Multi Agent Pathfinding for autonomous vehicles navigating the roads of a city

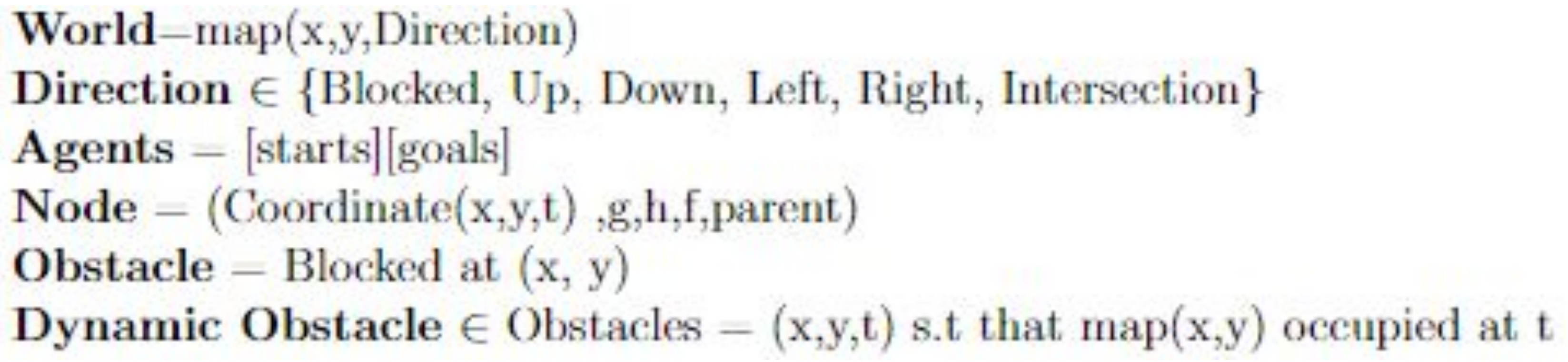
Andrew Chong, Harshul Singh, Darren Lee

Motivation

Once fully autonomous driving becomes feasible its eventual deployment is likely to lead to autonomous fleets of vehicles that are serving routes within an area. A centralized planner to optimize the high level path that minimizes the total cost of the paths for all cars would be helpful.

Current navigational tools seem to only consider route length and/or traffic in the present instead of considering car’s routes in the future.

Definition of the Problem



World=map(x,y,Direction)

Direction ∈ {Blocked, Up, Down, Left, Right, Intersection}

Agents = [starts][goals]

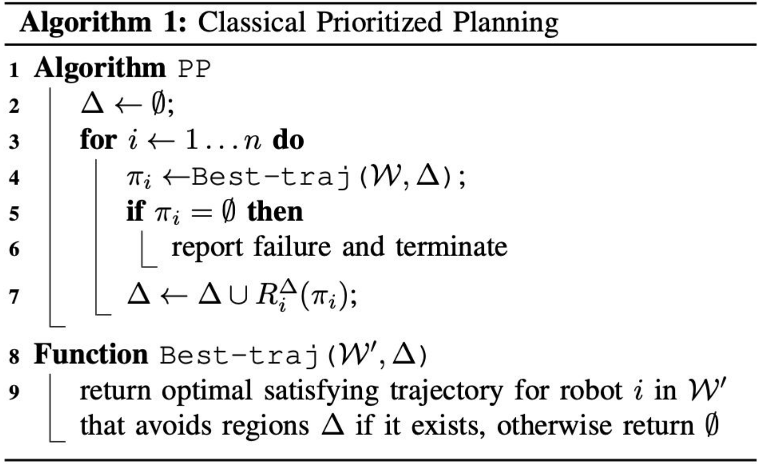
Node = (Coordinate(x,y,t) ,g,h,f,parent)

Obstacle = Blocked at (x, y)

