2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

Optimizing Topologies for Probabilistically Secure Multi-Robot Systems

Remy Wehbe and Ryan K. Williams

Abstract— **In this paper, we optimize the interaction graph of a multi-robot system (MRS) by maximizing its probability of security while requiring the MRS to have the fewest edges possible. Edges that represent robot interactions exist according to a probability distribution and security is defined using the control theoretic notion of left invertibility. To compute an optimal solution to our problem, we first start by reducing our problem to a variation of the rooted k-connections problem using three graph transformations. Then, we apply a weighted matroid intersection algorithm (WMIA) on matroids defined on the edge set of the interaction graph. Although the optimal solution can be found in polynomial time, MRSs are dynamic and their topologies may change faster than the rate at which the optimal security solution can be found. To cope with dynamic behavior, we present two heuristics that relax optimality but execute with much lower time complexity. Finally, we validate our results through Monte Carlo simulations.**

I. Introduction

Communication and sensing play an integral role in the successful operation of Multi-Robot Systems (MRSs). Essentially, tasks such as consensus, swarming, monitoring, etc., are possible through the reliable exchange of information. The notion of reliability in communication is often taken for granted. Unfortunately, real systems may have their interactions compromised by natural phenomena such as noise and interference [1], or by malicious agents [2]. Ideally, one would like to have an MRS where all robots can interact with each other. However, this is not always possible due to limited communication range [3]. Additionally, excess communication adds cost in the form of sensors and transmitters, increases the burden on already limited power budgets, and can overly constrain a system spatially. Motivated by these limitations, we seek to perform topology optimization where we limit robot interactions to a minimum, while maximizing a topological metric of security.

Multi-Robot Systems are vulnerable to a variety of attacks, ranging from attacks directly targeting communications like a denial of service attack [4], to attacks that manipulate the system dynamics through false data like a false data injection attack [5], to even stealthier attacks like replay [6] or covert [7] attacks that try to mask malicious behavior by manipulating sensors and actuators to mimic nominal operation. Fortunately, recent work has focused on methods to counteract or prevent malicious behavior in MRSs. Some of these works include [8]–[10] which study conditions for the detection and identification of malicious behaviour. Similarly, [11], [12] explore the probability of detecting malicious behavior when interactions are assumed to be probabilistic. [13]–[15] present algorithms for resilient consensus of MRSs in the presence of malicious agents. Topology optimization has also been extensively studied in the literature. Works such as [16]–[18] look at methods for optimizing graph topologies to minimize communication cost while maximizing system reliability. Similarly, [19], [20] tackle the problem of topology optimization with communication constraints to maximize consensus rates. [21] looks at the problem of deploying mobile sensors in a network to enhance the system's connectivity and coverage. A specific optimization problem of interest in this work is finding a subgraph of minimum cost such that there exists k-vertex-disjoint paths from a given node to all other nodes in the graph. This is one of a family of problems known as the rooted k-connections problem. The mentioned version of this problem was first solved by [22] using submodular flows. [23] showed that the complicated approach involving submodular flows is avoidable, and instead proved that the problem can be solved as a weighted matroid intersection problem. Matroids are a generalization of the notion of linear independence with various applications in combinatorial optimization, graph theory, etc., [24]. Recently, matroids have found application in multi-robot systems modeling constraints of combinatorial optimization problems [25], [26].

In this work, we assume that robot interactions are probabilistic and adopt the notion of probabilistic security from [11]. The concept of security is based on system left invertibiltiy which can be characterized as a topological property of the interaction graph using vertex separators [8]. As such, considering probabilistic interactions as a cost, we aim to find the interaction graph with the fewest edges that maximizes the metric of probabilistic security. To solve this combinatorial optimization problem, we derive a series of graph transformations to reduce our problem to the rooted k-connections problem. Then we apply results from matroid theory and recent developments in solving the rooted k-connections problem to find an optimal solution to our problem. The optimal algorithm runs in polynomial time, however, it is still too slow to run real-time for dynamic MRSs. To this end, we present two fast and efficient heuristics to solve the problem while sacrificing optimality. Finally, we show the validity of our results through Monte Carlo simulations and an environmental monitoring task.

