Erin Tanaka
Madison Dhanens
CS 331 Assignment #1 Report

Methodology:

With the three provided test cases, we ran four different search algorithms in order to find the solution. The breadth first search algorithm searches across the graph first. This function does not require any additional parameters to the start state and the goal state. In contrast, the depth first algorithm first explores down to the terminal nodes of the graph. This search method does not require additional parameters to the start and goal state. The third search algorithm was the iterative deepening depth first search. This algorithm has one function that does a limited depth search, stopping at a predetermined depth that was given as a parameter.

The third function calls this depth limited search, incrementing the limit by one each time the search returns without finding a goal state, rebuilding the graph each time. This function has no additional parameters other than the start and the goal state.

The final algorithm type is the A* search method. This method takes the cost of each path into account. It uses a heuristic function to calculate the weight of each path. The heuristic function calculates the difference of the number of animals on the right bank of the current state and the number of animals on the right bank of the goal state. The absolute value of this number became the weight for the current state. We chose this heuristic function because it may not find the overall optimal path, but it will make a sufficient decision for the immediate goal. It will pick the node that gets it immediately closest to the goal of all the chickens and all the wolves on one side.

The first test case had three chickens and three wolves. The second had nine chickens and 8 wolves. The final test case had 100 chickens and 95 wolves. The different sizes of the tests shows the efficiency of these functions when they run at varying capacities.

Results:

Number of nodes expanded:

Search method	Test File 1	Test File 2	Test File 3
BFS	13	52	1341
DFS	11	45	1230
IDDFS	105	1758	1391460
A*	12	45	761

Number of nodes on the solution path:

Search method	Test File 1	Test File 2	Test File 3
BFS	12	32	388

DFS	12	32	390
IDDFS	12	32	390
A*	12	32	388

Discussion:

We expected the breadth first and depth first search algorithms to be more efficient when the game ran with a small number of chickens and wolves. This is illustrated in the chart above, with breadth expanding 13 nodes and depth expanding 11. The iterative deepening depth first search function was not optimal at this size, expanding more than 9 times as many nodes, coming in at 105. We expected this because IDDFS trades off its time efficiency for space efficiency.

As the test cases got larger, the DFS and BFS functions began to expand more and more nodes, but they were still more efficient than IDDFS. It expanded way more nodes than any other algorithm.

A* performs well with both a small and a large number of animals. This is due to the fact that it is an informed search, where the other three are uniformed. It was able to look at the weight of the path calculated by the heuristic function and make an immediately optimal decision.

The number of nodes on the solution path were either the same or similar for all the different algorithms.

Conclusion:

The running the uninformed and informed search on the wolves and chickens problem gave us insight on how the different algorithms performed based off of different sized inputs. Overall A * search expanded the fewest nodes with larger inputs. Our various test cases further enforced that with smaller inputs the algorithms performed similarly and with large inputs the differences in time and space complexities were noticeably better or worse.