IEEE Robotics and Automation Letters (RAL) paper presented at the
2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Mixed-Integer Linear Programming Models for Multi-Robot Non-Adversarial Search

Beatriz A. Asfora, Jacopo Banfi and Mark Campbell

Abstract— **In this letter, we consider the Multi-Robot Efficient Search Path Planning (MESPP) problem, where a team of robots is deployed in a graph-represented environment to capture a moving target within a given deadline. We prove this problem to be NP-hard, and present the first set of Mixed-Integer Linear Programming (MILP) models to tackle the MESPP problem. Our models are the first to encompass multiple searchers, arbitrary capture ranges, and false negatives simultaneously. While state-of-the-art algorithms for MESPP are based on simple path enumeration, the adoption of MILP as a planning paradigm allows to leverage the powerful techniques of modern solvers, yielding better computational performance and, as a consequence, longer planning horizons. The models are designed for computing optimal solutions offline, but can be easily adapted for a distributed online approach. Our simulations show that it is possible to achieve 98% decrease in computational time relative to the previous state-of-the-art. We also show that the distributed approach performs nearly as well as the centralized, within 6% in the settings studied in this letter, with the advantage of requiring significant less time – an important consideration in practical search missions.**

I. INTRODUCTION

In this letter, we consider the Multi-Robot Efficient Search Path Planning (MESPP) problem introduced by Hollinger et al. in [1]. In this problem, a team of robots is deployed in an environment represented as an undirected graph with the aim of capturing a moving non-adversarial target within a given deadline. In [1], the authors propose to tackle the MESPP problem with a receding horizon approach that can be implemented either in a centralized or in a distributed fashion.

In the centralized case, all the possible joint paths are enumerated over a given planning horizon h, and the best set of paths is executed until the next planning step. Considering a graph of n vertices and a team of m searchers, this approach has a worst-case complexity of $O(n^{mh})$, i.e. exponential in both the team size and the planning horizon. This approach is dubbed "explicit coordination". In the distributed case, instead, at each planning step and for $i = 1, \ldots, m$ (according to a lexicographic order), the path of the i-th robot is computed by leaving the paths of the remaining robots $j \neq i$ fixed (robots with $j > i$ are initially assumed to remain at their starting position). The optimization of the i-th path is again performed by enumerating all feasible paths over a given horizon h. This approach is dubbed "implicit coordination".

All the authors are with the Sibley School of Mechanical and Aerospace Engineering, Cornell University, Ithaca NY 14850, USA. Email: {ba386, jb2639, mc288}@cornell.edu

Compared to explicit coordination, implicit coordination scales better w.r.t. the number of robots, with a worst-case complexity of $O(mn^h)$. Implicit coordination also provides an approximation factor $(1 + \kappa)$ along a single planning horizon, where κ is the approximation achieved by the solver for the single searcher problem [1], whenever the search objective function can be formulated as a nondecreasing submodular set function [2]. However, the optimization of the single paths still requires the expansion of a search tree with depth h. For this reason, practical real-time implementations limit the planning horizon to a few steps ahead (5-6 for a typical indoor environment [1], [3]).

Main contributions. First, we prove that the MESPP problem is NP-hard even for two-dimensional grid environments with a static target and a single searcher. Second, we present the first set of Mixed-Integer Linear Programming (MILP) models for tackling the MESPP problem, the most general of which is able to encompass multiple searchers, arbitrary capture ranges, and false negatives simultaneously. Our proposed MILP models for the MESPP problem allow path enumeration to be performed in a much more efficient way by leveraging the sophisticated branching and pruning techniques of modern MILP solvers [4]. The models are primarily designed to compute optimal solutions offline, for relatively short missions ($h \leq 10$), but they can be used to plan single-robot paths in the same receding-horizon planning scheme introduced in [1], for longer missions. Our simulation results show that a receding-horizon distributed approach can yield results within 6% of an optimal offline solution in a matter of seconds. Moreover, the adoption of MILP as a planning paradigm in an online setting can provide 98% decrease in computational time relative to the state-of-the-art.

This letter is structured as follows. Section II frames the MESPP problem within the multi-robot search literature. Section III introduces the MESPP problem and Section IV proves its NP-hardness. Section V presents the MILP models to compute optimal solutions offline, and Section VI shows how the models are adapted for an online distributed implementation. Simulation results are presented in Section VII, and Section VIII concludes the paper.

II. RELATED WORK

Target search problems have traditionally been a subject of study in operations research [5] and game theory [6] communities. The past two decades have witnessed an ever-increasing interest in these problems by researchers in mobile robotics, under flavors that make them more suitable to cope with the inherent constraints – computation, sensing, mobility, communication – of mobile robots. Chung et al [7] provide an overview of how target search problems can be tackled from a robotic perspective, introducing a rigorous taxonomy with several classification dimensions. For example, the target can be static [8] or dynamic [9]; adversarial [10], non-adversarial or cooperative [11]; in case of known target's motion model, this can be a random walk [12] or Markovian [1]; the environment can be continuous, unbounded [13] or bounded (typically represented as a polygon [14]), or discrete and represented by a finite graph [15]. For what concerns the sensing model, this can be assumed to be perfect, with detection events happening within a given range [16] or when in line-of-sight with the target [17], or affected by false negatives [1] and/or false positives [18].

In this paper, we consider the target search problem introduced as MESPP in [1]. This problem version deals with a dynamic, non-adversarial target which moves in a graph-represented environment according to a known Markovian motion model. After its introduction, the MESPP model was used in further studies in data fusion [19] and connectivity problems [20], most recently in [21].

The approach presented in [22] for searching a target on graph-represented environments is also based on a MILP formulation. However, it is restricted to a single searcher and capture events in a single graph vertex. In this letter, we provide a more general formulation able to encompass multiple searchers, arbitrary capture ranges, and false negatives simultaneously.

