Optimal Task Assignment and Path Planning using Conflict-Based Search with Precedence and Temporal Constraints

Extended Abstract

Yu Quan Chong
Carnegie Mellon University
Pittsburgh, PA, USA
yuquanc@andrew.cmu.edu

Jiaoyang Li
Carnegie Mellon University
Pittsburgh, PA, USA
jiaoyangli@cmu.edu

Katia Sycara
Carnegie Mellon University
Pittsburgh, PA, USA
sycara@andrew.cmu.edu

ABSTRACT

This paper examines the Task Assignment and Path Finding with Precedence and Temporal Constraints (TAPF-PTC) problem. We augment Conflict-Based Search (CBS) to generate task assignments and collision-free paths that adhere to precedence and temporal constraints for agents to maximize a user-defined objective.

KEYWORDS

Multi-Agent Task Assignment; Multi-Agent Path Finding; Precedence Constraints; Temporal Constraints

ACM Reference Format:
Yu Quan Chong, Jiaoyang Li, and Katia Sycara. 2024. Optimal Task Assignment and Path Planning using Conflict-Based Search with Precedence and Temporal Constraints: Extended Abstract. In *Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 3 pages.

1 INTRODUCTION

The Multi-Agent Path Finding (MAPF) problem [7] entails finding collision-free paths for a set of agents, guiding them from their start to goal locations. However, MAPF does not account for several practical task-related constraints. For example, agents may need to perform actions at goal locations with specific execution times, adhering to predetermined orders and timeframes. Moreover, goal assignments may not be predefined for agents, and the optimization objective may lack an explicit definition. To incorporate task assignment, path planning, and a user-defined objective into a coherent framework, this paper examines the Task Assignment and Path Finding with Precedence and Temporal Constraints (TAPF-PTC) problem. We augment Conflict-Based Search (CBS) [6] to simultaneously generate task assignments and collision-free paths that adhere to precedence and temporal constraints, maximizing an objective quantified by the return from a user-defined reward function in reinforcement learning (RL).

2 PROBLEM DEFINITION

TAPF-PTC is characterised by an undirected graph $G = (V, E)$, a set of N agents $\{a_1, \ldots, a_N\}$, and a task T, which is a set of M goals $\{g^1, \ldots, g^M\}$ with given temporal constraints. Each agent has a start vertex $s_i \in V$. Each goal g^j is a tuple comprising a goal vertex $g^j.v$ and a goal action $g^j.act$ and needs to be assigned to an agent a_i, requiring a_i to reach $g^j.v$ and perform $g^j.act$ for $\alpha(g^j)$ timesteps while waiting there. We use $[g_i^1, \ldots, g_i^{l_i}]$ to denote the sequence of l_i goals assigned to agent a_i; so $\sum_{i=1}^{N} l_i = M$. We further use $\mu(g_i^j)$ and $\tau(g_i^j)$ to denote the timestep when agent a_i start and finish performing action $g_i^j.act$ at vertex $g_i^j.v$, respectively, so $\tau(g_i^j) - \mu(g_i^j) = \alpha(g_i^j)$. A path segment for an agent a_i consists of the sequence of vertices and actions from the completion of g_i^{j-1} to the completion of g_i^j, with the full path being the sequential concatenation of the path segments of l_i goals. A solution to TAPF-PTC consists of a goal assignment and a set of conflict-free paths. Conflicts occur when certain constraints are violated: (1) We use vertex/edge constraints in [6] to avoid collisions, where no two agents can occupy the same vertex/edge at the same timestep. (2) We expand on the precedence constraints defined in [9], which consists of two goals and requires the execution or completion timestep of one to precede the other. (3) We introduce absolute temporal range constraints to fix the execution or completion timestep of goals within specified temporal ranges. (4) We introduce inter-goal temporal constraints to specify an upper bound between the difference in the execution or completion timesteps of two goals.

3 CBS-TA-PTC

We propose Conflict-Based Search with Task Assignment, Precedence, and Temporal Constraints (CBS-TA-PTC), an extension of CBS-TA [3] and CBS-PC [9], as shown in Algorithm 1. In cases where M is significantly larger than N, we partition T into subtasks $\{T_{sub}\}$, forming each T_{sub} as a TAPF-PTC instance and solved by CBS-TA-PTC.

