Representation-Optimal Multi-Robot Motion Planning Using Conflict-Based Search

Irving Solis, James Motes, Read Sandström, and Nancy M. Amato

***Abstract*—Multi-Agent Motion Planning (MAMP) is the problem of computing feasible paths for a set of agents each with individual start and goal states within a continuous state space. Existing approaches can be split into coupled methods which provide optimal solutions but struggle with scalability or decoupled methods which provide scalable solutions but offer no optimality guarantees. Recent work has explored hybrid approaches that leverage the advantages of both coupled and decoupled approaches in an easier discrete subproblem, Multi-Agent Pathfinding (MAPF). In this work, we adapt recent developments in hybrid MAPF to the continuous domain of MAMP. We demonstrate the scalability of our method to manage groups of up to 32 agents, demonstrate the ability to handle up to 8 high-DOF manipulators, and plan for heterogeneous teams. In all scenarios, our approach plans significantly faster while providing higher quality solutions.**

***Index Terms*—Path planning for multiple mobile robots or agents, motion and path planning.**

I. Introduction

In AUTOMATED manufacturing, high degree-of-freedom (DOF) manipulators working in tight coordination must avoid colliding with each other while planning efficient motions. Heterogenous teams used on construction sites or in search-and-rescue missions must similarly coordinate collision free motions. Video games with large multi-agent teams must also produce feasible plans. These are only a few examples of the multi-agent motion planning (MAMP) problem.

Given the complexity of MAMP, most of the research related to multi-agent systems has focused on solving an easier subproblem, multi-agent pathfinding (MAPF). The pathfinding problem operates in a discrete state space as opposed to the continuous space considered in the motion planning problem. The assumption of shared state representations for all agents and uniformity of state transition duration often made by MAPF methods prevent them from being directly applied to the continuous space MAMP domain. Thus many real world problems, such as the high DOF manipulator and heterogenous multi-agent teams, cannot be solved by discrete MAPF techniques as the agents depend on a unique world representation. Some MAMP methods, such as an environment grid-discretization [1], map their motions to a common environment representation to handle the disjoint agent's state-space representation. Unfortunately, the solution quality of these types of approaches is heavily dependent upon the environment representation.

There are two standard approaches to MAMP/PF: coupled and decoupled. Coupled methods can provide optimal solutions but search over the joint state space. Decoupled methods search independently over the individual state spaces of each robot but cannot guarantee completeness or optimality as they explore individual agent state spaces in isolation before combining them later. Hybrid approaches try to leverage the benefits of both approaches.

In this work, we present an efficient and scalable MAMP solver that generalizes a recent efficient and optimal MAPF technique, Conflict-Based Search (CBS) [2], to continuous state spaces. We show our new MAMP method, CBS-MP, offers a significant improvement over state-of-the-art MAMP solvers both in terms of planning time and solution quality. We validate our approach against both standard coupled and decoupled Probabilistic Roadmap (PRM) variants as well as a detailed comparison to ECBS-MP, another CBS extension to solve MAMP problems, specifically state-lattice problems.

The results show that our proposed approach has improved performance for a variety of scenarios. We show an increased scalability, planning for up to 32 agents, while the PRM variants were unable to plan for 16 agents, and ECBS-CT was only able to handle some 16 agent for some trials. We planned for up to 8 high DOF manipulators over 60 times faster on average than the fastest PRM variant run. Additionally, we demonstrate the flexibility of the approach on a heterogeneous problem with a manipulator, mobile robot, and aerial robot operating in the same workspace. In addition to significantly improved planning times, we show lower cost solutions than both PRM variants. Our contributions are as follows:

- a uniform time discretization process to efficiently extend CBS to sampling-based motion planning,
- an optimal MAMP method which improves performance and scalability over existing methods,
- theoretical analysis of the quality of the technique, and

Manuscript received October 15, 2020; accepted February 25, 2021. Date of publication March 25, 2021; date of current version April 13, 2021. This letter was recommended for publication by Associate Editor B. Wang and Editor M. A. Hsieh upon evaluation of the reviewers' comments. This work was supported in part by CONACYT. *(Corresponding author: James Motes.)*

Irving Solis is with the Department of Computer Science and Engineering, Texas A&M University, College Station, TX 77840 USA (e-mail: irvingsolis-89@tamu.edu.).

James Motes and Nancy M. Amato are with the Department of Computer Science, University of Illinois Urbana-Champaign, Urbana, IL 61801 USA (e-mail: jmotes2@illinois.edu; namato@illinois.edu).

Read Sandström is with the iRobot Corporation, (e-mail: rsandstrom@fastmail.com).

Digital Object Identifier 10.1109/LRA.2021.3068910

TABLE I
A BRIEF OVERVIEW OF RELATED METHODS

Algorithm	MAPF/MAMP	Coordination	Optimal	State-Representation	Scalability
Composite-A* [3]	MAPF	Coupled	X	grid	
Decoupled-A* [4]	MAPF	Decoupled		roadmap	X
CBS [2]	MAPF	Hybrid	X	grid	X
MA-CBS [5]	MAPF	Hybrid	X	grid	X
ECBS [6]	MAPF	Hybrid	X	grid	X
M* [7]	MAPF	Hybrid	X	grid	X
MRdRRT [8]	MAMP	Coupled	X	composite roadmap	
Composite-PRM [9]	MAMP	Coupled	X	composite roadmap	
Decoupled-PRM [9]	MAMP	Decoupled		roadmap	
MRP-IC [10]	MAMP	Decoupled		composite roadmap	
ECBS-CT [1]	MAMP	Hybrid	X	state-lattice	X
CBS-MP	MAMP	Hybrid	X	roadmap	X

- experimental validation on mobile, high-dof and heterogeneous multi-robot systems.

