Intelligent Planning for Large-Scale Multi-Agent Systems

Hang Ma

This article summarizes the New Faculty Highlights talk with the same title at AAAI 2021.

Intelligent agents such as different types of robots will soon become an integral part of our daily lives. In real-world multi-agent systems, the most fundamental challenges are assigning tasks to multiple agents (task-level coordination problems) and planning collision-free paths for the agents to task locations (motion-level coordination problems). This article surveys four directions of our research on using intelligent planning techniques for the above multi-agent coordination problems.

Self-driving cars, autonomous drones, autonomous aircraft towing vehicles, automated warehouse robots, automated-guided port vehicles, home and office service robots, and other intelligent agents will become an integral part of our daily lives. For example, in the coming years, hundreds of autonomous aircraft towing vehicles will tow their assigned aircraft between runways and terminal gates (Morris et al., 2016). Today, hundreds of warehouse robots already navigate fully autonomously in automated fulfillment and sortation centers to deliver inventory shelves and express parcels to fulfill online orders (Wurman, D'Andrea, and Mountz, 2008; Kou et al., 2020). For these real-world applications of large-scale multi-agent systems, the basic building blocks include assigning tasks to agents and planning paths for the agents to reach task locations. Agents must avoid collisions in a congested environment while reaching their task locations as promptly as possible and completing a large number of tasks as quickly as possible. The resulting task- and motion-level coordination problems, which model the task-assignment and path-planning operations of the agents, are fundamental for these multi-agent systems but, at the same time, computationally challenging since there are typically many agents in such a system and the operating time of the system is long.

In the artificial intelligence (AI) community, much attention has been placed on a simplified one-short version of the path-planning problem in the above multi-agent systems, known as *Multi-Agent Path Finding (MAPF)* (Ma and Koenig, 2017; Stern et al., 2019; Ma, 2022). The problem of MAPF is to move multiple agents from their start vertices to their target vertices in discrete time steps on a given graph that models the environment and let the agents wait in their target vertices. During each time step, each agent can wait in its current vertex or move to an adjacent vertex. Agents are not allowed to collide. Two agents collide if and only if, during the same time step, they move to the same vertex or traverse the same edge in opposite directions. A MAPF solution consists of a set of collision-free paths, one for each agent that specifies the vertex occupies by the agent at each time step. The objective is to minimize the flowtime (also known as the sum-of-costs (Felner et al., 2017)), that is, the sum of the numbers of time steps for the agents to reach their target vertices, or the makespan, that is, the earliest time step when all agents are at their target vertices. MAPF is NP-hard to solve optimally for makespan (Surynek, 2010) or flowtime (Yu and LaValle, 2013c) minimization. MAPF algorithms include reductions to Boolean Satisfiability (Surynek et al., 2016), Integer Linear Programming (Yu and LaValle, 2013b), and Answer Set Programming (Erdem et al., 2013) and specialized combinatorial search algorithms (Standley and Korf, 2011; Sharon et al., 2013; Wagner and Choset, 2015; Sharon et al., 2015).

Nevertheless, the following concerns should be addressed when we generalize MAPF to the above real-world applications of large-scale multi-agent systems (Ma et al., 2016a): 1. MAPF algorithms need to be made more efficient, namely they need to compute a solution faster for the same number of agents or scale to a larger number of agents in the same amount of time. 2. New variants and extensions of MAPF need to be studied to tackle problems arising in practice, such as executing MAPF solutions with unmodeled motion delays and robot constraints, combining target assignment and path finding, and long-term and online planning.

This article surveys our research on intelligent planning for large-scale multi-agent systems in four research directions that addresses the above concerns:

- **Improved MAPF algorithms:** We propose several improvements to

state-of-the-art optimal MAPF algorithms based on heuristic search techniques and leveraging insights from solvers of other combinatorial problems. The resulting improved MAPF algorithms are several orders of magnitude faster. We develop bounded-suboptimal and suboptimal MAPF algorithms that produce close-to-optimal MAPF solutions for hundreds of agents in seconds of computation time. Furthermore, we demonstrate the benefits of our algorithms by applying them to the navigation of drones and video characters.

