2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

AnyGeometry-CBS: Any Geometry Conflict-Based Search for Multi-Agent Path Finding

Yichen Li[1], Xuebo Zhang[1], Jingjin Yu[2], Yaonan Wang[3]

***Abstract*— The Multi-Agent Path Finding (MAPF) problem seeks to find conflict-free paths for multiple agents. However, most existing MAPF methods simplify agents to points or uniform circles, a model that fails when agents have diverse geometries or carry oversized loads. This oversimplification can lead to undetected collisions or the failure to find feasible paths. To address this, we propose AnyGeometry-CBS (AG-CBS), a novel extension of Conflict-Based Search (CBS) that accommodates agents of arbitrary, non-convex shapes. AG-CBS represents each geometry of agent via a set of grid cells and introduces enriched conflict definitions to handle complex interactions. To improve search efficiency, we develop a Multi-Constraint (MC) technique and a Shape Heuristic (SH) for suboptimal variants. Experimental results demonstrate that our method reduces runtime by up to 84.43% against optimal baselines and 88.24% against bounded-suboptimal ones, providing a general and effective solution to complex MAPF problems.**

I. INTRODUCTION

In recent years, robotic automation has become ubiquitous across numerous industries. A key challenge in this domain is Multi-Agent Path Finding (MAPF), which aims to compute conflict-free paths for multiple agents from their start to goal configurations within a shared environment. MAPF has widespread applications, ranging from industrial automation and warehouse logistics [1], [2] to character navigation in video games [3], [4].

Most existing MAPF solvers simplify agents to points or a single grid cell, which is insufficient for modern robotic applications involving agents with diverse sizes and profiles. This challenge was formally defined as MAPF for Large Agents (LA-MAPF) by Li et al [5]. Their seminal work, LA-CBS, and its powerful enhancement, Multi-Constraint CBS, demonstrated that by adding multiple constraints at once, the search efficiency for large agents could be dramatically improved. The key limitation of LA-CBS is the assumption that agents cannot rotate and it primarily addresses regular shapes like rectangles. This assumption does not hold in many real-world scenarios where agents must rotate to navigate tight spaces or avoid collisions with each other.

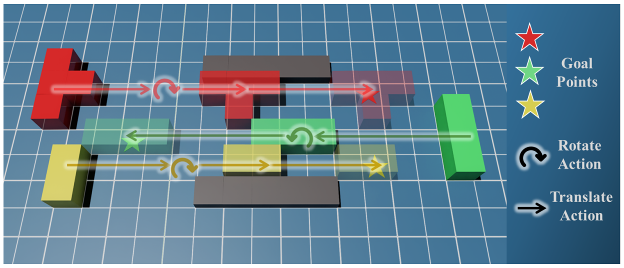


Fig. 1. Using AG-CBS as a solver for the MAPF problem, paths for agents of arbitrary geometric shapes can be derived.

We propose AnyGeometry-CBS (AG-CBS)[1] to address these gaps. Our method models agents by multiple grid cells that approximate their true physical profiles, including diverse and non-convex shapes, most importantly, integrates rotation movements and its conflict detection, as illustrated in Fig. 1. We build upon CBS, rather than methods like PIBT [6] or LaCAM [7], to retain the guarantee of optimality. The detailed geometric model significantly increases computational complexity. To mitigate this, we introduce two enhancements: 1) a Multi-Constraint (MC) method is designed to proactively resolve multiple potential conflicts at once, including the new rotation conflict, and 2) a Shape Heuristic (SH) for suboptimal variants like ECBS [8] that leverages geometric information to guide the search. Our approach provides a scalable framework applicable to other graph-based MAPF solvers.

In this paper, the main contributions are as follows:

- We introduce AnyGeometry-CBS, an optimality and complete framework that extends the LA-MAPF problem to include arbitrarily shaped, rotating agents, featuring a new class of shape conflict types.
- We design a Multi-Constraint (MC) method that proactively resolves sets of future shape conflicts to drastically reduce the search effort.
- We propose a Shape Heuristic (SH) that utilizes agent shape information to guide the search avoid from more complex shape conflicts.

II. RELATED WORK

The essence of the MAPF problem is to find a solution of paths for all agents, allowing them to move together without conflicts [9]. Finding an optimal solution to the MAPF problem is NP-hard [10]. Centralized MAPF methods, such as PBS [11], [12] or CBS [13], which use a centralized solver

---

*This work was supported in part by the NSFC (62293513/62293510). *(Corresponding author: Xuebo Zhang.)*

[1] Yichen Li and Xuebo Zhang are with the Tianjin Key Laboratory of Intelligent Robotics, the College of Artificial Intelligence, and the Institute of Robotics and Automatic Information System, Nankai University, Tianjin 300350, China `spike.li@mail.nankai.edu.cn`; `zhangxuebo@nankai.edu.cn`

[2] Jingjin Yu is with the Department of Computer Science, Rutgers, the State University of New Jersey, Piscataway, NJ, USA. `jingjin.yu@rutgers.edu`

[3] Yaonan Wang is with the College of Electrical and Information Engineering, Hunan University, Changsha 410082, China `yaonan@hnu.edu.cn`

[1] https://github.com/NKU-MobFly-Robotics/AnyGeometry-CBS.

