Tecnológico de Monterrey Campus Querétaro

Laboratory: Implementing Uninformed Search Algorithms

Cinthia Valdez Guillén A01203141
Sergio Antonio Juárez Benítez A01270410
Eduardo Josué Contreras Álvarez A01064882

Artificial Intelligence
Ruben Stranders
October 4th 2016

Laboratory: Implementing Uninformed Search Algorithms

The Problem

The objective of this Laboratory were to build a solver for “the container crane problem” where the containers were labeled from A to E and the stacks labeled from 0 to 2.
The solver returns the sequence of movements that it takes for the crane to move the stack of containers from the initial state to the goal state.

The cost of function is calculated is based on time:

- Picking a container takes 0.5 minutes.
- Moving the container one stack to the left or right takes 1 minute (1 minute per movement).
- Puting the container down takes 0.5 minutes.

A* algorithm

For the A* algorithm the laboratory required the solution to use two different heuristics, one consistent and one non-consistent.

- For the Consistent heuristic we used “*The number of the misplaced containers in the three stacks.”*
    - It’s admissible because it never overestimates the cost of reaching the goal.
    - It’s consistent because the number of our heuristic is less than or equal to the estimated distance from any state next to the goal, plus the step of cost of reaching the goal. This means that the estimated final cost of a partial solution is non-decreasing along the best path to the goal.
- For the Non-consistent heuristic we used “*Twice the number of the misplaced containers in the three stacks.”*
    - Because it can give a lower bound on the cost to get to the goal than its parent node is giving. This means that the estimated final cost of a partial solution can decrease along the best path to the goal.

The next table compares the problems with each algorithm and shows the number of nodes that are searched to find the answer.

Problem	BFS	DFS	A* Cons	A* Non-cons
Dummy 1	1	1	1 (cost 2)	1 (cost 2)

Dummy 3	3	7	3 (cost 6)	3 (cost 6)
Dummy 4	3	123	3 (cost 12)	3 (cost 12)
Problem 1	10	295	10 (cost 23)	10 (cost 23)

There were two algorithms that gave the same results: BFS and A*, and definitely the one that took the most nodes was DFS.

In this case it happened that this two algorithms (BFS and A*) gave the same number of nodes but that is not always gonna happen because BFS is not optimal but A* Is an algorithm that always warranty the optimal result.

For the DFS case, it expands as many nodes as possible trying to reach the deepest node but the goal is not always in that branch or the deepest node causing this algorithm to open more nodes.

A* is always optimal, and BFS is only optimal when the step cost is constant, like in this case this is why we had the same results in BFS and A*.

In our opinion the complexity of implementation and understanding it’s easier for the simpler algorithms and as we saw for small search spaces they can give similar results. This can be considered as an advantage over the complex ones.