- **Safe execution of MAPF solutions:** We study the problem of handling motion delays when robots execute MAPF solutions. We propose to use execution policies to guarantee safe execution of MAPF solutions. These execution policies determine the correct timing for robots to follow each step of a given MAPF solution. We showcase how we have combined insights from this study and MAPF techniques to win a railway scheduling competition that was held at a top-tier machine learning conference. We also develop a hierarchical framework for generating and executing MAPF solutions for real-world multi-robot systems. This framework makes use of an efficient procedure that accounts for kinematic constraints of robots to transform a MAPF solution to a continuous-time plan-execution schedule that is safe for robots to execute. We demonstrate our framework using different types of simulated and real robots.
- **Combined target assignment and path finding:** We formalize and study a variant of MAPF, called *Combined Target-Assignment and Path Finding (TAPF)* (Ma and Koenig, 2016), that models the joint problem of (1) which locations the robots go to next and (2) how the robots go to the locations. The problem of TAPF is to assign target vertices to multiple agents and move the agents from their start vertices to their target vertices on a given graph without collisions. We develop an optimal TAPF algorithm that computes solutions for hundreds of agents in minutes of computation time. We apply our TAPF algorithm to solving the formation control problem with teams of real robots.
- **Long-term task and path planning:** We formalize and study an extension to MAPF and TAPF, called *Multi-Agent Pickup and Delivery (MAPD)* (Ma et al., 2017b). MAPD models a long-term problem where a system needs to repeatedly assign incoming tasks to a set of agents and plan paths for the agents to the targets of their assigned tasks. We develop complete and deadlock-free MAPD algorithms for both online and offline settings and techniques to account for kinematic constraints of robots when solving MAPD. These algorithms made decisions for hundreds of agents and thousands of tasks in seconds of computation time. We applies our techniques for solving MAPD to automated parcel sorting with warehouse robots and demonstrate the benefits of our techniques on an industrial simulator with real-world data.

Improved MAPF Algorithms

Modern large-scale real-world multi-agent systems such as the automated fulfillment and sortation centers constructed by Amazon (Wurman, D'Andrea, and Mountz, 2008) and Alibaba (Kou et al., 2020) require solving MAPF efficiently, namly planning high-quality paths for hundreds of agents in a short computation time. However, it is NP-hard to solve MAPF optimally (Surynek, 2010; Yu and LaValle, 2013c). In our theoretical study (Ma et al., 2016b), we further prove that MAPF is NP-hard to approximate within any constant factor less than 4/3 for makespan minimization by a reduction from a specialized NP-complete version (Tovey, 1984) of the Boolean satisfiability problem.

Nevertheless, a state-of-the-art MAPF algorithm, called *Conflict-Based Search (CBS)* (Sharon et al., 2015) can compute optimal MAPF solutions for dozens of agents in a few minutes of computation time. CBS is a two-level combinatorial search algorithm. It first finds individually optimal paths for all agents (ignoring collisions). On the high level, CBS performs a best-first search to resolve each collision of the computed paths by imposing constraints on both the agents involved in the collision. The constraints forbid the agents from occupying a vertex or traversing an edge at a given time step. On the low level, CBS uses an A* search

in both the space and the time dimensions to find a path for an agent that obeys its constraints. The high-level search of CBS branches on which collision to resolve.

We have explored three directions to make CBS more efficient. First, we have developed admissible heuristics for the high-level best-first search of CBS to significantly reduce the size of the search tree (Felner et al., 2018; Li et al., 2019a). Experimental results show that the resulting algorithm CBSH (Li et al., 2019a) is up to 50 times faster than CBS and can thus compute solutions for up to 3 times more agents than CBS within one minute of computation time. Second, we have proposed a high-level branching rule for CBS called Disjoint Splitting (Li et al., 2019b). Disjoint Splitting guarantees that CBS solution spaces explored under the child nodes resulting from each node expansion are disjoint, thereby reducing duplicate search effort. Experimental results show that CBS with Disjoint Splitting is up to 2 orders of magnitude faster than CBS. Third, we have developed several techniques (Li et al., 2019c, 2020a, 2021b) to tackle ''pairwise path symmetry'', which occurs when two agents each has multiple paths of the same cost but any paths of the two agents are pairwise incompatible since they result in collisions. Our recent work (Li et al., 2021b) experimentally demonstrates that the combination of several of the above directions result in an improved version of CBS that is up to 4 orders of magnitude faster than CBS and can thus compute solutions for up to 30 times more agents than CBS within one minute of computation time.

