AAAI-25 / IAAI-25 / EAAI-25

FEBRUARY 25 – MARCH 4, 2025 | PHILADELPHIA, USA

Windowed MAPF with Completeness Guarantees

Rishi Veerapaneni, Muhammad Suhail Saleem, Jiaoyang Li, Maxim Likhachev

Carnegie Mellon University

Carnegie Mellon University Robotics Institute

Typical Heuristic Search MAPF Methods

- Many solvers
  - Conflict-Based Search: CBS, ECBS, EECBS, W-EECBS
  - Greedy: PIBT, LaCAM, LaCAM*
  - Others: MAPF-LNS2, BCP
- Majority plan full trajectories

May not be possible in applications with tight runtime requirements!

Windowed MAPF

- Robot Runners competition had a runtime cutoff of 1 second
- With severe time constraints, ***all MAPF works*** utilize Windowed planning
  - Only plan a partial collision free path
  - Execute certain # of steps
  - Replan
- Existing works:
  - Windowed Cooperative A*
  - RHCR
  - exPBS
  - Windowed Parallel PIBT-LNS (WPPL)
  - …

Pros and Cons of Windowed MAPF

Pros:

- Faster planning
- More adaptable to changes
- Enables decentralized planning

Cons:

- Theory: No solution guarantee
- Practice: Myopic reasoning = Deadlock / Livelock

Livelock

Deadlock

Contribution 1: Windowed MAPF with Guarantees!

- We design a framework for windowed MAPF with completeness!
    - **Win**dowed **C**omplete MAPF, or WinC-MAPF
- Theory: Completeness guarantees for certain windowed MAPF solvers
- Practice: Overcomes deadlock / livelock

Main ideas:

- Leverage heuristic updates from single-agent real-time search literature
- Leverage agent semi-independence from MAPF literature

Background: Real-Time Heuristic Search (RTHS)

- A problem formulation in single-agent planning where agents can only plan for a finite time (or horizon)
- Their key idea: Maintain completeness by updating the heuristic
    - Suppose at state S, we find S' which minimizes c(S,S') + h(S')
    - Update: h(S) = c(S,S') + h(S')
        - This heuristic update "penalizes" visited states
        - Heuristic penalty: $h(S) = h_{original}(S) + h_p(S)$
            - $h_p(S) = c(S,S') + h(S') - h_{original}(S)$
    - RTHS algorithms prove completeness via the update/penalty

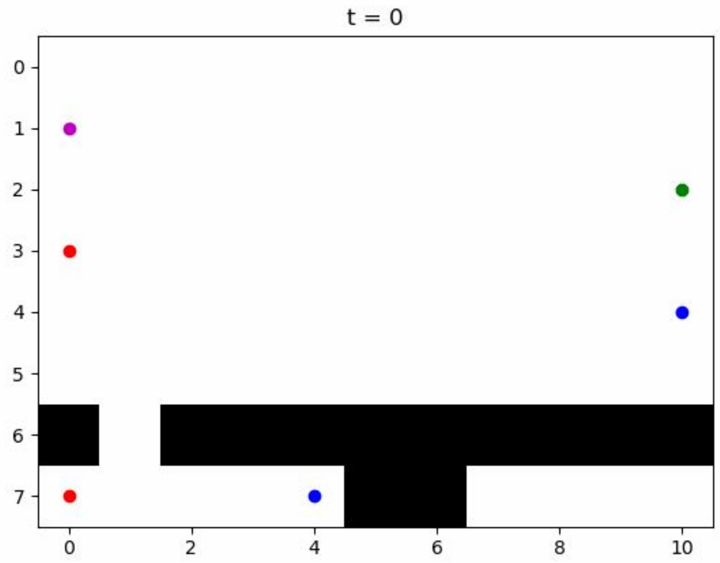
Our key idea: We can reuse this if we view MAPF in the joint configuration

Naive W-MAPF Framework

1. Query "Action Generator" to get limited horizon plan
   - S -> S_1 -> S_2 -> ..... -> S_last
   - AG minimizes c(S, S_last) + $h_{original}$(S_last) + h_p(S_last)
2. Update the heuristic using the standard RTHS update
   - h_p(S) = c(S,S_last) + h(S_last) - $h_{original}$(S)
3. Execute actions & repeat

Our key idea: View MAPF in the joint configuration

Problem: Heuristic Updates in Joint Space



Has to explore all joint configurations

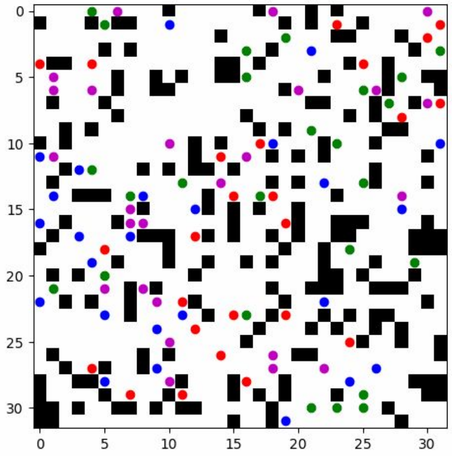
- Involves irrelevant agents!

