GROUP: LUANSONGJIAN, Raja Petroff
ONID: luans@oregonstate.edu , petroffr@oregonstate.edu
CLASS: CS331
DATE: 2017-04-19

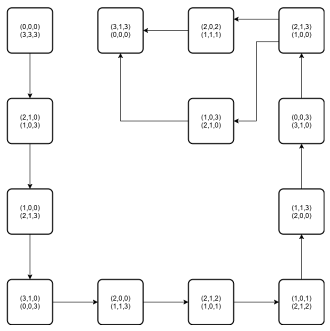
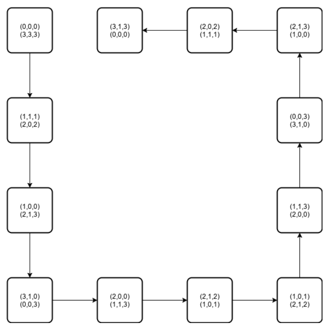
Methodology

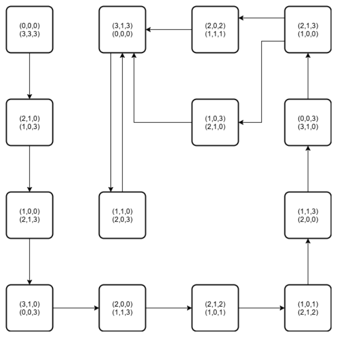
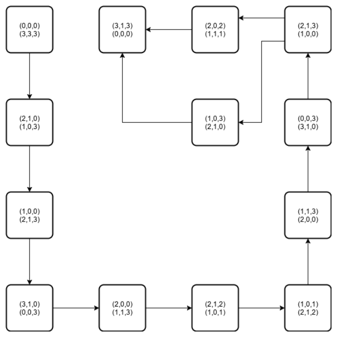
BFS: Nodes are expanded at a given depth before moving on to the next level. Nodes are stored in a queue.
DFS: Nodes are expanded along a path until a terminal node is reached. Nodes are stored in a stack. No depth limit.
IDDFS: Similar to DFS, but limit is incremented until a goal is found. Depth is incremented one at a time.
ASTAR: Uses an admissible heuristic which means that it will never overestimate the cost to reach the goal. Our heuristic is the difference of the cannibals and difference of the missionaries added together calculated as the cost.

Result

These graphs below are four kinds of search for the first test case. Because other two test cases will generate a few lines of nodes, we don't draw those graphs. Following those results written into files, the DFS mostly generate only one path, but other search methods can generate more paths as many as possible.

	Case 1	Case 2	Case 3
BFS	12	34	378
DFS	12	38	1548
IDDFS	12	38	1534
ASTAR	12	34	757

Test Case 1 Solution Graph	
 (0,0,0) (3,3,3) (3,1,3) (0,0,0) (2,0,2) (1,1,1) (2,1,3) (1,0,0) (2,1,0) (1,0,3) (1,0,3) (2,1,0) (0,0,3) (3,1,0) (1,0,0) (2,1,3) (1,1,3) (2,0,0) (3,1,0) (0,0,3) (2,0,0) (1,1,3) (2,1,2) (1,0,1) (1,0,1) (2,1,2)	 (0,0,0) (3,3,3) (3,1,3) (0,0,0) (2,0,2) (1,1,1) (2,1,3) (1,0,0) (1,1,1) (2,0,2) (0,0,3) (3,1,0) (1,0,0) (2,1,3) (1,1,3) (2,0,0) (3,1,0) (0,0,3) (2,0,0) (1,1,3) (2,1,2) (1,0,1) (1,0,1) (2,1,2)
BFS	DFS

 (0,0,0) (3,3,3) (3,1,3) (0,0,0) (2,0,2) (1,1,1) (2,1,3) (1,0,0) (2,1,0) (1,0,3) (1,0,3) (2,1,0) (0,0,3) (3,1,0) (1,0,0) (2,1,3) (1,1,0) (2,0,3) (1,1,3) (2,0,0) (3,1,0) (0,0,3) (2,0,0) (1,1,3) (2,1,2) (1,0,1) (1,0,1) (2,1,2)	 (0,0,0) (3,3,3) (3,1,3) (0,0,0) (2,0,2) (1,1,1) (2,1,3) (1,0,0) (2,1,0) (1,0,3) (1,0,3) (2,1,0) (0,0,3) (3,1,0) (1,0,0) (2,1,3) (1,1,3) (2,0,0) (3,1,0) (0,0,3) (2,0,0) (1,1,3) (2,1,2) (1,0,1) (1,0,1) (2,1,2)
IDDFS	ASTAR

Discussion

Mode	Case	Expected	Description
BFS	start1.txt & goal1.txt	Full	We think all of our solutions are correct. After comparing each solution, BFS, DFS, IDDFS, and ASTAR search can generate the same path as a solution. However, the DFS always creates only one solution in the case 2 and 3. The thing we feel surprised by the most is the solution created by IDDFS has an exception in the last two states. Based on the checking validation function implemented, it shouldn't generate this path. But it does. Therefore, we feel very confused. We don't check each line from other result files, but we think it is possible to have the same problems in those results. In general, the more of the numbers of cannibals and missionaries is, all four search method will take more time to run and find out possible solutions. Also, there are more solutions in test cases with more numbers of cannibals and missionaries. During testing each case, we find the IDDFS and ASTAR are better than the BFS and DFS.
	start2.txt & goal2.txt		
	start3.txt & goal3.txt		
DFS	start1.txt & goal1.txt	Some	
	start2.txt & goal2.txt		
	start3.txt & goal3.txt		
IDDFS	start1.txt & goal1.txt	Some	
	start2.txt & goal2.txt		
	start3.txt & goal3.txt		
ASTAR	start1.txt & goal1.txt	Full	
	start2.txt & goal2.txt		
	start3.txt & goal3.txt		

Conclusion

We can conclude from these results that the search algorithm that performs the best is A* search. A* search uses a heuristic and this heuristic uses problem-specific knowledge. IDDFS also found an optimal solution, but took longer because of all the nodes it had to expand.