979-8-3315-8160-2/26/$31.00 ©2026 IEEE

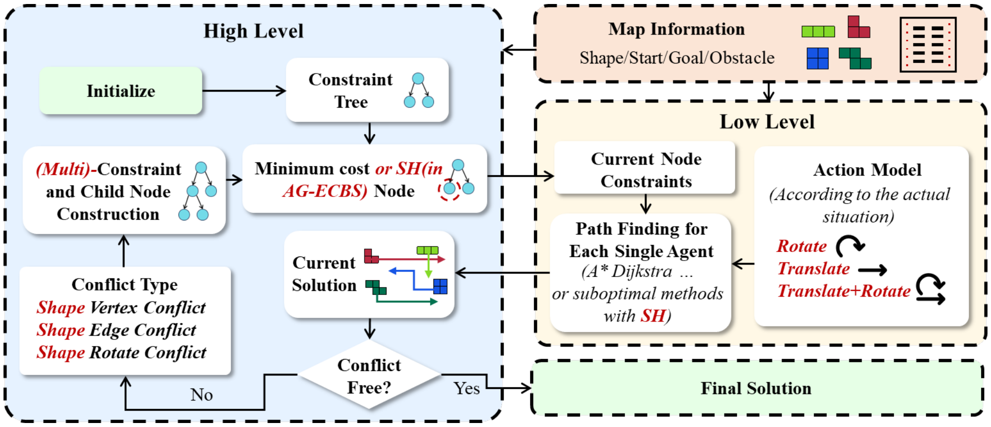


Fig. 2. An overview of the AG-CBS method framework, which is a hierarchical method. At the high level, it maintains a constraint tree and identifies conflicts within the solution. At the low level, it searches for paths for each agent under constraints.

to compute solutions, are commonly employed. Compared with decentralized methods [14], centralized methods can handle potential conflicts without requiring communication between robots but at the cost of increased computational demands on the solver. The CBS framework has been enhanced with various techniques, such as disjoint splitting [15], target assignment strategies [16], merging and restarting child nodes and conflict prioritization [17], cardinality-based heuristics [18], all aimed at improving search efficiency.

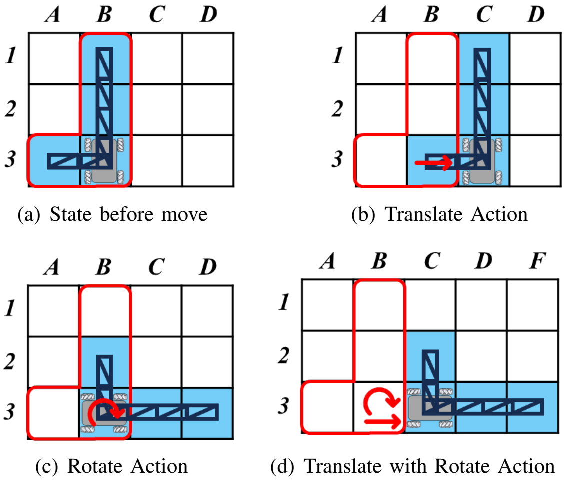
The search frameworks have also been extended to accommodate a wider range of application scenarios. Andreychuk et al. [19] extend CBS to continuous time (CCBS) while maintaining optimality. Yakovlev et al. [20] further develop CCBS into an any-angle version (AA-CCBS). The MO-CBS method is proposed to solve the MAPF problem with multiple objectives while ensuring search performance [21]. Some suboptimal variants significantly reduce the search time required to obtain an effective solution [22], [23].

Considering the shape of the robot can reduce the gap between the solutions of MAPF methods and the actual movement paths of robots and improve the effectiveness of the path. MAPF methods between heterogeneous robots have also been proposed, expanding the application scenarios of MAPF methods [24], [25].

III. PROBLEM STATEMENT

In AG-CBS, we model the working space as a 2-dimensional undirected grid graph $W = (V, E)$, where V represents all vertices and E represents edges of each four-connected grid. The set of obstacles is denoted by $\mathcal{O}$. All k agents are represented as $\{a_1, ..., a_k\}$, with their start points $S = \{s_1, ..., s_k\}$ and goal points $G = \{g_1, ..., g_k\}$.

Each agent occupies one or more vertices according to its actual shape. The center grid cell, representing the moving center of agent (usually the robot chassis center). Other shape grid cells (the load or profile of the robot) have fixed relative coordinates to the center grid. Fig. 3(a) shows an agent carrying a steel frame, with its moving center at $B3$, which is the center grid cell, and the steel frame occupies three additional grid cells at $B1$, $B2$, and $A3$, which are shape grid cells. When an agent reaches its goal, it means its center cell has reached the goal point vertex. Agents do not disappear after reaching their goals; they continue to occupy the grid cells defined by their shapes at the goal configurations.



