Sarthak Aggarwal
2020101008
Assignment -2
machine & data learning

Q1 As per the model discussed in class,

Initial state → it is the initial configuration of the robot.

Goal state → it is the final configuration we want of the robot.

Actions → Possible moves listed below that can be done by robot:-

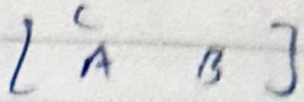
a) P~~lace~~ Place (x) → Place x on table.
b) Pop (x,y) → remove x from y block.
c) push (x,y) → keeping x on y block.
d) pickup (x) → pick x block.

States → it is the configuration of two blocks or block and the table
Following are the states:-

a) Tabletop (x) → x block is on table
b) Emptyarm () → robot's arm is void/empty.
c) Inhand (x) → x block in robot's arm
d) clear (x) → top of x block is empty.
e) Top (x,y) → x block on y block.

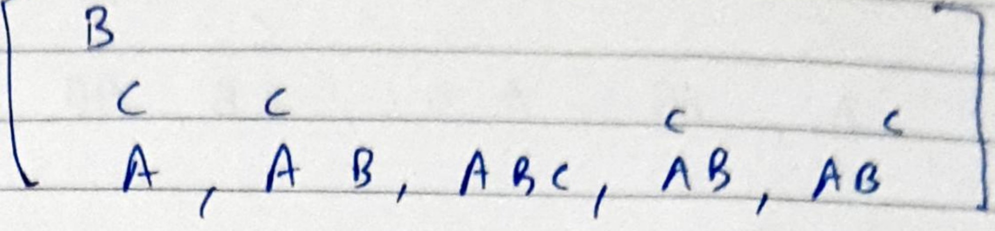
The path cost is one for every action, optimal solution is therefore, a solution with min. cost / least actions.