We have also explored five directions to make MAPF algorithms more suitable for practical applications of large-scale multi-agent systems. First, we have developed an anytime bounded-suboptimal version of CBS (Cohen et al., 2018) that can be stopped at any time and return a MAPF solution with a guaranteed suboptimality bound. The suboptimality bound become smaller as the algorithm runs longer. Second, we have developed *Prioritized-Based Search (PBS)* (Ma et al., 2019a), a MAPF algorithm that searches in the space of all possible orderings of agents. PBS is a two-level algorithm similar to CBS but performs a depth-first search on the high level. Unlike CBS, PBS is suboptimal and complete only for a realistic family of MAPF instances. However, empirical study shows that PBS always returns close-to-optimal solutions (with a cost no more than 105% of the optimal cost) in our experimental setting and can compute solution for 600 agents in half a minute of computation time. Third, We have addressed the MAPF problem where a deadline is given for agents to reach their target vertices (Ma et al., 2018b,a). We have developed an Integer Linear Programming based algorithm and a CBS-based algorithm that both maximize the number of agents reaching target vertices before the deadline. Fourth, we have addressed the MAPF problem for large-size agents with different geometric shapes and volumes of agents (Li et al., 2019d). We tackle this challenge by allowing each agent to occupy more than one vertex and generalizing the definition of collisions since the agent can intersect with multiple vertices and edges at each time step. We have developed a version of CBS to solve this problem optimally and demonstrate it for the navigation of a drone fleet. Fifth, we have shown how a combination of swarm-based approaches from the robotics community and an adapted version of CBS can be applied to the navigation of multiple game characters that needs to keep a desired formation while moving (Li et al., 2020b). This adapted version of CBS balances between the makespan and the cost of deviating from the desired formation.

Safe Execution of MAPF Solutions

MAPF algorithms can be used to compute collision-free paths for multiple agents in real-world multi-agent systems. However, real-world agents such as warehouse robots cannot perfectly execute the computed solutions, namely following their paths, since they can be delayed unexpectedly when they move, they have unmodeled kinematic constraints (for example, they do not move at the same speed), etc. existing AI research has not studied how the agents can safely execute the computed MAPF solutions.

In our recent work (Ma, Kumar, and Koenig, 2017), we study a variant of MAPF where each agent is delayed and stays in its current vertex with a given probability

whenever it intends to move to another vertex during plan execution. We propose several decentralized execution policies to guarantee safe execution of MAPF solutions under such delay uncertainty. Decentralized execution policies use a GO or STOP command to control, at each time step, whether an agent should follow its planned path to move to the next vertex or not. For example, a naive policy is to move all agents in locked time steps according to a given MAPF solution, namely it stops all other agents if an agent is delayed, which requires all agents to communicate with each other at every time step. We proposed *Minimal Communication Policy (MCP)* to make the execution more efficient. The key idea of MCP is to respect the precedence constraints that, if two different agents visit the same vertex, they have to visit it in the same order as specified by the given MAPF solution. To do so, MCP constructs a directed acyclic graph whose nodes represent events of agents visiting vertices and whose arcs represent precedence constraints. During execution, MCP thus stops an agent only when a precedence constraint is not satisfied until it receives a signal from another agent for the constraint. MCP also attempts to minimize the number of arcs between events of different agents, namely precedence constraints between agents, and thus the communication cost. Furthermore, we have developed a version of CBS that computes MAPF solutions with small expected makespan for given delay probabilities, assuming that MCP is used for plan execution. In our most recent work (Li et al., 2021a), we combine MAPF algorithms and MCP to develop a software that won the NeurIPS-20 Flatland Challenge[1], a railway scheduling competition which was held in partnership with German, Swiss, and French railway companies. Our software uses automated planning and combinatorial search techniques only but outperformed all other entries, including all reinforcement learning entries, to win overall first place in both rounds of the competition in the top machine learning conference NeurIPS 2020. The problem in the competition is a MAPF variant similar to above where the agents can be stopped at a random time step for a random duration during plan execution.