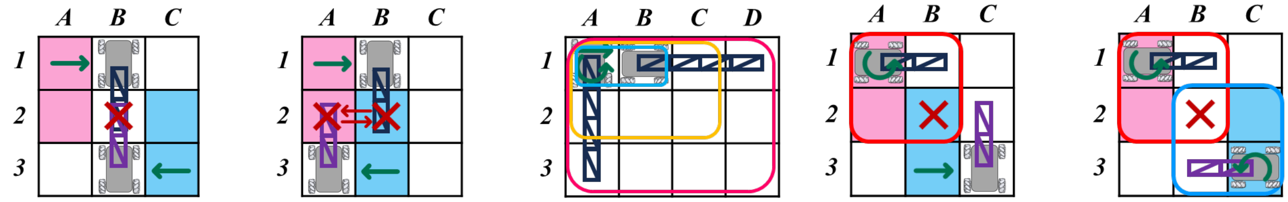
(a) State before move (b) Translate Action

(c) Rotate Action (d) Translate with Rotate Action

Fig. 3. Executable actions in AG-CBS. The red rounded polygon is the outline before move.(a) State before movement with robot moving center at $B3$, carrying a steel frame occupied $B1$ $B2$ and $A3$. (b) Agent translates one step to the right. (c) Rotates 90° clockwise around the center grid cell. (d) Translates one step to the right while rotating 90° clockwise simultaneously.

We discretize time into uniform time steps $\{t_0, ..., t_n\}$. In this paper, we take the McNamm wheel car model as an example, an agent can perform actions: translate, rotate, translate with rotation, or wait at each time step. The agent can rotate 90° clockwise or counterclockwise in one time step. Fig. 3(b)-3(d) illustrate these moving actions. Clearly, executable actions can be determined on the basis of the actual motion model of the robot such as differential wheel cars and Ackermann cars.

The state of agent a_i at time step t is represented as (v, o, t), where v means the vertex where the current center grid is located and $o \in \{N, S, W, E\}$ indicates the current orientation. In MAPF algorithms, conflicts between agents



(a) Shape Vertex Conflict (b) Shape Edge Conflict (c) AABB of Rotate Action (d) Shape Rotation Conflict A (e) Shape Rotation Conflict B

Fig. 4. The broadened conflict types and AABB in AG-CBS. (a) Shape vertex conflict at $B2$. (b) Shape edge conflict at edge $A2 - B2$. (c) The AABB of three grid cells generated during an action are represented by round-corner rectangles of different colors. (d) Shape rotation conflict and two AABB overlap at $B2$. (e) Shape rotation conflict and an AABB overlap with another agent at $B2$.

are of two types: vertex conflicts and edge conflicts. These lead to vertex and edge constraints. In AG-CBS, a vertex constraint for a_i is denoted by $\langle a_i, v, o, t\rangle$, prohibits the center grid of a_i from occupying vertex v with oriented towards o at time t. An edge constraint for a_i is denoted by $\langle a_i, v_1, v_2, o_1, o_2, t\rangle$, which means that the center grid of a_i cannot traverse the edge $v_1 - v_2$ with orientation from o_1 to o_2 at time t. Note that there is no requirement for the same orientation in our edge constraint, which allows agent rotation during edge traversal.

When constructing agent grid cells, the safety distance should be taken into account. Generally, all cells should be larger than the actual range of the robot like in Fig. 3(a).

The planning results are evaluated as the sum of cost (SOC), the total time taken by all the agent paths.

IV. ANY GEOMETRY CONFLICT-BASED SEARCH

Conflict-Based Search (CBS) is a two-level algorithm that finds optimal solutions to MAPF. The high level performs a best-first search on a constraint tree (CT), resolving conflicts between paths of agents one by one. The low level replans a path for a single agent subject to a new set of constraints.

AG-CBS is built upon the CBS method. However, conventional vertex and edge conflicts are insufficient to capture all interactions between complex shapes. We therefore broaden the definition of a conflict in AG-CBS. An overview of the AG-CBS framework is shown in Fig. 2.

A. Shape Conflicts and Constraints in AG-CBS

1) Shape Vertex Conflicts: In Fig. 4(a), at the $t-1$ time step, the center grid cell of the pink agent a_i is at $A1$ and the shape grid cell at $A2$, while the center grid cell of the blue agent a_j is at $C2$ and the shape grid cell at $C3$. When a_i translates one step to the right and a_j translates one step to the left, a shape vertex conflict occurs at $B2$. In the CT, the shape vertex constraint for a_i can be expressed as $\langle a_i, B1, o_i, t\rangle$ and for a_j as $\langle a_j, B3, o_j, t\rangle$. Even though the conflicting shape grid cell of a_i is not directly restricted, it is sufficient to restrict the center grid cell due to the fixed relative coordinates.

2) Shape Edge Conflicts: In Fig. 4(b), the shape grid cell of a_i translates from $A2$ to $B2$, while the center grid cell of a_j translates from $B2$ to $A2$, resulting in a shape edge conflict. The shape edge constraint for a_i is expressed as $\langle a_i, A1, A2, o_i, o_i, t\rangle$, and $\langle a_j, B2, B1, o_j, o_j, t\rangle$ for a_j.

3) Shape Rotation Conflicts: To detect rotation conflicts, we approximate the area swept by each cell of the agent during an action. Using Axis-Aligned Bounding Box (AABB) for this approximation to enable efficient conflict checks. As illustrated in Fig. 4(c), an agent performing a translate and rotate action in a step, generates a distinct AABB for each of its cells, representing their individual swept areas.

