Proceedings of the Fourteenth International Symposium on Combinatorial Search (SoCS 2021)

Cooperative Multi-Agent Path Finding: Beyond Path Planning and Collision Avoidance

Nir Greshler,[1] Ofir Gordon, [2] Oren Salzman, [2] Nahum Shimkin [1]
[1] Viterbi Faculty of Electrical & Computer Engineering, Technion, Haifa, Israel
[2] Department of Computer Science, Technion, Haifa, Israel
nirgreshler@campus.technion.ac.il, ofirgo@campus.technion.ac.il, osalzman@cs.technion.ac.il, shimkin@ee.technion.ac.il

Abstract

We introduce the *Cooperative Multi-Agent Path Finding (Co-MAPF)* problem, an extension to the classical MAPF problem, where cooperative behavior is incorporated. In this setting, a group of autonomous agents operate in a shared environment and have to complete *cooperative tasks* while avoiding collisions with the other agents in the group. This extension naturally models many real-world applications, where groups of agents are required to collaborate in order to complete a given task. To this end, we formalize the Co-MAPF problem and introduce *Cooperative Conflict-Based Search (Co-CBS)*, a CBS-based algorithm for solving the problem optimally for a wide set of Co-MAPF problems. Co-CBS uses a cooperation-planning module integrated into CBS such that cooperation planning is decoupled from path planning. Finally, we present empirical results on several MAPF benchmarks demonstrating our algorithm's properties.

Introduction and Setting

While the classical MAPF problem is inherently cooperative, we term the setting of Co-MAPF as *truly cooeprative*, as we may want agents not just to "not interrupt" each other, but also help each other achieve their goals. This extension naturally models many real-world applications, where groups of *heterogeneous* agents are required to collaborate in order to complete a given task.

Our motivating problem is taken from the warehouse-automation domain (Wurman, D'Andrea, and Mountz 2008). In this problem, storage locations host inventory pods that hold goods of different kinds. Robots operate autonomously in the warehouse, picking up and carrying inventory pods to designated drop-off locations, where goods are manually taken off the pods for packaging. In this scenario, the robot's main task is to transport the pods around the warehouse. Research in a different, yet closely-related area, has studied the problem of autonomous robotic arms capable of picking-up a specific item from an inventory pod. This motivates the investigation of an improved warehouse scenario, where robots of two types (namely, grasp and transfer robots), can work together in coordination (for example, by scheduling a meeting between them) to improve some optimization objective. This motivating example is depicted in Figure 1.

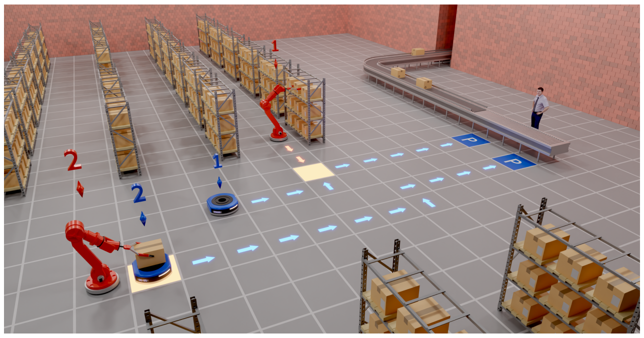

Figure 1: Two pairs of robots operate in a warehouse–two grasp units and two transfer units. Grasp unit #1 arrived at the task start location, i.e., next to the shelf. It will pick up the box and then drive to the meeting location (marked with a yellow square) to transfer the box to transfer unit #1. The transfer unit has a path (marked with blue arrows) to the meeting point, and from there to the task goal (the P square), where the box will be picked by a human employee. The second pair of robots (#2) are at their meeting location.

We now formalize the Co-MAPF problem and then introduce *Cooperative Conflict-Based Search (Co-CBS)*, a CBS-based algorithm for solving the problem optimally for a wide set of Co-MAPF problems. More details can be found in our full paper (Greshler et al. 2021).