III. Multi-Robot Search of a Non-Adversarial Object

This section formalizes the Multi-Robot Efficient Search Path Planning (MESPP) problem. A team of cooperative robots efficiently searches for a non-adversarial target in a known graph-represented environment within a specified deadline, and the problem goal is to place the searchers where they, as a team, are most likely to intercept the target. The deadline is defined due to a practical reason, i.e., the target must always be located within a certain time in practical situations. This formulation assumes that the robots have path planning and obstacle avoidance capabilities, allowing them to follow a sequence of waypoints (the graph vertices). Time evolves in discrete steps: each step encompasses the robots' transition between graph vertices, followed by a sensing action at the new vertex. In this letter, the terms capture, interception, and detection are used interchangeably, as well as the terms object and target.

A. Environment and Searchers' Paths

Let $G = (V, E)$ be an undirected, connected, and simple graph representing a known environment, with $V = \{1, 2, ..., n\}$. The graph can be obtained by discretizing the environment (for example, a floorplan) by hand, or by means of automated discretization techniques, such as grids [21] or constrained Delaunay triangulation [23]. We use $\delta(v)$ to denote the neighbors of $v \in V$, while $\delta'(v)$ represents $\delta(v) \cup \{v\}$. Let $d(u, v)$ be the length of the shortest path between any two vertices $u, v \in V$.

Each searcher is represented by $s \in S$, where $S = \{1, 2, ..., m\}$. Time $t \in T$ evolves in discrete steps until the deadline τ, $T = \{1, 2, ..., \tau\}$. Note that in the offline centralized approach, planning horizon $h = \tau$. A searcher's path π^s is defined as an ordered sequence of $\tau + 1$ vertices $\pi^s = [v_o^s, v^{s,1}, \ldots, v^{s,\tau}]$, where v_o^s denotes the starting vertex of s. We use $\mathcal{P}^s$ to denote the set of all the possible paths for searcher s, and $\mathcal{P} = \prod_{s=1}^{m} \mathcal{P}^s$ to denote the set of all the possible joint paths. Each path must respect the following: at each step, the searcher can either stay still at the current vertex, or move to a neighboring vertex. Formally, $\forall \{v^{s,t}, v^{s,t+1}\} \in \pi^s$, $\{v^{s,t}, v^{s,t+1}\} \in \delta'(v)$.

B. Object's Motion and Capture

The object moves probabilistically in the graph, with motion encoded by a Markov chain specified by the stochastic matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$. Specifically, the entry $\mathbf{M}_{uv}$ represents the probability that the object will move from u to v between time-steps t and $t + 1$.

At each step t, the objects state, resulting from its interactions with searchers executing a set of joint paths $\boldsymbol{\pi} \in \mathcal{P}$, is represented by the belief vector,

$$\mathbf{b}^{\boldsymbol{\pi}}(t) = [b_{\mathrm{c}}(t),\ b_1(t), ..., b_n(t)]. \tag{1}$$

The first element, $b_{\mathrm{c}}(t)$, represents the probability that the searchers have located the object by time t. The remaining elements $b_1(t), ..., b_n(t)$ represent the probability that the object is in the corresponding vertices at time t, such that $b_{\mathrm{c}}(t) + \sum_{i=1}^{n} b_v(t) = 1$. Note that such probabilities can describe the state of the object in all the possible realizations of the world. In the remaining of this letter, we will simply denote $\mathbf{b}^{\boldsymbol{\pi}}(t)$ by $\mathbf{b}(t)$, to reduce the notation burden as the particular set of joint paths will always be clear from the context.

Capture events are described by matrices $\mathbf{C}^{s,u} \in [0, 1]^{(n+1)\times(n+1)}$, $\forall s \in S$, $u \in V$. Their effect is to connect the probability of the object being at a particular location with its capture state. In other words, the capture matrix $\mathbf{C}^{s,u}$ encodes which vertices of the graph fall within the sensing range of searcher s, when such is located in vertex u.

For the moment, assume the searcher has perfect sensing capabilities. Its capture matrix is constructed as follows. Initialize the capture matrix as an identity matrix $\mathbf{C}^{s,u} = \mathbf{I}_{n+1}$. Then, for each possible object location $v \in V$ that allows a detection when s is placed in u, null the v-th column of the capture matrix by switching the 1 at $\mathbf{C}^{s,u}_{vv}$ with the 0 at $\mathbf{C}^{s,u}_{v0}$. Note that the first column of $\mathbf{C}^{s,u}$ is denoted by index 0 to avoid confusion with vertex 1.

Now consider the presence of false negatives in the searcher's sensing actions. False negative rates can vary across the different team members, but we assume they remain constant for each searcher regardless of its position. Let $\zeta^s \in [0, 1)$ be the false negative probability of searcher s. When accounting for false negatives, capture matrices are essentially constructed as above: initialize it as an identity matrix $\mathbf{I}_{n+1}$, but now replace the 1 at $\mathbf{C}^{s,u}_{vv}$ for ζ^s, and the 0 at $\mathbf{C}^{s,u}_{v0}$ for $1-\zeta^s$.

The belief update equation links the current belief, the probabilistic object motion, and the searchers' paths $\boldsymbol{\pi} = (\pi^1, \dots, \pi^m)$ with the associated capture events, as follows:

$$\mathbf{b}(t+1) = \mathbf{b}(t) \begin{bmatrix} 1 & \mathbf{0}_{1\times n} \\ \mathbf{0}_{n\times 1} & \mathbf{M} \end{bmatrix} \prod_{s=1}^{m} \mathbf{C}^{s,\pi^{s,t+1}}. \quad (2)$$

C. Optimization Problem

The MESPP problem seeks to optimize the following objective, subject to Eq. (2):

$$\boldsymbol{\pi}^* = \arg\max_{\boldsymbol{\pi}\in\mathcal{P}} \sum_{t=0}^{\tau} \gamma^t b_{\mathrm{c}}(t), \quad (3)$$

where $\gamma \in (0,1]$ is a discount factor.

IV. NP-Hardness

In this section we show the hardness of the MESPP problem by proving that its decision version, which we dub MESPP-D, is NP-hard even when the graph is a two-dimensional grid, the target is stationary, and there is a single searcher. We reduce from the Hamiltonian-Path Between Two Points (2HP), proven in [24] to be NP-complete on grid graphs[1].

MESPP-D
INSTANCE: MESPP instance with reward function $F_\tau = \sum_{t=0}^{\tau} \gamma^t b_{\mathrm{c}}(t)$, and bound value $B \geq 0$.[2]
QUESTION: Is there a search plan such that $F_\tau \geq B$?

