Mutex Propagation in Multi-Agent Path Finding for Large Agents

Han Zhang, Yutong Li, Jiaoyang Li, T. K. Satish Kumar, Sven Koenig

University of Southern California
{zhan645, yli81711, jiaoyanl}@usc.edu, tkskwork@gmail.com, skoenig@usc.edu

Abstract
Mutex propagation and its concomitant symmetry-breaking techniques have proven useful in Multi-Agent Path Finding (MAPF) with point agents. In this paper, we show that they can be easily generalized to richer MAPF problems. In particular, we demonstrate their application to MAPF with “Large” agents (LA-MAPF). Here, agents can occupy multiple points at the same time according to their fixed shapes and sizes. While existing rule-based symmetry-breaking techniques are difficult to generalize from point agents to large agents, mutex-based symmetry-breaking techniques can be generalized easily. In a Conflict-Based Search (CBS) framework for LA-MAPF, we also develop a mutex-based conflict-selection strategy to further enhance the efficiency of the search. Through experiments on various maps, we show that our techniques significantly improve MC-CBS, a state-of-the-art optimal LA-MAPF algorithm, in terms of both success rate and runtime.

Introduction and Related Work
The Multi-Agent Path-Finding (MAPF) problem is a generalization of the single-agent path-finding problem to multiple agents. Each agent is required to move from a given start vertex to a given goal vertex on a given graph while avoiding collisions with other agents. Solving the MAPF problem optimally is known to be NP-hard for various objective functions (Yu and LaValle 2013; Ma et al. 2016). Although the MAPF problem arises in many real-world application domains (Wurman, D’Andrea, and Mountz 2008; Morris et al. 2016), MAPF research has mostly focused on point agents, i.e., agents that have no shape or size.

MAPF with Large Agents (LA-MAPF) is a generalization of MAPF that bestows shape and size to the agents to make them more realistic. However, generalizing existing algorithmic techniques for MAPF to LA-MAPF can be non-trivial. For example, while Conflict-Based Search (CBS) is designed to solve the MAPF problem, generalizing it to solving the LA-MAPF problem efficiently requires more sophisticated reasoning, as encapsulated in Multi-Constraint CBS (MC-CBS) (Li et al. 2019b).

Another important paradigm for enhancing search in MAPF with point agents is symmetry breaking. Existing symmetry-breaking techniques fall into two categories: (a) rule-based techniques (Li et al. 2019a, 2020) and (b) mutex-based techniques (Zhang et al. 2020; Walker et al. 2021). However, none of these techniques have been generalized to LA-MAPF. Rule-based techniques are human-generated, handle only limited kinds of symmetries, and are difficult to generalize to LA-MAPF. Mutex-based techniques are derived from constraint propagation and handle more general kinds of symmetries.

In later sections, we describe how mutex propagation and its concomitant symmetry-breaking techniques can be generalized to LA-MAPF. In fact, the generalization makes these techniques applicable to other richer MAPF problems as well. We also develop a mutex-based conflict-selection strategy to further enhance the efficiency of search. In the experimental results, we show that our techniques significantly improve MC-CBS, a state-of-the-art optimal LA-MAPF algorithm, in terms of both success rate and runtime.

Preliminaries
In this section, we provide background material related to MAPF, LA-MAPF, and CBS.

MAPF and LA-MAPF
The MAPF problem is defined by an undirected graph $G = (V, E)$ and a set of $m$ agents $\{a_1, \ldots, a_m\}$. Each agent $a_i$ has a start vertex $s_i \in V$ and a goal vertex $g_i \in V$. In each timestep, an agent either moves to a neighboring vertex, waits at its current vertex, or terminates at its goal vertex (that is, does not move anymore). A path of an agent is a sequence of actions that leads it from its start vertex to its goal vertex and ends with a terminate action. The path cost of a path is the number of timesteps from beginning to termination. A vertex conflict happens when two agents stay at the same vertex simultaneously, and an edge conflict happens when two agents traverse the same edge in opposite directions simultaneously. A solution is a set of conflict-free paths of all agents. In this paper, we focus on minimizing the Sum of path Costs (SoC), that is, the sum of the path costs of the paths of all agents.

LA-MAPF generalizes MAPF to agents with different shapes and sizes. In LA-MAPF, $G$ is embedded in a $d$-dimensional Euclidean space (usually $d = 2, 3$). Each agent has a fixed shape around a reference point and can occupy
multiple vertices at the same time. A vertex conflict happens when the shapes of two agents overlap at some timestep, and an edge conflict happens when the shapes of two agents overlap at some time when they move to their respective next vertices. In this paper, we focus on 2-dimensional grids. However, mutex propagation does not make any assumption of the space embedding of $G$ and hence can be applied in other settings as well.

