WPI

Decentralized Multi Agent Path Finding with I-CBS

Presented By:
Abhijeet Sanjay Rathi
Anuj Jagetia

Instructor:
Constantinos Chamzas

Problem Description:

The problem involves real-time multi-agent path planning system for a robot with ICBS within known environment, The robots needs to navigate ensuring safe and collision-free movement with efficiency for every individual agents without explicit communication between them.

Methodology

Implementations of

- A star algorithm
- Single agent planner.
- ICBS algorithm
  - Disjoint Splitting
  - Standard Splitting

ICBS Algorithm

Algorithm 1: High-level of ICBS

```
1  Main(MAPF problem instance)
2      Init R with low-level paths for the individual agents
3      insert R into OPEN
4      while OPEN not empty do
5          N ← best node from OPEN // lowest solution cost
6          Simulate the paths in N and find all conflicts.
7          if N has no conflict then
8              return N.solution // N is goal
9          C ← find-cardinal/semi-cardinal-conflict(N) // (PC)
10         if C is semi- or non-cardinal then
11             if Find-bypass(N, C) then // (BP)
12                 Continue
13         if should-merge(a_i, a_j) then // Optional, MA-CBS:
14             a_ij = merge(a_i, a_j)
15             if MR active then // (MR)
16                 Restart search
17             Update N.constraints()
18             Update N.solution by invoking low-level(a_ij)
19             Insert N back into OPEN
20             continue // go back to the while statement
21         foreach agent a_i in C do
22             A ← Generate Child(N, (a_i, s, t))
23             Insert A into OPEN
24 Generate Child(Node N, Constraint C = (a_i, s, t))
25     A.constraints ← N.constraints + (a_i, s, t)
26     A.solution ← N.solution
27     Update A.solution by invoking low level(a_i)
28     A.cost ← SIC(A.solution)
29     return A
```

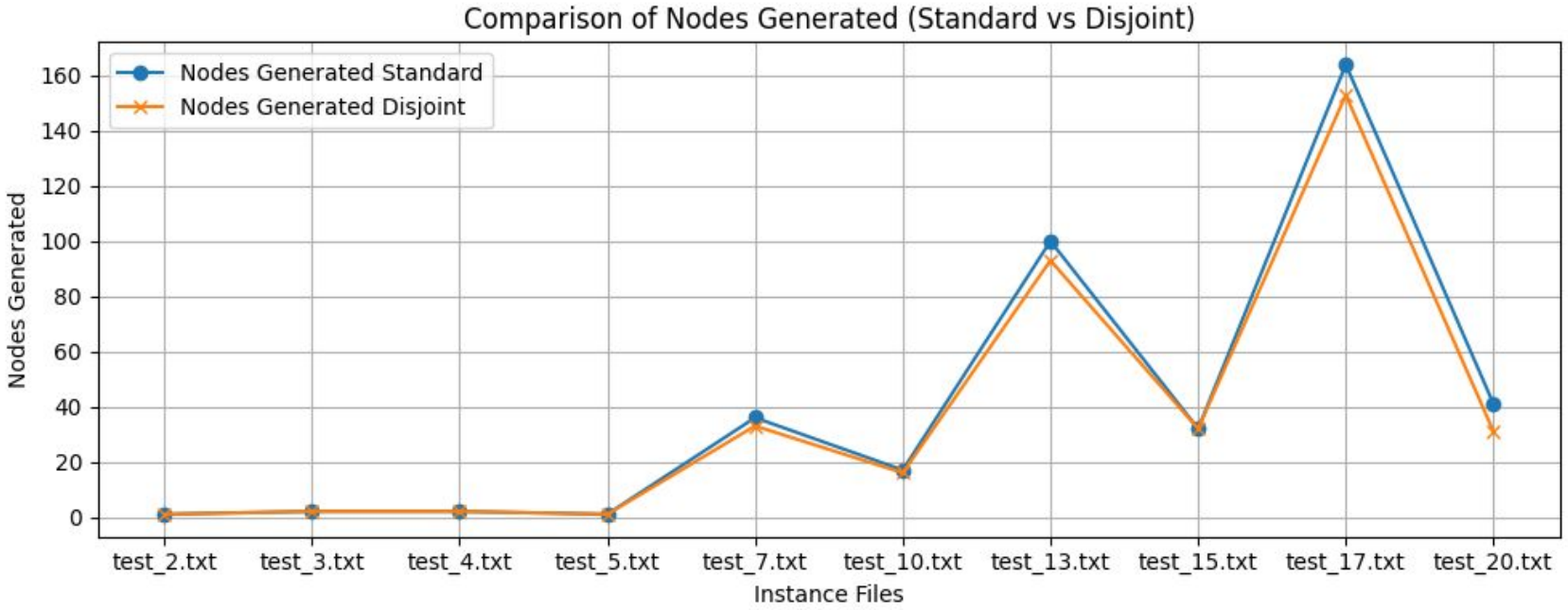
Disjoint vs Standard Splitting

- Standard Splitting: Two negative constraints split
  - Results in inefficiencies
- Disjoint Splitting: One positive and one negative constraint
  - Results in improved efficiency