II. Preliminaries

A. Modeling MRSs and their Interactions

We model our MRS as a group of n robots in the set $\mathcal{I}_r = \{1, 2, ..., n\}$, observed by one of m observers in

R. Wehbe and R. K. Williams are with the Department of Electrical and Computer Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA USA, E-mail: {*rewehbe, rywilli1*}*@vt.edu*.

the set $\mathcal{I}_o = \{1, 2, ...m\}$. The number of observers cannot exceed the number of robots, i.e., $n \geq m$, and each observer can only measure one robot. Deterministic interactions (communication/sensing) are modeled using a directed graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$. The node set $\mathcal{V} = \{v_1, v_2, ..., v_{n+m}\}$ represents robots and observers, and the edge set $\mathcal{E} = \{e_{ij} | v_i, v_j \in \mathcal{V}\}$ represents interactions, where node i can interact with node j if $e_{ij} \in \mathcal{E}$. The notation e_i denotes the i^{th} edge of some set. Probabilistic interactions are modeled using a probabilistic graph denoted by $\mathscr{G}(\mathscr{V}, \mathscr{E})$, where $\mathscr{V}$ is a *deterministic* vertex set and $\mathscr{E}$ is a *probabilistic* edge set. A probabilistic edge is an edge that does not always exist, rather it exists according to a probability p_{ij} associated with e_{ij}. This probability models uncertainty, interference, etc., variables that lead to random edge failures. We assume that at every time instance $t \in \mathbb{R}_{\geq 0}$, the probabilistic graph is realized yielding an instance of a deterministic graph with $\mathcal{V} = \mathscr{V}$ and $\mathcal{E} \subseteq \mathscr{E}$.

Assumption 1: All edges are assumed to be independent such that the probability of edges e_1 and e_2 being realized in an instance of $\mathcal{E}$ simultaneously is $P(e_1 \cap e_2) = P(e_1)P(e_2)$, where the symbols $\cap$ and $\cup$ respectively represent the Boolean operations of conjunction/AND and disjunction/OR.

A simple path between two nodes i and j, denoted by $l_{ij} = \{e_1, e_2, ..., e_{|l|}\}$, is a finite set of edges that connects vertices such that no vertex is repeated along the path. A set of α-vertex-disjoint paths between two nodes i and j, denoted by $\mathcal{H}_{ij}^{\alpha} = \{l_1, l_2, ..., l_\alpha\}$, is defined as a set of paths not sharing any vertices except the source and terminal nodes. Unless stated otherwise, any mention of disjoint paths strictly refers to vertex-disjoint paths. Since different sets of disjoint paths may exist between two nodes, define by $\mathcal{R}_{ij}^{\alpha}$ the set of all possible α-disjoint paths between i and j. A vertex separator $\mathcal{S}_{ij}$ is a set of vertices whose removal disconnects node i from j. Menger's theorem [27] states that the minimum size of a vertex separator between two non-adjacent nodes i and j, denoted by $|S_{ij}|_{min}$, is equal to the maximum number of vertex disjoint paths between these nodes.

B. Attack Model and Conditions for System Security

To model the state of the MRS, we adopt the control theoretic approach presented in [8]. Denote by $x_i(t)$ the scalar state associated with robot i, and by $y_j(t)$ the measurement obtained by observer j. Define $\mathbf{x}(t) \triangleq [x_1(t), x_2(t), ..., x_n(t)] \in \mathbb{R}^n$ as a stacked vector of $x_i(t)$ and $\mathbf{y}(t) \triangleq [y_1(t), y_2(t), ..., y_m(t)] \in \mathbb{R}^m$ as a stacked vector of $y_j(t)$. Then the dynamics of the MRS under attack become,

$$\begin{aligned} \mathbf{x}^a(t+1) &= A\mathbf{x}^a(t) + B^a\mathbf{u}^a(t) + \mathbf{w}(t) \\ \mathbf{y}^a(t) &= C\mathbf{x}^a(t) + D^a\mathbf{u}^a(t) \end{aligned} \tag{1}$$

where $\mathbf{u}^a(t)$ and $\mathbf{w}(t)$ are respectively stacked vectors of malicious input and process noise. Setting $\mathbf{u}^a = 0$ results in the MRS's control law under nominal conditions. Note that the topology of the interaction graph is reflected in the system matrices in (1) through structural zeros, i.e., zeros associated with robots $i, j \in \mathcal{I}_r$ such that $e_{ij} \notin \mathcal{E}$ for some realized edge set $\mathcal{E}$. State estimation is performed by a centralized detector using a linear filter based on observer measurements. The attackers aim to inject a malicious input $\mathbf{u}^a$ that goes undetected by the centralized detector. Specifically, an attack is called *perfect* if an attacker can maintain $\Delta\mathbf{z}(t) \triangleq \mathbf{z}(t) - \mathbf{z}^a(t) = 0$ while $\mathbf{u}^a(t) \neq 0$, where $\mathbf{z}(t)$ and $\mathbf{z}^a(t)$ are respectively the nominal and attacked filter residuals. Then, an MRS is said to be *secure* if it can avoid perfect attacks. A necessary and sufficient condition to avoid perfect attacks is system left invertibility [8]. Assume that an MRS is under attack by p attackers, where the number of attackers cannot exceed the number of observers, i.e., $p \leq m$. Additionally, denote by ([A],[C]) a structural system where certain properties (e.g. left invertibility) hold for almost all admissible realizations of system matrices A, B, C, D possessing a fixed pattern of zero and non-zero entries. Then a graph-theoretic condition for structural left invertibility which implies *deterministic* detection of attacks is given by,

Theorem 1 ([8], Graph-theoretic Security): A structural system ([A],[C]) as defined in (1) is secure from p attackers for all feasible sets of malicious nodes if and only if for each robot $i \in \mathcal{I}_r$, the minimum number of vertex separators between robot i and the node o satisfies $|S_{io}|_{min} \geq p$.