**CBS**

CBS (Sharon et al. 2015) is an optimal two-level MAPF algorithm. On the high level, CBS performs a best-first search on a Constraint Tree (CT). Each CT node contains (1) a set of constraints and (2) a set of paths, one for each agent, that satisfy all these constraints. The cost of a CT node is the SoC of the paths. CBS starts with the root CT node, which has an empty set of constraints and a path for each agent that has the minimum path cost when ignoring conflicts. When expanding a CT node, CBS returns the paths of it as a solution if the paths are conflict-free. Otherwise, CBS picks a conflict to resolve, splits the CT node into two child CT nodes, and adds a constraint to each child CT node to prohibit either one or the other of the two conflicting agents from using the conflicting vertex or edge at the conflicting timestep. For the newly constrained agent in each child CT node, CBS then calls its low level to find an individual minimum-cost path, that is, a path that has the minimum cost while satisfying all constraints of the CT node but ignoring conflicts.

**Multi-Decision Diagrams (MDDs):** An MDD (Sharon et al. 2013, 2015) $MDD^i_l$ for agent $a_i$ in a CT node is an $(l+1)$-level directed acyclic graph that consists of all paths of cost $l$ for agent $a_i$ that satisfy all constraints of the CT node. Cost $l$ is usually set to the individual minimum cost (that is, the cost of the individual minimum-cost path) of $a_i$, but not necessarily so. Each MDD node $n$ of $MDD^i_l$ at level $t$ correspond to a vertex of agent $a_i$ at timestep $t$ in these paths. We use $n.loc$ and $n.level$ to denote the corresponding vertex and $t$, respectively. Slightly abusing the notation, we use $n \in MDD^i_l$ to denote that $n$ is an MDD node of $MDD^i_l$. At level 0, $MDD^i_0$ has a single source MDD node corresponding to agent $a_i$ occupying its start vertex $s_i$ at timestep 0. At level $l$, $MDD^i_l$ has a single sink MDD node corresponding to agent $a_i$ occupying its goal vertex $g_i$ at timestep $l$.

**Cardinal Conflicts:** Two agents have a cardinal conflict in a CT node if there does not exist a pair of conflict-free individual minimum-cost-paths for both agents (that, by definition, satisfy all constraints of the CT node). That is, the SoC of an optimal solution for the two agents is larger than the SoC of the individual minimum-cost-paths of them. CBS cannot resolve all cardinal conflicts efficiently since it needs to check all combinations of paths whose SoC is less than the SoC of an optimal solution, which can necessitate many CT node expansions.

**Mutex Propagation for LA-MAPF**

In this section, we describe a mutex-based symmetry-breaking technique for LA-MAPF. It first finds mutexes between MDDs of a pair of agents using mutex propagation, and then it uses these mutexes to identify and automatically resolve cardinal conflicts.

Figure 1a shows an example of cardinal conflicts in LA-MAPF, which is similar to corridor symmetry in MAPF (Li et al. 2020; Lam et al. 2019). In this example, both agents are of size $2 \times 2$, and there is a “corridor” of width 3 and length $L = 3$ in the middle of the map. In any optimal solution of this problem instance, either $a_1$ or $a_2$ needs to wait for the other agent to traverse the corridor. One needs to change the existing rule-based technique for corridor symmetries carefully so that it can be used here. In MAPF, corridor symmetries consider only corridors of width 1, which can be easily identified by finding vertices of degree 2. However, in LA-MAPF, corridor symmetries need to consider corridors of different widths depending on the sizes of the agents.

Mutex propagation takes the MDDs $MDD^i_l$ and $MDD^j_l$ as inputs (where $l_i$ and $l_j$ are determined by the algorithm and are not less than the individual minimum costs of agents $a_i$ and $a_j$, respectively). For ease of presentation, we assume that $l_i \leq l_j$. We first find initial mutexes, which correspond to vertex and edge conflicts in LA-MAPF:

1. Two MDD nodes $n_i \in MDD^i_l$ and $n_j \in MDD^j_l$ are initial mutex iff $n_i.loc = n_j.loc$ and agents $a_i$ and $a_j$ have a vertex conflict when they are at $n_i.loc$ and $n_j.loc$, respectively, simultaneously.
2. Two MDD edges $e_i = (n_i, n'_i)$ and $e_j = (n_j, n'_j)$ with $n_i, n'_i \in MDD^i_l$ and $n_j, n'_j \in MDD^j_l$ are initial mutex iff $n_i.loc = n_j.loc$ and agents $a_i$ and $a_j$ have an edge conflict when agent $a_i$ moves from $n_i.loc$ to $n'_i.loc$ and agent $a_j$ moves from $n_j.loc$ to $n'_j.loc$ simultaneously.

Propagated mutexes can be found using the following mutex-propagation rules:

1. **Forward propagation for MDD nodes:** Two MDD nodes $n_i$ and $n_j$ are propagated mutex iff neither $n_i.loc$ nor $n_j.loc$ is 0 and any MDD edge that points to $n_i$ is either initial mutex or propagated mutex with any MDD edge that points to $n_j$. 