Solution: Apply Heuristic Updates to Agent Groups

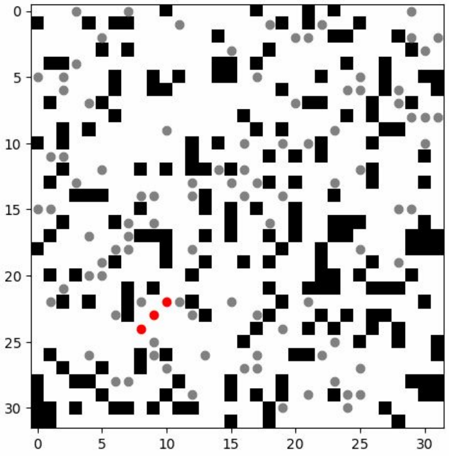
- All performant MAPF methods utilize agent semi-independence
  - I.e. if agents were far apart, they would plan independently
- Examples:
  - CBS: Plans agents independently, then uses conflicts
  - LNS2: Uses prioritized planning where agents plan independently conditioned on others
  - PIBT: Each agent acts greedily, avoids earlier agents
- Solution: Only apply heuristics to "coupled" agent groups
  - Ignore independent agents
  - Note: Paper discusses more formally "disjoint agent groups"

Only apply heuristic penalties to coupled agents

Visualizing Groups in Action



Current configuration



Dynamically Detected Disjoint Agent Groups

WinC-MAPF Framework

Algorithm 1: Windowed Complete MAPF Framework

procedure WINDOWED COMPLETE MAPF($\mathcal{C}^{cur}$)
    $\mathcal{H} \leftarrow \emptyset$ ▷ Heuristic Penalties
    while $\mathcal{C}^{cur} \neq$ Goal **do**
        $\mathcal{C}'$, ListOfGroups = AG($\mathcal{C}^{cur}$, $\mathcal{H}$)
        for $Gr \in$ ListOfGroups **do** ▷ For each group
            $h_{new} = c(\mathcal{C}^{cur}_{Gr}, \mathcal{C}'_{Gr}) + h(\mathcal{C}'_{Gr})$
            $penalty = h_{new} - h_{BD}(\mathcal{C}'_{Gr})$
            if $penalty > 0$ **then**
                $\mathcal{H}$.insert($\mathcal{C}^{cur}_{Gr}$, $penalty$)
        $\mathcal{C}^{cur} \leftarrow \mathcal{C}'$ ▷ Move agents

Key idea 1 for **completeness**: Apply heuristic updates on joint-space!

Key idea 2 for **efficiency**: Agent Groups!