HAMILTONIAN PATH BETWEEN 2 POINTS (2HP)
INSTANCE: Graph $G = (V,E)$, vertices v_A and v_B.
QUESTION: Does G contain a Hamiltonian path beginning with v_A and ending with v_B?

Theorem 1. *MESPP-D is NP-hard even when the following conditions hold simultaneously:*
1. G is a grid graph,
2. the target is stationary, and
3. there is only one searcher.

Proof. We reduce 2HP to MESPP-D in polynomial time as follows. The MESPP-D *grid graph* is the original 2HP grid graph $G = (V,E)$ with vertices labeled in a lexicographic order, restricting only $v_1 = v_A$, $v_n = v_B$. We place at v_1 a *single searcher* with perfect sensing capabilities (no false negatives) and able to capture the target only from its current vertex. The *target is stationary*, $\mathbf{M} = \mathbf{I}_n$. The initial belief is set to $b_{\mathrm{c}} = b_1 = 0$, $b_n = \dfrac{1}{(n-1)^2}$ and $b_v = \dfrac{1}{n-1} + \dfrac{1}{(n-1)^2}$, $2 \leq v \leq n-1$. We define the deadline $\tau = n-1$ and bound value $B = \sum_{v=1}^{n} \gamma^v \sum_{u=1}^{v} b_u$. The discount factor value does not influence the proof, and is arbitrarily chosen as $\gamma = 0.99$.

Stating G has a Hamiltonian path between vertices v_A and v_B means that it is possible to start at v_A, pass through all vertices exactly once and reach v_B within $n-1$ steps. Note that in the transformation to MESPP-D, v_B maps to the lowest non-zero probability vertex v_n, as $b_n < b_v$, for $v = 2, ..., n-1$. Thus the search plan with the maximum capture probability by deadline (Eq. 3) requires the searcher to visit each vertex exactly once and lastly the vertex with lowest probability of reward (v_n). Following this path yields a reward of exactly $F_\tau = B$, and MESPP-D is therefore a yes-instance.

Conversely, if MESPP-D is a yes-instance, i.e. $F_\tau \geq B$, the searcher must have started at a particular vertex v_1 and reached the lowest probability vertex v_n at the last time step. Otherwise the searcher would have collected a lower reward by the deadline, $F_\tau < B$, due to visiting a vertex more than once (zero reward) or visiting the vertex with the lowest probability early on (cumulative effect of collecting a smaller reward at $t < \tau$). This implies that if $F_\tau \geq B$, 2HP must be a yes-instance. □

Corollary 1.1. *For a static target, MESPP-D is NP-complete.*

Proof. For a graph with n vertices, a searcher needs n^2 steps to visit all vertices in an arbitrary order. In the case of a static target, the searcher will have collected all the possible reward by $t = n^2$ even if $\tau > n^2$. Thus the solution depends only on n and not on τ. The solution is of polynomial size and verifiable in polynomial time, placing the problem in NP. □

V. Mixed-Integer Linear Programming Models

This section presents three MILP models for solving the MESPP problem defined in Section III. Legal searchers' paths and object's motion are modeled in Sections V-A and V-B, respectively. These first sets of variables and constraints are common across all models. We then introduce the constraints for different types of capture events. In particular, capture events limited to the same graph vertex without false negatives are presented in Section V-C; capture events with arbitrary capture range are shown in Section V-D; finally, capture events with arbitrary range and false negatives are introduced in Section V-E.

A. Legal Paths for Searchers

We use $V^{s,t} = \{(v,t) \in V \times T \mid d(v_o^s, v) \leqslant t,\ s \in S\}$ to denote, for each searcher s, the set of all the possible (v,t) states that are compatible with its starting position v_o^s (i.e. the searcher can actually be in v at time t). With a slight abuse of notation, we also define $V^{s,t}(t) = \{v \in V \mid (v,t) \in V^{s,t}\}$ and $V^{s,t}(v) = \{t \in T \mid (v,t) \in V^{s,t}\}$. Let us also introduce a dummy goal vertex v_g, which can be thought of as connecting to all vertices of G in a fictitious manner.

[1]Itai et al. [24] define grid graphs as finite, vertex induced subgraphs of an infinite graph where (a) the vertex set consists of all points of the plane with integer coordinates and (b) two vertices are connected by an edge if and only if the Euclidean distance between them is equal to 1.

[2]As customarily done, we assume that all the numbers used in the instance are rational [25].

Consider two sets of binary variables: $x_v^{s,t}$ denotes the presence of searcher s in vertex v at time t, and $y_{uv}^{s,t}$ conveys the fact that searcher s will move from u to v between steps t and $t+1$. The following constraints enforce the legality of the paths for the searchers:

$$x_{v_o^s}^{s,0} = \sum_{j \in \delta'(v_o^s)} y_{v_o^s j}^{s,0} = \sum_{j \in V^{s,t}(\tau)} y_{jv_g}^{s,\tau} = 1, \ \forall s \in S, \tag{4}$$

$$x_v^{s,t} = \sum_{\substack{j \in \delta'(v) \\ \cap V^{s,t}(t-1)}} y_{jv}^{s,t-1} = \sum_{i \in \delta'(v)} y_{vi}^{s,t}, \ \forall (v, t < \tau) \in V^{s,t}, \tag{5}$$

$$x_v^{s,\tau} = \sum_{\substack{j \in \delta'(v) \\ \cap V^{s,t}(\tau-1)}} y_{jv}^{s,\tau-1} = y_{vv_g}^{s,\tau}, \ \forall v \in V^{s,t}(\tau) \ . \tag{6}$$

Equations (4) set the searchers' starts and goal vertices, while Eqs. (5)-(6) ensure path consistency. The variables are formally defined as

$$x_v^{s,t} \in \{0,1\}, \ \forall s \in S, (v,t) \in V^{s,t}, \tag{7}$$

$$y_{uv}^{s,t} \in \{0,1\}, \forall s \in S, (u, t < \tau) \in V^{s,t}(t), v \in \delta'(u), \tag{8}$$

$$y_{uv_g}^{s,\tau} \in \{0,1\}, \ \forall s \in S, u \in V^{s,t}(\tau). \tag{9}$$