Dynamic Obstacle ∈ Obstacles = (x,y,t) s.t that map(x,y) occupied at t

Algorithm

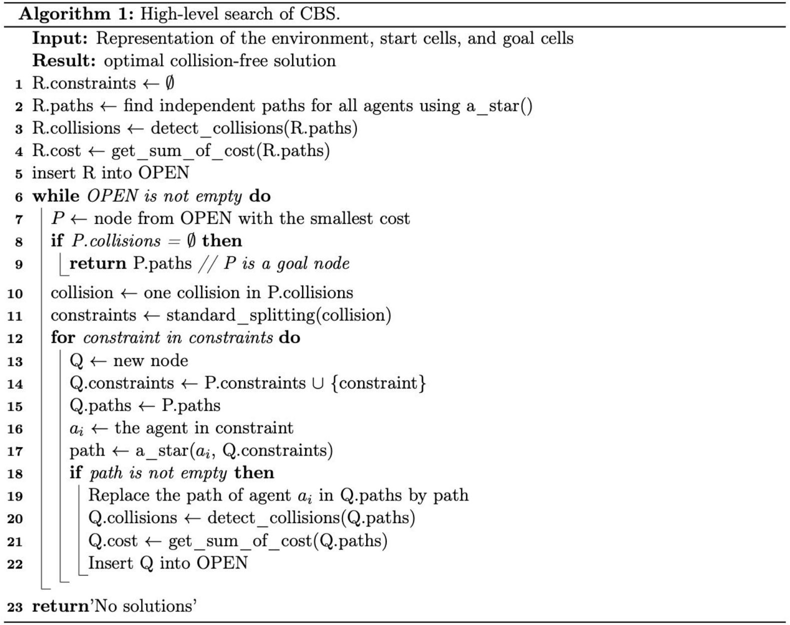
Prioritized planning with x, y, time a* search



Algorithm 1: Classical Prioritized Planning

```
Algorithm PP
    $\Delta \leftarrow \emptyset$;
    for $i \leftarrow 1 \ldots n$ do
        $\pi_i \leftarrow$Best-traj($\mathcal{W}, \Delta$);
        if $\pi_i = \emptyset$ then
            report failure and terminate
        $\Delta \leftarrow \Delta \cup R_i^{\Delta}(\pi_i)$;
Function Best-traj($\mathcal{W}', \Delta$)
    return optimal satisfying trajectory for robot $i$ in $\mathcal{W}'$
    that avoids regions $\Delta$ if it exists, otherwise return $\emptyset$
```

Constraint based planning with x, y, time a* search



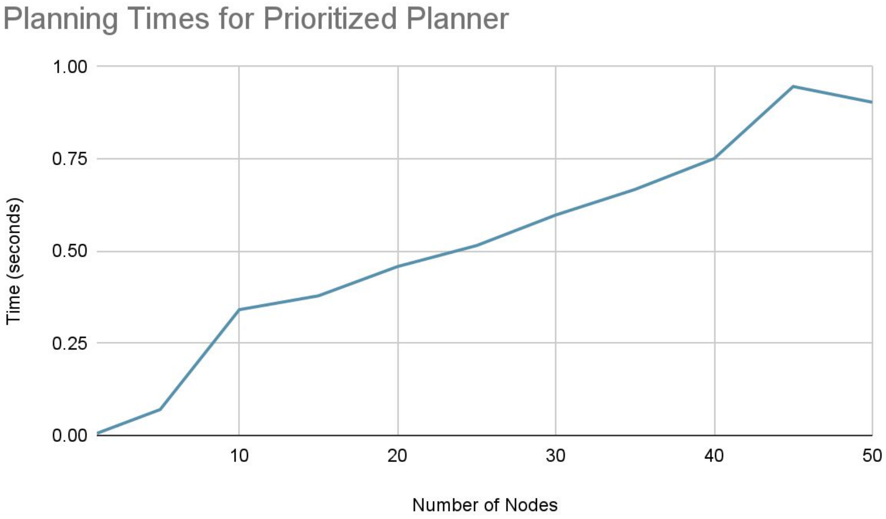
Algorithm 1: High-level search of CBS.