Where $o \in \mathscr{V}$ is an added vertex with incoming directed edges from all observers as shown in Figure 1a. Figure 1b shows an MRS with the minimum number of deterministic interactions satisfying the conditions of Theorem 1 for $p = 1$. In the present work, robot interactions are assumed to be probabilistic and our previous work [11] extends the concept of deterministic security into the probabilistic domain.

Theorem 2 ([11]), Probabilistic Security): The probability that system ([A],[C]) as defined in (1) is secure for all feasible sets of malicious nodes p, is equal to the probability of the conjunction of $\mathcal{R}_{io}^p \quad \forall i \in \mathcal{I}_r$, where $\mathcal{R}_{io}^p$ is calculated using the disjunction of all the events $\mathcal{H}_{io,l}^p \in \mathcal{R}_{io}^p$. That is,

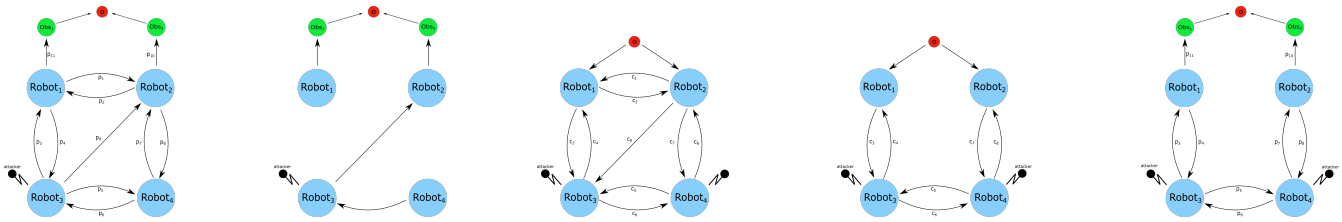
$$P_{secure} = P(\bigcap_{i=1}^{n} \bigcup_{l=1}^{|\mathcal{R}_{io}^p|} \bigcap_{\zeta=1}^{p} \ell_\zeta) \tag{2}$$

where $\bigcap_{\zeta=1}^{p} \ell_\zeta$ represents the set $\mathcal{H}_{io,l}^p$ and $\ell_\zeta = \{e_1, e_2, ..., e_\beta\}$ represents the ζth path in $\mathcal{H}_{io,l}^p$.

Solving for (2) is at least NP-Complete [11] and requires the use of Binary Decision Diagrams [28]. An approximation of this problem can be found in [29]. Additionally, a machine learning approach to solving (2) can be found in [12].

C. Matroid Theory

To solve the optimization problem defined in Section III, we need to introduce the concept of *matroids*. First, we give a few definitions from set theory. Given two sets I and J, the notation $I \wedge J$ denotes set intersection, $I \vee J$ denotes set union, $I - J$ denotes the set of elements in I but not in J, and $I \Delta J$ denotes the set $(I - J) \vee (J - I)$. For a singleton set $\{x\}$, the notation $I + \{x\}$, abbreviated as $I + x$, denotes adding the element x to the set I ($I - x$ is similarly defined). Matroids are denoted by $\mathcal{M}(\mathcal{S}, \mathcal{I})$ where $\mathcal{S}$ is the ground set of the matroid and $\mathcal{I}$, formally known as the set



(a) Interaction graph (b) Min. for $p = 1$ (c) Transformed graph (d) 2-rooted graph (e) Min. for $p = 2$

Fig. 1: Multi-Robot Systems with robots colored in blue, observers in green, node 'o' in red, and attackers in black

of independent sets, is a non-empty collection of subsets of $\mathcal{S}$ satisfying the following axioms:

1) $\emptyset \in \mathcal{I}$
2) if $I \in \mathcal{I}$ and $J \subseteq I$, then $J \in \mathcal{I}$
3) if $I, J \in \mathcal{I}$ and $|I| < |J|$, then $\exists z \in J - I \mid I + z \in \mathcal{I}$

Generally, to establish whether a set belongs to the independent set $\mathcal{I}$, one uses a function called an *oracle*. Assuming the elements of the ground set have an associated cost/weight function $w : \mathcal{S} \to \mathbb{R}$, then a problem of interest is finding an independent set of maximum weight; $I \in \mathcal{I} \mid w(I)\ is\ max$. For this problem, the greedy algorithm is guaranteed to be optimal. Another problem of interest, particularity for this work, is finding a minimum weight common independent set of fixed cardinality between two matroids $\mathcal{M}_1(\mathcal{S}_1, \mathcal{I}_1)$ and $\mathcal{M}_2(\mathcal{S}_2, \mathcal{I}_2)$; $I \in \mathcal{I}_1 \wedge \mathcal{I}_2 \mid |I| = cst, w(I)\ is\ min$. This problem is solvable in polynomial time given that the matroids have oracles that run in polynomial time [24]. The intersection of three or more matroids is NP-hard [24].

