1.1 Searching in Space-Time Domain
CPU time (s): 0.00
Sum of costs: 6

1.2 Handling Vertex Constraints
CPU time (s): 0.00
Sum of costs: 7
[[(1, 1), (1, 2), (1, 3), (1, 4), (1, 4), (1, 5)], [(1, 2), (1, 3), (1, 4)]]

1.4 Handling Goal Constraints
At timestep 10, agent 0 is at cell (1, 4). Utilizing the earliest_goal_timestep variable to handle goal constraints by setting its value to be either zero or the timestep of a goal location constraint. In order for the goal test condition to pass, the current location must be equal to the goal location as well as the current timestep must be greater or equal than the earliest_goal_timestep.

1.5 Designing Constraints

```
constraints = [
            {'agent': 1, 'loc':[(1,3), (1,4)], 'timestep': 2},
            {'agent': 1, 'loc':[(1,3)], 'timestep': 2},
            {'agent': 1, 'loc':[(1,2)], 'timestep': 2}
        ]
```

CPU time (s): 0.00
Sum of costs: 8
[[(1, 1), (1, 2), (1, 3), (1, 4), (1, 5)], [(1, 2), (1, 3), (2, 3), (1, 3), (1, 4)]]

2.4 Addressing Failures
For instance exp2_3.txt, the solver does not report ‘no solutions’. Agent 0 is at the goal cell while Agent 1 waits one cell before Agent 0’s goal cell. Eventually Agent 1 collides with Agent 0 at timestep 8 on its way to its own goal cell as it must pass over Agent 0’s goal cell.

2.5 Showing that Prioritized Planning is Incomplete and Suboptimal
Design a MAPF instance for which prioritized planning does not find an (optimal or suboptimal) collision-free solution, no matter which ordering of the agents it uses.

```
4 7
@ @ @ @ @ @ @
@ . . . . . @
@ @ @ . @ @ @
@ @ @ @ @ @ @
2
1 1 1 4
1 5 1 2
```

(Bonus: 0.5pt) Design a MAPF instance for which prioritized planning does not find an (optimal or suboptimal) collision-free solution for a given ordering of the agents even if an ordering of the agents exists for which prioritized planning finds an optimal collision-free solution.

```
4 7
@ @ @ @ @ @ @
@ . . . . . @
@ @ @ . @ @ @
@ @ @ @ @ @ @
2
1 4 1 2
1 1 1 4
```

3.3 Implementing the High-Level Search

```
PS D:\CMPT 417\Individual Project\code> python run_experiments.py --instance
instances/exp2_1.txt --solver CBS
***Import an instance***
Start locations
@ @ @ @ @ @ @
@ 0 1 . . . @
@ @ @ . @ @ @
@ @ @ @ @ @ @

Goal locations
@ @ @ @ @ @ @
@ . . . 1 0 @
@ @ @ . @ @ @
@ @ @ @ @ @ @

***Run CBS***
Generate node 0
[{'a1': 0, 'a2': 1, 'loc': [(1, 4)], 'timestep': 3}]
[{'agent': 0, 'loc': [(1, 4)], 'timestep': 3}, {'agent': 1, 'loc': [(1, 4)],
'timestep': 3}]
Expand node 0
Generate node 1
Generate node 2
Expand node 1
Generate node 3
Generate node 4
Expand node 2
Generate node 5
Generate node 6
Expand node 3
```

```
Generate node 7
Generate node 8
Expand node 6
Generate node 9
Generate node 10
Expand node 10
Generate node 11
Generate node 12
Expand node 12
Generate node 13
Generate node 14
Expand node 14
Generate node 15
Generate node 16
Expand node 16

 Found a solution!

CPU time (s):    0.01
Sum of costs:    8
Expanded nodes:  9
Generated nodes: 17
***Test paths on a simulation***
```

4.3 Adjusting High-Level Search

```
CPU time (s):    0.01
Sum of costs:    11
Expanded nodes:  9
Generated nodes: 17
***Test paths on a simulation***
```