We have also extended the above idea to a hierarchical framework (Ma et al., 2017a) for plan generation and execution to account for kinematic constraints of agents during planning and other system dynamics during plan execution. This framework uses a post-processing procedure called *MAPF-POST* (Hönig et al., 2016a) that transforms a MAPF solution into a plan-execution schedule in continuous time using a simple temporal network (STN). The STN is a also directed acyclic graph similar to the one constructed by MCP. The STN takes into account important kinematic constraints such as various edge lengths, different agent sizes, and translational and rotational velocity limits of agents by representing the kinematic constraints as temporal constraints between agents, which are arcs annotated with time bounds in the directed acyclic graph, in the computation. The resulting plan-execution schedule guarantees a user-specified safety distance among agents and avoids replanning, namely resolving MAPF problems, in many cases during execution. We have verified our framework using different types of simulated and real robots. The pipeline of our framework is as follows. First, our framework runs an optimal MAPF algorithm such as CBS to compute a MAPF solution. Second, it then uses MAPF-POST to construct an STN that transforms the MAPF solution into a plan-execution schedule, which specifies the expected execution time when a robot should arrive at each location. Third, all robots move along their paths to meet the expected execution time at each location. If the execution deviates from the plan in the third step, the framework constructs a new STN based on the real execution times and computes a new plan-execution schedule for the rest of the plan. The framework needs to perform the first step to replan, namely to solve a new MAPF instance, only when MAPF-POST fails to compute a plan-execution schedule in the second step. In practice, our experimental results show that the robots can almost always execute the given MAPF solution safely without replanning.

Combined Target Assignment and Path Finding

Many real-world applications of multi-agent systems require the coordination of not only path-planning operations but also target-assignment operations. For example, an automated warehouse system needs to decide which robots to deliver which parcels and on what routes the robots should move. In our recent work (Ma and Koenig, 2016), we formalize and study a variant of MAPF called TAPF that couples the target-assignment and the path-finding problems for multiple teams of agents. In TAPF, agents are partitioned into teams and each team is given the same number of target vertices as there are agents in the team. The problem of TAPF is to assign the target vertices of each team to agents in the same team and plan collision-free paths for the agents to their target vertices so that each agent reaches exactly one target vertex and each target vertex is reached by an agent. Existing AI research has considered only two extremes of TAPF. On one hand, the special case of TAPF with one team of agents can be solved optimally in polynomial time (Yu and LaValle, 2013a) by solving a max-flow problem. On the other hand, MAPF algorithms assume that each agent forms a single-agent team, namely the agent is assigned the only target vertex in its team. It remains unclear how and how well one can solve the general case of TAPF with multiple teams of agents.

Therefore, we have developed an optimal TAPF algorithm called Conflict-Based MinCost Flow (CBM) (Ma and Koenig, 2016). CBM breaks TAPF down to the NP-hard sub-problem of coordinating different teams of agents and the polynomial-time solvable sub-problems of coordinating the agents in every team. It then tackles these sub-problems by using a combination of CBS for the NP-hard sub-problem and a min-cost max-flow algorithm (Goldberg and Tarjan, 1987) for the polynomial-time solvable sub-problems. Our experimental results demonstrate that CBM can compute optimal TAPF solutions for more than 400 agents in minutes of runtime, showcasing its potential for large-scale multi-agent systems. In our empirical study (Hönig et al., 2016b), we verify CBM using both simulated and real robots. We demonstrate how CBM can be combined with MAPF-POST to generate plan-execution schedules that allow for the safe execution of TAPF on these real-world agents. We apply the resulting techniques to solve a formation control problem where drones in different colors move to desired locations to display an English word in 3D space with each letter in a unique color.