B. Object's Motion

We introduce two sets of continuous variables: β_i^t, representing the entries of the belief vector at time t, and α_v^t, representing the result of the application of the object's motion model. The constraints below respectively set the initial belief and evolve the object's location based on the previous belief:

$$\beta_i^0 = b_i(0), \ \forall i \in V \cup \{\text{c}\}, \tag{10}$$

$$\alpha_v^t = \sum_{u \in V} \mathbf{M}_{uv} \beta_u^{t-1}, \ \forall v \in V, t \in T. \tag{11}$$

The variables are formally defined as

$$\beta_i^t \in [0,1], \ \forall i \in V \cup \{c\}, \ t \in \{0\} \cup T, \tag{12}$$

$$\alpha_v^t \in [0,1], \ \forall v \in V, \ t \in T. \tag{13}$$

C. Capture Events in Same Vertex, Binary Detection

Define same-vertex capture with binary detection (0 or 1) as the searcher being in vertex v at time t, and able to determine with certainty if the object is also in v. This entails the following property: if no searcher is at vertex v, no new information is available about that vertex, and the belief in v is simply the probability that the object might have moved there between $t-1$ and t, denoted by Eq. (11). On the other hand, if there is at least one searcher at v and no object was detected, one can infer the object is not in v.

Define then, for each time-step, a binary variable ψ_v^t that equals one if and only if there is at least one searcher located in v at time t. The belief vector entries can be expressed as

$$\beta_v^t = \alpha_v^t \left(1 - \psi_v^t\right), \ \forall t \in T, v \in V, \tag{14}$$

which translates to $\beta_v^t = \alpha_v^t$ if $\psi_v^t = 0$, or $\beta_v^t = 0$ if $\psi_v^t = 1$.

The above constraint is nonlinear and can not be applied directly in a MILP model, but it can be formulated in a linearized manner [26]. The following constraints substitute Eq. (14) for the belief,

$$\beta_v^t \leqslant 1 - \psi_v^t, \ \forall v \in V, \ t \in T, \tag{15}$$

$$\beta_v^t \leqslant \alpha_v^t, \ \forall v \in V, \ t \in T, \tag{16}$$

$$\beta_v^t \geqslant \alpha_v^t - \psi_v^t, \ \forall v \in V, \ t \in T, \tag{17}$$

$$\psi_v^t \in \{0,1\}, \ \forall v \in V, \ t \in T. \tag{18}$$

The relationship between the capture variable ψ_v^t and the searchers' positions variables $x_v^{s,t}$ is expressed as

$$\sum_{\substack{s \in S \ s.t. \\ v \in V^{s,t}(t)}} x_v^{s,t} \leqslant m\psi_v^t, \ \forall v \in V, \ t \in T, \tag{19}$$

$$\psi_v^t \leqslant \sum_{\substack{s \in S \ s.t. \\ v \in V^{s,t}(t)}} x_v^{s,t}, \ \forall v \in V, \ t \in T, \tag{20}$$

which means that, if all searchers are in vertex v at time t, $\psi_v^t = 1$, and the sum of the searchers positions $x_v^{s,t} \ \forall s \in S$ equals the number of searchers in the team, $m\psi_v^t$. If however no searcher is in v, $\psi_v^t = 0$ and so is the sum of $x_v^{s,t} \ \forall s \in S$.

Finally, the probability of the object being intercepted within step t is the remaining probability after the belief update on all vertices,

$$\beta_c^t = 1 - \sum_{v \in V} \beta_v^t, \ \forall t \in T. \tag{21}$$

D. Capture Events Within Given Range, Binary Detection

Generalizing the capture event, let us say that a searcher positioned in vertex u is able to detect, with certainty, the presence of the object in vertex v located within some arbitrary capture range. As before, this assumption entails that the team does not gain additional knowledge about the object's true position unless the latter can be intercepted. This rationale is again expressed by Eq. (14) and linearized in Eqs. (15)-(18). Equations (19)-(20) must, however, be replaced by the following:

$$\sum_{s \in S} \sum_{\substack{u \in V^{s,t}(t) \\ s.t. \ \mathbf{C}_{v0}^{s,u}=1}} x_u^{s,t} \leqslant m\psi_v^t \quad \forall v \in V, \ t \in T, \tag{22}$$

$$\psi_v^t \leqslant \sum_{s \in S} \sum_{\substack{u \in V^{s,t}(t) \\ s.t. \ \mathbf{C}_{v0}^{s,u}=1}} x_u^{s,t} \quad \forall v \in V, \ t \in T. \tag{23}$$

Now, when v is within range, $\psi_v^t = 1$ and the sum of $x_u^{s,t}$ such that $\mathbf{C}_{v0}^{s,u} = 1 \ \forall s \in S$ is at most the number of searchers, or $m\psi_v^t$. The opposite logic also applies. The probability of the object being intercepted within step t is enforced by Eq. (21).

E. Capture Events with False Negatives

In order to account for the false negatives in detection (see Sec. III), one must modify Eq. (14). First, consider a team of only one searcher. If the capture range can reach a particular vertex, the probability the object might be there is no longer zero, but rather $\beta_v^t = \zeta\alpha_v^t$, to account for the chance that the object is actually in that vertex, but has not been detected. If the searcher can not reach vertex v, no new information is available, and the belief is the probability the target has moved there, as before.