3.0.1 Goal Assignment. As our objective is defined by an RL return, determining the cost of assigning a goal to an agent requires knowledge of the agent's entire path. So optimal assignment algorithms such as Hungarian [5] cannot be applied. Hence, we enumerate all possible combinations of agents and the goals in T_{sub} and generate a root Constraint Tree (CT) node for each of them (line 1–7).

3.0.2 Conflict Resolution. Conflict resolution priority ordering: 1) Absolute temporal range, 2) Precedence, 3) Inter-goal temporal, and 4) Vertex/Edge. Absolute temporal range conflicts are the priority as they are based on temporal ranges independent from other goals. Precedence conflicts take priority over inter-goal temporal conflicts as actions should be performed in the correct order before being correctly spaced apart temporally. It is also likely that resolving

Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand. © 2024 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

Algorithm 1 CBS-TA-PTC()

Input: Graph G, starts $\{s_i\}$, subtask T_{sub}, past solutions
Output: Path for each agent for given T_{sub}

for assignment in Combinations(T_{sub}) **do**
  R ⟵ GenerateRootCTNode(S, past solutions, assignment)
  if R.return is maximum return **then**
    return R.solution
  end if
  insert R to OPEN
end for
while OPEN not empty **do**
  P ⟵ node from OPEN with the highest return
  if P.return is maximum return **then**
    return P.solution
  end if
  conflict ⟵ conflict in P w.r.t conflict resolution order
  constraints ⟵ ResolveConflict(conflict)
  for constraint in constraints **do**
    Q ⟵ GenerateCTNode(S, past solutions, P, constraint)
    insert Q to OPEN
  end for
end while

a precedence conflict would resolve any related inter-goal temporal conflicts. For example, the inter-goal temporal constraint with $\tau(g_{i'}^{j'}) - \tau(g_i^j) \leq t_{inter}$ is denoted by $\langle\tau(g_i^j), \tau(g_{i'}^{j'}), t_{inter}\rangle$. When $\langle\tau(g_i^j), \tau(g_{i'}^{j'}), t_{inter}\rangle$ is violated, i.e., $\tau(g_{i'}^{j'}) - \tau(g_i^j) > t_{inter}$, two child CT nodes with either one of the following constraints are generated, where $\tau(g_{i'}^{j'}) = t'$ in the parent CT node:

(1) $\tau(g_i^j) \geq t' - t_{inter}$: Agent a_i is to complete g_i^j at or after timestep $t' - t_{inter}$. Agent a_i's path in the corresponding child CT node is replanned and the inter-goal temporal conflict is resolved.
(2) $\tau(g_i^j) < t' - t_{inter}$: Agent a_i is to complete g_i^j before timestep $t' - t_{inter}$, which is already satisfied. Agent $a_{i'}$ must complete $g_{i'}^{j'}$ at least one timestep earlier than before when replanning its path in its child CT node, i.e. $\tau(g_{i'}^{j'}) \leq t' - 1$. Hence, $\tau(g_{i'}^{j'})$ is guaranteed to decrease by one timestep.

3.0.3 Low-Level Path Planning. Multi-Label A* (MLA*) [1] is used to generate a solution for a task assignment under the constraints on the CT node. A linear programming module using one of the HiGHS solvers [2, 4] is used to prune CT nodes that are unsolvable given their constraints from various conflicts before the MLA* search.

3.0.4 Theoretical Properties of CBS-TA-PTC. By decomposing the task into subtasks, CBS-TA-PTC is an incomplete and suboptimal algorithm. However, with the entire task as a subtask, CBS-TA-PTC can be shown to be optimal and complete by adapting Theorems 1 and 3 in [6] respectively, with the assumption that every trajectory is a Markov game with a fixed terminal timestep to ensure that the set of constraints that can be added to CT nodes is finite.

