CS312, Lab-2 Report

R.Chidaksh(200010046),R.Sai Krupakar(200010047)

January 1, 2022

1 Objective:

Blocks World Domain Game starts with an initial state consisting of a fixed number of blocks arranged in 3 stacks and we can move only top blocks of the stacks and we have to achieve a goal state that is a particular arrangement of blocks by moving these blocks. Blocks World is a planning problem where we know the goal state beforehand and the path to the Goal state is more important.

2 Brief description about the domain:

2.1 State space

The "state space" is the Euclidean space in which the variables on the axes are the state variables. The state of the system can be represented as a state vector within that space. To abstract from the number of inputs, outputs and states, these variables are expressed as vectors.

2.2 Start node and Goal node

Start node:
We can say that Start node is the configuration in which the blocks are arranged initially. Generally we start from here and go to the goal state which we have to reach.
Example of the Start node is:

$[E, B, F]$
$[D, A]$
$[C]$

Goal node:
We can say that Goal node is the configuration which we should arrive after implementing the code.
Example of the Goal node is:
$[A, D, B]$
$[E, F, C]$
[ ]

3 Pseudo codes

3.1 MOVEGEN(current state)

Main use of this function is to generate the neighbours of the present state and storing it in a list.

Algorithm 1 MOVEGEN(current state)

```
procedure MOVEGEN(CURRENT STATE)
    nextStates ← ()
    for neighbour n of state in order(HeuristicValue) do
        pop(List)
        nextStates.append(n)
     return nextStates
```

3.2 GOALTEST(current state)

The main use of this function of this function is to check whether the present node matches with the goal node or not.

Algorithm 2 GOALTEST(current state)

```
procedure GOALTEST(CURRENT STATE)
    if currentstate.value == goalstate then
        return True
    return False
```

3.3 Heuristic Functions

3.3.1 Heuristic Function-1

In an intermediate state, depending upon number of matching-positions of blocks with goal state , If a block is in correct position with respect to the goal state it is assigned a value of +1 or else it is assigned -1, heurestic function is the sum of all such values for all the blocks.

Algorithm 3 heuristic_1

```
if CurrentState(Block) == GoalState(Block) then
    for all blocks do
        h(block) += 1
else
        h(block) -= 1
```

3.3.2 Heuristic Function-2

In an intermediate state, the heurestic of a block is $|x_1 - x_2| + |y_1 - y_2|$ where x_1, y_1 are x and y coordinates of the block in start state, x_2, y_2 are the x and y-coordinates of the block in goal state. And the heurestic function is the sum of heurestics of all the blocks.

Algorithm 4 heuristic_2

```
for all blocks do
    h(block) += |x1 − x2| + |y1 − y2|
```

3.3.3 Heuristic Function-3

In an intermediate state,If a block is not in the correct position with the respect to the goal state, then it's heurestic is $sqrt(|x_1 - x_2|^2 + |y_1 - y_2|^2)$.

Algorithm 5 heuristic$_3$

```
if CurrentState(Block) == GoalState(Block) then
    for all blocks do
        h(block) += sqrt(|x1 − x2|^2 + |y1 − y2|^2)
```

3.4 Best First Search Analysis and Observations

Analysis:
BeFS is a greedy algorithm that searches all the outcomes until it reaches the goal state. BeFS takes up more space and time than Hill Climbing.

Observations:
For our heuristic functions the BeFS performed very differently. We got the best result for the 1st approach and the worst result for the manhattan based approach.

4 Hill Climbing and BeFS comparison

4.1 States explored

The number of states explored by best first search are more than that of hill climbing. This happens because hill climbing considers only the best state based on the heuristic (minimum if it's a minimization problem or maximum if it's a maximization problem) out of all possible next states, whereas BeFS takes all possible states.

4.2 Time taken

As Hill Climbing explores only the best state at each level, it explores less number of states than BeFS, so Hill Climbing is faster than BeFS. In general,

Best first search = $O(N3)$ where N is the number of possible states Hill climbing = $O(N2)$ where N is number of possible states

4.3 Reaching the optimal solution

Hill Climbing hardly reaches the goal state, most of the time it gets stuck on the local minima, hence we have many variations of hill climbing to reach global maxima or minima. BeFS won't stop until it reaches the goal state, and searches for each and every state.

5 Conclusion

From the results above, it is seen that Best First Search (BeFS) always finds an optimal solution with the trade off of time, as it explores all possible N! states in the solution space.
Conversely, on the other hand, the Hill Climbing Algorithm, has lesser execution time due to recursive greedy selection in iterations but it cannot guarantee an optimal solution in all cases.
So it is better to use Befs than plane Hill climbing or an good option would be implementing Hill Climbing with proper variations increasing the exploration power of the algorithm.