Multi-Goal Multi-Agent Path Finding via Decoupled and Integrated Goal Vertex Ordering

Pavel Surynek *
Czech Technical University in Prague, Faculty of Information Technology, Thákurova 9, 160 00 Praha 6, Czechia
pavel.surynek@fit.cvut.cz

Abstract

We introduce multi-goal multi-agent path finding (MG-MAPF) which generalizes the standard discrete multi-agent path finding (MAPF) problem. While the task in MAPF is to navigate agents in an undirected graph from their starting vertices to one individual goal vertex per agent, MG-MAPF assigns each agent multiple goal vertices and the task is to visit each of them at least once. Solving MG-MAPF not only requires finding collision free paths for individual agents but also determining the order of visiting agent’s goal vertices so that common objectives like the sum-of-costs are optimized. We suggest two novel algorithms using different paradigms to address MG-MAPF: a heuristic search-based algorithm called Hamiltonian-CBS (HCBS) and a compilation-based algorithm built using the satisfiability modulo theories (SMT), called SMT-Hamiltonian-CBS (SMT-HCBS).

Introduction

Multi-agent path finding (MAPF) (Silver 2005; Ryan 2008; Surynek 2009; Luna and Bekris 2011; Wang and Botea 2011; Ma and Koenig 2017) is an abstraction for many real-life problems where agents, both autonomous or passive, need to be moved (see (Felner et al. 2017; Ma and Koenig 2017) for a survey). The environment in MAPF is modeled as an undirected graph where vertices represent positions and edges define the topology.

The standard variant of MAPF assumes that each agent starts in a specified starting vertex and its task is to reach a specified goal vertex. While such formalization encompass many real-life navigation tasks (Cáp et al. 2013; Ma et al. 2017a) there still exist problems especially in logistic domain where the standard MAPF lacks expressiveness.

Such problems that cannot be expressed as MAPF include situations where agents have multiple goals so that instead of reaching single goal location agents need to perform a round-trip to service a set of goals. Many real-life applications requires that agents perform certain task at each of multiple goal locations such as performing maintenance or pickup operation (Pansart, Catusse, and Cambazard 2018; Briant et al. 2020).

Having multiple goals per agent adds a significant new challenge to the problem consisting of determining the order of visiting agent’s goal vertices. Hence the ordering of goals as well as non-conflicting path finding are subject to decision making which in addition to this aims on the optimization of various objectives such as commonly adopted sum-of-costs (Standley 2010; Sharon et al. 2013).

We introduce a problem we call a multi-goal multi-agent path finding (MG-MAPF). The problem shares movement rules with the standard MAPF but generalizes it by allowing each agent to have multiple goal vertices. Each of the agent’s goal vertices must be visited at least once to successfully solve MG-MAPF.

Contribution

We introduce two novel solving algorithms for MG-MAPF: a search-based Hamiltonian CBS (HCBS), a derivative of the CBS algorithm (Sharon et al. 2015), and a compilation-based algorithm, called SMT-Hamiltonian-CBS, derived from SMT-CBS (Surynek 2019b).

The important feature of HCBS is that it decouples the goal vertex ordering from collision free path finding, making HCBS a three level search algorithm where at the high-level, the standard CBS conflict resolution search is performed and the low level search for collision-free circuit is further divided into two levels. The higher level of circuit-finding determines the goal vertex ordering while the lower level finds a collision free circuit visiting all goals following the order determined by the higher level.

The compilation-based approach uses the ideas from solvers for satisfiability modulo theories (SMT) (Barrett, de Moura, and Stump 2005; Barrett et al. 2009) namely lazy construction of formulae. A significant feature of this approach is that collision avoidance and goal vertex ordering are integrated and solved at once.

The paper is organized as follows: basic definitions from MAPF and its properties are introduced first. Then search-based HCBS and related MG-MAPF specific heuristics are developed. Compilation-based SMT-HCBS follows in the next section. Finally both new algorithms are experimentally evaluated and compared.
Background and Definitions

We first recall the concepts from the standard multi-agent path finding (MAPF) that we inherit into MG-MAPF. Agents in MAPF are placed in vertices of an undirected graph so that there is at most one agent per vertex. Formally, $s : A \mapsto V$ is a configuration of agents in vertices of the graph. A configuration can be transformed instantaneously to the next one by valid movements of agents; the next configuration corresponds to the next time step. An agent can move into another vertex across an edge provided no collision occurs. A collision occurs if two agents are located in the same vertex (vertex collision) or if two agents simultaneously cross the same edge in opposite directions (edge collision) (Sharon et al. 2015). The configuration at time step $t$ is denoted $s_t$.