Considering the capture variable ψ_v^t as previously presented, the belief update equation for one searcher becomes

$$\beta_v^t = (1-\zeta)\,\alpha_v^t\left(1-\psi_v^t\right) + \zeta\alpha_v^t. \tag{24}$$

For multiple searchers, however, the detection uncertainty must decrease as more robots are in locations where they can potentially detect the object. This is expressed by updating the belief in an iterative manner, one searcher at a time. To this aim, we define capture and belief variables $\psi_v^{s,t}$ and $\beta_v^{s,t}$ for each searcher, and impose the following constraints:

$$\begin{aligned}\beta_v^{s,t} = (1-\zeta^s)\,\beta_v^{s-1,t}\left(1-\psi_v^{s,t}\right) + \zeta^s\beta_v^{s-1,t},\\ \forall s\in S, t\in T, v\in V,\end{aligned} \tag{25}$$

where

$$\beta_v^{0,t} = \alpha_v^t,\ \forall t\in T, v\in V. \tag{26}$$

The variables are formally defined as

$$\beta_v^{s,t} \in [0,1],\ \forall t\in T, v\in V, s\in S, \tag{27}$$

$$\psi_v^{s,t} \in \{0,1\},\ \forall t\in T, v\in V, s\in S. \tag{28}$$

To linearize Eq. (25), we define the auxiliary variable,

$$\delta_v^{s,t} = \beta_v^{s-1,t}\left(1-\psi_v^{s,t}\right), \tag{29}$$

$$\delta_v^{s,t} \in [0,1] \quad \forall t\in T, v\in V, s\in S, \tag{30}$$

and use the same technique as before to yield linear constraints:

$$\delta_v^{s,t} \leqslant 1-\psi_v^{s,t},\ \forall s\in S, t\in T, v\in V, \tag{31}$$

$$\delta_v^{s,t} \leqslant \beta_v^{s-1,t},\ \forall s\in S, t\in T, v\in V, \tag{32}$$

$$\delta_v^{s,t} \geqslant \beta_v^{s-1,t} - \psi_v^{s,t},\ \forall s\in S, t\in T, v\in V. \tag{33}$$

Equation (25) can therefore be rewritten as

$$\beta_v^{s,t} = (1-\zeta^s)\,\delta_v^{s,t} + \zeta^s\beta_v^{s-1,t},\ \forall t\in T, v\in V, s\in S. \tag{34}$$

The capture events must now be expressed separately for each searcher:

$$\sum_{\substack{u\in V^{s,t}(t)\\ s.t.\ \mathbf{C}_{v0}^{s,u}>0}} x_u^{s,t} \leqslant \psi_v^{s,t} \quad \forall t\in T, v\in V, s\in S, \tag{35}$$

$$\psi_v^{s,t} \leqslant \sum_{\substack{u\in V^{s,t}(t)\\ s.t.\ \mathbf{C}_{v0}^{s,u}>0}} x_u^{s,t} \quad \forall t\in T, v\in V, s\in S. \tag{36}$$

Equation (21) is again used to express the probability that the object has been captured by time t, noting that

$$\beta_v^t = \beta_v^{m,t},\ \forall t\in T, v\in V. \tag{37}$$

F. Complete MILP Models

For same-vertex capture, no false negatives:

$$\text{(SV-MILP)} \quad \max \quad \sum_{t\in T}\gamma^t b_{\mathrm{c}}(t) \quad \text{s.t.}$$
$$\text{Eqs. (4)-(13), (15)-(21).}$$

For arbitrary capture range, no false negatives:

$$\text{(MV-MILP)} \quad \max \quad \sum_{t\in T}\gamma^t b_{\mathrm{c}}(t) \quad \text{s.t.}$$
$$\text{Eqs. (4)-(13), (15)-(18), (21)-(23).}$$

Finally, the most general model, encompassing arbitrary capture ranges and false negatives:

$$\text{(FN-MV-MILP)} \quad \max \quad \sum_{t\in T}\gamma^t b_{\mathrm{c}}(t) \quad \text{s.t.}$$
$$\text{Eqs. (4)-(13), (26)-(28), (30)-(37).}$$

VI. Distributed Online Implementation

The MILP models presented on Section V are primarily designed to compute optimal or near-optimal solutions offline. However, for large values of mission deadline τ and team size m, a centralized approach does not scale well computationally.

An implicit coordination approach is inherently more scalable than explicit coordination. Recall the algorithm proposed in [1]: at each planning step and for $i = 1,\ldots,m$ (according to a lexicographic order), the path of the i-th robot is computed by leaving the paths of its teammates $j \neq i$ fixed; robots with $j > i$ are initially assumed to remain at their starting position, i.e., $\pi^{j,t} = v_o^s,\ \forall t = 1,\ldots,h$. In [1], the paths of the single robots are optimized by enumeration. Exploring the same search space by leveraging modern solver techniques ensures better scalability in general. To this aim, we can easily adapt our a models for this task. We adopt the implicit coordination algorithm of [1], using our MILP-based approach to perform a more efficient iterative optimization of single paths.

We can solve a sequence of m models, one for each searcher, while assigning a deterministic value to the variables associated with the paths of the teammates. When planning the path for searcher i, a deterministic value is assigned for $\forall j \neq i \in S$:

$$x_v^{j,t} = \begin{cases} 1, & \text{iff } v = \pi^{j,t} \\ 0, & \text{otherwise.}\end{cases} \tag{38}$$

A distributed approach decreases the number of variables to be optimized on the MILP model and thus the complexity of the problem, which generally yields a smaller solution time than a centralized planning scheme for $m > 1$. Although communication constraints are not addressed here, these could also be incorporated into the proposed model by leveraging recent work which proposes MILP-based approaches to connected multi-robot path planning [21], [27].

VII. Simulations

A. General Setup

We use GUROBI [28] to solve the MILP models[3] on a machine equipped with Intel-Core i9-9900K and 32 GB RAM. The maximum number of used threads is set to eight and the presolve level is kept as default (automatic). The solver timeout is set to 30 min for the offline (centralized) approach, and 10 sec for the distributed. Except when stated otherwise, default values are used for the remaining parameters.

We consider three graph environments: OFFICE and MUSEUM, both used in [1] and shown in Fig. 1; and a 10x10 4-connected GRID graph. For OFFICE and MUSEUM, we assume perfect sensing capability and same-vertex capture. For the GRID environment, we assume that the robots' sensing range spans the current vertex plus its 1-hop neighbors, and consider two settings: without (GRID-NOFN) and with (GRID-FN) false negatives, with $\zeta^s = 0.3,\ \forall s \in S$ for the latter. We use SV-MILP on MUSEUM and OFFICE, MV-MILP on GRID-NOFN, and FN-MV-MILP on GRID-FN.