When one agent rotates and the other translates, as shown in Fig. 4(d) a_i rotates 90° counterclockwise around its center grid cell at $A1$. The AABB formed by with the shape grid cell of a_i is shown as the red rounded rectangle. Even if a_j translates to the right in the next step, a shape rotation conflict will occur at $B2$. In such cases, asymmetric constraints need to be applied to them, which means a_i requires a constraint that restricts its rotation, and a_j requires a constraint that restricts the vertex it is in. The shape edge constraint for a_i is expressed as $\langle a_i, A1, A1, o_{i1}, o_{i2}, t\rangle$. Although a_i does not translate, this can still be represented as a shape edge constraint. Meanwhile, the shape vertex constraint for a_j is expressed as $\langle a_j, B2, o_j, t\rangle$.

If both agents rotate, as shown in Fig. 4(e), two shape grid cell bounding boxes of them overlap at $B2$. Consequently, symmetric shape edge constraints are imposed on both agents. The shape edge constraint for a_i is $\langle a_i, A1, A1, o_{i1}, o_{i2}, t\rangle$ and $\langle a_j, C2, C2, o_{j1}, o_{j2}, t\rangle$ for a_j.

We use AABB for its computational efficiency in conflict checking, though it can be a conservative approximation. Future work could explore tighter yet still efficient bounding volumes like Oriented Bounding Box (OBB) or convex hull to potentially improve search performance.

B. Low Level Path Finding in AG-CBS

In low level, it will find a new path of for each agent according to its constraints and action model. It is necessary to check not only whether the center grid cell is passable for each move, but also the shape grid cells, so that it can ensure that the whole agent is passable to next location. Except for the changes in the search space of available actions, the low level search does not affect the optimality and completeness.

C. Optimality and Completeness of AG-CBS

In discrete and bounded grid maps, AG-CBS maintains the optimality and completeness from CBS, but this is difficult to guarantee in continuous space. Theorem can be given and the reasons follow.

Theorem 1. *AG-CBS maintains optimality and completeness in a discrete and bounded grid map.*

1) Optimality: The high level search of AG-CBS performs a best-first search on a CT, always expanding the node with the minimum SOC. The cost of any node is a lower bound on the cost of any valid solution that can be derived from it. When a conflict is resolved, the algorithm branches into two child nodes, each inheriting constraints from the parent plus one or more new prohibitive constraints for one of the conflicting agents. This ensures that the child node cost is always greater than or equal to its parent node cost. Since this branching rule covers all possibilities for resolving the conflict, no possible optimal solution is pruned. Therefore, the first conflict-free solution found by the best-first search is guaranteed to be optimal.

2) Completeness: The state of any agent (position and orientation) is discrete and the map is bounded. For any finite solution cost, the number of possible states and potential conflicts is finite. AG-CBS guarantees that any collision will be detected through its expanded conflict definitions. Each high level expansion resolves at least one conflict, and that conflict will not reappear in its child nodes of the CT. As the search space is reduced, the search must terminate in a finite number of steps, either by finding a solution if one exists or running out all the search space and stopping.

V. MULTI-CONSTRAINT IN AG-CBS

When a conflict occurs, two child nodes will be expanded, due to the optimality, the agent always prefers the shortest path and may reach this location soon with a different orientation or state, they may have conflict many times near the same conflict vertex or edge, generating numerous redundant nodes in the constraint tree and requiring costly path recalculations at each node.

To improve the situation, we improved Multi-Constraint (MC) method to adapt to AG-CBS, which adds multiple constraints to nodes that may have conflicts in the future. Due to this method, the number of CT nodes is significantly reduced and the discovery of the goal solution is accelerated.

A. MC State Selection Rules

In Fig. 5(a), a_i and a_j have a conflict at $B2$ in node N, then two child nodes N_i and N_j are generated. In N_j, the constraint of the conflicting state of a_j in Fig. 5(a) is added, and all other states that a_j occupies $B2$ at the same time step (Figs. 5(b)-(h)) are identified and added to the shape vertex constraints of a_j in N_j, totaling eight shape vertex constraints. Similarly, in the child node N_i, the constraints on a_i are performed like before. As for the shape edge conflict, the MC follows the same process steps.

For a shape rotation conflict where both agents rotate as in Fig. 4(e), identify states where each agent rotates or rotates while translating, resulting in bounding boxes covering $B2$, and add these to shape edge constraints of each agent.

In summary, the rules of MC in AG-CBS are as follows:

1) Only search for the same conflict type.
2) Search for MC only at the same time step.
3) Only add MC of a_i in child node N_i and add MC of a_j in child node N_j.

Algorithm 1 Forming MC for AG-CBS

```
    Input: conflict vertex $v_0$, state $(v_i, o_i, t)$, state $(v_j, o_j, t)$, node $N$
    Output: child node $N_i$ and $N_j$
 1: create child node $N_i$, $N_j$ from $N$
 2: $N_i.constraint \leftarrow N.constraint$
 3: $N_j.constraint \leftarrow N.constraint$
 4: $MC_i \leftarrow \text{GETMC}(v_0, a_i, t)$
 5: $MC_j \leftarrow \text{GETMC}(v_0, a_j, t)$
 6: $N_i.constraint.add(MC_i)$
 7: $N_j.constraint.add(MC_j)$
 8: return $N_i, N_j$
 9: function GETMC(agent $a_i$, vertex $v_0$, time step $t$)
10:     $n \leftarrow$ number of grid cells of $a_i$
11:     $m \leftarrow$ number of orientation
12:     for $p = 1$ to $n$ do
13:         for $q = 1$ to $m$ do
14:             $c \leftarrow \text{GETSTATE}(p, q, a_i)$
15:             $MC_i.add(c)$
16:     return $MC_i$
```

First, some states found by the MC might be unreachable due to exceeding the range that can be reached at this time step. But we do not distinguish these states further, as it does not impact the low level path finding, only takes extra time.