II. RELATED WORK

In this section, we examine relevant work to the multi-agent pathfinding and motion planning domains. An overview of the state-of-the-art is shown in Table I.

There are two problem classes: Multi-Agent Pathfinding (MAPF) and Multi-Agent Motion Planning (MAMP). MAPF problems are defined by a set of agents, a graph, and a start and goal location for each agent. A solution consists of a set of collision-free paths on the graph, moving each agent from the start to the goal. Solution quality is traditionally measured by *sum-of-costs* (summed cost of all paths) or *makespan* (maximum individual path). An optimal solution minimizes the desired metric. MAMP is the problem of finding a feasible path between a start and goal for each agent in a continuous state space that is usually intractable to represent explicitly. This is a superset of the pathfinding problem that searches over a discretized state space.

Methods can be classified into three categories: coupled, decoupled, and hybrid. In coupled approaches, all agent paths are computed in unison. These approaches work in the joint space of all agent states. They tend to provide stronger guarantees on feasible paths and minimum cost by exploring the joint space. However, they have a high computational cost as the dimensionality of the joint C-space increases with the number of robots and their degrees of freedom.

computational cost, attention has turned to decoupled algorithms. Instead of planning all paths in unison, each path is planned individually. Once all paths are computed, they are adjusted according to defined priorities to avoid inter-agent collisions. Decoupled approaches work in single-agent spaces allowing to rapidly compute feasible paths for problems with a large number of agents. However, individual agent state spaces are explored in isolation, and later solutions are combined. This prevents ensuring completeness and optimality, and there is often the possibility of failing to find feasible paths on solvable problems.

Due to the tradeoff between faster computation times and finding optimal cost solutions, researchers explored new ways of leveraging the strengths of both coupled and decoupled techniques. These techniques are known as hybrid approaches. The method we present, CBS-MP, is a hybrid approach to MAMP problems. This approach leverages the techniques developed in the MAPF space to create an optimal and scalable solution for the continuous space MAMP.

A. Multi-Agent Pathfinding

Generally, all MAPF methods can be classified into three major groups: coupled, decoupled, and hybrid.

1) Coupled Approaches: A simple MAPF solver can be implemented by concatenating all the single agent states into a joint state and then using a generic search algorithm like A* to traverse the joint space to get to the joint state solution [3]. Unfortunately,the number of states grows exponentially with respect to the number of agents. As a result, these approaches are only practical for a small number of agents.

To improve and speed up the search over the joint space, other methods attempt to prune the search by using some heuristics and expand fewer nodes than regular A* [11]–[13]. Alternative approaches have modeled the multi-agent pathfinding problem as Integer Linear Programming (ILP) and Boolean Satisfiability (SAT) problems. In [14], [15] the problem is mapped to a network-flow and apply ILP algorithms to solve MAPF instances optimally. However, all these techniques are still susceptible to increased computational cost as the number of robot increases or even a few robots with high DOF as they also plan in joint state spaces.

2) Decoupled Approaches: To achieve group coordination of multiple robots, decoupled approaches first compute individual paths over a given graph by using a single-agent pathfinding algorithm such as A* and Djkstra's, minimizing a multi-robot cost metric such as *makespan* or *sum-of-costs* with respect to traversal distance or time. Then inter-agent conflicts are resolved. In [16] collisions are avoided by fixing all agents' velocities. In [4], [17]–[20], collisions are avoided by assigning priorities to incrementally compute each agent path, treating higher priority agents as dynamic obstacles. By partially exploring the joint-space, decoupled planners are *incomplete*.

3) Hybrid Approaches: In [7], a hybrid MAPF method, *M**, solves the MAPF problem by initially planning a set of individual policies in a fully decoupled manner. These policies are then used to guide a coupled search over the joint state space. When an inter-agent conflict arises, the coupled search is backtracked until the last collision-free joint state, and the conflicting agents are merged into a coupled meta-agent. New collision-free paths are computed using a coupled planner for the meta-agent. If all agents are in collision at the same place and time, M* may become a fully coupled planner as long as the inter-robot conflict remains unresolved.

CBS [2], produces optimal solutions on a grid world representation. To ensure completeness, it considers all the possible ways in which conflicts between individual paths can be resolved. CBS uses a low-level search to find individual agent paths and a high-level search to find and resolve conflicts between paths. To avoid increasing the dimensionality of the search like M*, CBS

poses constraints to low-level searches to avoid previously detected conflicts. This enables CBS to resolve conflicts for agents independently and prevents elevating the dimensionality of the planner. This is the method we extend to the sampling-based motion planning domain in this letter. CBS and recent variants are discussed in detail in the following section.

B. Multi-Agent Motion Planning

In motion planning, the state space is the set of all possible agent configurations known as the *configuration space* ($\mathcal{C}_{space}$) [21]. A solution to the motion planning problem is a continuous path in a subset of $\mathcal{C}_{space}$ called *free space* ($\mathcal{C}_{free}$) consisting of valid configurations. In response to the complexity of motion planning [22], sampling-based motion planners were developed as an efficient means of discovering valid paths in $\mathcal{C}_{free}$. These methods, such as the Probabilistic Roadmap Method (PRM) [23] attempt to create a *roadmap*, or graph, approximating $\mathcal{C}_{free}$. Paths are found by querying this roadmap.

Not much work has been proposed for sampling-based multi-agent motion planning (MAMP). Much that has been done modifies or extends two widely used single-agent sampling-based motion planning algorithms, PRM [23] and RRT [24]. In [9], [25], PRM is used to solve MAMP instances in both coupled and decoupled manners. In [10], MRP-IC uses individual PRMs to solve the MAMP problem incrementally. MRP-IC use single-agents paths to guide an incremented state-space coupled search to find the path of the next agent. In [8], [26]–[29] optimizations and improvements to regular RRT are proposed to enable efficient constructions and rapid explorations of the joint $\mathcal{C}_{space}$. MRdRRT is one the most relevant RRT-based techniques.