Definition 1 Multi-goal multi-agent path finding (MG-MAPF) is a 4-tuple $\Theta = (G = (V, E), A, s_0, g)$ where $G = (V, E)$ is an undirected graph, $A = \{a_1, a_2, ..., a_k\}$ with $k \in \mathbb{N}$ is a set of agents where $k \leq |V|$, $s : A \mapsto V$ represents agents starting vertices (starting configuration), and $g : A \mapsto 2^V$ assigns a set of goal vertices to each agent.

Each agent in MG-MAPF has the task to visit its goal vertices. Agent’s goal vertices can be visited in an arbitrary order but each goal vertex must be visited by the agent at least once. Various objectives can be taken into account. We develop all concepts here for the sum-of-costs objective commonly adopted in MAPF but different cumulative objectives can be used as well (Surynek et al. 2016b).

Formally, an MG-MAPF solution is a sequence of configurations that can be obtained via valid moves from the starting configuration $s_0$ following the MAPF movement rules such that each agent visits each of its goal vertices at least once:

Definition 2 A solution to MG-MAPF $\Theta = (G = (V, E), A, s_0, g)$ is a sequence of configurations $S = \{s_0, s_1, ..., s_t_M\}$ such that $s_i$ results from $s_{i-1}$ for all $i \in \{1, 2, ..., t_M\}$ via valid MAPF movements and given that $g(a_i) = V$ it holds that $(\forall v_g \in g(a_i))(\exists h \in \{0, 1, ..., t_h\})(s_t(a_i) = v_g)$.

Given a solution $S = \{s_0, s_1, ..., s_t_M\}$, $t_M$ is the makespan, denoted $\text{Make}(S)$. We define the sum-of-costs as the sum of costs for individual agents: $\text{SoC}(S) = \sum_{i=1}^{k} \text{Cost}(a_i)$, where the individual cost for agent $a_i$ is defined as: $\text{Cost}(a_i) = \min \{t_c | (\forall v_g \in g(a_i))(\exists h \in \{0, 1, ..., t_h\})(s_t(a_i) = v_g)\}$.

Let us note that the $\text{SoC}(S)$ accumulates unit costs of actions including wait actions until all goals are visited. After that we do not care about agent’s movements. This is one possible definition of MG-MAPF. We can alternatively require that all agents should return to their starting vertices and/or stay in the final goal. The developed solving algorithms require only minor modifications to be applicable for any of such MG-MAPF variants.

MG-MAPF as well as MAPF is NP-hard (Ratner and Warmuth 1990; Surynek 2010; Yu and LaValle 2013, 2015). A difference can be observed for a special case with one agent. Finding a sum-of-costs optimal solution to MAPF with one agent is in P as it corresponds to finding a shortest path in $G$. On the other hand sum-of-costs optimal MG-MAPF with one agent is still NP-hard as it corresponds to finding a Hamiltonian path (Rahman and Kaykobad 2005). An example of MG-MAPF is shown in Figure 1.

Hamiltonian Conflict-based Search: HCBS

We suggest a novel algorithm called Hamiltonian Conflict-based Search (Hamiltonian CBS, HCBS) which shares the high level structure with the CBS algorithm (Sharon et al. 2015), one of the most commonly used algorithm for the standard MAPF (Li et al. 2019b,a; Zhang et al. 2020; Li et al. 2020; Boyarski et al. 2020).

When trying to use CBS for MG-MAPF, the significant challenge is represented by the fact that at the low level there is no longer search for a minimum cost path with respect to the set of conflicts, a polynomial-time problem, but rather the search for a minimum cost Hamiltonian path (which even without a set of conflicts is a difficult problem).

Definition 3 A Hamiltonian path (HP) in $G$ starting at $u \in V$ covering a subset of vertices $U \subseteq V$ is a sequence of vertices denoted $H_P(u, U) = \{h_0, h_1, ..., h_{t_H}\}$ such that $h_1 = u, \{h_t, h_{t+1}\} \in E$ for $t \in \{0, 1, ..., t_H - 1\}$, and for each $v \in U \exists t \in \{0, 1, ..., t_H\}$ such that $h_t = v$. The cost of Hamiltonian path corresponds to the number of its edges: $\text{Cost}(H_P(u, U)) = t_H - 1$.