Second, only search for the same shape conflict type and same time step due to efficiency and time concerns. Searching for all types of possible shape conflict would yield many results and reduce search efficiency. Limiting the search to the same time step for the same reason.

Then, like the MC-CBS in [5], the MC in AG-CBS is also *mutually disjunctive* because we add MC for the two agents respectively in the two child nodes, and a conflict free path will not simultaneously violate two constraints from two different branch nodes.

The pseudo-code for MC is presented by Alg. 1, taking shape vertex conflict as an example. The process begins by creating two child nodes from the current node N, and inheriting constraints from N (Lines 1-3). Then, it generates MC for each child node (Lines 4-5). For a_i, there are n grid cells and m orientations (four in this paper) (Lines 10-11), then gets all the states of a_i that occupy v_0 and adds them to the MC (Lines 12-16). Finally, N_i and N_j incorporate the MC into their constraints sets and return (Lines 6-8).

B. Optimality and Completeness for AG-CBS with MC

The node adds the multi-constraint sets to its constraints is akin to pre-merging its child nodes. Conflicts that may occur in the future will be found and avoided in advance. The cost of the solution remains minimal, the heuristic function is not affected, and the search space will be checked more quickly. These factors underpin the proposition on MC in AG-CBS.

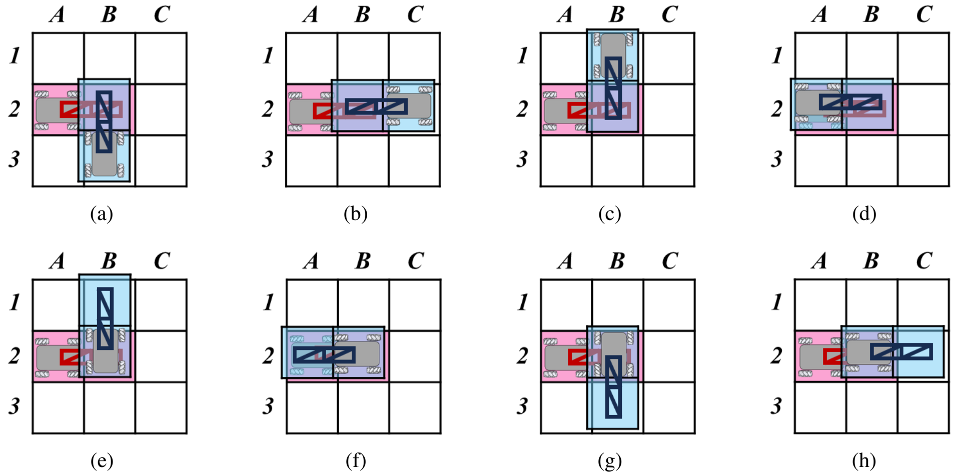
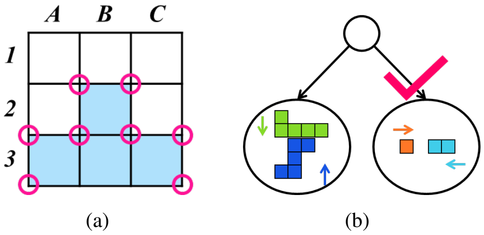


Fig. 5. MC method in AG-CBS. (a) A shape vertex conflict is detected between the pink agent a_i and the blue agents a_j at $B2$. (b)-(h) Check all other states of a_j that may cause same conflicts with a_i again and add them to the constraints of a_j. The two agents are slightly staggered for clarity.



(a) (b)

Fig. 6. Shape Heuristic in AG-ECBS, (a) Eight corner vertices of a_i are marked by pink circles, and a_i occupies 4 grid cells, namely $C(i)$=8, $K(i)$=4. (b) In focal list, there are two nodes with conflicts, SH will prioritize choosing the node with the less complex shape conflicts(the light blue and orange agents).

Proposition 1. *Incorporation of the MC method into AG-CBS enhances search efficiency while guaranteeing preservation of optimality and completeness.*

VI. SHAPE HEURISTIC FOR BOUNDED SUBOPTIMAL VARIANT

In some real-time planning scenarios, making all the robots start moving as soon as possible, i.e. getting a valid and acceptable solution is more important than pursuing the optimal solution. In this case, the optimality requirement could be relaxed slightly. Therefore, we introduce ECBS, a suboptimal variant of CBS, into AG-CBS, effectively balancing the speed and quality of the solution. This makes AG-ECBS more suitable for time-sensitive applications.

We also propose shape heuristic (SH) as a focal heuristic, which avoids conflicts based on the shape information.

A. Enhanced-CBS

ECBS is a bounded suboptimal variant of CBS that uses a focal list to accelerate search. It maintains a list of nodes whose costs are within a factor w of the minimum cost and uses a secondary focal heuristic to select the most promising node from this list, often one with fewer or simpler conflicts.