Nonetheless, despite the optimal guarantees of composite planners, the solutions they produce are often much more expensive than decoupled approaches. This is because they explore the joint $\mathcal{C}_{space}$ of the multi-agent team. As the size of the joint $\mathcal{C}_{space}$ grows exponentially with the number of robots, the probability of sampling joint configurations useful relative to the individual agent paths becomes increasingly low. This issue can be mitigated by increasing the size of the composite roadmap. However, this involves increasing the computational cost and memory usage.

Several ideas of how discrete MAPF techniques can be adapted to continuous problems, "Relocating Roadmap Nodes", "Merging nodes" and "Addressing Node/Node and Node/Edge interactions" are presented in [30]. However, no experimentation is provided to validate their applicability.

III. Conflict-Based Search

Conflict-Based Search (CBS) finds optimal solutions for MAPF problems where the underlying graph is a grid roadmap which is shared by all agents [2]. The core idea of CBS is to efficiently explore possible single agent path combinations in which a MAPF instance can be solved. This is achieved by using a two-level framework that grows and maintains a set of path constraints for the various agents. This set is later used for finding new paths that are consistent with these constraints. A constraint $< a_i, v, t >$ is defined as the vertex v and timestep t at which an agent a_i must not traverse. At the high-level of CBS, individual paths are checked for conflicts. A conflict $< v, t, a_i, a_j >$ is defined as an object representing when both an agent i and agent j occupy a vertex v at timestep t. In other words, a conflict represents the location and time when an inter-agent collision occurs. All vertices are one unit distance from their neighbors, and moving between neighboring vertices constitutes one timestep. A different type of conflict occurs when two agents located on a pair of neighboring vertices swap their positions, and then they collide during their motions. This event is identified as an *edge conflict*. An edge conflict then has the form $< a_i, a_j, v_1, v_2, t >$ where a_i moves from v_1 to v_2 while a_j moves from v_2 to v_1) during the time frame $[t, t+1]$. An edge constraint is defined as $< a_i, v_1, v_2, t >$, where agent a_i is prohibited of traversing the edge (v_1, v_2) at timestep t for not reaching v_2 at timestep $t+1$. Edge conflicts are treated in the same manner as vertex conflicts. When a conflict is detected, the high-level generates corresponding path constraints for resolving it. Then, the low level searches for individual paths that are consistent with the new constraints.

The high-level search utilizes a data structure called Conflict Tree (*CT*). Each node in *CT* contains a solution, a set of path constraints for the various agents, and a total cost. The solution is a set of paths that contains one path per agent. Each path is consistent with the corresponding constraints obtained from the set of constraints. The low-level search computes these paths. The total cost is relative to either *sum-of-costs* or *makespan* of the set of paths.

The CT is searched in a best-first manner, with respect to node solution cost. This results in optimal solutions. In the beginning, the root node of CT contains no constraints. Thus, the set of paths in this node is freely computed. After this, the root node serves as the basis for growing the CT. This process mainly involves adding, expanding, and validating new nodes in CT. There are two essential tasks within this process, Conflict Detection and Conflict Resolution.

1) Conflict Detection: CT node solutions are evaluated for path conflicts by the high-level planner. The conflict detection iterates through all the individual paths and matches the locations and times traversed by all agents. If no conflict is detected, the CT node is declared as a goal node.

2) Conflict Resolution: Whenever a conflict c is detected, the high-level planner generates a path constraint for each agent involved. A constraint for an agent a_i consists of the time t of the collision, and the vertex v. A child *CT* node is generated for each constraint. Each child node additionally copies all of the constraints of the parent node.

The low-level planner computes individual paths that are consistent with current constraint sets. In MAPF, all agents use the same graph, and a constraint can be easily mapped to an "invalid" state to be avoided by the low-level search.

The high-level search continues to evaluate the next lowest cost unexplored node in *CT*. The search stops upon finding the first conflict free or goal node. As the tree generates all possible plans and nodes cost at least as much as their parent, the first goal node discovered holds an optimal solution.

3) Suboptimal Variant: In ECBS [6], best-first searches of both high- and low-level planners are aided by a focal search. Secondary heuristics enable pruning and exploration of a subset of both high-level and low-level nodes. ECBS solutions have cost no more than ω-times optimal solution.

4) Continuous Variant: In [1], a recent ECBS extension to the continuous-time field is proposed. This technique, ECBS-CT, aims to solve the MAMP problem in the state lattice world representation, where the workspace is discretized into a grid, and then grid cells are connected using a predefined set of single-agent motion primitives. ECBS-CT produces optimal and sub-optimal solutions and is considered a state-of-the-art MAMP solver. It leverages using a state-lattice representation to map the agents' motions to a common workspace discretization. Thus, all the agents' motions can be incorporated into the same state-space representation.

Conflicts occur when two or more agents sweep a grid cell at the same timestep. Conflicts are resolved by posting a constraint $< c, t >$ on the cell c a timestep t where the conflict occurred. This constraint prohibits the corresponding agent from being at cell c at timestep t.

Relying on the workspace grid discretization can lead to unfavorable situations for non-state lattice approaches like PRMs and RRTs. Mapping motions to grid cells leads to a dependency on grid cell size. Large grid cells can lead to false positive conflicts as two agents may be at the same cell and timestep, but no real collision occurs. This results in a loss of precision, thus optimality and completeness of ECBS-CT dependent upon grid cell size choice. Additionally, extra conflicts results in more CT nodes increasing the planning time. On the other hand, when the size of grid cells is small, mapping agent motions to grid cells may significantly increase the computational cost and memory usage.

Algorithm 1: $CBS - MP$.