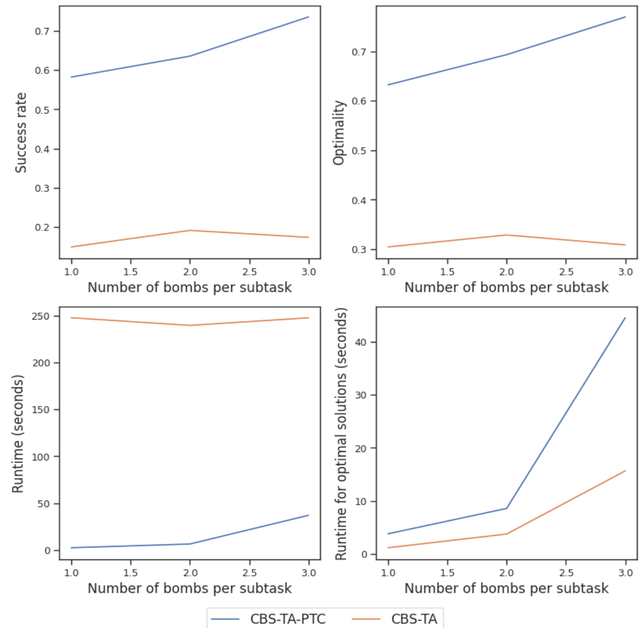

Figure 1: Results showing the success rate, optimality ratio (ratio between return and optimal return) and runtime.

3.1 Results

3.1.1 Environment. Three agents have to defuse all bombs within a time limit. Each bomb has an ordered sequence of three colors with a sequence length of [1, 3], a countdown timer that resets upon a correct defusing step, where the next sequence must be defused before the countdown, and a fuse timer that indicates the time when it must be fully defused. Certain bombs have dependencies, which dictates that the bomb it depends on must be fully defused or have exploded before any defusing steps on itself. Each agent has tools with 2 out of the 3 colors that remove the matching color from the bomb sequence. Failure to adhere to the above would result in the bomb exploding. A fully defused bomb gives a team reward proportionate to the sequence length of the bomb.

3.1.2 Baselines. We augment CBS-TA [3] to maximize return with vertex/edge constraints only, where, given the explosion of a bomb, CT nodes are generated in a naive manner with vertex conflicts for each agent for that timestep. It solves TAPF-PTC with its underlying precedence and temporal conflicts through a more inefficient best-first search through the CT relative to CBS-TA-PTC. For CBS-TA and CBS-TA-PTC, the return is generated by evaluating the solution from the low level, with past solutions from previous subtasks, on an oracle that simulates the environment based on the user-defined reward function and dynamics, which is typically implemented as a user-designed RL environment in practice [8]. Note that a maximum return solution is conflict-free by design.

ACKNOWLEDGMENTS

This work was partially supported by DARPA award HR001120C0036 and AFOSR award FA9550-18-1-0097.

REFERENCES

[1] Florian Grenouilleau, Willem-Jan van Hoeve, and John N Hooker. 2019. A multi-label A* algorithm for multi-agent pathfinding. In *Proceedings of the International Conference on Automated Planning and Scheduling*, Vol. 29. 181–185.

[2] J Hall, I Galabova, L Gottwald, and M Feldmeier. [n.d.]. HiGHS–high performance software for linear optimization.

[3] Wolfgang Hönig, Scott Kiesel, Andrew Tinka, Joseph Durham, and Nora Ayanian. 2018. Conflict-based search with optimal task assignment. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems*.

[4] Qi Huangfu and JA Julian Hall. 2018. Parallelizing the dual revised simplex method. *Mathematical Programming Computation* 10, 1 (2018), 119–142.

[5] Harold W Kuhn. 1955. The Hungarian method for the assignment problem. *Naval research logistics quarterly* 2, 1-2 (1955), 83–97.

[6] Guni Sharon, Roni Stern, Ariel Felner, and Nathan R Sturtevant. 2015. Conflict-based search for optimal multi-agent pathfinding. *Artificial Intelligence* 219 (2015), 40–66.

[7] Roni Stern, Nathan Sturtevant, Ariel Felner, Sven Koenig, Hang Ma, Thayne Walker, Jiaoyang Li, Dor Atzmon, Liron Cohen, TK Kumar, et al. 2019. Multi-agent pathfinding: Definitions, variants, and benchmarks. In *Proceedings of the International Symposium on Combinatorial Search*, Vol. 10. 151–158.

[8] Mark Towers, Jordan K. Terry, Ariel Kwiatkowski, John U. Balis, Gianluca de Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Arjun KG, Markus Krimmel, Rodrigo Perez-Vicente, Andrea Pierré, Sander Schulhoff, Jun Jet Tai, Andrew Tan Jin Shen, and Omar G. Younis. 2023. Gymnasium. https://doi.org/10.5281/zenodo.8127026

[9] Han Zhang, Jingkai Chen, Jiaoyang Li, B Williams, and Sven Koenig. 2022. Multi-Agent Path Finding for Precedence-Constrained Goal Sequences. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems*. 1464–1472.