B. Shape Heuristic

When two agents conflict, those with simple shapes and fewer grid cells are less likely to reconflict in child nodes, while complex shapes with more grid cells are not. The SH evaluates the shape complexity of conflicting agents, serves as a suitable focal heuristic function and is calculated as:

$$sh(i) = C(i) + K(i), \tag{1}$$

$$SH(i,j) = sh(i) + sh(j), \tag{2}$$

where $C(i)$ is the number of corner vertices of the profile of a_i, $K(i)$ is the grid cell number of a_i as shown in Fig. 6(a), $sh(i)$ is the shape heuristic of a_i. $SH(i,j)$ represents the shape heuristic for conflicting agents a_i and a_j. A larger $K(i)$ means that it occupies more grid cells and a more dispersed shape results in a larger $C(i)$.

This linear combination (Eq. 1) is an empirical choice that intuitively captures shape complexity from both its boundary and area. We leave the exploration of more complex heuristic formulations for future work.

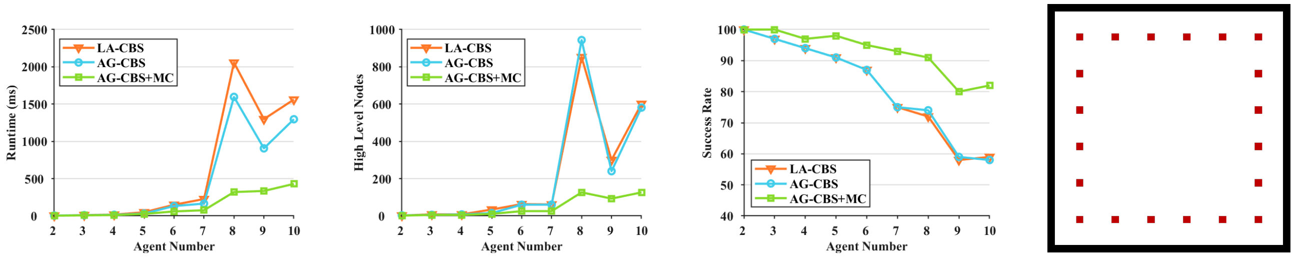
For a node N in AG-ECBS, the shape heuristic is the sum of $SH(i,j)$ for each conflicting agent pair:

$$SH(N) = \sum SH(i,j), \tag{3}$$

similar to other focal heuristics, the node with the lowest SH value in the focal list is expanded first, because conflicts between simple shape agents are easier to find alternative paths than complex shapes, as shown in Fig. 6(b). SH can be applied simultaneously to both high and low level searches.

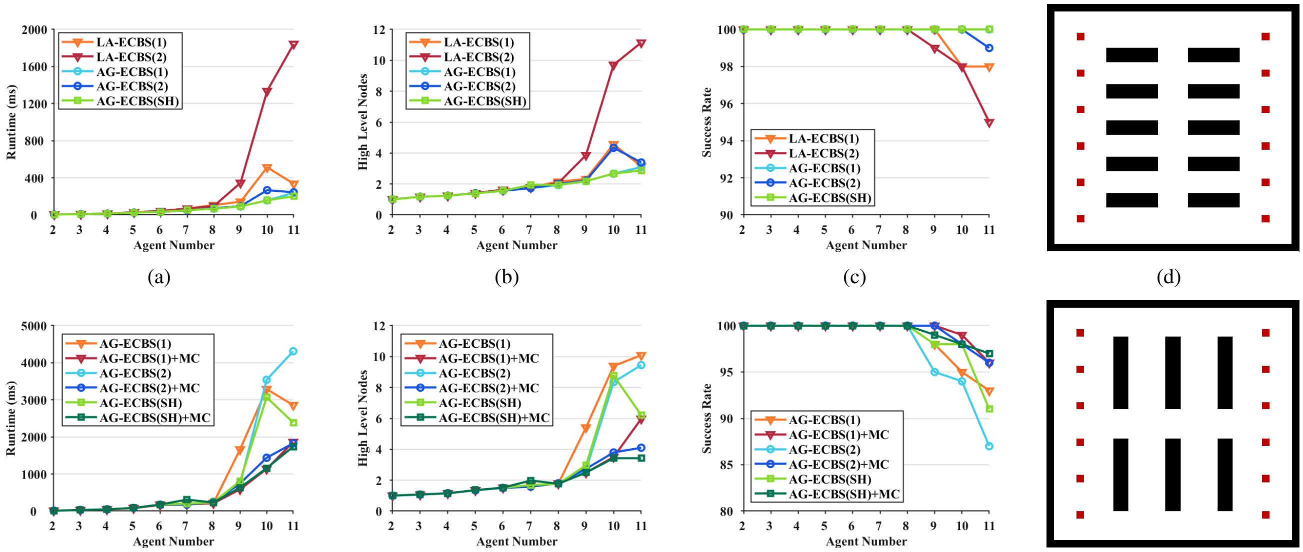
VII. EXPERIMENTAL RESULTS

LA-CBS and its variants are implemented to compare with AG-CBS and its variants. For fairness, the same rotation action and collision detection are added to LA-CBS. Algorithms are implemented in C++, run on a laptop with



(a) (b) (c) (d)

Fig. 7. Performance comparison of LA-CBS, AG-CBS, and AG-CBS+MC on a 32×32 empty grid map, the red grid cells in (d) indicate start/goal points, (a) shows average runtime across different agent numbers, (b) shows average high level nodes, (c) shows success rates under a 100 s time limit.