Input: Environment E, Set of agents A, Set of queries Q. **Output:** Set of motion plans P
Initial motion plans $P \leftarrow \emptyset$; Sampling iterations $S \leftarrow 1$
for each agent $a_i \in A$ **do**
  Sample E to build a roadmap $r_i \in R$
  $p_i \leftarrow$ `MotionPlan`$(q_i, r_i, \emptyset)$
  $P \leftarrow P \cup \{p_i\}$
end for
while true **do**
  Constraint Set $C \leftarrow \emptyset$; Node `root`$\leftarrow (P, C)$
  Conflict Tree $CT \leftarrow \{$`root`$\}$; Node count $N \leftarrow 0$;
  while $CT \neq \emptyset$ **do**
    if `ShouldResample`(N, S) **then**
      break
    end if
    $n = CT$.`GetMinNode()`; $N \leftarrow N + 1$
    $x =$ `FindConflict`$(n.P)$
    if $x == \emptyset$ **then**
      return $n.P$
    else
      Obtain new constraints c_i, c_j from x
      Node $n_1 \leftarrow (n.P, n.C \cup \{c_i\}$
      Node $n_2 \leftarrow (n.P, n.C \cup \{c_j\})$
      $n_1.P \leftarrow n_1.P \cup \{$`MotionPlan`$(q_i, r_i, n_i.C)\}$
      $n_2.P \leftarrow n_2.P \cup \{$`MotionPlan`$(q_j, r_j, n_j.C)\}$
      $CT \leftarrow CT \cup \{n_1, n_2\}$
      continue
    end if
  end while
  Sample E to expand roadmaps $r \in R$; $S \leftarrow S + 1$
end while
return $P \leftarrow \emptyset$

IV. Conflict-based Search in Sampling-Based Multi-Agent Motion Planning

We first formally define the sampling-based MAMP problem. Next, we introduce our method CBS-MP, the CBS extension to sampling-based MAMP problems. Finally, we provide its theoretical properties.

A. Problem formulation

This subsection formally defines the Multi-agent Motion Planning problem and what constitutes a solution for it.

Definition 1: A multi-agent motion planning (MAMP) problem consists of an environment E, set of agents A, and initial and goal conditions for each agent $a_i \in A$. Let the free configuration space for agent a_i be denoted as $\mathcal{C}_i$. Then for each agent $a_i \in A$ the initial state is a configuration $s_i \in \mathcal{C}_i$ and the goal condition is reaching a set $G_i \subset \mathcal{C}_i$.

Definition 2: Let R be a set of roadmaps r_i for each agent $a_i \in A$. We assume each $r_i \in R$ is sufficiently representative for containing a valid path from s_i to G_i.

Definition 3: Let P be a set of paths $\rho_i(t) : [0, t_{final}] \rightarrow C_i$ for each agent $a_i \in A$. We say that P is a solution to an MAMP problem if for all times $t \in [0, t_{final}]$ there are no collisions between the robot configurations $\rho_i(t)$.

Definition 4: Let P be a solution to an MAMP problem, P is said to be *representation-optimal* with respect to R and a cost metric $C(R, P) = C^*$ if there is no set of valid paths P' in R with lower cost $C(R, P') < C^*$.

Definition 5: A *representation-optimal* team solution is optimal with respect to the current representation of each individual agent's state space.

Many MAPF methods claim to provide optimal solutions. This optimality is with respect to their input state representation or roadmap which is assumed to be perfect. While some sampling-based motion planning methods provide asymptotic optimality, this never achieves true optimality in any real application. In this work, we focus on providing a team path which is optimal with respect to the current representation of each individual agent's state space.

B. Adapting CBS to Sampling-Based Motion Planning

In general planning problems, heterogeneous agents operate in distinct continuous spaces. Our approach, CBS-MP, adapts

the MAPF CBS method, which assumes a shared discrete representation, to this more general scenario.

Initially, we sample individual roadmaps for each agent using standard PRM techniques. Each roadmap is grown until it contains a valid path for the corresponding agent. Vertices represent valid configurations, and edges represent valid transitions between a pair of vertices. Each edge is validated by a local planner that discretizes it into intermediate points of fine enough granularity to validate the corresponding continuous motion [31]. Edges can also be represented as a chain of configurations known as *intermediates*, and a path can be represented as a longer chain of intermediates. A path intermediate is a configuration and can be either be a roadmap vertex or an edge intermediate.

Then we use the two-level framework of CBS to query the roadmaps. While the high- and low-level framework remains the same, MAPF CBS cannot directly handle the PRM discretization. This is due to disjoint individual state representations and the non-uniformity of edge length and duration. These issues manifest themselves during the conflict detection and resolution stages of CBS.

1) Conflict Detection: Since agent motions are represented by edges, they may have different distances and duration. Then, individual paths will also exhibit the same notion. Thus, before checking the paths for conflicts, they have to be put in the same time context. To achieve this, we introduce the uniform time discretization process. Which essentially consists of discretizing all the individual paths into uniform time resolution segments. These segments can be of different lengths in different agents' roadmaps, but each segment takes the same time duration for the corresponding agent to traverse. This allows the sequence of segment endpoints along each agent's path to be synchronized. The endpoints can also be seen as path intermediates that correspond to configurations for the associated robots [31]. The path intermediates are checked for conflict with standard motion panning collision detection methods. When a conflict is detected, it consists of a pair of intermediates p_i, p_j and a timestep t, $c =< t, p_i, p_j >$. Regardless of the type of conflicts we found, vertex conflicts or edge conflicts, our method provides a simpler and more efficient way to find and treat conflicts equally without distinction.