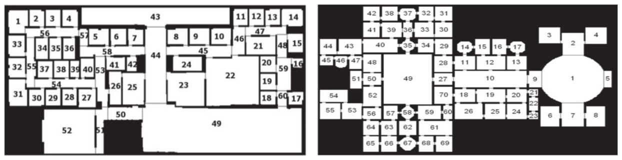

Fig. 1. Environments from [1] used in evaluations, each room is associated with the corresponding vertex number. Left: OFFICE; Right: MUSEUM.

We perform **five experiment sets**, each consisting of 100 instances per environment, except for set 4 in which we double this quantity. The initial configurations (searchers and object positions) are randomly chosen. In all instances, the initial capture belief is zero, i.e., the searchers are not able to detect the object at the start of the mission. For the initial object's location belief, we assume an uniform probability between an assorted number of vertices, chosen randomly. For experiment sets $1-3, 5$, there are five possible initial vertices; for experiment set 4, we vary the number of vertices randomly from two to fifteen; particularly for GRID-FN in set 5, we consider four possible vertices, drawn from each of the 3×3 corner regions of the grid graph, while the initial position of the searchers is drawn from the central portion of the grid.

B. Results

1) Scalability of the MILP models for centralized approach w.r.t. planning horizon length: Fig. 2 shows the solution times for one searcher ($m = 1$) and varying horizons h, for OFFICE, GRID-NOFN and GRID-FN.

The complexity of the model, and therefore the time required to solve it, increases with the complexity of the graph, the robots' sensing range, and the presence of false negatives. OFFICE instances are simpler than GRIDs due to

[3]Code is open source and available at `https://github.com/basfora/milp_mespp.git`

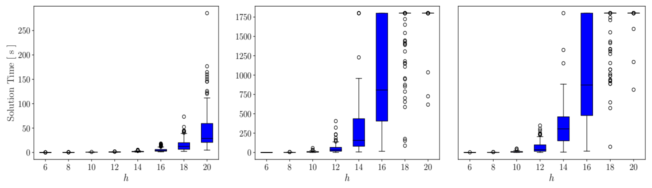

Fig. 2. Solution times with MILP centralized approach, $m = 1$ and varying h. Left: OFFICE. Center: GRID-NOFN. Right: GRID-FN.

the smaller average graph degree, plus capture events limited to the same vertex. As a result, all OFFICE instances are solved to optimality within a couple of minutes (see Fig. 2, left). On the other hand, GRID instances with $h > 16$ tend to hit the solver time limit before an optimal solution can be found. In these sub-optimal cases, the median MIP gap values (not shown in plot) for GRID-FN and $h = 18, 20$ were respectively 11% and 23%. The presence of false negatives in the centralized approach does not have a significant impact for $m = 1$ (Fig. 2 center, right). As defined in Eq. (25), additional intermediate variables are necessary for each searcher and time-step, causing the increase in complexity to become relevant for multiple searchers. This is confirmed by experiment set 2 (Figs. 4-3).

2) Performance of the centralized MILP approach for different team sizes: We choose a planning horizon of $h = 10$, shown previously to be optimally solvable within our time limit for a single searcher in all instances. Figs. 3-4 show the solution times and corresponding MIP gaps for OFFICE, GRID-NOFN and GRID-FN.

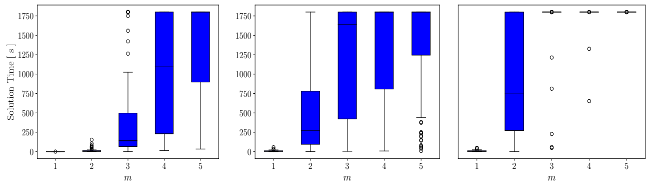

Fig. 3. Solution times with centralized approach, $h = 10$ and varying m. Left: OFFICE. Center: GRID-NOFN. Right: GRID-FN.

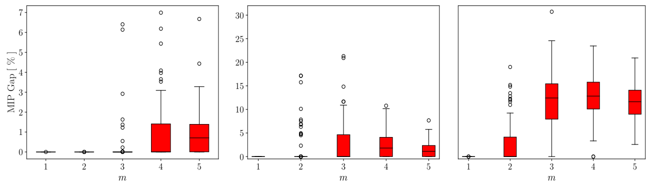

Fig. 4. MIP gaps with centralized approach, $h = 10$ and varying m. Left: OFFICE. Center: GRID-NOFN. Right: GRID-FN.

A larger search team increases the model complexity (see Fig. 3), and consequently even OFFICE instances hit the time limit for $m \geq 3$. These sub-optimal solutions, however, present small MIP gap values ($\leq 7\%$), with median gaps of less than 0.8% (Fig. 4, left). On GRID-NOFN, median gap values are still low ($< 1.2\%$), although we have some outliers (max. 22%). Overall higher gaps are found on GRID-FN, with median values around 12% for $m \geq 3$ (Fig. 4, right). As it becomes comparatively easier to intercept the object with an extra searcher in an environment of this size, the higher MIP gaps in both GRIDs arise when $m = 3, 4$.

3a) Scalability of the MILP models for distributed approach w.r.t. team size: We implement the implicit coordination algorithm described in Sec. VI, replanning at each time step. Figure 5 shows the solution times with a planning horizon $h = 10$ for OFFICE and GRID-FN.

The distributed solution presents a significantly better scalability than the centralized approach under equivalent conditions, which can be seen by comparing the solution times in Fig. 5 (left, right) and Fig. 3 (respectively left, right). The following results from OFFICE illustrate this claim: although for $m = 1$ the median computational times of the centralized and distributed approaches are similar, respectively 0.52 sec and 0.54 sec, for $m = 5$ these values increase to approximately 1800 sec (centralized) and 1.46 sec (distributed). In comparative terms, a 5x increase in the team size caused the median solution time to increase 3x for the distributed algorithm, against a drastic 3400x increase in solution time for the centralized approach.

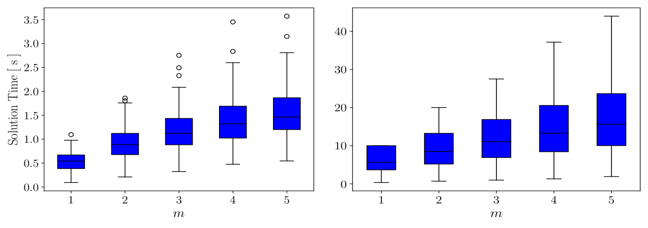

Fig. 5. Solution times with distributed approach, $h = 10$ and varying m. Left: OFFICE. Right: GRID-FN.

