AI Assignment-2 Report

Team -14
Tabish Khalid Halim ,
200020049
Anand Hegde ,
200020007

Department of Computer Science, IIT Dharwad

January 2, 2022

Contents

1 Approach

The basic approach we used for the A.I assignment is to use the concept of Heuristic Search Algorithms and apply BFS {Best First Search} and Hill-climbing techniques.

State Space

A state space is the set of all configurations that a given problem and its environment could achieve .

We have been given with the fact that each node has exactly not more than 6 neighbors . The Set Space Set (S) consists of all possible configuration of the blocks in the set of 3 stacks .

Start & Goal node

We have taken the start node , the goal node and the algorithm mode as inputs which are provided in input.txt file and the corresponding output is displayed in CLI .

The heuristic value for each visited node is calculated with reference to the goal node using the heuristic function .

One such example is the following :
Start Node :
E B F
D A
C
Goal Node :
A D B
E F C

2 Pseudo Code

The main pseudo code used in our assignment is as follows :

move_gen function

```
def move_gen(curr_state):
    global closed, open,parent
    state = copy.deepcopy(curr_state)
    neighbors = [ ]
    for i in range(len(state)):
      temp = copy.deepcopy(state)
      if length of temp[i] is greater than 0:
        elem = temp[i].pop()
        for j in range(len(temp)):
          temp1 = copy.deepcopy(temp)
          if j not equal to i:
            temp1[j] = temp1[j] + [elem]
            if(parent.get(stringify(temp1)) is empty):
              neighbors.append(temp1)
    return neighbors
```

goal_state function

```
def goal_state(cur,i):
    global goal
    if(heuristic(cur,i) is equal to heuristic(goal,i)):
      return 1
    return 0
```

3 Heuristic functions

3.1 Heuristic 1

- This function turns the Block problem into a maximization problem for the Best First Search Algorithm .
- A block is in its correct position iff it is present in the correct stack with appropriate height .
- This heuristic function assigns the value :
  1. +1 to the blocks of the current stack which are present in their correct position with respect to the goal state's blocks .
  2. -1 to the blocks which are not present in their right position with respect to the goal state .
- The sum of the assigned values to each of the blocks gives us the heuristic value for that state .
- The higher the heuristic value of a given state, the more priority is given to it during the BFS search .

3.2 Heuristic 2

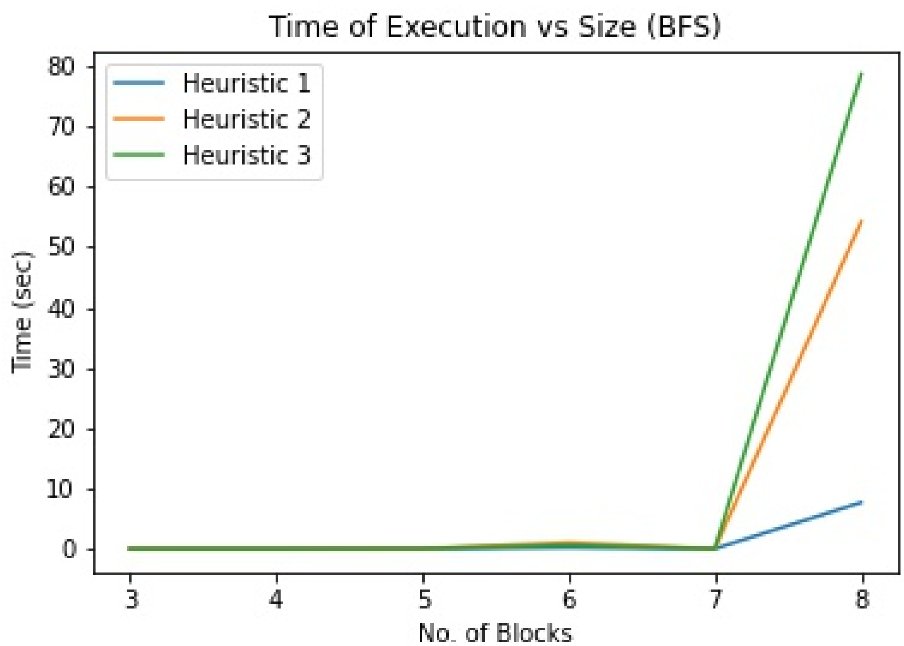
- This function turns the Block problem into a maximization problem for the Best First Search Algorithm .
- A block is in its correct position iff it is present in the correct stack with appropriate height .
- This heuristic function assigns the value :
  1. +h to those blocks of the current stack which have the same height as of that to the goal state's blocks , where h is the height of the block.
  2. $-h$ to the blocks which do not have the same height with respect to the goal state's blocks , where h is the height of the block .
- The sum of the assigned values to each of the blocks gives us the heuristic value for that state .
- The higher the heuristic value of a given state, the more priority is given to it during the BFS search .

