LAB-2 REPORT
GROUP: 2

SHUBH AGARWAL AND SAKSHAM CHHIMWAL

1. Domain Description

State Space: Our Implementation accepts the states as tuple (x, y, label) for each block in the state. For example: (1, 2, C)

Start Node and End Node: Check input.txt for initial node and goal.txt for final node. Below is the graphical representation of the initial and final nodes.

E	
A	**C**
B	**D**

Table 1. Initial State

C	
A	**E**
B	**D**

Table 2. Final State

MOVEGEN Algorithm: We are using 3 `stack` to find the next generations. First, we will convert our state representation into the `stacks` (x will be stack number, y will be index of block in that stack and label will be the representation), then the top element of non empty stack is moved to other stacks. It gives us all the possible next states. Now each possible state is converted back to tuple representation from stack representation. We are using `Hill Climb(Greedy)` approach to find the solution using one of the four heuristics:

- Manhattam Distance Heuristic
- XNOR Heuristic
- XNOR-height Heuristic
- ASCII-Code Heuristic

Pseudocode:

```
Initialize stack in a 2-D Array from the List of Tuples
Initialize List holding all possible States that can be achieved
FOR EACH Stack:
    Get the TopBlock
    FOR different Stack:
        Add the TopBlock to Pile2
        PUSH this state in Possible State List
        POP the TopBlock from Pile2
    PUSH the TopBlock in Pile
return the List containing the possible states
```

Date: January 24, 2023.

This paper is in final form.

GOALTEST Algorithm: It is just a simple comparision test, simple comparing given_state and goal_state.
Pseudocode:

```
Get the currentState and the finalState
If currentState.SORT(key=LABEL) == finalState.SORT(key=LABEL)
    return true
return false
```

2. Heuristic Functions Considered

`xnor_heuristic.` This heuristic is the one basic heuristic discussed in class. This gives a value **+1** to the blocks that are on the correct position w.r.t to the GOALSTATE. And assigns a value of **-1** to the ones that are on the incorrect position w.r.t the GOALSTATE.
This heuristic has a high possibility of getting stuck and was not able to reach the GOALSTATE many a times.

`xnor_heuristic_modified.` This heuristic combines the xnor_heuristic with height. It works as follows:

(1) If the item is at the correct position and has a height `h` then it will assign it a value of `+h`
(2) If the item is not at the correct position and has a height `h` then it will assign it a value of `-h`.
(3) The height starts form 1 at the bottom of the stack. That is the lowest block has a height of `1`.

This heuristic was better than the `xnor_heuristic` and was able to reach the GOALSTATE for some of the inputs.

manhattan_heuristic. It sums the distance between block in current state and the final state. We return additive inverse of the sum because our hill climb maximizes the heuristic value.

ascii_heuristic. It multiplies the height of block by the `ASCII` value of label and add it if the position of the block is correct and substract if the position is incorrect.

Heuristic	Goal Reached	States Explored
XNOR	Flase	0
XNOR_MOD	True	3
MANHATTAN	False	1
ASCII	True	3

Table 3. Final Results

3. Hill Climbing

The Hill Climbing approach works as follows:

(1) It calculates the value of the heuristic of the initial state which is being used currently by the program.
(2) It then calculates the heuristic values for all possible state that can be reached from the current state and stores them in a list.
(3) It chooses the state that has the highest value in the list.
(4) If the chosen value if larger than the heuristic value of the current state then the current state is transformed into the chosen state.
(5) This loop continues till the GOALSTATE is reached or the program halts because of indeterminism.

The program halts under the following conditions:

(1) If the maximum heuristic value available among the possible states is lower than the current state.
(2) If all the heuristic values are same among the possible states.

Time Taken: We have already shared the number of steps by our heuristics, which is a good rough estimate of time taken by the heuristics. Time details are shared below:

Heuristic	Goal Reached	Time Taken(in ms)
XNOR	Flase	0.0036
XNOR_MOD	True	0.132
MANHATTAN	False	0.00926
ASCII	True	0.135

TABLE 4. Time Taken

Optimal Solution: We are effectively doing restricted `BFS`, solution if obtained will be optimal given the heuristics.

Email address, Shubh: `210020047@iitdh.ac.in`
URL: `https://shubhagarwal-dev.github.io/`

Email address, Saksham: `210010046@iitdh.ac.in`