Such CBS hence consists of potentially exponential-sized conflict resolution at the high-level where each conflict leads to exponential-time search for a fresh Hamiltonian path, resulting in an algorithm with high exponential factor. Despite prohibitive theoretical complexity such algorithm could be feasible provided that the low level search is fast enough.

Decoupled Goal Ordering

The key to adapt CBS for MG-MAPF is to decouple the goal vertex ordering from conflict avoidance. The search for a Hamiltonian path going through agent’s goal vertices with respect to a set of conflicts is done in two level fashion. At the high-level (of this low level) we are trying to determine optimal ordering of agent’s goal vertices. To this purpose we made use of the A* algorithm (Hart, Nilsson, and Raphael 1968) that searches the space of possible permutations of agent’s goals. After determining the next goal vertex to visit, the algorithm searches for the shortest path observing the conflicts that interconnects the next goal vertex with the current one. The search for the shortest path is done by another instance of A*. Another important factor for the performance of the decoupled approach are the heuristics.
Minimum Spanning Tree Heuristic

We define a variant of spanning tree with respect to a subset of vertices of undirected graph \( G = (V, E) \).

**Definition 4** A spanning tree (ST) of an undirected graph \( G = (V, E) \) with respect to a subset of vertices \( U \subseteq V \), denoted \( T_{S}(U) = (V_{U}, E_{U}) \) is a tree covering \( U \), that is, \( U \subseteq V_{U} \subseteq V \) and \( E_{U} \subseteq E \). The cost of a spanning tree is defined as the number of edges included in the tree: \( \text{Cost}(T_{S}(U)) = |E_{U}| \). A minimum spanning tree (MST) with respect to \( U \) is a spanning tree with minimum cost.

We define a variant of Minimum Spanning Tree Heuristic (MSTH) (line 2-5). At a general step, HCBS takes a node \( u \) and the sum-of-costs \( N:u \) and \( N:h \) to the actual cost of partially constructed Hamiltonian path finishing in \( N:u \), and \( N:h \) is a lower bound estimation of the cost for the remaining part of the Hamiltonian path calculated as the cost of MST starting in \( N:u \) and covering the remaining goals. Finally, \( N:circuits \) is the partial Hamiltonian path itself.

The very low-level search is done by another instance of A* (line 33) which interconnects the current vertex \( N:u \) with the candidate for the next goal \( v \) by a shortest path taking into account conflict avoidance constraints from the very high-level CBS-style search.

Altogether HCBS consists of three levels of search in three different spaces: (i) space of conflicts, (ii) goal ordering space, and (iii) path space.

Combining soundness and optimality of CBS with properties of MST-based heuristic for goal vertex ordering we can state the following proposition.

**Proposition 2** The HCBS algorithm returns sum-of-costs optimal solution for given input MG-MAPF \( \Theta \).

Compilation-based Approach: SMT-HCBS

The second approach for solving MG-MAPF is based on the reduction of MG-MAPF to a series of Boolean formulae that are decided by an external SAT solver (Audemard and Simon 2018). Our new algorithm called SMT-Hamiltonian-CBS combines existing SMT-CBS (Surynek 2019b) with MG-MAPF specific generation of target Boolean formulae.

**Time Expansion and Decision Diagrams**

Construction of a Boolean formula corresponding to solvability of MAPF as used in SMT-CBS relies on the time expansion of \( G \). Having SoC, the basic variant of time expansion determines the maximum number of time steps (makespan) \( t_{M} \) such that every possible solution with the sum-of-costs less than or equal to SoC fits in \( t_{M} \) timesteps.

The time expansion makes copies of vertices \( V \) for each timestep \( t = 0, 1, 2, \ldots, t_{M} \). That is, we have vertices \( v^{t} \) for each \( v \in V \) and time step \( t \). Edges from \( G \) are converted to directed edges interconnected timesteps in the time expansion. Directed edges \( (v^{t}, v^{t+1}) \) are introduced for \( t = 0, 1, \ldots, t_{M} - 1 \) whenever there is \( \{u, v\} \in E \). Wait actions are modeled by introducing edges \( (v^{t}, v^{t+1}) \). A directed path in the time expansion corresponds to the trajectory of an agent in \( G \). Hence the modeling task now consists in construction of a formula in which satisfying truth-value assignments correspond to directed paths from \( s^{0}_{0}(a_{i}) \) to \( g^{t_{M}}(a_{i}) \) in the time expansion.