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 8. Evaluation of LA-ECBS and AG-ECBS with MC across different focal heuristics, (a) and (e) average runtime difference, (b) and (f) average high level nodes difference, (c) and (g) success rate difference between methods and (d) warehouse-1 map, (h) warehouse-2 map.

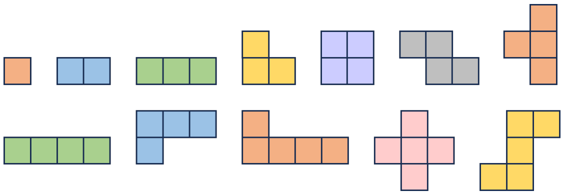


Fig. 9. Various shapes available for agents.

an Intel i9-13900K 3.0GHz CPU and 16GB RAM. Three 32×32 size maps (empty space, warehouse-1 and warehouse-2) are designed to simulate high-density environments. We intentionally chose compact maps to focus the evaluation on the complexity of shape conflicts resolution rather than low level path finding over long distances. For each test, 100 instances are randomly generated with a 100 s runtime limit.

A. Optimal Methods

We compare three optimal methods LA-CBS, AG-CBS and AG-CBS with MC on empty grid map. All start and goal points are randomly selected from the red points in Fig. 7(d) with no overlapping. The geometric shape of the agents is chosen from the first ten in Fig. 9.

The average runtime, high level nodes, and success rate are shown in Fig. 7(a)-(c). Compared to LA-CBS, AG-CBS achieves a 30.21% reduction in average runtime and a 20.0% reduction in average high level nodes at nine agents. AG-CBS with MC further decreases the average runtime by up to 84.43% and reduces high level nodes by up to 85.17% at eight agents. The runtime of LA-CBS and AG-CBS fluctuates when the number of agents is between 8 and 10. This may be due to the varying difficulty of individual instances, leading to a longer average runtime.

B. Suboptimal Methods

Bounded suboptimal methods are compared on two warehouse-like maps (Fig. 8(d), (h)) with suboptimal factor $w = 1.25$. Choosing start and goal points on opposite sides for agents, selecting shapes from Fig. 9 without repetition.

Except SH, two other focal heuristics are applied on both high and low level: ECBS(1) is the number of conflicts in a node, and ECBS(2) is the number of conflicting agents.

In warehouse-1, four suboptimal methods are compared. As shown in Fig. 8(a)-(c), for 2 to 8 agents, they show similar performance. When the number of agents is 9 to 11, the average runtime and high level nodes of LA-ECBS(2) has increased dramatically, with AG-ECBS(SH) having the lowest runtime. At 10 agents, AG-ECBS(SH) reduces the runtime by 69.37% compared to LA-ECBS(1) and by 88.24% compared to LA-ECBS(2). AG-ECBS(SH) achieve a success rate of 100%.

To assess the performance differences of the three focal heuristics and MC on AG-ECBS, we evaluate them in warehouse-2 which is harder to get through. As shown in Fig. 8(e)-(g), the runtimes of AG-ECBS(SH) and AG-ECBS(SH)+MC are slightly lower than the other two focal heuristics overall. The high level nodes of AG-ECBS(SH) and AG-ECBS(SH)+MC are relatively lower in Fig. 8(f). The success rate of AG-ECBS(SH) is higher than AG-ECBS(1) and (2) when the number of agents is 10.

To ensure the performance gains of SH will not compromise solution quality, we also compare the average SOC. Across all suboptimal variants, the average path costs are nearly identical. When normalized against the AG-ECBS(1) baseline, the average SOC for each method varied by less than 0.6% on the warehouse-1 map and by less than 0.2% on the warehouse-2 map. Therefore, we conclude that SH provides a substantial improvement in search efficiency without any meaningful sacrifice in solution quality.

VIII. CONCLUSION AND FUTURE WORK

In this paper, we propose AG-CBS, an MAPF method for agents of any geometry. We introduce an MC method that incorporates future conflict constraints based on agent shape. For the bounded suboptimal variant AG-ECBS, we propose SH based on shape information. Experimental results demonstrate that AG-CBS outperforms LA-CBS. Incorporating the MC method significantly improves the success rate and reduces runtime. Meanwhile, the SH improves the performance of runtime and high level nodes, exhibits considerable potential for further enhancement.

In future work, we will explore more efficient MAPF methods for agents of any geometric shape, such as those involving continuous time or continuous angle space, and further refine the definition and resolution of conflicts.

REFERENCES

[1] Y. Hua, Y. Wang, and Z. Ji, "Adaptive lifelong multi-agent path finding with multiple priorities," *IEEE Robotics and Automation Letters*, vol. 9, no. 6, pp. 5607–5614, 2024.

[2] W. Hönig, S. Kiesel, A. Tinka, J. W. Durham, and N. Ayanian, "Persistent and robust execution of mapf schedules in warehouses," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1125–1131, 2019.

