to the heuristic function of *H(n) = 2*(n-2) + 1 = 2n - 3* where *n* is the number of animals on the starting bank. We tested our function by commenting out the line of code that enforces the restriction where the number of wolves must be less than or equal to the number of chickens, and verified that the *nodes in solution* was equal to what our function predicted. The fringe is a minimum priority queue. So, the node that has the shortest path cost will be expanded next.

2 Results

	BFS	**DFS**	**IDDFS**	**A-Star**
Test 1	Solution: 11 Expanded: 14	Solution: 11 Expanded: 11	Solution: 11 Expanded: 84	Solution: 11 Expanded: 29
Test 2	Solution: 31 Expanded: 54	Solution: 31 Expanded: 45	Solution: 31 Expanded: 891	Solution: 31 Expanded: 50
Test 3	Solution: 387 Expanded: 1344	Solution: 953 Expanded: 1230	Solution: 951 Expanded: 713088	Solution: 387 Expanded: 1281

3 Discussion

From the table above, we notice that all four searching algorithms give the same solution path number for the first two tests. In addition, BFS, DFS, and A-Star out performed IDDFS as the nodes expanded are a lot less than the ones in IDDFS. IDDFS took a very long time on the larger data sets (both test 2 and test 3). Even though it found the best solution it expanded 713088 nodes in test 3, which is much more than any other algorithm.

We were surprised by how well A-Star search worked. It was fast and efficient. It is also interesting to think that A-Star search could have performed worse or better depending on the heuristic used, where the other algorithms don't need that sort of ”thinking”.

4 Conclusion

Among the 4 algorithms, theoretically the A-star search will perform the best, if given a perfect heuristic function. However, there is no such algorithm that can solve everything. BFS is fast and is guaranteed to find a solution if there is one, but it takes more memory. DFS saves memory but will not always give the optimal answer. IDDFS will provide an optimal path but is much slower. In conclusion, every searching algorithm has its own pros and cons, but we decided a A-Star was the best to use.