Long-Term Task and Path Planning

Agents in many multi-agent systems need to constantly attend to new tasks after they finish their current tasks. For example, a warehouse robot in an Amazon fulfillment center (Wurman, D'Andrea, and Mountz, 2008) needs to pick up and deliver another inventory shelf after it finishes delivering its current inventory shelf. Existing AI research on MAPF and TAPF has focused mostly on *one-shot* problems only where each agent has one target vertex and the agents stop moving after they all reach their target vertices. Therefore, in our recent work (Ma et al., 2017b), we formalize and study MAPD that generalizes the one-shot problems MAPF and TAPF to a *long-term* problem. In MAPD, agents have to attend to a stream of incoming tasks. Each task enters the system at an unknown time and is characterized by two target vertices, namely a pickup vertex and a delivery vertex. A free agent, namely one that is currently not executing any task, can be assigned an unexecuted task. To execute the task, the agent has to first move from its current vertex to the pickup vertex of the task, become occupied and start to execute the task upon reaching the pickup vertex of the task, and then move from the pickup vertex to the delivery vertex of the task, while avoiding collisions with other agents.

There are three benefits of modeling ''pickup-and-delivery'' tasks that have an intermediate target vertex (the pickup vertex) and a final target vertex (the delivery vertex) each instead of ''navigation'' tasks that have only one target vertex each. 1. Modeling "pickup-and-delivery" tasks results in a mix of both single-agent teams (each consisting of an occupied agent) and a team of free agents, while modeling "navigation" tasks results in only one team of agents, which limits its generalizability. 2. The resulting MAPD algorithms still apply directly to

''navigation'' tasks because each such task is a special case of a ''pickup-and-delivery'' task with the pickup and delivery vertices being the same vertex. (c) Modeling "pickup-and-delivery" tasks also makes it easy to explain the resulting MAPD algorithms, even though these algorithms can easily be generalized to cases where the tasks have multiple (ordered) intermediate target vertices and a subsequent final target vertex each.

We have considered an *online* setting (Ma et al., 2017b) of MAPD where tasks can enter the system at any time and are not known until they have been added to the system. We develop decentralized and centralized MAPD algorithms that are deadlock-free. The key idea of these online MAPD algorithms is to decouple the long-term problem of MAPD into a sequence of one-shot sub-problems at each time and repeatedly apply task-assignment, MAPF, and TAPF algorithms to these sub-problems. Experimental results show that these MAPD algorithms can determine the tasks and paths for 500 agents in seconds of computation time. We also demonstrate how one of our MAPD algorithms can account for kinematic constraints of robots directly during planning (Ma et al., 2019b) in an online setting. The resulting algorithm can compute an executable solution for 30 minutes of operation of 250 robots and 2,000 tasks in less than 10 seconds of computation time, which is more efficient than using MAPF-POST in a post-processing phase.

We have also considered an *offline* setting (Liu et al., 2019) of MAPD where all tasks are known a priori, thus affording opportunity to optimize the order in which agents execute tasks. We develop an offline MAPD algorithm that models the task scheduling problem as a specialized asymmetric version of the Traveling Salesman problem to compute a chronologically ordered task sequence for each agent. Experimental results show that the offline MAPD algorithm can compute solutions where agents finish all tasks by up to 46% sooner than online MAPD algorithms.

Finally, we have applied MAPD algorithms to an application of parcel sorting with warehouse robots in an automated sortation center(Kou et al., 2020) where the robots need to move to sorting stations to obtain an express parcel and deliver it to a correct sorting bin associated with the postal code of the shipping address of the parcel. A machine in each sorting station scans the shipping address of a parcel, determines the sorting bin the parcel should be delivered to, and load the parcel onto a robot as long as there are robots waiting in the queue of the sorting station. Our goal is to optimize the idle time of the sorting stations, namely the duration when there are no robots queuing, since it is often the throughput bottleneck of such automated sortation centers. The problem is to assign sorting stations to robots that are not delivering parcels and plan paths for all the robots. We develop an algorithm that solves the TAPF sub-problem for robots that are not delivering parcels and solves the MAPF sub-problem for robots that are delivering parcels. We test our algorithm using an industrial simulator with real-world data of online orders. Experimental results show that our algorithm can make decisions for 350 agents in no more than 2 seconds of computation time and improve throughput (measured in the average number of parcels obtained by robots per second) of a sortation center by up to 12%.