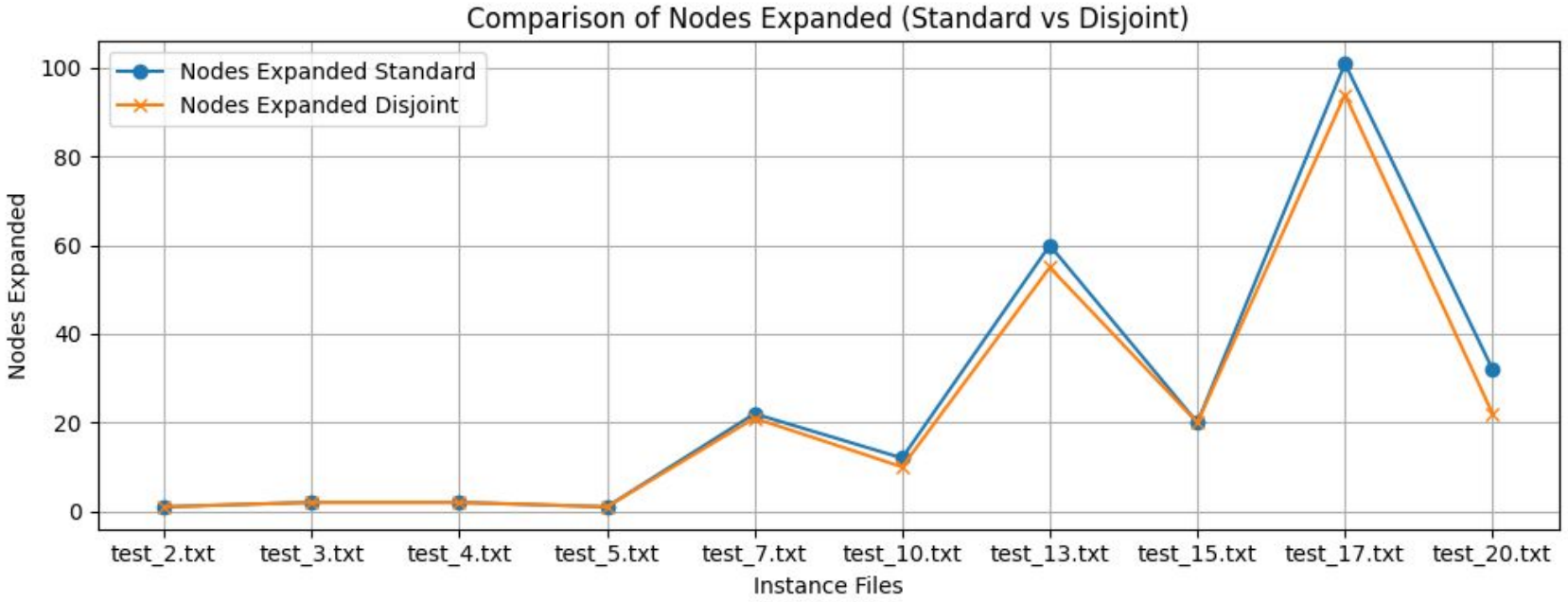
Results and Outcome

- It is able to find the solution without collision
- Impact of incrementing the agents.
- Evaluation of Efficiency
- Comparison of Disjoint and Standard Splitting