[3] X. Hong, X. Hu, and Y. Lin, "Three real-time pathfinding strategies in games: From A* ideas to machine learning approaches," in *2024 3rd International Conference on Artificial Intelligence and Software Engineering (ICAISE)*, 2024, pp. 78–89.

[4] H. Ma, J. Yang, L. Cohen, T. Kumar, and S. Koenig, "Feasibility study: Moving non-homogeneous teams in congested video game environments," in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 13, no. 1, 2017, pp. 270–272.

[5] J. Li, P. Surynek, A. Felner, H. Ma, T. S. Kumar, and S. Koenig, "Multi-agent path finding for large agents," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 7627–7634.

[6] K. Okumura, M. Machida, X. Défago, and Y. Tamura, "Priority inheritance with backtracking for iterative multi-agent path finding," *Artificial Intelligence*, vol. 310, p. 103752, 2022.

[7] K. Okumura, "LaCAM: Search-based algorithm for quick multi-agent pathfinding," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 10, 2023, pp. 11 655–11 662.

[8] M. Barer, G. Sharon, R. Stern, and A. Felner, "Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem," in *Proceedings of the international symposium on combinatorial Search*, vol. 5, no. 1, 2014, pp. 19–27.

[9] R. Stern, N. Sturtevant, A. Felner, S. Koenig, H. Ma, T. Walker, J. Li, D. Atzmon, L. Cohen, T. Kumar *et al.*, "Multi-agent pathfinding: Definitions, variants, and benchmarks," in *Proceedings of the International Symposium on Combinatorial Search*, vol. 10, no. 1, 2019, pp. 151–158.

[10] J. Yu and S. LaValle, "Structure and intractability of optimal multi-robot path planning on graphs," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 27, no. 1, 2013, pp. 1443–1449.

[11] X. Zhang, G. Xiong, Y. Wang, S. Teng, and L. Chen, "D-PBS: Dueling priority-based search for multiple nonholonomic robots motion planning in congested environments," *IEEE Robotics and Automation Letters*, vol. 9, no. 7, pp. 6288–6295, 2024.

[12] S.-H. Chan, R. Stern, A. Felner, and S. Koenig, "Greedy priority-based search for suboptimal multi-agent path finding," in *Proceedings of the International Symposium on Combinatorial Search*, vol. 16, no. 1, 2023, pp. 11–19.

[13] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant, "Conflict-based search for optimal multi-agent pathfinding," *Artificial intelligence*, vol. 219, pp. 40–66, 2015.

[14] T. Guo and J. Yu, "Decentralized lifelong path planning for multiple ackerman car-like robots," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 7041–7047.

[15] J. Li, D. Harabor, P. J. Stuckey, A. Felner, H. Ma, and S. Koenig, "Disjoint splitting for multi-agent path finding with conflict-based search," in *Proceedings of the international conference on automated planning and scheduling*, vol. 29, 2019, pp. 279–283.

[16] Z. Ren, S. Rathinam, and H. Choset, "CBSS: A new approach for multiagent combinatorial path finding," *IEEE Transactions on Robotics*, vol. 39, no. 4, pp. 2669–2683, 2023.

[17] E. Boyarski, A. Felner, R. Stern, G. Sharon, O. Betzalel, D. Tolpin, and E. Shimony, "ICBS: The improved conflict-based search algorithm for multi-agent pathfinding," in *Proceedings of the International Symposium on Combinatorial Search*, vol. 6, no. 1, 2015, pp. 223–225.

[18] A. Felner, J. Li, E. Boyarski, H. Ma, L. Cohen, T. S. Kumar, and S. Koenig, "Adding heuristics to conflict-based search for multi-agent path finding," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 28, 2018, pp. 83–87.

[19] A. Andreychuk, K. Yakovlev, E. Boyarski, and R. Stern, "Improving continuous-time conflict based search," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 13, 2021, pp. 11 220–11 227.

[20] K. Yakovlev, A. Andreychuk, and R. Stern, "Optimal and bounded suboptimal any-angle multi-agent pathfinding," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 7996–8001.

[21] Z. Ren, S. Rathinam, and H. Choset, "A conflict-based search framework for multiobjective multiagent path finding," *IEEE Transactions on Automation Science and Engineering*, vol. 20, no. 2, pp. 1262–1274, 2022.

[22] Z. Pan, R. Wang, Q. Bi, X. Zhang, and J. Yu, "RH-ECBS: enhanced conflict-based search for MRPP with region heuristics," *Robotica*, pp. 1–11, 2024.

[23] J. Li, W. Ruml, and S. Koenig, "EECBS: A bounded-suboptimal search for multi-agent path finding," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 14, 2021, pp. 12 353–12 362.

[24] Y. Bai, S. Kotpalliwar, C. Kanellakis, and G. Nikolakopoulos, "Multi-agent path planning based on conflict-based search (CBS) variations for heterogeneous robots," *Journal of Intelligent & Robotic Systems*, vol. 111, no. 1, p. 26, 2025.

[25] W. Lin, W. Song, Q. Zhu, and S. Zhu, "Multi-agent path finding with heterogeneous geometric and kinematic constraints in continuous space," *IEEE Robotics and Automation Letters*, vol. 10, no. 1, pp. 492–499, 2024.