

CMPT417

Individual Project Report

Guangcheng Lin

Q1:

1.1 Result is the same as the example given.

```
ksl@ksl-virtual-machine:~/sfu/cmpt417/project_individual/code/code$ python3 run_
experiments.py --instance instances/exp1.txt --solver Independent
***Import an instance***
Start locations
@ @ @ @ @ @ @
@ 0 1 . . . @
@ @ @ . @ @ @
@ @ @ @ @ @ @

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

***Run Independent***

Found a solution!

CPU time (s):    0.00
Sum of costs:    6
```

1.2: Test in prioritized.py by adding constraints of

#constraints.append ({'agent': 0, 'loc': [(1, 5)], 'time\_step': 4}), agent 0 stops at (1,4) at timestep 4.

```
***Import an instance***
Start locations
@ @ @ @ @ @ @
@ 0 1 . . . @
@ @ @ . @ @ @
@ @ @ @ @ @ @

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

***Run Prioritized***
test is [{'agent': 0, 'loc': [(1, 5)], 'time_step': 4}]
loc is (1, 3) [(1, 5)]
loc is (1, 5) [(1, 5)]
loc is (1, 4) [(1, 5)]

Found a solution!

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

```

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

***Run Prioritized***
test is  [{'agent': 0, 'loc': [(1, 5)], 'time_step': 10}]
loc is  (1, 5) (1, 4) [(1, 5)]
loc is  (1, 5) (1, 5) [(1, 5)]
loc is  (1, 4) (1, 3) [(1, 5)]
loc is  (1, 4) (1, 5) [(1, 5)]
loc is  (1, 4) (1, 4) [(1, 5)]
earliest_goal_timestep is 11
earliest_goal_timestep is 0

Found a solution!

CPU time (s):    0.00
Sum of costs:    13
[[ (1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1, 5), (1, 5), (1, 5), (1, 5), (1, 5),
  (1, 4), (1, 5)], [(1, 2), (1, 3), (1, 4)]]

```

Agent 0 is moving from (1,5) to (1,4) at timestep = 10.

```

if curr['loc'] == goal_loc and curr['time_step'] >= earliest_goal_timestep:
    #print("earliest_goal_timestep is", earliest_goal_timestep)
    return get_path(curr)

```

I add condition of “curr[‘time\_step’] >= earliest\_goal\_timestep” in order to let the agent stay at their goal point until timestep is larger than the earliest goal timestep. In this case, earliest goal timestep is 10.

1.5

The set of constraints are

```
constraints.append({'agent': 1, 'loc': [(1, 3), (1, 2)], 'time_step': 2})
constraints.append({'agent': 1, 'loc': [(1, 3), (1, 4)], 'time_step': 2})
constraints.append({'agent': 1, 'loc': [(1, 3)], 'time_step': 2})
constraints.append({'agent': 1, 'loc': [(1, 4)], 'time_step': 3})
constraints.append({'agent': 1, 'loc': [(1, 2)], 'time_step': 2})
#print("test is ", constraints)
```

The sum of path length is 8. The solution is shown below.

```
Sum of costs:      8
[[ (1, 1), (1, 2), (1, 3), (1, 4), (1, 5)], [(1, 2), (1, 3), (2, 3), (1, 3), (1, 4)]]
```

## Question 2:

2.4: My solution is “Found a solution” instead of “No solution”. My timestep in constraint\_table is up to 9 in this case. Therefore, there is no obstruction and constraints when timestep is larger than 9. Agent 1 waits until timestep 9 at location (1, 3) and go through agent 0 and make collision towards (1, 4) which is the goal location of agent 0 at timestep = 10.

```
Found a solution!