Q2 Breadth-first search :-

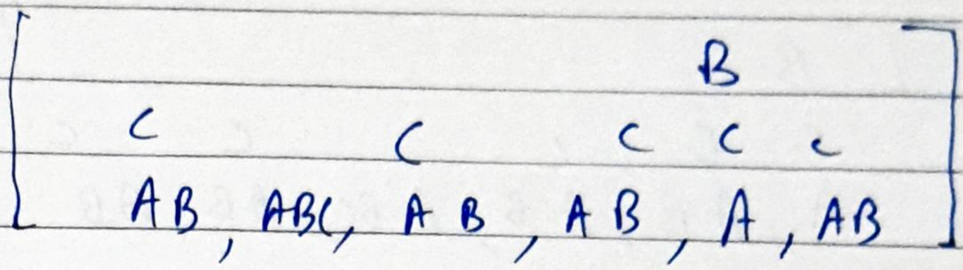


← initial state

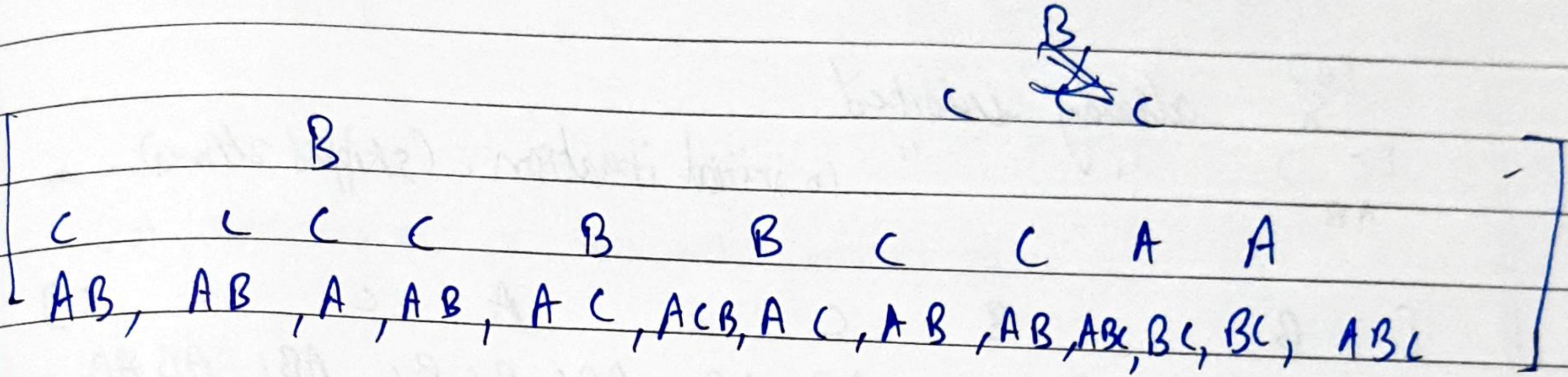
Iteration ① :-



Iteration ② :-



Iteration ③ :-


~~B~~
C ~~C~~ C
B
C C C C B B C C A A
AB, AB , A , AB , A C , ACB, A C , A B , AB, ABC, BC, BC, ABC

## Depth first search:-

iteration ①:-

```
                                   ]
 [ B
    C      c                C     c
  A ,    AB  ,  A B C,  A B ,  AB
```

iteration ②:-

```
                                      ]
 [ B
   C   C    c               C       c
   A , A B , A B , A BC,  AB  , AB
```

iteration ③ :-

```
[B]
 C
  A   already visited.
[C  ]          "          "   in initial iteration. (skipped 2times).
AB
```

```
                                          A       C        C
[    B          B      C       c                                C]
  A C , A C B , A C , AB , AB , ABC, BC, BC , ABC , AB, AB
```

Uniform cost search :- (no. of steps req. to obtain goal state from initial state).

iteration ① :-

```
 [                      A          ]
        C      C      C     C
     A  B  ,  A B  , A B , B, C A B
       1        0       0   1    1
```

iteration ② :-

```
  [                                              A             A    ]
                  C              C      C  C  C         C  C
       A B C  ,  A B  ,  A B ,  A B ,  B,  A B ,  A B , B,  C A B
          1         0       1      1   1    0      0    1     1
```

On removing all visited nodes from the list :-

```
Iteration ③ :-   [ A          ]
                   C C
                   B, BA, A B C
                   1   1    1
```

Q3 ①. Count the blocks which are not in the correct position as per there goal state and the block which is ~~as~~ present in the arm of the robot is not taken into account.

②. Measuring the distance b/w initial state and goal state while looking at details of each block

Example::

initial state::

[      C ]

A   B

Goal state::

[ C ]

A

B

As per ①, A & C are at incorrect position.

∴ Count = 2.

As per ②, A should be above B & below C, but it can be seen that its ~~the~~ opp. B & C are at incorrect pos. Therefore count = 2.

For C, no block is at top ∴ only 1 is added.

For B, only above block doesnt match, ∴ count = 1

Q4 Initial state :-

```
[ C  ]
  A B
```

Heuristic
Heuristic
1 mentioned in
previous question

Goal state

```
[A]
 B
 C
```

Iteration ① :-

```
   [ B                         ]
     c    c            c     c
     A  , AB  , ABC  , AB , AB
     3     3     2      3   3
```

Iteration ② :- (following the taken heuristic)

```
[                                                                           B             ]
             B                   B    c      c           A     A            c  c   c    c
       A     C,     A CB,        A C, AB,   AB,   ABC,   BC,   BC,   ABC,   A, AB, AB, AB
             1        2           2   3      3     2      2     2      2    3   3  3    3
```

Iteration ③ :- (removing visited nodes)

```
  [                                   A                                                                      B]
                              B       B         B       B             c           c      A        A         c
     ACB            ,        AC   ,   C  ,     AC   ,  AC        ,   AB   ,      AB   ,  Bc    ,  BC    ,   A
      2                       1       0        1        2             3           3      2        2         3
```