Based on the MAPF problem formulation, in the Co-MAPF problem we are given an undirected graph $G = (V, E)$, and a set of agents A that consists of two distinguishable sets, i.e., $A = \mathcal{A} \cup \mathcal{B}$. Each set includes k agents of a specific type, namely $\mathcal{A} = \{\alpha_1, \ldots, \alpha_k\}$ and $\mathcal{B} = \{\beta_1, \ldots, \beta_k\}$. The two types of agents may differ in their traversal capabilities or possible actions in a location (for instance, picking up an object). We are also given a set of tasks $\mathcal{T} = \{\tau_1, \ldots, \tau_k\}$ s.t. each task τ_i is assigned to a pair of agents (α_i, β_i). We refer to α_i and β_i as the *initiator* and *executor* agents, respectively. Each task $\tau_i \in \mathcal{T}$ is defined by a start location s_i and a goal location g_i.

The cooperation between agents is restricted to the form of *meetings*, where agents have to schedule a meeting location and time to complete a task. Specifically, a task $\tau_i = (s_i, g_i)$ assigned to agents (α_i, β_i) is composed of the following steps: (i) moving the initiator agent α_i to the task's start location s_i, (ii) moving both agents to a so-called meeting $m_i = (v_i^m, t_i^m)$ where $v_i^m \in V$ is the meeting location and t_i^m is the meeting time step, both of which are computed by the algorithm (and not specified by the task), (iii) moving the executor agent to the task's goal location g_i.

Cooperative Conflict-Based Search

Co-CBS is an optimal three-level algorithm for solving the introduced Co-MAPF problem, based on two previously-suggested optimal algorithms: the well-known Conflict-Based Search (CBS) (Sharon et al. 2015) for solving a classical MAPF problem and the Conflict-Based Search with Optimal Task Assignment (CBS-TA) (Hönig et al. 2018) for solving the anonymous MAPF problem, where we also need to assign goals (or tasks) to each agent.

Co-CBS considers the cooperative aspect of the problem and consists of three levels of search in three different spaces: (i) the *meetings space*, (ii) the *conflicts space* and (iii) the *paths space*. The meetings space contains all possible combinations of meetings, one for each task.

Co-CBS simultaneously searches over all possible meetings and for each meeting, over all possible paths. To perform this search in a systematic and efficient manner, we need to consider an *ordering* of the meetings. This is done by defining a meeting's cost which is dependent both on the meeting's location and time. To efficiently traverse the set of possible meetings, we introduce the notion of a *Meetings Table* which stores for each meeting location the currently-best meeting time. This table will allow us to iterate over all meetings in a best-first manner, thus guarantee the obtained solution is optimal.

In contrast to CBS that constructs a single conflicts-tree (CT), Co-CBS creates a forest of CTs, similar to (Hönig et al. 2018). Each CT starts in a *root* node and corresponds to a specific set of meetings (a specific meeting for each task).

Co-CBS starts with a single root node, with the optimal set of meetings, while ignoring possible conflicts between agents. In each iteration, a lowest-cost node is selected from the OPEN list (either a root or regular node), in a best-first approach similar to CBS. Whenever a root node is selected, in addition to splitting the tree due to a conflict, Co-CBS also expands it in the meetings space by generating the next best sets of meetings. Namely, new root nodes are created only on demand. For each expanded node, given its set of meetings and constraints, Co-CBS computes a solution by planning the different steps a task solution is composed of.

Experimental Evaluation

We evaluated Co-CBS on several 2D grid-based benchmarks (Sturtevant 2012; Stern et al. 2019). Co-CBS is implemented in Python[1]. All simulations were performed on an Intel Xeon Platinum 8000 @ 3.1Ghz machine with 64.0 GB RAM.

[1]Our code is publicly available at: https://github.com/CRL-Technion/Cooperative-MAPF.