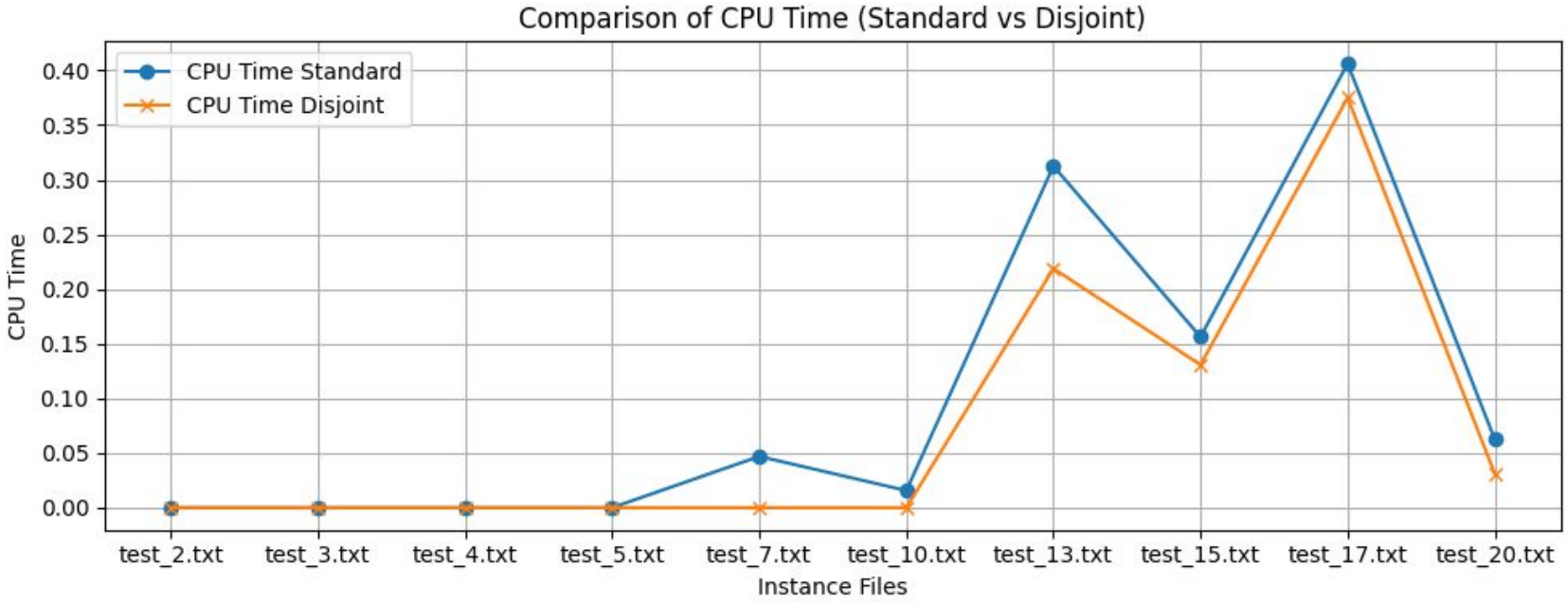
Result 1: Nodes Generated

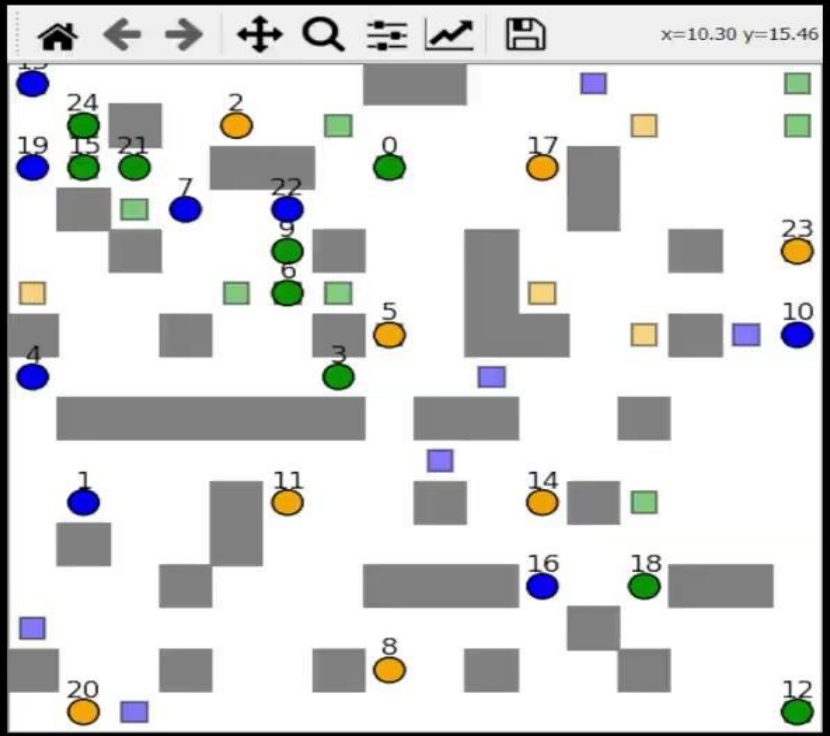


Result 2: Nodes Expanded



Result 3: CPU Time




x=10.30 y=15.46
24
2
19
15
21
0
17
7
22
9
23
6
5
10
4
3
1
11
14
16
18
8
20
12

Thank You