Brandon Lee
CS 331 – Dr. Rebecca Hutchinson
Intro to Artificial Intelligence
15 April 2016

Assignment Report 1

Methodology

For each of the three test cases, I ran BFS, DFS, IDDFS, and A* to test the traits of every search algorithm under various intensity levels of Cannibals and Missionaries. Because both the uninformed and informed algorithms were implemented in accordance with the design of GraphSearch and Expand functions from lecture, the algorithms all share numerous commonalities with each other.

Method Description

Firstly, for the implementation of the actual parameters in each search function – I included the fringe, the given initial state, and the final goal state. The fringe for uninformed searches is implemented with a deque object from the Python's *collections* library as the *deque.popleft()* and *deque.pop()* functions allow for easy access to the data structure from either the left or right sides. As for A*, I use a priority queue implemented with Python's *heapq* to push and pop the state and state cost. The initial/goal state parameters are implemented through Node objects which contain necessary attributes to iterate through the search and expand functions including: the state of the left and right banks, parent state, action, state depth, and cost. A closed list implemented with a hash table is used to store states that have already been checked. Solutions are stored within a result object capturing the state of the puzzle along with the action.

Algorithm Parameters

For DFS and IDDFS, I chose to set a depth limit of 400 as a problem past this size would be fairly time consuming to compute.

For A*, I am using a heuristic based on if the restriction that the game ends when there exist more cannibals than missionaries on one bank is removed. With this restriction lifted, it is observable that the heuristic would be admissible in the following situation: for each trip from one bank to another, the boat would need at least one individual to move it back. Thus because each trip at most can only bring one person to the ending bank due to a boat size of two, an admissible heuristic would be:

$$h(n) = \ Individuals\ on\ Starting\ Bank\ /\ Boat\ Size$$

where in our situation:

$$h(n) = n\ /\ 2$$

Results

	Test 1	Test 2	Test 3
Breath First Search	(11, 14)	(23, 42)	(333, 1620)
Depth First Search	(11, 11)	(23, 33)	(401, 76552)
Iterative Deepening Depth First Search	(11, 106)	(23, 664)	(333, 9900958)
A*	(11, 14)	(23, 41)	(333, 1619)

*** Key = (Solution Path Length, Total Nodes Expanded)

Discussion

When approaching this assignment after the past lectures, I believed that with a strong heuristic, A* would be the best search algorithm due to having an existing knowledge of non-goal states. Closely followed by this are the uninformed searches: BFS, IDDFS, and DFS from most efficient to least. My estimations for efficiency come from variables such as completeness, optimality, and time/space complexity. The results obtained from the tests above were somewhat similar to what I would have expected.

Firstly, I was surprised with BFS and how well it performed under the larger test cases. With an exponential time and space complexity of $O(b^{d+1})$, I assumed it would've performed much worse. However, due to the completeness and optimality that BFS brings, the results were satisfactory in addition to being time efficient.

The incompleteness and lack of optimality that DFS has is evident in the results above. If we take a look at the trends among the various searches, it is observable that for almost every case, each algorithm was able to arrive at the same solution path as the next. However, one noticeable anomaly resides in Test 3's depth first search, which returns a solution path inconsistent to the values obtained by the other three algorithms. After testing this issue a couple more iterations with similar results, my hypothesis for this behavior is supported by the lack of optimality and completeness in DFS.

For IDDFS, the results were similar to the other searches, as I expected because of the algorithm's completeness and optimality. However, the time and space complexity was a bit of a toss up for me. Because IDDFS is essentially DFS being iteratively performed through each step, the exponential time complexity of $O(b^d)$ was a bit long for me, spanning multiple minutes – which brings me back to my surprise with BFS and how much faster it was even with the same time complexity.

And finally A* search, with all the advantages of being informed and incorporating knowledge of non-goal states, was one of the most efficient algorithms tested. Given an appropriate heuristic, the search results were as expected – quick, optimal, and complete.

Conclusion

At the end of the day, we can conclude that A* remains the best search algorithm of this set when given an admissible, consistent heuristic. The other uninformed algorithms performed a bit below the efficiency of A*. While both BFS and IDDFS obtained the same optimal path, DFS's lack of completeness/optimality resulted in a different solution. In addition, IDDFS was by far the slowest – taking several minutes to compute. All in all, the results obtained from these set of tests were fairly representative of what I expected, other than how much more efficient BFS was compared to the other uninformed algorithms.