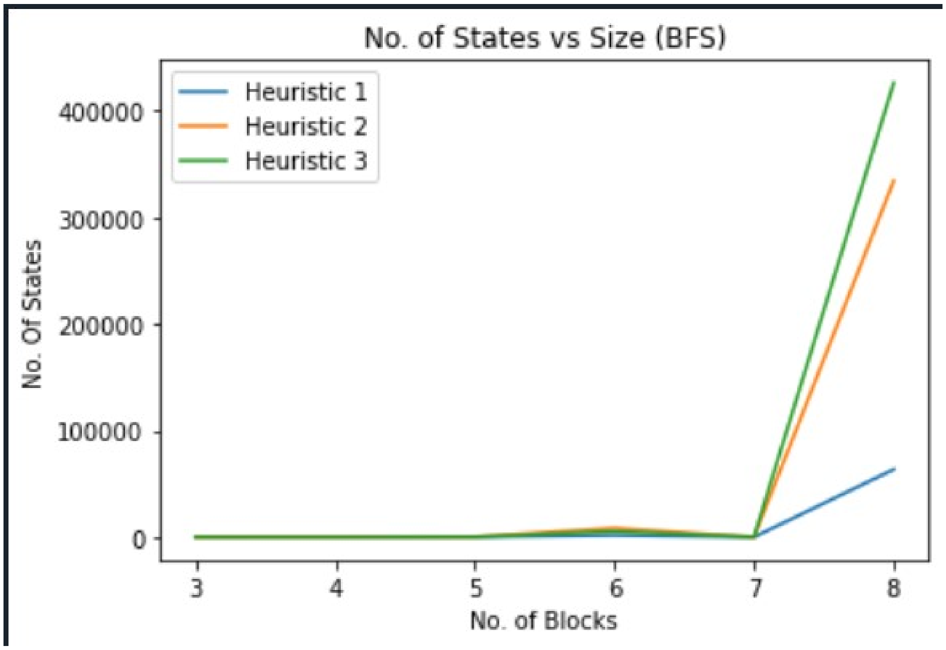
3.3 Heuristic 3

- This function turns the Block problem into a minimization problem for the Best First Search Algorithm .
- A block is in its correct position iff it is present in the correct stack with appropriate height .
- The sum of the Euclidean distances of each of the blocks in a state gives us the heuristic value for that state .
- The higher the heuristic value of a given state, the more priority is given to it during the BFS search .

4 Analysis & Observations

Based on observation , we have plotted the following graphs for Best First Search Algorithm :





From the above graphs , we have analysed that the best first search algorithm always finds the best and optimal solution but at the cost of time consumed is high as BFS has a time complexity of O(b^d) .

5 Results

The following conclusions have been made after evaluation of our program :

States Explored : The Hill-Climbing algorithm in general will have lesser no. of states explored than that of BFS . In the worst scenario, the best first search algorithm will explore all the states but Hill-Climbing algorithm will explore uptill $O(bd)$,
where b = the maximum beam width
d = the depth of the search graph from the start state.

Time Complexity : The time complexity for the Best First Search Algorithm is $O(b^d)$ and for Hill-Climbing algorithm it is $O(bd)$.

Optimal Solution : For reaching to the optimal solution , the Best First Search Algorithm is the better algorithm as it explores all the nodes/states but in the case of Hill-Climbing algorithm it doesn't explore all the states and it may get stuck in the local extremum while finding the goal state .

Completeness : The Best First Search Algorithm is Complete as it explores all the states sooner or later. But the Hill Climbing Algorithm is not complete. As it almost always gets stuck in local minima if we do not come up with a really good heuristic.

Conclusion

The Best First Search Algorithm is the more optimal choice as it helps to solve the given problem efficiently .

Although it takes more time than the Hill-Climbing algorithm, it ensures that the best solution has been found to reach the goal state by exploring all the nodes/states in $O(b^d)$, whereas in Hill-Climbing algorithm not all the nodes/states are visited as sometimes the algorithm gets stuck at a local extremum while finding the optimal path to the goal state.

6 References

- http://geeksforgeeks.com
- https://wikipedia.org
- https://stackoverflow.com