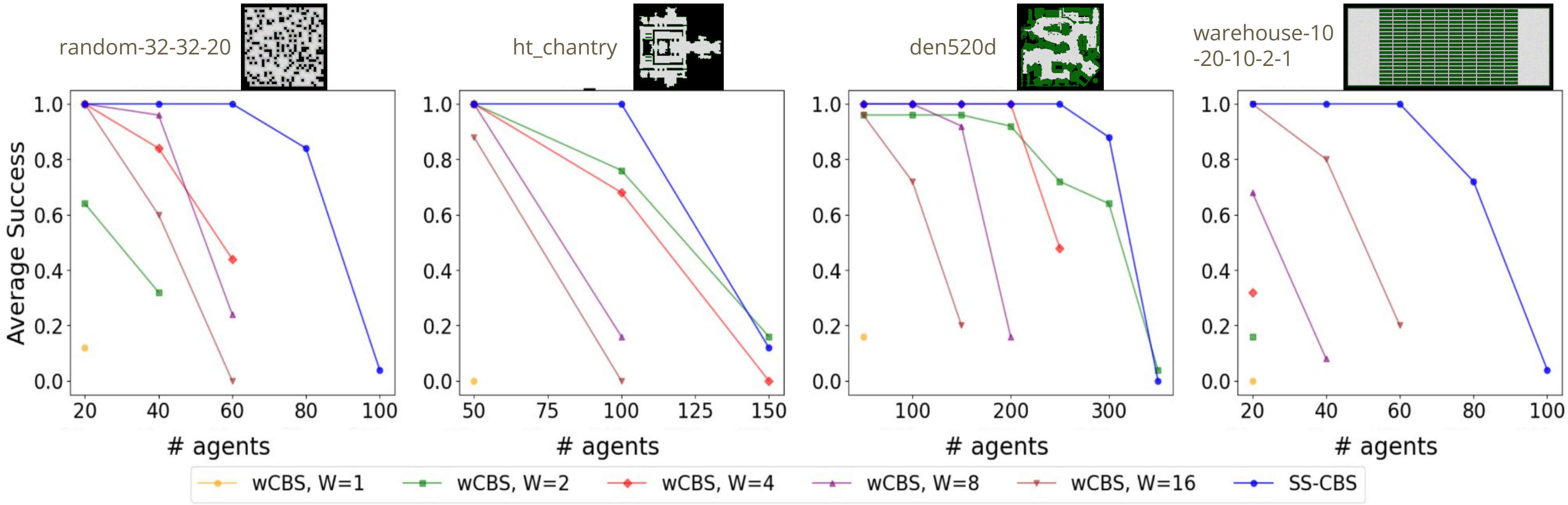
Contribution 2: Single-Step CBS

A concrete instantiation of our framework using CBS

- Detecting Disjoint Agent Groups
- Incorporating Heuristic Penalties

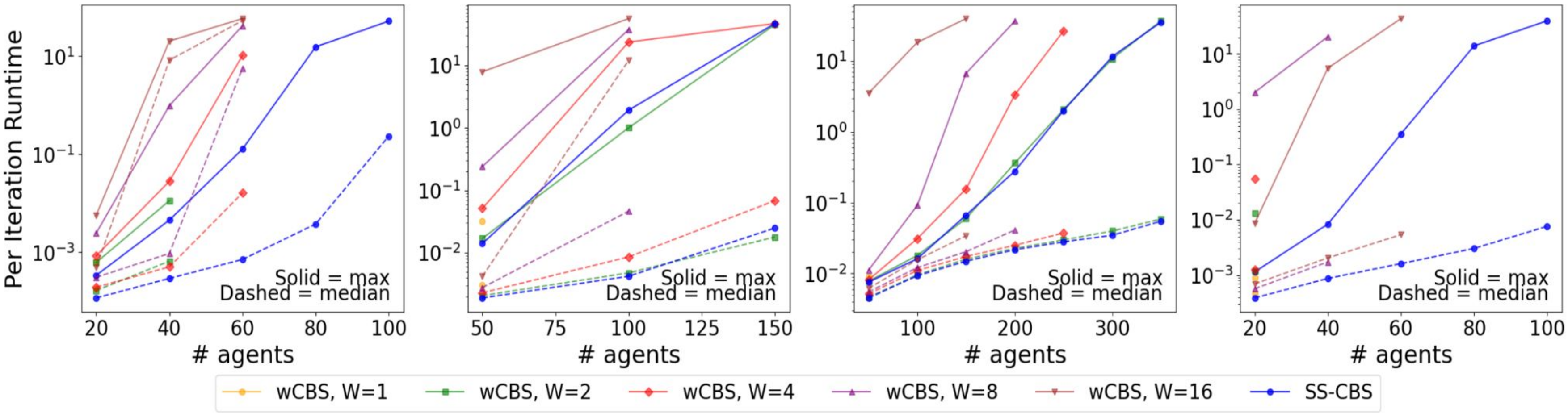
Results

Large Map Success Rate: SS-CBS vs Windowed CBS



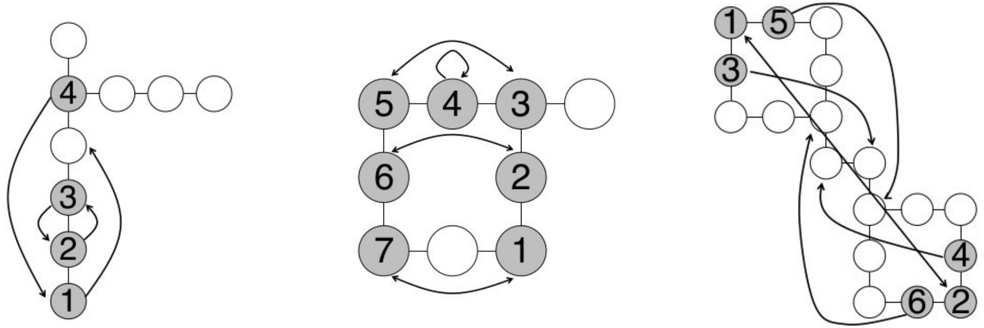
SS-CBS has a higher success rate than existing windowed CBS across multiple window sizes

SS-CBS Runtime



The median iteration is very fast, but certain iterations are bottlenecks

Small Tough Maps



Method	Horizon	Tunnel		Loopchain		Connector	
		3	4	6	7	5	6
wCBS	1,2,4,8,16	-	-	-	-	-	-
CBS+	∞	0.9	-	-	-	-	-
SS-CBS	1	1	1	1	0.95	1	1

+ SS-CBS can solve tough congestion that windowed approaches cannot

- Note: Solution cost is very poor (100s-1000s)

Takeaways

- Existing Windowed MAPF methods are necessary for runtime performance, but are incomplete
- Our WinC-MAPF Framework enables theoretical completeness
  - And practical improvements
- SS-CBS is an initial WinC-MAPF method
  - Better success rate in small and large instances
- Lots of future work!
  - Multi-step solvers
  - Faster or suboptimal solvers (requires proving completeness)
  - Incorporate single agent heuristic updates as well
  - Lifelong MAPF variants

This work opens up a new line of MAPF research!