CPU time (s):    0.00
Sum of costs:    13
[[ (1, 2), (1, 3), (1, 4) ], [ (1, 1), (1, 2), (1, 3), (1, 3), (1, 3), (1, 3), (1, 3), (1, 3), (1, 3), (1, 3), (1, 3), (1, 3), (1, 3), (1, 3), (1, 3), (1, 4), (1, 5) ]]
```

After I set the upper bound for the travelling time and the program would return “No solution”.

## 2.5:

i. No solution with condition of ordering agents:

```
4 7
@ @ @ @ @ @ @
@ . . . . @
@ . . @ . . @
@ @ @ @ @ @ @
2
1 2 1 3
1 1 2 3
```

Start locations

```
@ @ @ @ @ @ @
@ 1 0 . . . @
@ . . @ . . @
@ @ @ @ @ @ @
```

Goal locations

```
@ @ @ @ @ @ @
@ . . 0 . . @
@ . . 1 . . @
@ @ @ @ @ @ @
```

```
Hit upper bound: 10 time_steps
Traceback (most recent call last):
  File "/home/ksl/sfu/cmpt417/project individual/code/code/run_experiments.py", line 105, in <module>
    paths = solver.find_solution()
  File "/home/ksl/sfu/cmpt417/project individual/code/code/prioritized.py", line 49, in find_solution
    raise BaseException('No solutions')
BaseException: No solutions
```

ii. No solution no matter what order the agent is

4 7		
@ @ @ @ @ @ @		
@ . . . . . @	Start locations	Goal locations
@ . . @ . . @	@ @ @ @ @ @ @	@ @ @ @ @ @ @
@ @ @ @ @ @ @	@ . 0 . . 1 @	@ . 1 . 0 . @
2	@ . . @ . . @	@ . . @ . . @
1 2 1 4	@ @ @ @ @ @ @	@ @ @ @ @ @ @
1 5 1 2		

iii. (bouns): Raise no solution if:

4 7		
@ . @ @ @ . @		
@ . @ @ @ . @	Start locations	Goal locations
@ . . . . . @	@ 1 @ @ @ . @	@ . @ @ @ 1 @
@ . @ @ @ . @	@ 0 @ @ @ . @	@ . @ @ @ . @
2	@ . . . . . @	@ . 0 . . . @
1 1 2 2	@ . @ @ @ . @	@ . @ @ @ . @
0 1 0 5		

```
Hit upper bound: 11 time_steps
Traceback (most recent call last):
  File "/home/ksl/sfu/cmpt417/project individual/code/code/run_experiments.py",
line 105, in <module>
    paths = solver.find_solution()
  File "/home/ksl/sfu/cmpt417/project individual/code/code/prioritized.py", line
49, in find_solution
    raise BaseException('No solutions')
BaseException: No solutions
```

And if switching two agents, it finds solutions:

4 7		
@ . @ @ @ . @		
@ . @ @ @ . @	Start locations	Goal locations
@ . . . . . @	@ 0 @ @ @ . @	@ . @ @ @ 0 @
@ . @ @ @ . @	@ 1 @ @ @ . @	@ . @ @ @ . @
2	@ . . . . . @	@ . 1 . . . @
0 1 0 5	@ . @ @ @ . @	@ . @ @ @ . @
1 1 2 2		



```

Found a solution!

CPU time (s):    0.00
Sum of costs:    16
[[ (0, 1), (1, 1), (2, 1), (2, 2), (2, 3), (2, 4), (2, 5), (1, 5), (0, 5) ], [ (1, 1), (2, 1), (3, 1), (2, 1), (2, 2), (2, 2), (2, 2), (2, 2), (2, 2) ]]
***Test paths on a simulation***

```

Question 3:

3.3

```

ksl@ksl-virtual-machine:~/sfu/cmpt417/project_individual/code/code$ python3 run_
experiments.py --instance instances/exp2_1.txt --solver CBS
***Import an instance***
Start locations
@ @ @ @ @ @ @
@ 0 1 . . . @
@ @ @ . @ @ @
@ @ @ @ @ @ @

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

***Run CBS***
path is  [[ (1, 1), (1, 2), (1, 3), (1, 4), (1, 5) ], [ (1, 2), (1, 3), (1, 4) ]]
Generate node 0
3.1 [ {'time_step': 3, 'loc': [ (1, 4) ], 'a1': 0, 'a2': 1} ]
3.2 [ {'agent': 0, 'loc': [ (1, 4) ], 'time_step': 3}, {'agent': 1, 'loc': [ (1, 4) ], 'time_step': 3} ]
Expand node 0
Generate node 1

```

testing task 1 to task 3.

```

Hit upper bound: 6 time_steps
Generate node 24
Expand node 24
Paths:  [[ (1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1, 5) ], [ (1, 2), (1, 3), (2, 3), (1, 3), (1, 2), (1, 3), (1, 4) ]]
Expanded Nodes are  20

```