3b) Comparative performance of online (distributed) and offline (centralized) MILP search plans: As basis for comparison, we introduce two metrics: the *average mission time*, defined as the time-step the mission ends due to the expiration of the deadline or capture of the object; and the *relative reward loss*, defined as the percentage difference between the distributed and centralized reward functions computed at time $t = 0$. Figure 6 shows the relative reward loss (left) and the average mission time (right) for OFFICE and GRID-FN, for a mission deadline $\tau = 50$ with $h = 10$ and varying m.

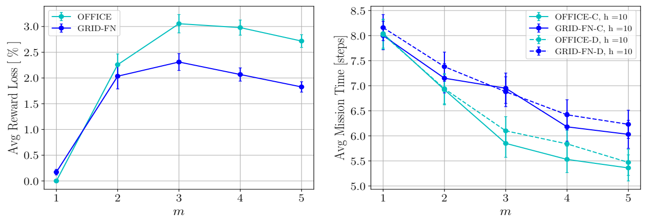

Fig. 6. Performance of distributed and centralized MILP approaches for OFFICE and GRID-FN, $h = 10$. Left: Relative reward loss computed at $t = 0$. Right: Average mission time. Bars show the std. error of the mean.

The relative reward loss for the environments studied in this letter is minimal, as shown in Fig. 6 (left). The higher loss is seen for the OFFICE environment (within 3% of optimal reward) and slightly lower for GRID-FN (2% difference). Recall from Fig. 3 (right) that the centralized approach often fails to solve the GRID-FN problem to optimality in the time given, which might result in a sub-optimal offline plan, however with a higher reward when compared to the proposed distributed plan. This small difference in reward translates into an overall shorter, if at times irrelevant[4], average mission time for the centralized approach (see Fig. 6, right). Given the same planning horizon, the distributed approach often performs nearly as well as the centralized, both w.r.t. reward (within 3%) and mission time (within 6%), with the advantage of requiring significant less time.

4) Comparison between MILP approach and previous state-of-the-art (SoA) algorithm: The solution times for varying planning horizons and $m = 3$ are shown in Fig. 7 for MUSEUM, OFFICE and GRID-FN.

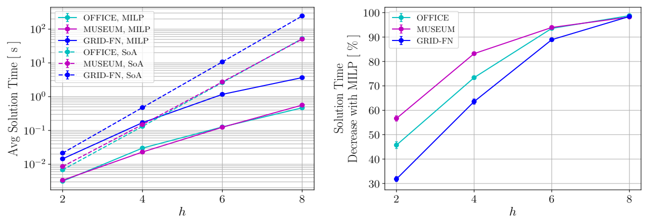

Fig. 7. Comparison of MILP and SoA [1] distributed approaches for OFFICE, MUSEUM and GRID-FN, $m = 3$ and varying h. Left: Average solution time (log scale). Right: Solution time decrease with MILP (relative change with SoA as the reference value). Bars show the std. error of the mean.

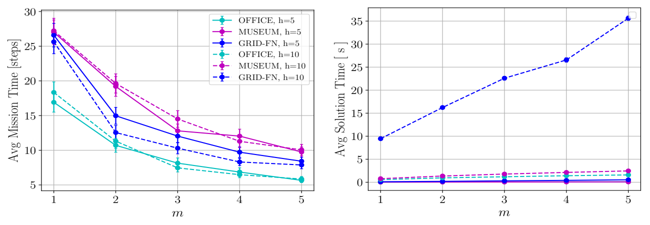

Fig. 8. Performance of MILP distributed approach, $h = 5, 10$ for OFFICE, MUSEUM and GRID-FN. Left: Average mission time. Right: Average solution time. Bars show the standard error of the mean.

The implicit coordination algorithm (SoA) was implemented in C++ by the authors[5] as presented in [1]. The same machine is used to run the MILP and SoA experiments, and no time limit is imposed. While the algorithms provide interchangeable solutions (same computed reward), the computational time required to do so varies greatly between them. The MILP paradigm outperforms the previous SoA w.r.t. computational time in all cases, and this difference becomes more expressive as the planning horizon increases

[4]Note for $m = 3$ in GRID-FN, the false negative causes the actual detection of the target in the distributed, but not in the centralized instance.

[5]Code is open source and available at `https://github.com/jacoban/implicit_coordination`.

(see Fig. 7, left). In average terms, for $h = 8$ and $m = 3$ in the environments tested in this letter, the MILP models allow for a solution time decrease of 98% compared to the previous SoA (see Fig. 7, right).

5) Performance of the MILP distributed approach with different planning horizons: Figure 8 shows the average mission time for $\tau = 50$ and the solution time with $h = 5, 10$ for OFFICE, MUSEUM and GRID-FN.

For both OFFICE and MUSEUM there is virtually no difference in performance for the planning horizons tested (Fig. 8, left). For GRID-FN, the imposed restriction on searchers and object's initial positions (Sec. VII-A) creates a more challenging planning scenario given their relative initial distance. For this case, a longer planning horizon yields better performance, however at the expense of a greater computing time, which grows expressively with the number of searchers (see Fig. 8, right).

VIII. Discussion

In this letter, we proved the MESPP problem to be NP-hard even on seemingly simple instances, i.e. grid graphs, static target, and single searcher. We also presented the first set of MILP models able to encompass multiple searchers, arbitrary capture ranges, and false negatives simultaneously. Our results show that the adoption of MILP as a planning paradigm outperforms the previous state-of-the-art approach, both in terms of planning horizon and computational performance.

Leveraging the powerful techniques and tools used by modern solvers comes with a minor challenge: very rarely (three instances in total), the presolver might deal poorly with small probabilities and deem the problem infeasible. This numerical issue is fixed either by turning the presolver off and increasing the solver timeout, or by keeping the searchers' in their current positions and re-planning on next time-step (always a feasible solution and the one we adopted). We believe this is just a small inconvenience, given the benefits provided by the MILP models.

