1. Model the puzzle as a search problem

- **States:** $(f + e)$ x 4 matrices where f is the number of full bottles and e is the number of empty bottles. The matrices consist of integers where a cell has -1 if it is empty, c_i if it has ball with color i.
- **Initial State:** A $(f + e)$ x 4 matrix where the first f rows come from input and the last e rows contain -1s.
- **Successor State Function:** Takes a state (matrix of $(f + e)$ x 4) as parameter. The action is performed between two rows. The rightmost (not (-1)) value of a row is changed with the leftmost -1 value of another row (if colors match).
- **Step Cost:** It is equal to 1 for each state transition (each ball move).
- **Goal Test:** A matrix is at goal state if e many rows contains only -1s and other rows contain the same color in all of their columns.

3. Heuristic Function

In my A* Search algorithm heuristic cost of a state -which is a matrix- is calculated as follows:

h = number of rows that contain more than one color

For example, if a node's state has 4 bottles where 1st one is full of c_i colored balls, and the others have balls from different colors, then h(node) = 3.

My heuristic function is admissible. I.e. It never overestimates the true cost. Because, for any row that contains multiple colors, number of actions to move different colored ball to another bottle will be at least 1 (if it is the top-most ball). If the ball to be moved is not at the top, or if there are more than 2 colors in a bottle, true cost will be greater than 1. However, it can never be smaller than estimated cost.

5. Experimental Results

I run both algorithms in 3 example games. The input and time-consumption of each algorithm for these three examples can be seen below:

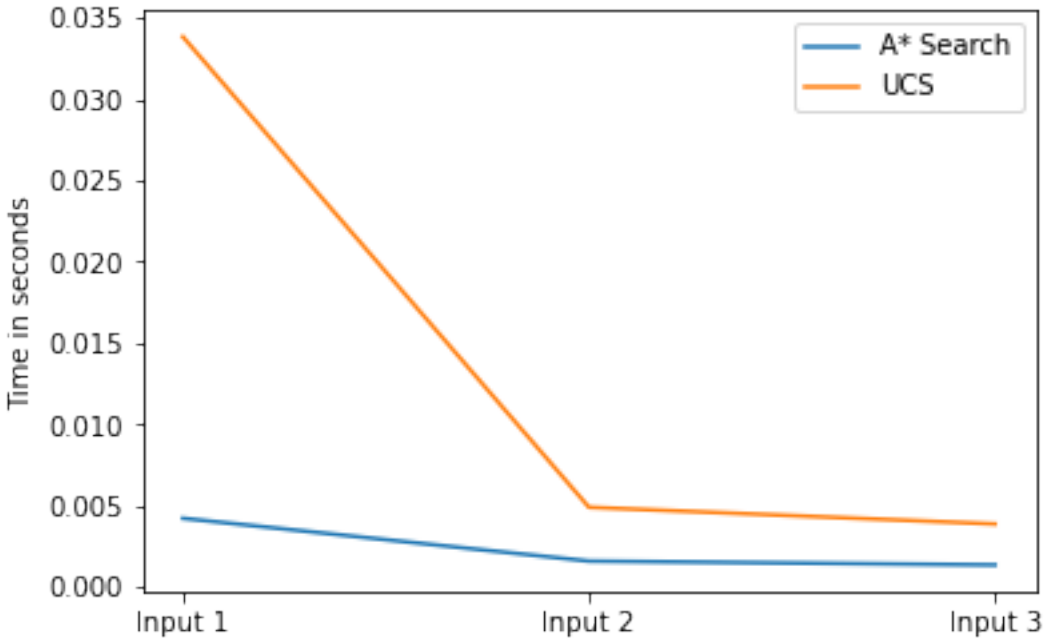
- Example Input 1:
  4 2
  2 2 2 1
  1 1 1 4
  3 3 3 2
  4 4 4 3
  A* Search - Time: 0.00424 s
  UCS - Time: 0.03382 s
  A* Search - Memory Peak: 91547
  UCS - Memory Peak: 1097677

- Example Input 2:
  3 1
  2 2 2 2
  1 1 3 1
  3 3 1 3
  A* Search: 0.001612 s
  UCS Search: 0.00491 s
  A* Search - Memory Peak: 16446
  UCS - Memory Peak: 138429

- Example Input 3:
  2 1
  1 2 1 2
  2 1 2 1
  A* Search: 0.00137 s
  UCS Search:0.003891 s
  A* Search - Memory Peak: 15847
  UCS - Memory Peak: 133101

As it can be clearly seen in the below table, A* Search algorithm took less time for all three examples. It is expected since A* uses heuristic function. i.e. It is an informed search algorithm. It can guess that choosing a certain action over others is cheaper to get to goal. With my heuristic function, A* chooses the actions which led less number of rows that contain multiple colors. If a bottle is solved (has only same colored balls), as a rule, A* will not try to move balls from it (because this would increase the expected cost.)

Uniform-Cost Search algorithm, on the other hand, is not informed. So, it will expand the search graph without any guess on expected cost. It will -in most cases- explore greater number of nodes until it reaches the goal.



The memory usage statistics for two algorithms is also as expected because the A* Search is more selective when expanding the search graph. It explores the nodes that are closer to the goal first. However, UCS does not perform any such selection. Therefore, it consumes more memory as below graph shows.