![Figure 1: An example of a corridor conflict for two agents. Both agents are of $2 \times 2$ square shapes, and we pick their top-left corners as reference points.](image-url)
of constraints \( C_i \) and \( C_j \), one for each agent, that are used for CBS splitting. The two types of cardinal conflicts and their corresponding algorithms are:

1. **Pre-goal cardinal Conflicts (PCs):** CBS identifies a PC between agent \( a_i \) and \( a_j \) iff the sink MDD node of \( MDD_i^n \) is mutex with all MDD nodes of \( MDD_j^n \) at level \( l_i \). To resolve a PC, CBS uses two constraint sets \( C_i \) and \( C_j \) for agents \( a_i \) and \( a_j \), respectively. Constraint set \( C_i \) contains the vertex constraints for every MDD node of \( MDD_i^n \) that is mutex with all MDD nodes of \( MDD_j^n \) at the same level. Similarly, constraint set \( C_j \) contains the vertex constraints for every MDD node of \( MDD_j^n \) that is mutex with all MDD nodes of \( MDD_i^n \) at the same level.

2. **After-goal cardinal Conflicts (ACs):** CBS identifies an AC between agents \( a_i \) and \( a_j \) iff, for any MDD node \( n \in MDD_j^n \) at level \( l_i \) that is not mutex with the sink node of \( MDD_i^n \), every path from \( n \) to the sink node of \( MDD_i^n \) traverses an MDD node \( n' \) such that agents \( a_i \) and \( a_j \) have a vertex conflict when they are at vertices \( g_i \) and \( n'.loc \) simultaneously, respectively. To resolve an AC, constraint set \( C_i \) contains a constraint that forces the path cost of agent \( a_i \) to be larger than \( l_i \). Constraint set \( C_j \) contains the constraints for all MDD nodes of \( MDD_j^n \) at level \( l_i \) that are mutex with the sink MDD node of \( MDD_i^n \) and the constraints for all MDD nodes \( n' \in MDD_j^n \) such that \( n'.level > l_i \) and agents \( a_i \) and \( a_j \) have a vertex conflict when they are at vertices \( g_i \) and \( n'.loc \) simultaneously, respectively.

**Example 1.** Figure 2 shows MDDs \( MDD_1^n \) and \( MDD_2^n \) for the problem instance in Figure 1a. Green dashed lines indicate initial mutexes, which correspond to conflicts between agents \( a_1 \) and \( a_2 \), and red dashed lines indicate propagated mutexes. A PC can be identified between these two MDDs because the sink node of \( MDD_1^n \) is mutex with all MDD nodes of \( MDD_2^n \) at level 9, that is, the sink node \( MDD_2^n \). The MDD nodes that are used to generate the constraint sets are filled with color.

The mutex-based technique can generate different constraint sets for different choices of \( l_i \) and \( l_j \) as long as a cardinal conflict can be identified using \( MDD_i^{l_i} \) and \( MDD_j^{l_j} \). Intuitively, larger values of \( l_i \) and \( l_j \) result in “stronger” constraints for CBS splitting. For example, the constraint sets in Example 1 will force agent \( a_1 \) to take a path with a cost of 10 in one CT node and agent \( a_2 \) to take a path with a cost of 10 in the other CT node, which is insufficient to resolve all conflicts between the agents. Zhang et al. (2020) proposed a greedy approach for determining the values of \( l_i \) and \( l_j \), which we also use in our implementation. Due to the space limit, we skip the details of this approach.

For a cardinal conflict between agents \( a_i \) and \( a_j \), let \( MDD_i^{l_i} \) and \( MDD_j^{l_j} \) denote the MDDs used for generating the constraint sets, \( l_i^* \) and \( l_j^* \) denote the individual minimum cost of agents \( a_i \) and \( a_j \), and \( \delta_i \) and \( \delta_j \) denote \( l_i - l_i^* \) and \( l_j - l_j^* \), respectively. The larger \( \delta_i \) and \( \delta_j \), the more the path
In this paper, we showed that mutex-based symmetry-breaking techniques can be easily generalized to richer MAPF problems than the classical one, as we demonstrated for LA-MAPF. We also presented a mutex-based conflict-selection strategy. Our experimental results showed that our techniques significantly improve MC-CBS, a state-of-the-art optimal LA-MAPF algorithm, in terms of both success rate and runtime.

While our mutex-based technique can handle generalized vertex and edge conflicts, it still requires mutexes to connect pairs of MDD nodes or edges at only the same level and thus might not apply to problems like k-robust MAPF (Atzmon et al. 2018), where a conflict happens if an agent occupies a vertex that was occupied by another agent at most k timestep ago. Such conflicts can be modeled using a mutex between two MDD nodes on different levels. An interesting direction for future work is therefore to generalize mutex propagation and mutex-based symmetry-breaking techniques to mutexes between MDD nodes on different levels.

1https://movingai.com/benchmarks/mapf.html
Acknowledgments

The research at the University of Southern California was supported by the National Science Foundation (NSF) under grant numbers 1409987, 1724392, 1817189, 1837779, 1935712, and 2112533 as well as a gift from Amazon. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the sponsoring organizations, agencies, or the U.S. government.

References