As shown in our simulations, the trade-off between expected mission time and required computational time is a challenging choice. Specially for practical situations, such choice is dependent upon the desired search mission's goals and is fundamental for its success. Future work will continue investigating MILP as a planning paradigm, towards the generalization of the presented models to handle heterogeneous teams of searchers (humans, ground and aerial vehicles), and the incorporation of connectivity constraints.

Acknowledgment

Funding for this research was provided by the National Robotics Initiative Program of the National Science Foundation, award number #1830497 and PERISCOPE MURI Grant N00014-17-1-2699.

References

[1] G. Hollinger, S. Singh, J. Djugash, and A. Kehagias, "Efficient multi-robot search for a moving target," *Int. J. Robot. Res.*, vol. 28, no. 2, pp. 201–219, 2009.

[2] A. Singh, A. Krause, C. Guestrin, W. J. Kaiser, and M. A. Batalin, "Efficient planning of informative paths for multiple robots." in *Proc. IJCAI*, vol. 7, 2007, pp. 2204–2211.

[3] G. Hollinger, "Search in the physical world," Ph.D. dissertation, Carnegie Mellon University, 2010.

[4] K. Bernhard and J. Vygen, "Combinatorial optimization: Theory and algorithms," *Springer, Third Edition, 2005.*, 2008.

[5] L. Stone, *Theory of optimal search*. Elsevier, 1976, vol. 118.

[6] S. Alpern and S. Gal, *The theory of search games and rendezvous*. Springer Science & Business Media, 2006, vol. 55.

[7] T. Chung, G. Hollinger, and V. Isler, "Search and pursuit-evasion in mobile robotics," *Auton. Robot.*, vol. 31, no. 4, p. 299, 2011.

[8] J. Tisdale, Z. Kim, and J. Hedrick, "Autonomous uav path planning and estimation," *IEEE Robot. Autom. Mag.*, vol. 16, no. 2, pp. 35–42, 2009.

[9] A. Kolling, A. Kleiner, M. Lewis, and K. Sycara, "Computing and executing strategies for moving target search," in *Proc. ICRA*, 2011, pp. 4246–4253.

[10] A. Kolling and S. Carpin, "Pursuit-evasion on trees by robot teams," *IEEE Trans. Robot.*, vol. 26, no. 1, pp. 32–47, 2009.

[11] F. Riccio, E. Borzi, G. Gemignani, and D. Nardi, "Multi-robot search for a moving target: Integrating world modeling, task assignment and context," in *Proc. IROS*, 2016, pp. 1879–1886.

[12] M. Adler, H. Räcke, N. Sivadasan, C. Sohler, and B. Vöcking, "Randomized pursuit-evasion in graphs," in *Comb. Probab. Comput.*, vol. 12, no. 3, 2003, pp. 225–244.

[13] S. Alexander, R. Bishop, and R. Ghrist, "Capture pursuit games on unbounded domains," *lEnseignement Mathematique*, vol. 55, pp. 103–125, 2009.

[14] D. Bhadauria and V. Isler, "Capturing an evader in a polygonal environment with obstacles," in *Proc. IJCAI*, 2011, pp. 2054–2059.

[15] V. Isler and N. Karnad, "The role of information in the cop-robber game," *Theor. Comput. Sci.*, vol. 399, no. 3, pp. 179–190, 2008.

[16] R. Murrieta-Cid, T. Muppirala, A. Sarmiento, S. Bhattacharya, and S. Hutchinson, "Surveillance strategies for a pursuer with finite sensor range," *Int. J. Robot. Res.*, vol. 26, no. 3, pp. 233–253, 2007.

[17] L. Guibas, J. Latombe, S. LaValle, D. Lin, , and R. Motwani, "Visibility-based pursuit-evasion in a polygonal environment," *Int. J. Comput. Geom. Appl.*, vol. 9, no. 5, pp. 471–494, 1999.

[18] M. Kress, K. Y. Lin, and R. Szechtman, "Optimal discrete search with imperfect specificity," *Math. Method. Oper. Res.*, vol. 68, no. 3, pp. 539–549, 2008.

[19] G. Hollinger, S. Yerramalli, S. Singh, U. Mitra, and G. Sukhatme, "Distributed data fusion for multirobot search," *IEEE Trans. Robot.*, vol. 31, no. 1, pp. 55–66, 2015.

[20] G. Hollinger and S. Singh, "Multirobot coordination with periodic connectivity: Theory and experiments," *IEEE Trans. Robot.*, vol. 28, no. 4, pp. 967–973, 2012.

[21] J. Banfi, N. Basilico, and F. Amigoni, "Multirobot reconnection on graphs: Problem, complexity, and algorithms," *IEEE Trans. Robot.*, vol. 34, no. 5, pp. 1299–1314, 2018.

[22] M. Morin, I. Abi-Zeid, P. Lang, L. Lamontagne, and P. Maupin, "The optimal searcher path problem with a visibility criterion in discrete time and space," in *Proc. FUSION*, 2009, pp. 2217–2224.

[23] J. Shewchuk, "Delaunay refinement algorithms for triangular mesh generation," *Comp. Geom. Theor. Appl.*, vol. 22, no. 1–3, pp. 21–74, 2002.

[24] A. Itai, C. H. Papadimitriou, and J. L. Szwarcfiter, "Hamilton paths in grid graphs," *SIAM Journal on Computing*, vol. 11, no. 4, pp. 676–686, 1982.

[25] O. Madani, S. Hanks, and A. Condon, "On the undecidability of probabilistic planning and related stochastic optimization problems," *Artif Intell*, vol. 147, no. 1-2, pp. 5–34, 2003.

[26] F. Glover, "Improved linear integer programming formulations of nonlinear integer problems," *Management Science*, vol. 22, no. 4, pp. 455–460, 1975.

[27] J. Banfi, N. Basilico, and S. Carpin, "Optimal redeployment of multirobot teams for communication maintenance," in *Proc. IROS*. IEEE, 2018, pp. 3757–3764.

[28] G. Optimization, "Inc.,gurobi optimizer reference manual, 2019," 2019.