The time expansion is often further improved so that unreachable nodes are removed which reduces the subsequent search effort done on top of the time expansion (Bryant 1995; Sharon et al. 2013; Surynek et al. 2016a). A structure called multi-valued decision diagram (MDD) is a subset of the time expansion as described above, that is, it is a directed graph \( MDD_{t} = (V_{t}, E_{t}) \) for each agent \( a_{i} \). A vertex \( v^{t} \) is included in \( MDD_{t} \) if is reachable with respect to the given makespan bound \( t_{M} \). That is, \( v^{t} \) is included if and only if \( \text{Dist}(s_{0}(a_{i}), v) \leq t \) (agent \( a_{i} \) has enough time to reach \( v \) at time step \( t \)) and \( \text{Dist}(v, g(a_{i})) \leq t_{M} - t \) (agent \( a_{i} \) can
Algorithm 1: HCBS algorithm for MG-MAPF.

1. HCBS_conflicts(Θ = (G = (V, E), A, so, g))
2. N.constraints ← ∅
3. N.circuits ← \{circuit*(a_i) a shortest Hamiltonian path from s_0(a_i) covering g(a_i) \mid 1 = 1, 2, ..., k\}
4. N.SoC ← \sum_{k=1}^{\infty} \text{Cost}(N.circuits(a_i))
5. insert (SoC, N) into OPEN
6. while OPEN ≠ ∅ do
7. (key, N) ← min-Key(OPEN)
8. remove-Min-Key(OPEN)
9. collisions ← validate(N.circuits)
10. if collisions = ∅ then
11. return N.circuits
12. let (a_c, a_j, v, t) ∈ collisions
13. for each a ∈ {a_c, a_j} do
14. N’.constraints ← N.constraints \cup (a_v, t)
15. N’.circuits ← N.circuits
16. N’.circuits(a_i) ← HCBS_ordering(Θ, a, N’.constraints)
17. SoC’ ← \sum_{k=1}^{\infty} \text{Cost}(N’.circuits(a_i))
18. insert (SoC’, N’) into OPEN
19. HCBS_ordering(Θ, a, constraints)
20. let Θ = (G = (V, E), A, so, g)
21. N.u ← \text{so}(a_i); N.finished ← ∅
22. T_S(N.u, g(a_i)) ← construct-MST(\text{so}(a_i), g, G)
23. N.g ← 0; N.h ← \text{Cost}(T_S(N.u, g_R))
24. N.circuit ← []
25. insert (N.g + N.h, N) into OPEN
26. while OPEN ≠ ∅ do
27. (key, N) ← min-Key(OPEN)
28. remove-Min-Key(OPEN)
29. if N.finished = g(a) then
30. return N.circuit
31. else for each v ∈ g(a_i) \ N.finished do
32. path ← A*(N.u, v, constraints, N.g)
33. if path ≠ Fail then
34. N’.u ← v
35. N’.finished ← N.finished \cup \{v\}
36. g_R ← g(a_i) \ \text{N’.finished}
37. T_S(N’.u, g_R) ←
38. construct-MST(N’.u, g_R, G)
39. N’.g ← N.g + \text{Cost}(path)
40. N’.h ← \text{Cost}(T_S(N’.u, g_R))
41. insert (N’.g + N’.h, N) into OPEN
42. return Fail

reach its goal in the remaining time) where Dist(u, v) is the length of the shortest path between u and v in G.

Hamiltonian Multi-value Decision Diagram

The idea of MDD can be adapted for MG-MAPF. We introduce a Hamiltonian MDD for individual agents denoted MDD^H_i which is almost identical to MDD_i in the terms of expanding G. However, the reachability of vertices at given time steps in MDD^H_i is treated in a different way to reflect that each agent in MG-MAPF has multiple goal vertices.

A node v^t is included in MDD^H_i if the following conditions hold:
1. Dist(s_0(a_i), v) ≤ t and
2. a Hamiltonian path H_P = H_P(s_0(a_i), g(a_i) \cup \{v\}) exists in G such that Cost(H_P) ≤ t_M.

In other words, time t must be enough to reach v from the start and there must be a Hamiltonian path starting at s_0(a_i) visiting both v and all agent’s goals of the cost not exceeding the number of levels of expansion t_M. Without proof let us summarize important property of MDD^H_i.