2) Conflict Resolution: In CBS and most of the variations, a constraint inherits directly from the conflict found and represent specific states that must be avoided during the low-level search. In our approach, a constraint $< a_i, t, p_j >$ consists of an agent a_i, timestep t, and the other agent's conflicting configuration p_j. Motion plans for an individual agent a_i satisfying constraints of this form must avoid collision with the constraint configuration p_j at the corresponding timestep t. Constraints are then mapped to the edges of agent a_i's roadmap that physically collide against p_j. This means a constraint will then encompass a set of edges that must not be traversed at timestep t, not just a single state to avoid during the low-level search. Doing this involves performing a large number of collision detection calls. To alleviate this issue, we lazily evaluate edges against the constraints while searching the roadmap by first checking if the constraint timestep overlaps the edge, and if so, we perform the collision detection against the other agent configuration. Otherwise we proceed with the search.

3) Roadmap Expansion: In MAMP problems, it is possible to verify individual roadmaps contain valid single-agent solutions in a decoupled manner, but it is unknown if the current set of roadmaps contain a valid team solution without exploring the composite space. In CBS-MP, the composite space is explored by expanding the conflict tree.

A deep exploration of the tree (and thus a longer planning time) may be caused by either a problem that requires very tight coordination between agents or by insufficient roadmaps. A choice has to be made to either let the query keep expanding the CT or to quit and expand the roadmaps before restarting the query. As either option may be correct for solving the problem, we define a probability function

$$p = 1 - X^{\alpha \frac{N}{S}} \tag{1}$$

to define the likelihood of expanding the roadmaps at any given point in the exploration of the CT. The equation is designed to increase the probability of quitting the query to expand the roadmaps as the number of CT nodes N increases. Additionally, as the number of times S the method has quit to resample increases, the probability of quitting decreases, making a deeper exploration of the CT more likely. The parameters X and α are user-defined constants and allow the function to be biased towards or away from additional roadmap expansion. Setting $X{=}1$ will force the query to search the entire CT without expanding the roadmaps.

C. Theoretical Properties

We will show that the CBS-Query reduces to CBS with a different conflict resolution mechanism. We assume that $X = 1$ in equation 1, so the entire CT is explored.

Lemma 1. Validity: If the CBS-MP query finds a solution, it will be collision-free.

Proof: CBS-MP maps the agents' heterogeneous representations to a common representation with uniform time discretization. For any common representation where conflicts can be detected, the inverse map can be used to assign the conflict to the associated state in the agents' individual representations. These are the generalized requirements for conflict resolution in CBS, and any produced solution will be free of conflicts. □

Lemma 2. Completeness: If a solution exists in the set of roadmaps R, then the CBS-MP query will find it.

Proof: In the worst case the CBS-MP query will explore all possible paths in R. On discovery of the first valid set of paths P, no conflicts will be detected by collision checking with uniform time discretization, and P will be identified as a solution. □

Lemma 3. Optimality: If a *representation-optimal* path exists in R, then CBS-MP query will find it in finite time.

Proof: The individual agent paths are generated with Dijkstra's algorithm on a state space representation with strictly non-negative weights, which represent motion time. The individual costs will thus increase monotonically on each replan and are admissible. For any admissible group cost metric, the

CBS-MP query explores group paths in best-first cost order. A discovered solution will therefore have the best possible cost of those encoded by R. By the *completeness* lemma, we will find such a path in finite time if it exists. □

Theorem 1: By lemmas 1,2,3, the CBS-MP query reduces to CBS with a different conflict resolution mechanism. It thus is complete, and provides a representation-optimal solution.

We now discuss the implications the properties of the CBS-MP query has on the full motion planner. The value of X in equation 1 lies within (0,1).

Lemma 4: If a solution exists, we will find roadmaps that encode it eventually.

Proof: As the CT is expanded infinitely, the probability of expanding the roadmaps goes to 1. Thus the CT will either find a solution (in which R must contain a solution), or the roadmaps will be expanded. Due to probabilistic completeness of the decoupled PRM used to generate the individual roadmaps in CBS-MP, the set of roadmaps R will eventually contain all possible paths. □

Lemma 5: If a solution exists, we will eventually find the representation-optimal set of paths for a set of roadmaps R.

Proof: Assume a solution exists and that it will be the N^{th} CT node evaluated for the current set of roadmaps R. If the CT is explored to the N^{th} node, then by Theorem 1, the query will return the representation-optimal path for the current representation. As the number of roadmap expansions S increases, the probability of further roadmap expansion goes to 0 and the probability of exploring the CT to the solution node goes to 1. Thus we will eventually reach the CT node containing the representation-optimal solution. □

Theorem 2: The full planner is probabilistically complete and representation-optimal.

Proof: As any possible solution will be found by Lemma 4, and the query will return the representation-optimal path for the current roadmaps by Lemma 5, the full planner, CBS-MP, is probabilistically complete and representation-optimal. □

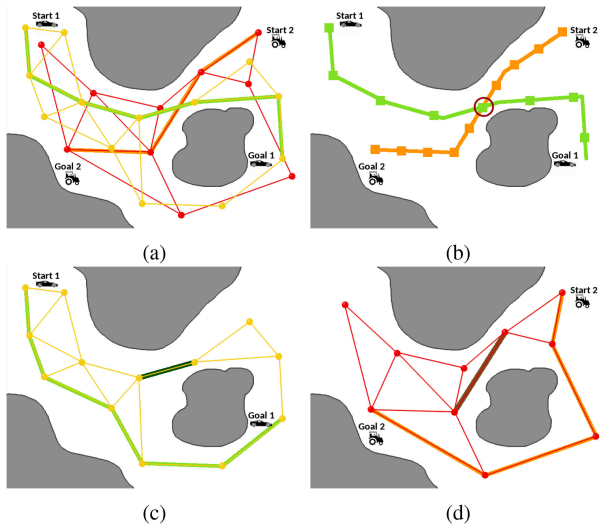
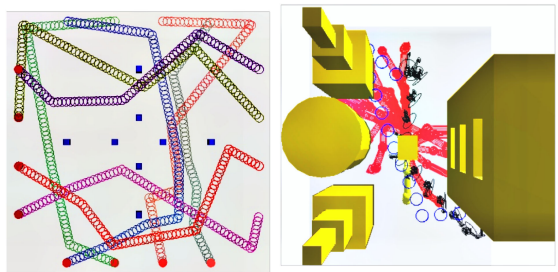


Fig. 1. (a) Individual roadmaps and paths are computed separately. (b) Paths are discretized into uniform motion time segments, enabling inter-agent collision detection. (c,d) Conflicts are mapped back to the corresponding edges, then new paths can be computed.