Summary

We described our research on using intelligent planning techniques to tackle multi-agent coordination problems. We outlined four directions that generalize task-assignment and MAPF research to real-world applications of large-scale multi-agent systems. For our ongoing research, we are currently developing a deeper theoretical understanding of using MAPF algorithms for long-term autonomy of such systems (Ma, 2021), a learning-based distributed MAPF algorithm (Ma, Luo, and Ma, 2021), and algorithms that can jointly solve MAPF and complex task-planning problems (Zhong et al., 2022; Xu et al., 2022). We hope that researchers working in this area can benefit from the insights provided in this article.

Acknowledgment

This work was supported by NSERC under grant number RGPIN2020-06540 as well as a Canada Foundation for Innovation John R. Evans Leaders Fund award and a gift from Huawei Technologies Canada.

Conflict of Interest

The author declares that there is no conflict.

Notes

[1]https://discourse.aicrowd.com/t/neurips-2020-flatland-winners/4010

References

Cohen, L.; Greco, M.; Ma, H.; Hernandez, C.; Felner, A.; Kumar, T. K. S.; and Koenig, S. 2018. Anytime focal search with applications. In *International Joint Conference on Artificial Intelligence*, 1434--1441.

Erdem, E.; Kisa, D. G.; Oztok, U.; and Schueller, P. 2013. A general formal framework for pathfinding problems with multiple agents. In *AAAI Conference on Artificial Intelligence*, 290--296.

Felner, A.; Stern, R.; Shimony, S. E.; Boyarski, E.; Goldenberg, M.; Sharon, G.; Sturtevant, N. R.; Wagner, G.; and Surynek, P. 2017. Search-based optimal solvers for the multi-agent pathfinding problem: Summary and challenges. In *International Symposium on Combinatorial Search*, 29--37.

Felner, A.; Li, J.; Boyarski, E.; Ma, H.; Cohen, L.; Kumar, T. K. S.; and Koenig, S. 2018. Adding heuristics to conflict-based search for multi-agent pathfinding. In *International Conference on Automated Planning and Scheduling*, 83--87.

Goldberg, A. V., and Tarjan, R. E. 1987. Solving minimum-cost flow problems by successive approximation. In *Annual ACM Symposium on Theory of Computing*, 7--18.

Hönig, W.; Kumar, T. K. S.; Cohen, L.; Ma, H.; Xu, H.; Ayanian, N.; and Koenig, S. 2016a. Multi-agent path finding with kinematic constraints. In *International Conference on Automated Planning and Scheduling*, 477--485.

Hönig, W.; Kumar, T. K. S.; Ma, H.; Ayanian, N.; and Koenig, S. 2016b. Formation change for robot groups in occluded environments. In *IEEE/RSJ International Conference on Intelligent Robots and System*, 4836--4842.

Kou, N. M.; Peng, C.; Ma, H.; Kumar, T. K. S.; and Koenig, S. 2020. Idle time optimization for target assignment and path finding in sortation centers. In *AAAI Conference on Artificial Intelligence*, 9925--9932.

Li, J.; Boyarski, E.; Felner, A.; Ma, H.; and Koenig, S. 2019a. Improved heuristics for multi-agent path finding with conflict-based search. In *International Joint Conference on Artificial Intelligence*, 442--449.

Li, J.; Harabor, D.; Stuckey, P. J.; Felner, A.; Ma, H.; and Koenig, S. 2019b. Disjoint splitting for conflict-based search for multi-agent path finding. In *International Conference on Automated Planning and Scheduling*, 279--283.

Li, J.; Harabor, D.; Stuckey, P. J.; Ma, H.; and Koenig, S. 2019c. Symmetry-breaking constraints for grid-based multi-agent path finding. In *AAAI Conference on Artificial Intelligence*, 6087--6095.

Li, J.; Surynek, P.; Felner, A.; Ma, H.; Kumar, T. K. S.; and Koenig, S. 2019d. Multi-agent path finding for large agents. In *AAAI Conference on Artificial Intelligence*, 7627--7634.