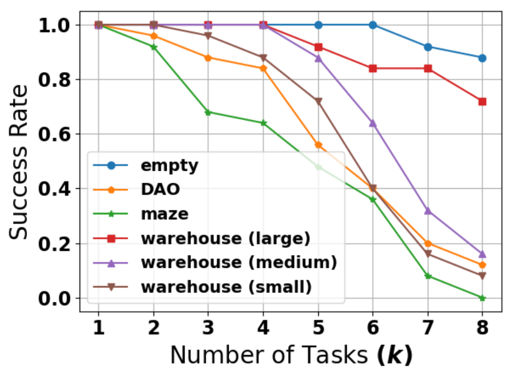

Figure 2: Success rates for the sum-of-costs objective.

More specifically, we tested the algorithm on maps of different types and sizes–empty grid (*empty-48-48*), dense game map (*DAO, den312d*), a maze (*maze-32-32-4*), large warehouse (*warehouse-10-20-10-2-1*) and custom small (20×10) and medium (26×15) warehouses. We ran 25 random scenarios for each benchmark both for the *makepsan* and *sum-of-costs* objectives with the number of tasks ranging from one task (two agents) to eight tasks (16 agents) and with a timeout of ten minutes each.

We present empirical results with a sum-of-costs objective. The sum of costs is the sum of time steps required by each agent, to complete all tasks. This objective is arguably more natural for our setting—it implicitly minimizes both the time it takes to complete a task, and the time the initiator finishes its part in the task.

Figure 2 shows the success rates of Co-CBS, successfully solving more than 80% of the instances (excluding the maze benchmark) given four tasks. The success rates drops below 80% for five tasks or more on smaller and denser maps (maze, DAO, small warehouse). In large and sparse environments (large warehouse and empty map), a feasible solution is quickly found, usually using the first set of meetings. The search in these cases is equivalent to running CBS with the first set of meetings. In smaller and denser maps, a more exhaustive meeting-space search is required to find an optimal solution, causing the success rates to drop.

Discussion and Future Work

We introduced the Cooperative Multi-Agent Path Finding (Co-MAPF) problem, an extension to the classical MAPF problem that incorporates cooperative behavior to agents. We introduced Co-CBS, a three-level search algorithm that optimally solves Co-MAPF instances. Please refer to (Greshler et al. 2021) for the full paper.

Many possible improvements for Co-CBS exist, such as improving the search in the meetings space, reusing information between conflict trees and applying existing CBS enhancements to Co-CBS. The Co-MAPF framework can also be extended to other forms of cooperative interaction and varying number of collaborating agents. Co-MAPF also presents a challenging *task-assignment* problem, that may be further investigated in this context.

References

Greshler, N.; Gordon, O.; Salzman, O.; and Shimkin, N. 2021. Cooperative Multi-Agent Path Finding: Beyond Path Planning and Collision Avoidance. *Computing Research Repository (CoRR)* abs/2105.10993.

Hönig, W.; Kiesel, S.; Tinka, A.; Durham, J. W.; and Ayanian, N. 2018. Conflict-Based Search with Optimal Task Assignment. In *Int. Conf. on Autonomous Agents and MultiAgent Systems (AAMAS)*, 757–765.

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

Stern, R.; Sturtevant, N. R.; Felner, A.; Koenig, S.; Ma, H.; Walker, T. T.; Li, J.; Atzmon, D.; Cohen, L.; Kumar, T. K. S.; Barták, R.; and Boyarski, E. 2019. Multi-Agent Pathfinding: Definitions, Variants, and Benchmarks. In *Int. Symp. on Combinatorial Search (SOCS)*, 151–159.

Sturtevant, N. 2012. Benchmarks for Grid-Based Pathfinding. *Transactions on Computational Intelligence and AI in Games* 4(2): 144 – 148. URL http://web.cs.du.edu/~sturtevant/papers/benchmarks.pdf.

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