(a) Multi-mobile robot team (b) Heterogeneous team

Fig. 2. (a) The agents move from bottom to top and left to right. The paths for each agent are shown in the various colors. (b) A crane occupies the central region while a ground-based and aerial robot must navigate in the same space.

V. Validation

We designed three different experiments to evaluate scalability, performance on high-DOF multi-robot systems, and performance on heterogeneous systems respectively. To test scalability, we plan for large teams of mobile robots in a crowded environment. Our method shows improved performance over existing approaches. It is the only one capable of consistently scaling to teams of 32 mobile robots. In the second experiment, we consider teams of 10-DOF manipulator robots of increasing number in close proximity. It is the only method that solved all trials for 8 manipulator robots while keeping low planning times and solution costs. In the last experiment, CBS-MP exhibits a better performance in coordinating a heterogeneous system. It is the only method that solved for the 6 robot team while producing lower planning times and sparser roadmaps for the 3 robot team.

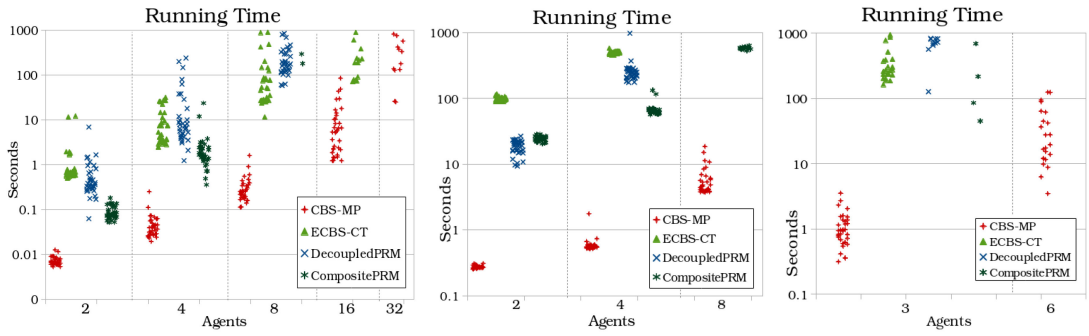
We evaluate our method against the canonical coupled and decoupled PRM as well as ECBS-CT. Decoupled PRM was implemented as a dynamic-obstacle-based approach and computes individual agent paths incrementally. As the performance of ECBS-CT is dependent upon the choice of grid cell size, we experimentally determine the best choice for each scenario. All methods are given 1000s seconds to plan at which the attempt is considered a failure.

A. Scenario I

Scenario I evaluates the scalability with respect to the number of agents in scenarios prone to high numbers of conflicts. In an open environment, half of the robots move from left to right and half go from bottom to top (Fig. 2(a)).

In addition to analyzing the scalability of the four methods, this scenario demonstrates the performance of our method when its *CT* grows significantly. Using simple robots inside a free environment minimizes extraneous factors that may affect the performance of the planners. We proportionally increased the size of the environment with the number of robots. This allows us to have the same density per robot.

CBS-MP exhibits improved scalability over Composite PRM, Decoupled PRM, and ECBS-CT (Fig. 3(a)). For small agent



(a) Scenario I (b) Scenario II (c) Scenario III

Fig. 3. Running time results: These results show the running time of all scenarios. (a) CBS-MP shows faster times and is the only method that solved problems with 32 robots. (b) CBS-MP shows faster running times and is the only method that solved the problem with eight robots in all trials. (c) CBS-MP shows faster running times and is the only method that solved all the trials.

TABLE II
THESE ARE THE RESULTS FOR THE MOBILE ROBOT SCALABILITY SCENARIOS. DATA IS OMITTED FOR METHODS THAT WERE UNABLE TO COMPLETE ANY SUCCESSFUL TRIALS IN THE ALLOTTED TIME FOR A PARTICULAR SCENARIO

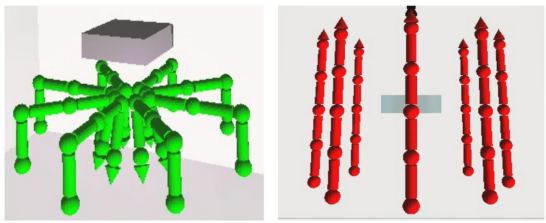
Algorithm	Agents	Path Cost		Roadmap size		Success
		Avg	Std Dev	Avg	Std Dev	
CBS-MP	2	9.3	1.0	12.4	1.9	100%
ECBS-CT		9.5	1.2	16.4	11.9	100%
Decoupled-PRM		9.1	1.1	22.4	25.7	100%
Composite-PRM		10.5	1.6	14.9	5.2	100%
CBS-MP	4	28.3	2.9	20.1	10.4	100%
ECBS-CT		30.3	3.3	47.3	23.6	100%
Decoupled-PRM		26.5	2.8	63.4	87.0	100%
Composite-PRM		60.1	19.5	112.0	167.0	100%
CBS-MP	8	92.2	10.5	26.1	12.7	100%
ECBS-CT		90.0	13.4	99.9	81.7	100%
Decoupled-PRM		85.8	5.7	65.8	36.6	100%
Composite-PRM		381.0	247.0	5000.0	1541.0	5.%
CBS-MP	16	443.4	69.6	36.5	36.1	100%
ECBS-CT		490.0	65.3	56.2	36.0	46%
CBS-MP	32	1626.0	273.3	34.9	23.91	43%

teams, the Composite PRM has low planning times, taking roughly three times as long as CBS-MP and planning faster than Decoupled PRM. For eight agents, the coupled approach becomes drastically more expensive as the state space grows too large. Despite the optimal guarantees of the composite approach, the solutions are often much more expensive than the other two methods. This is due to Composite PRM sampling the joint $\mathcal{C}_{space}$. The probability of sampling joint configurations that are useful relative the individual agent paths becomes increasingly low as the number of robots increase.