Li, J.; Gange, G.; Harabor, D.; Stuckey, P. J.; Ma, H.; and Koenig, S. 2020a. New techniques for pairwise symmetry breaking in multi-agent path finding. In *International Conference on Automated Planning and Scheduling*, 193--201.

Li, J.; Sun, K.; Ma, H.; Felner, A.; Kumar, T. K. S.; and Koenig, S. 2020b. Moving agents in formation in congested environments. In *International Conference on Autonomous Agents and Multiagent Systems*, 726--734.

Li, J.; Chen, Z.; Zheng, Y.; Chan, S.-H.; Harabor, D.; Stuckey, P. J.; Ma, H.; and Koenig, S. 2021a. Scalable rail planning and replanning: Winning the 2020 flatland challenge. In *International Conference on Automated Planning and Scheduling*, 477--485.

Li, J.; Harabor, D.; Stuckey, P. J.; Ma, H.; Gange, G.; and Koenig, S. 2021b. Pairwise symmetry reasoning for multi-agent path finding search. *Artifical Intelligence* 301:103574.

Liu, M.; Ma, H.; Li, J.; and Koenig, S. 2019. Target assignment and path planning for multi-agent pickup and delivery. In *International Conference on Autonomous Agents and Multiagent Systems*, 2253--2255.

Ma, H., and Koenig, S. 2016. Optimal target assignment and path finding for teams of agents. In *International Conference on Autonomous Agents and Multiagent Systems*, 1144--1152.

Ma, H., and Koenig, S. 2017. AI buzzwords explained: Multi-agent path finding (MAPF). *AI Matters* 3(3):15--19.

Ma, H.; Koenig, S.; Ayanian, N.; Cohen, L.; Hönig, W.; Kumar, T. K. S.; Uras, T.; Xu, H.; Tovey, C.; and Sharon, G. 2016a. Overview: Generalizations of multi-agent path finding to real-world scenarios. In *IJCAI-16 Workshop on Multi-Agent Path Finding*.

Ma, H.; Tovey, C.; Sharon, G.; Kumar, T. K. S.; and Koenig, S. 2016b. Multi-agent path finding with payload transfers and the package-exchange robot-routing problem. In *AAAI Conference on Artificial Intelligence*, 3166--3173.

Ma, H.; Hönig, W.; Cohen, L.; Uras, T.; Xu, H.; Kumar, T. K. S.; Ayanian, N.; and Koenig, S. 2017a. Overview: A hierarchical framework for plan generation and execution in multi-robot systems. *IEEE Intelligent Systems* 32(6):6--12.

Ma, H.; Li, J.; Kumar, T. K. S.; and Koenig, S. 2017b. Lifelong multi-agent path finding for online pickup and delivery tasks. In *International Conference on Autonomous Agents and Multiagent Systems*, 837--845.

Ma, H.; Wagner, G.; Felner, A.; Li, J.; Kumar, T. K. S.; and Koenig, S. 2018a. Multi-agent path finding with deadlines. In *International Joint Conference on Artificial Intelligence*, 417--423.

Ma, H.; Wagner, G.; Felner, A.; Li, J.; Kumar, T. K. S.; and Koenig, S. 2018b. Multi-agent path finding with deadlines: Preliminary results. In *International Conference on Autonomous Agents and Multiagent Systems*, 2004--2006.

Ma, H.; Harabor, D.; Stuckey, P. J.; Li, J.; and Koenig, S. 2019a. Searching with consistent prioritization for multi-agent path finding. In *AAAI Conference on Artificial Intelligence*, 7643--7650.

Ma, H.; Hönig, W.; Kumar, T. K. S.; Ayanian, N.; and Koenig, S. 2019b. Lifelong path planning with kinematic constraints for multi-agent pickup and delivery. In *AAAI Conference on Artificial Intelligence*, 7651--7658.

Ma, H.; Kumar, T. K. S.; and Koenig, S. 2017. Multi-agent path finding with delay probabilities. In *AAAI Conference on Artificial Intelligence*, 3605--3612.

