Armando Martínez Mandujano A01205786
Christopher Ojeda Rivera A01203980

Lab: Implementing (Un)informed Search Algorithms

Which heuristics did you use for the A* algorithm?

- In the first A* we used the position of the box on the stack, starting with 1 and adding 1 as the level increments and adding the distance to the goal state.
- In the second one, we used the quantity of columns of the space

Test your program with a couple of different problems. Increase the size of the problem to test the limits of your program. Make a table comparing how many nodes are searched to find the answer for each problem. For this table, you should compare a number of different problems (at least 3) to avoid a statistical bias. Which of the four algorithms searches the least nodes and which one take the most?

The performance of the algorithms was improved by making hashes of the visited nodes to avoid duplicates.

Test 1

3
(A); (B); (C)
(); (AB); (C)

	dfs	bfs	A*	A* inconsistence
Cost	6	6	6	6
Nodes	20	55	39	45
Path	(1, 2); (0, 1); (2, 1)	(1, 2); (0, 1); (2, 1)	(1, 2); (0, 1); (2, 1)	(1, 2); (0, 1); (2, 1)

In test 1 the algorithm with the minor quantity of nodes is depth first search, while breadth first search takes the most with 55 nodes. All the algorithms cover the same path to the solution with the cost of 6.

Test 2

4
(AE); (C); (D)
(E); (AC); (D)

	dfs	bfs	A*	A* inconsistence
Cost	-	12	12	12
Nodes	37	26	314	235
Path	No solution found	(0, 2); (1, 2); (0, 1); (2, 1); (2, 0)	(0, 2); (1, 2); (0, 1); (2, 1); (2, 0)	(0, 2); (1, 2); (0, 1); (2, 1); (2, 0)

In test 2 the depth first search shows no solution because the given depth is lower than the maximum permitted. The algorithm with the less quantity of nodes is breadth first search with 26 nodes while the one with the more quantity of nodes is A* with 314. In this test the algorithms that gave an answer obtained the same solution once again.

Test 3

5
(D, E); (A); (B, C)
(A, C); X; (E, B)

	dfs	bfs	A*	A* inconsistence
Cost	-	23	23	21
Nodes	28	13745	5786	6328
Path	No solution found	(0, 2); (0, 2); (1, 0); (2, 1); (2, 1); (2, 0); (2, 0); (1, 2); (0, 2)	(0, 2); (0, 2); (1, 0); (2, 1); (2, 1); (2, 0); (2, 0); (1, 2); (0, 2)	(2, 1); (2, 1); (0, 2); (1, 2); (1, 2); (0, 2); (1, 0); (2, 1); (2, 0)

In this test also the depth limit of the solution is less than the one given to the algorithm. The algorithm with the less quantity of nodes in this case was breadth first search and the one with the most was A* inconsistent but it gave the solution with less cost than the other two, giving a cost of 21..

Why does this happen?

In the case of the depth first, the algorithm founds no answer due to the limit of depth set to 5, the difference in solutions of the A* and A* inconsistent is caused by the heuristic implemented to each algorithm. The difference between these algorithms and the breadth first is that by implementing prioritization in the elements the result changes, and it is more probable that the less cost solution is found.

Which algorithms are optimal? Why?

According to the test cases the algorithm that results optimal is breadth first search, with the less quantity of nodes as the complexity of the search increased. Due to the heuristics applied to the A* algorithms their behaviour instead of being improved was reduced, but it got the most optimal solution in all of the cases regard of the quantity of nodes.

In your opinion, what are the benefits of simpler algorithms versus more complex ones?

- Simpler algorithms would be easier to implement but they would not solve the problem with the most efficient solution
- Complex algorithms may harder to implement but they can improve the efficiency
- Simpler algorithms are good to begin to understand a topic and complex ones help to fully understand the problem