Neither Decoupled PRM or Composite PRM is able to solve in the allotted time for 16 or 32 agents.

Decoupled PRM often produces higher quality plans than CBS-MP. This is expected as CBS-MP often finds solutions with sparser roadmaps as shown by the average roadmap size (Table II). Decoupled PRM continues to expand the roadmaps until it finds a solution as instead of resolving conflicts within the existing roadmap like CBS-MP. As CBS-MP is representation-optimal, it will always produce a plan at least as good as Decoupled PRM on the same roadmap.

ECBS-CT performs similarly to Decoupled PRM up to 8 agents. It is able to solve some seeds with 16 agents before timing out but failed in all 32 agent seeds. The cost of mapping configurations to the grid cells as the roadmaps grow outweighs the advantage of using the CBS framework. CBS-MP was run on 64 robot teams as well, though the CT grew too large and overflowed the memory.



(a) The start configuration. (b) The goal configuration.

Fig. 4. The set of manipulators starts in a convoluted configuration requiring tightly coupled maneuvering to reach the goal configuration.

B. Scenario II

This experiment shows planning performance on multiple high-DOF robots in a shared space. Several robotic arms start in a confined initial position where each arm is nearly in contact with the rest (Fig. 4). CBS-MP performs well in this problem even with eight robots (Fig. 3(b)). Due to its greedy-priority nature, DecoupledPRM always picks the shortest path for one of the manipulators, which usually goes through the contested middle space. This decreases the probability that the others will find a path because the majority of paths require their distal joints to use parts of the middle space. As the number of robots increases, so does the congestion in the middle. This prevents it from finding solutions where CBS-MP can select paths where all agents avoid each other while sharing the middle volume (Table III).

ECBS-CT took significantly longer to plan in all the manipulator experiments. In order to properly map the motions of the manipulators, the grid cell size is smaller relative the robot than the disc robots used in Scenario I resulting in increased cost mapping the configurations in robot paths to the swept cells in the grid discretization.

C. Scenario III

We evaluate our algorithm's ability to plan for heterogeneous multi-robot teams. A heterogeneous team, composed of aerial,

TABLE III
THESE ARE THE RESULTS FOR THE HIGH DOF MANIPULATOR SCENARIOS. DATA IS OMITTED FOR METHODS THAT WERE UNABLE TO COMPLETE ANY SUCCESSFUL TRIALS IN THE ALLOTTED TIME FOR A PARTICULAR SCENARIO

Algorithm	Agents	Path Cost		Roadmap size		Success
		Avg	Std Dev	Avg	Std Dev	
CBS-MP	2	4.4	0.38	27.0	0.00	100%
ECBS-CT		4.7	0.68	27.0	0.00	100%
Decoupled-PRM		4.4	0.38	27.0	0.00	100%
Composite-PRM		158.0	35.6	27.0	0.00	100%
CBS-MP	4	8.9	0.45	27.71	4.23	100%
ECBS-CT		8.9	0.63	32.0	0.00	100%
Decoupled-PRM		9.1	0.70	29.1	9.34	100%
Composite-PRM		499.0	151	28.4	5.89	100%
CBS-MP	8	15.7	0.49	60.5	19.12	100%
Composite-PRM		1203.0	630.0	52.0	0.00	42%

TABLE IV
THESE ARE THE RESULTS FOR THE HETEROGENEOUS SYSTEM SCENARIOS. DATA IS OMITTED FOR METHODS THAT WERE UNABLE TO COMPLETE ANY SUCCESSFUL TRIALS IN THE ALLOTTED TIME FOR A PARTICULAR SCENARIO

Algorithm	Agents	Path Cost		Roadmap size		Success
		Avg	Std Dev	Avg	Std Dev	
CBS-MP	3	135.3	16.5	34.5	18.3	100%
ECBS-CT		137	12.6	37.3	21.1	66%
Decoupled-PRM		124	8.8	35.7	20.9	29%
Composite-PRM		677	99.8	4641.0	5234.0	11%
CBS-MP	6	219.0	15.54	206.82	115.3	69%

ground-based, and crane-like robots, must coordinate their motions in a constrained environment (Fig. 2(b)).

CBS-MP, Decoupled PRM, and ECBS-CT solved the 3 robot heterogenous problem within the 1000 seconds. CBS-MP again showed improved performance (Fig. 3(c)) and solved all trials, while Decoupled PRM and ECBS-CT were successful on 20% and 65% respectively (Table IV). CBS-MP was the only method with any success on the 6 robot problem (69%). This demonstrates the effectiveness of the CBS-MP approach and the uniform-time discretization of roadmaps for finding and resolving conflicts in heterogeneous systems.

VI. CONCLUSIONS AND FUTURE WORK

We present CBS-MP, an extension of CBS to sampling-based motion planning.

We validate our work in different scenarios to show the strengths of our method to deal with sets of numerous agents, high-DOF agents, and heterogeneous agent teams. This work offers several interesting directions for future work. Within the presented framework, the motions considered can be extended to incorporate more interesting robot dynamics. Additionally, while the current framework replans paths at the global level, we are looking replanning at the local level as well.