Ma, Z.; Luo, Y.; and Ma, H. 2021. Distributed heuristic multi-agent path finding with communication. In *IEEE International Conference on Robotics and Automation*, (in press).

Ma, H. 2021. A competitive analysis of online multi-agent path finding. In *International Conference on Automated Planning and Scheduling*, 234--242.

Ma, H. 2022. Graph-based multi-robot path finding and planning. *Current Robotics Reports* 1--8.

Morris, R.; Pasareanu, C.; Luckow, K.; Malik, W.; Ma, H.; Kumar, T. K. S.; and Koenig, S. 2016. Planning, scheduling and monitoring for airport surface operations. In *AAAI-16 Workshop on Planning for Hybrid Systems*, 608--614.

Sharon, G.; Stern, R.; Goldenberg, M.; and Felner, A. 2013. The increasing cost tree search for optimal multi-agent pathfinding. *Artificial Intelligence* 195:470--495.

Sharon, G.; Stern, R.; Felner, A.; and Sturtevant, N. R. 2015. Conflict-based search for optimal multi-agent pathfinding. *Artificial Intelligence* 219:40--66.

Standley, T. S., and Korf, R. E. 2011. Complete algorithms for cooperative pathfinding problems. In *International Joint Conference on Artificial Intelligence*, 668--673.

Stern, R.; Sturtevant, N. R.; Felner, A.; Koenig, S.; Ma, H.; Walker, T. T.; Li, J.; Atzmon, D.; Cohen, L.; Kumar, T. K. S.; Boyarski, E.; and Barták, R. 2019. Multi-agent pathfinding: Definitions, variants, and benchmarks. In *International Symposium on Combinatorial Search*, 151--159.

Surynek, P.; Felner, A.; Stern, R.; and Boyarski, E. 2016. Efficient SAT approach to multi-agent path finding under the sum of costs objective. In *European Conference on Artificial Intelligence*, 810--818.

Surynek, P. 2010. An optimization variant of multi-robot path planning is intractable. In *AAAI Conference on Artificial Intelligence*, 1261--1263.

Tovey, C. 1984. A simplified NP-complete satisfiability problem. *Discrete Applied Mathematics* 8:85--90.

Wagner, G., and Choset, H. 2015. Subdimensional expansion for multirobot path planning. *Artificial Intelligence* 219:1--24.

Wurman, P. R.; D'Andrea, R.; and Mountz, M. 2008. Coordinating hundreds of cooperative, autonomous vehicles in warehouses. *AI Magazine* 29(1):9--20.

Xu, Q.; Li, J.; Koenig, S.; and Ma, H. 2022. Multi-goal multi-agent pickup and delivery. In *IEEE/RSJ International Conference on Intelligent Robots and System*, in press.

Yu, J., and LaValle, S. M. 2013a. Multi-agent path planning and network flow. In *Algorithmic Foundations of Robotics X, Springer Tracts in Advanced Robotics*, volume 86. Springer. 157--173.

Yu, J., and LaValle, S. M. 2013b. Planning optimal paths for multiple robots on graphs. In *IEEE International Conference on Robotics and Automation*, 3612--3617.

Yu, J., and LaValle, S. M. 2013c. Structure and intractability of optimal multi-robot path planning on graphs. In *AAAI Conference on Artificial Intelligence*, 1444--1449.

Zhong, X.; Li, J.; Koenig, S.; and Ma, H. 2022. Optimal and bounded-suboptimal multi-goal task assignment and path finding. In *IEEE International Conference on Robotics and Automation*, 10731--10737.

Hang Ma is an Assistant Professor in the School of Computing Science at Simon Fraser University. His research interests lie in the intersection of artificial intelligence, robotics, and machine learning. Specifically, he is interested in topics on automated planning, multi-agent/robot systems, spatio-temporal and constraint reasoning, and applications of probabilistic methods. His research work has won multiple awards, including the ICAPS 2016 Outstanding Paper Award in the Robotics Track, a runner-up for the 2020 Victor Lesser Distinguished Dissertation Award, the ICAPS 2021 Best Dissertation Award, and the ICAPS 2021 Best System Demonstration Award.