Proposition 3 A sequence of vertices visited by an agent a_i within any solution to MG-MAPF Θ with the sum-of-costs SoC corresponds to a directed path in MDD^H_i.

Computing costs of Hamiltonian paths for every vertex in G requires substantial computational effort as finding Hamiltonian path is an NP-hard problem (Garey, Johnson, and Tarjan 1976) and also confirmed by our preliminary tests. Hence we introduce a relaxation of MDD^H_i denoted MDD^TS_i in which the requirement of the existence of a Hamiltonian path is replaced with the requirement of existence of a spanning tree T_S(s_0(a_i), g(a_i) \cup \{v\}) of cost at most t_M. Computing a spanning tree can be done in polynomial time. The direct corollary of Proposition 1 is that using MDD^TS_i is sound for construction of the formula.

Corollary 1 A sequence of vertices visited by an agent a_i within any solution to MG-MAPF Θ with the sum-of-costs SoC corresponds to a directed path in MDD^TS_i.

See figure 2 for illustration how MDD^TS_i can be used to reduce the number of nodes in the time expansion. A spanning tree of cost 3 including vertices v_1, v_3 and v_7 exists so these vertices reachable at time step 3 from v_1 are included but v_2 not because no spanning tree of cost less than 4 including v_1, v_2, v_3 and v_7 exists in G (similarly for v_6).

As solutions to MG-MAPF are typically longer than those in the standard MAPF such pruning of the time expansion is crucial to keep the resulting formula reasonably small.

Incomplete Boolean Model

We construct a formula H(SoC) representing an incomplete model for MG-MAPF. Assume we have MDD^TS_i = (V_i, E_i)
for $a_i$ from which we derive the formula as follows. A Boolean variable $X^t_v(a_i)$ is introduced for every vertex $v^t$ in $V_i$. The semantics of $X^t_v(a_i)$ is that it is TRUE if and only if agent $a_i$ resides in $v$ at time step $t$. Similarly we introduce $\mathcal{E}^t_{u,v}(a_i)$ for every directed edge $(u^t, v^{t+1})$ in $E_i$ with the meaning that it is TRUE if and only if agent $a_i$ traverses edge $(u, v)$ between time steps $t$ and $t + 1$.

Constraints are added on top of these variables to encode the movement rules of MG-MAPF that are exactly the standard MAPF rules such that agents move across edges (do not jump) and preserve basic physical properties (do not disappear and do not spawn). The detailed list of constraints is given in (Surynek et al. 2016a) and (Surynek 2019b), for the sake of brevity we show here only illustrative examples:

$$X^t_u(a_i) \Rightarrow \bigvee_{(u^t, v^{t+1}) \in E_i} \mathcal{E}^t_{u,v}(a_i), \quad (1)$$

$$\sum_{v^{t+1} \mid (u^t, v^{t+1}) \in E_i} \mathcal{E}^t_{u,v}(a_i) \leq 1 \quad (2)$$

These constraints state that if $a_i$ is in $u$ at time step $t$ then it has to leave via exactly one edge $(u^t, v^{t+1})$.

MG-MAPF differs from MAPF in the encoding of the goal condition. An agent $a_i$ in MG-MAPF must visit each of its goals at least once. Hence we add for each $v \in g(a_i)$ the following constraint, that is we take all copies of $v$ in $MDD^{T_{SoC}}$ and require that at least one of the corresponding Boolean variables must be TRUE:

$$\sum_{v^t \mid v^t \in V_i} X^t_v(a_i) \geq 1 \quad (3)$$

The incompleteness of the model as suggested in (Surynek 2019b) consists in omitting certain constraints. Specifically collision avoidance constraints are omitted. Hence instead of the equivalence between the satisfiability of $H(\text{SoC})$ and the solvability of MG-MAPF under given $\text{SoC}$ we only establish an implication as follows:

**Definition 5 (incomplete Boolean model).** A Boolean formula $H(\text{SoC})$ is an incomplete Boolean model of MG-MAPF $\Theta$ if the following condition holds:

$$H(\text{SoC}) \in \text{SAT} \iff \Theta \text{ has a solution of sum-of-costs } \text{SoC}.$$  

**The Algorithm and Integrated Goal Ordering**

After obtaining a satisfiable formula we cannot immediately declare it to represent a valid solution but first the extracted candidate for MG-MAPF solution must be verified for collisions. If there are no collisions then we are finished and the candidate is a valid MG-MAPF solution. If not then a collision avoidance constraint must be added to refine $H(\text{SoC})$ and solving process continues with the next iteration. Such a tight integration of lazy formula construction and its solving using the SAT solver is often used in the *satisfiability modulo theories* (SMT) paradigm (Katz et al. 2016).