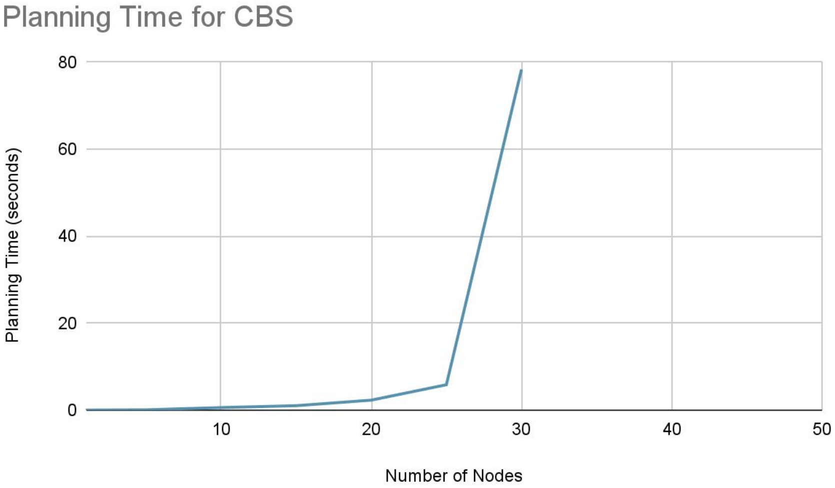
Input: Representation of the environment, start cells, and goal cells

Result: optimal collision-free solution

```
R.constraints $\leftarrow \emptyset$
R.paths $\leftarrow$ find independent paths for all agents using a_star()
R.collisions $\leftarrow$ detect_collisions(R.paths)
R.cost $\leftarrow$ get_sum_of_cost(R.paths)
insert R into OPEN
while OPEN is not empty do
    $P \leftarrow$ node from OPEN with the smallest cost
    if $P.collisions = \emptyset$ then
        return P.paths // P is a goal node
    collision $\leftarrow$ one collision in P.collisions
    constraints $\leftarrow$ standard_splitting(collision)
    for constraint in constraints do
        Q $\leftarrow$ new node
        Q.constraints $\leftarrow$ P.constraints $\cup$ {constraint}
        Q.paths $\leftarrow$ P.paths
        $a_i \leftarrow$ the agent in constraint
        path $\leftarrow$ a_star($a_i$, Q.constraints)
        if path is not empty then
            Replace the path of agent $a_i$ in Q.paths by path
            Q.collisions $\leftarrow$ detect_collisions(Q.paths)
            Q.cost $\leftarrow$ get_sum_of_cost(Q.paths)
            Insert Q into OPEN
return'No solutions'
```

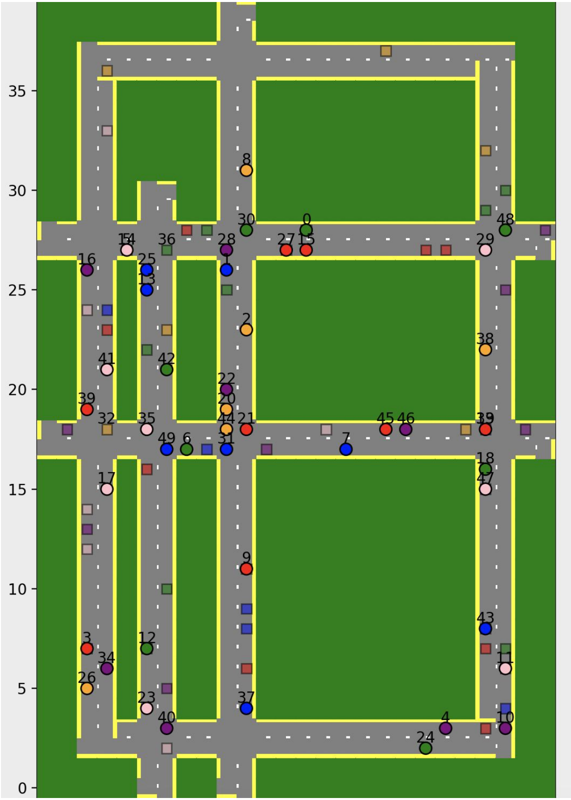
Results


Planning Times for Prioritized Planner
Time (seconds)
1.00
0.75
0.50
0.25
0.00
10
20
30
40
50
Number of Nodes


Planning Time for CBS
Planning Time (seconds)
80
60
40
20
0
10
20
30
40
50
Number of Nodes

Results

Demo



35
30
25
20
15
10
5
0
8
30
0
48
14
36
28
27
15
29
16
25
1
13
2
38
41
42
22
20
39
32
35
44
21
45
46
33
49
6
31
7
18
17
47
9
43
3
12
34
11
26
23
37
4
10
40
24