REFERENCES

[1] L. Cohen, T. Uras, T. S. Kumar, and S. Koenig, "Optimal and bounded-suboptimal multi-agent motion planning," in *Proc. 12th Int. Symp. Combinatorial Search*, 2019.

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

[3] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Trans. Syst. Sci. Cybern.*, vol. 4, no. 2, pp. 100–107, Jul. 1968.

[4] J. P. van denBerg and M. H.Overmars, "Prioritized motion planning for multiple robots," 2005, pp. 430–435.

[5] E. Boyarski, A. Felner, R. Stern, G. Sharon, D. Tolpin, O. Betzalel, and E. Shimony, "ICBS: Improved conflict-based search algorithm for multi-agent pathfinding," in *Proc. 24th Int. Joint Conf. Artif. Intell.*, 2015.

[6] M. Barer, G. Sharon, R. Stern, and A. Felner, "Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem," in *Proc. 7th Annu. Symp. Combinatorial Search*, 2014.

[7] G. Wagner and H. Choset, "Subdimensional expansion for multirobot path planning," *Artif. Intell.*, vol. 219, pp. 1–24, 2015.

[8] K. Solovey, O. Salzman, and D. Halperin, "Finding a needle in an exponential haystack: Discrete RRT for exploration of implicit roadmaps in multi-robot motion planning," *Int. J. Robot. Res.*, vol. 35, pp. 501–513, 2016.

[9] G. Sanchez and J. C. Latombe, "Using a PRM planner to compare centralized and decoupled planning for multi-robot systems," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2002, vol. 2, pp. 2112–2119.

[10] M. Saha and P. Isto, "Multi-robot motion planning by incremental coordination," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2006, pp. 5960–5963.

[11] A. Felner *et al.*, "Partial-expansion A* with selective node generation," in *Proc. 26th AAAI Conf. Artif. Intell.*, 2012.

[12] M. Goldenberg, A. Felner, R. Stern, G. Sharon, N. Sturtevant, R. C. Holte, and J. Schaeffer, "Enhanced partial expansion A*," *J. Artif. Intell. Res.*, vol. 50, pp. 141–187, 2014.

[13] R. E. Korf, "Depth-first iterative-deepening: An optimal admissible tree search," *Artif. Intell.*, vol. 27, no. 1, pp. 97–109, 1985.

[14] J. Yu and S. M. LaValle, "Optimal multirobot path planning on graphs: Complete algorithms and effective heuristics," *IEEE Trans. Robot.*, vol. 32, no. 5, pp. 1163–1177, Oct. 2016.

[15] E. Erdem, D. G. Kisa, U. Oztok, and P. Schüller, "A general formal framework for pathfinding problems with multiple agents," in *Proc. 27th AAAI Conf. Artif. Intell.*, 2013.

[16] R. Cui, B. Gao, and J. Guo, "Pareto-optimal coordination of multiple robots with safety guarantees," *Auton. Robots*, vol. 32, no. 3, pp. 189–205, 2012.

[17] M. Cáp, P. Novák, M. Seleckỳ, J. Faigl, and J. Vokffnek, "Asynchronous decentralized prioritized planning for coordination in multi-robot system," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 3822–3829.

[18] V. R. Desaraju and J. P. How, "Decentralized path planning for multi-agent teams with complex constraints," *Auton. Robots*, vol. 32, no. 4, pp. 385–403, 2012.

[19] R. Regele and P. Levi, "Cooperative multi-robot path planning by heuristic priority adjustment," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2006, pp. 5954–5959.

[20] A. Geramifard, P. Chubak, and V. Bulitko, "Biased cost pathfinding," in *AIIDE*. Marina Del Rey, California, 2006, pp. 112–114.

[21] T. Lozano-Pérez and M. A. Wesley, "An algorithm for planning collision-free paths among polyhedral obstacles," *Commun. ACM*, vol. 22, no. 10, pp. 560–570, Oct. 1979.

[22] J. H. Reif, "Complexity of the mover's problem and generalizations," in *Proc. IEEE Symp. Foundations Comput. Sci.*, San Juan, Puerto Rico, Oct. 1979, pp. 421–427.

[23] L. E. Kavraki, P. Švestka, J. C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Automat.*, vol. 12, no. 4, pp. 566–580, Aug. 1996.

[24] S. M. Lavalle, "Rapidly-exploring random trees: A. new tool for path planning," Tech. Rep., 1998.

[25] G. Sanchez and J.-C. Latombe, "On delaying collision checking in PRM planning application to multi-robot coordination," in *Int. J. Robot. Res.*, 2002.

[26] S. Carpin and E. Pagello, "On parallel RRTs for multi-robot systems," in *Proc. Italian Assoc. AI*, 2002, pp. 834–841.

[27] D. Ferguson, N. Kalra, and A. Stentz, "Replanning with RRTs," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2006, pp. 1243–1248.

[28] M. Čáp, P. Novák, J. Vokrínek, and M. Pěchouček, "Multi-agent RRT: Sampling-based cooperative pathfinding," in *Proc. Int. Conf. Auton. Agents Multi-Agent Syst. Int. Found. Auton. Agents Multiagent Syst.*, 2013, pp. 1263–1264.

[29] R. Shome, K. Solovey, A. Dobson, D. Halperin, and K. E. Bekris, "dRRT*: Scalable and informed asymptotically-optimal multi-robot motion planning," *Auton. Robots*, vol. 44, pp. 443–467, 2019.

[30] A. Krontiris, Q. Sajid, and K. E. Bekris, "Towards using discrete multiagent pathfinding to address continuous problems," in *Proc. Workshops 26th AAAI Conf. Artif. Intell.*, 2012.

[31] G. Song, S. L. Thomas, and N. M. Amato, "A General Framework for PRM Motion Planning," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2003, pp. 4445–4450.