Assume that a collision occurs at time step $t$ at vertex $v$ between agents $a_i$ and $a_j$ (denoted $(a_i, a_j, v, t)$). Eliminating this collision is to forbid that $a_i$ or $a_j$ appears at $v$ at time step $t$ which naturally corresponds to a binary clause:

$$\neg X^t_v(a_i) \lor \neg X^t_v(a_j).$$

Being aware of the construction of modern CDCL SAT solvers (Audemard and Simon 2018) we can see that this is quite fortunate as such short clauses promote *Boolean constraint propagation* (resolution, unit propagation) that significantly reduces the search effort.

The resulting algorithm called SMT-HCBS is described using pseudo-code as Algorithm 2. It is a modification of existing SMT-CBS, the major difference is the use of specific Boolean encoding for MG-MAPF (line 11) as described above. The algorithm tests the existence of a solution of the input instance $\Theta$ for increasing sum-of-costs until a positive answer is obtained (lines 5-9). The lower bound for sum-of-costs is obtained as the sum-of-costs of individual Hamiltonian paths for agents (lines 3-4).

The advantage against complete models (Surynek et al. 2016a) is that a valid solution can be obtained before the problem is fully specified in terms of constraints which often leads to faster solving process. Observe that an unsatisfiable formula in the case of incomplete model according to Definition 5 means that the input instance $\Theta$ is not solvable under the given cost $\text{SoC}$, leading to its incrementing of $\text{SoC}$ and the next iteration of the algorithm.

Let us note that both goal vertex ordering and conflict resolution are integrated at the same conceptual level - both problems are encoded in the target Boolean formula are decided by the SAT solver.
Experimental Evaluation

We evaluated HCBS and SMT-HCBS on standard benchmarks from movingai.com (Sturtevant 2012). Representative part of the results is presented in this section.

Benchmarks and Setup

HCBS and SMT-HCBS were implemented in C++. The SMT-HCBS solver is built on top of the Glucose 3.0 SAT solver (Audemard and Simon 2018) that ranks among the high performing SAT solvers according to recent SAT solver competitions (Balio, Heule, and Järvsalo 2017).

The experimental evaluation has been done on diverse instances consisting of 4-connected grid maps ranging in sizes from small to large. Random MAPF scenarios from movingai.com are used to generate MG-MAPF instances. To obtain instances of various difficulties we varied the number of agents while the number of goal vertices per agent was set as constant. As defined in the benchmark set, 25 different instances are generated per number of agents.

Starting positions of agents are taken directly from the scenario. Since only one goal is defined per agent in each MAPF scenario the set of goals for an agent is generated by making a specified number of random picks among goal positions of all agents in the scenario. This results in instances whose goal vertices are equally distributed across the map.

The tests we focused on comparing HCBS and SMT-HCBS in the terms of success rate and runtime. All experiments were run on system consisting of 200 Xeon 2.8 GHz cores, 1TB RAM, running Ubuntu Linux 18. 2 The success rate is the ratio of the number of instances out of 25 per number of agents that the solver managed to solve under the time limit of 5 minutes.

Runtime Results

We divided the tests into three categories with respect to the size of maps. Results for small instances derived from the empty-16-16 map are shown in Figure 3. Three different cases with the number of goal vertices per agent: 1 (corresponds to MAPF), 2, 4, and 8 are tested.

We can observe that for 1, 2 and 4 goals per agent SMT-HCBS dominates. This is an expectable result since similar behavior can be observed for SMT-CBS vs. CBS for the standard MAPF (Surynek 2019b). However the clearly visible trend it that HCBS scales better for increasing number of agents while the number of goal vertices per agent was set as constant. As defined in the benchmark set, 25 different instances are generated per number of agents.

Starting positions of agents are taken directly from the scenario. Since only one goal is defined per agent in each MAPF scenario the set of goals for an agent is generated by making a specified number of random picks among goal positions of all agents in the scenario. This results in instances whose goal vertices are equally distributed across the map.

