Esteban Quintana

Javier Rodríguez

Lab 3

(Un)Informed Search

- Which heuristics did you use for the A* algorithm?

  It compares all the values of the states with the goal state. It adds one to te heuristic for each stack that differs.

```

example:
Goal state: [ [B], [A], [C] ]
State [[A], [C], [B]]

````
herustic_value = 0
for index, element in state:
  if (element == X): cotinue
  else if (element != goal_state[index]): herustic_value++
return herustic_value

In this case the result is 3
```
````

- 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 three algorithms (UCS, A with consistent and and A with an inconsistent heuristic) searches the least nodes and * which one take the most?

	A Star Cons	A Star incons	UCS
1 (a);(b);() (a,b);x;x	13	13	13
3			

(a);(b,c,d);() (a,b);(c);(d)	55	59	77
4 (a,b);(d);(c) (b,a,c);();(d)	675	687	669
4 (a);(b);(c,d,e) (a,d);();(b,c,e)	1719	1909	2471

- Why does this happen?
- The consistent A* algorithm is a more optimal algorithm because it can assume (through the herustic algorithm) it's going in the right direction to reach the goal. The A* inconsistent cannot be always optimal since there are external factors that can alter the path the algorithm is going to take to reach certain goal. The difference between A* and UCS is that USC expands the node with lowest path cost whereas best expand the node with closest to the goal, but since it is an uninformed algorithm, it can't really tell if it is going in the right direction in all the cases.

- Which algorithms are optimal?
- Why? The three of them optimal because at the end, they reach the expected goal. But in memory/time to reach the expected goal, the best one is the consistent A* algorithm. If the heuristic is well posed, the algorithm can reach a solution efficiently having the certainty that it is following the correct path.

- In your opinion, what are the benefits of simple algorithms versus more complex ones?
- We don't believe simple algorithms are better than complex ones, it is fully dependent on the result, if a complex one provides a better solution but takes more time than a simple one but both give an accurate result you would need to choose which one to use depending on the needs and resources of the current problem.