The tests we focused on comparing HCBS and SMT-HCBS in the terms of success rate and runtime. All experiments were run on system consisting of 200 Xeon 2.8 GHz cores, 1TB RAM, running Ubuntu Linux 18. 2 The success rate is the ratio of the number of instances out of 25 per number of agents that the solver managed to solve under the time limit of 5 minutes.

Explanation of Results

There are multiple factors explaining the observed results. SMT-HCBS often needs to iterate through many unsatisfiable sum-of-costs before the optimal sum-of-cost is met on instances with greater number of goal vertices. This is due to the fact that agents interact more in such cases which results in a greater difference between the true optimal sum-of-costs and its lower bound estimation. Moreover node pruning in the spanning tree MDDs is less efficient for greater number of goals since the cost of spanning tree deviates more from the true Hamiltonian path cost.

In contrast to this, the effect of domain specific heuristics is more direct in HCBS which can more effectively determine the optimal ordering of goal vertices for individual agents - the choice of the next vertex can be immediately assessed by the heuristic. Similar effect cannot be easily achieved in SMT-HCBS since during MDD generation phase we need to represent all possible goal vertex orderings and the SAT solver itself has no domain specific information.

Figure 3: Success rate and runtime comparison on small-sized maps.

For the full reproducibility of the presented results we provide a complete source code of our solvers and detailed experimental data on author’s web: http://users.fit.cvut.cz/~surynpav/research and in git repository: https://github.com/surynek.
Related Work

The most closely related problem to MG-MAPF is multi-agent pickup and delivery (MAPD) (Ma et al. 2017b; Liu et al. 2019), defining a set of tasks \( T = \{ t_1, t_2, \ldots, t_m \} \) where each task \( t_j = (p_j, d_j) \) is characterized by a pickup location \( p_j \in V \) and a delivery location \( d_j \in V \). Agents can freely select tasks to fulfill. The freedom in choice of tasks and the order of their fulfilling makes the problem very hard. The contemporary solving approaches for MAPD first assign tasks to agents and followed by determining the ordering of tasks per agent ignoring collisions. Then collision free paths are planned according to the task ordering. As there is no feedback between the phases the resulting plan is sub-optimal.

Another related work is represented by order picking problem (OPP) (Pansart, Catusse, and Cambazard 2018) which is a variant of traveling salesman problem in a rectangular warehouse. Various integer programming and flow-based formulations of OPP have been studied. The important difference from MG-MAPF is that it is typically regarded as a single agent problem, hence collisions between agents are not considered in OPP.

A generalization of MAPF where an agent is assigned multiple goal vertices and is successful if it reaches any of its goals is suggested in (Surynek 2013). Similar setup has been studied for target assignment in MAPF (Hönig et al. 2018) where each agent selects one goal from multiple choices.

A compilation-based approach to MAPF that uses a similar clause generation scheme as SMT-HCBS but derived from CSP has been proposed in (Gange, Harabor, and Stuckey 2019).

Conclusion

We introduced a novel multi-goal multi agent path finding (MG-MAPF) problem. MG-MAPF generalizes MAPF by assigning each agent a set of goal vertices instead of one goal. The task is to visit each of agent’s goal vertices at least once while we aim on finding sum-of-costs optimal solutions. Two algorithms generating sum-of-costs optimal solutions are suggested. A search-based algorithm called Hamiltonian CBS (HCBS) derived from CBS and a compilation-based approach SMT-HCBS that reduces MG-MAPF to Boolean logic and solves the problem in the target formalism by the SAT solver.

HCBS introduces three level search in which conflict resolution is done at the high level and goal vertex ordering and path planning are done at the low level. The key technique is decoupling the vertex ordering from collision-free path planning. CBS provides greater room for integrating heuristics that we made use of when adapting it for MG-MAPF.

In contrast to this, SMT paradigm provides more powerful search but integration of domain specific heuristics is more complicated. We increased the informedness of the SMT-based algorithm SMT-HCBS through the concept of Hamiltonian and spanning tree MDD which eventually led to significant improvements so that SMT-HCBS is able to solve some of the standard benchmarks. However decoupling of the goal vertex ordering from conflict resolution turned out to be the crucial factor behind the better performance of HCBS. The worse support of domain specific heuristics shows limitations of the SMT approach generally.

For future work we plan investigate possibility of generating the formulae lazily not only in the phase of conflict resolution but also in the phase of MDD generation. A possible future direction is also to address MG-MAPF via the DPLL(MG-MAPF) framework (Surynek 2019a).