III. Problem Formulation

Given an interaction graph $\mathscr{G}(\mathscr{V}, \mathscr{E})$, we aim to find a subgraph $\bar{\mathscr{G}}(\mathscr{V}, \bar{\mathscr{E}})$ that maximizes the probability that the MRS is secure from adversarial attacks, subject to the constraint of fixing the cardinality of the edge set to be the minimum number of edges possible while still obtaining a secure graph. Formally, the combinatorial optimization problem is given by,

$$\begin{aligned} \underset{\bar{\mathscr{E}} \subseteq \mathscr{E}}{\text{maximize}} \quad & P\Big(\bigcap_{i=1}^{n} \bigcup_{l=1}^{|\bar{\mathcal{R}}_{io}^{p}|} \bigcap_{\zeta=1}^{p} \ell_\zeta\Big) \\ \text{subject to} \quad & |\bar{\mathscr{E}}| = np \end{aligned} \tag{3}$$

where np, number of robots multiplied by number of attackers, is the theoretical lower bound for the number of edges that need to exist for an MRS to be secure [12], and $\bar{\mathcal{R}}_{io}^{p}$ is the set of disjoint paths formed by $\bar{\mathscr{E}}$. Since the size of the set $\bar{\mathcal{E}}$ is constrained to be the minimum possible size $|\bar{\mathcal{E}}| = np$, then every edge in the set $\bar{\mathcal{E}}$ is necessary for the MRS to satisfy the conditions of security given in Theorem 1 (if we remove any edge, we immediately lose security). As a result, maximizing the probability of disjoint paths is the same as maximizing the probability of the edge set that forms these disjoint paths [1]; i.e. maximizing $P(\bigcap_{i=1}^{n} \bigcup_{l=1}^{|\bar{\mathcal{R}}_{io}^{p}|} \bigcap_{\zeta=1}^{p} \ell_\zeta)$ is equivalent to maximizing $P(\bar{\mathcal{E}})$. Since edges are assumed to be independent, then $P(\bar{\mathcal{E}}) = P(e_1 \cap e_2 \cap ... \cap e_{|\bar{\mathcal{E}}|}) = P(e_1) \times P(e_2) \times ... \times P(e_{|\bar{\mathcal{E}}|})$. The optimization then becomes:

$$\begin{aligned} \underset{\bar{\mathscr{E}} \subseteq \mathscr{E}}{\text{maximize}} \quad & \prod_{e_{ij} \in \bar{\mathscr{E}}} P(e_{ij}) \\ \text{subject to} \quad & |\bar{\mathscr{E}}| = np \\ & \exists\ \bar{\mathcal{H}}_{io}^{p}\ \forall\ i \in \mathcal{I}_r \end{aligned} \tag{4}$$

Note, however, that by reducing (3) to (4), we dropped the notion of the existence of disjoint paths in the maximization. As such, we added the necessity of having disjoint paths as a second constraint in (4), where $\bar{\mathcal{H}}_{io}^{p}$ is the set of disjoint paths formed by $\bar{\mathscr{E}}$. An optimal solution to (4) can be obtained by reformulating our problem into a variation of the rooted k-connections problem which can itself be solved with the help of a weighted matroid intersection algorithm. In our case, k should be set as the number of attackers, p. We will use the terms p-connections and p-disjoint paths from here on.

The rooted p-connections problem [23] is a family of problems that deals with finding a minimum cost subgraph with minimum edges such that there exists p-disjoint paths from a given root node, to every other node in the graph. For example, Figure 1d is a 2-rooted minimal subgraph of Figure 1c. Disjoint paths can be vertex disjoint, edge disjoint, or have an upper bound on the number of times a vertex that can be repeated. In this work, we are specifically interested in vertex-disjointness. To solve problem (4) we need to find a minimum set of edges that maximizes the probability that there exists p-disjoint paths from every robot in the MRS to the target node 'o'. Note that in our problem, edges only have an associated probability and not an explicit cost[2]. Additionally, the rooted p-connections problem applies to deterministic graphs. To enable us to select edges that maximize probability, we can think of edges as deterministic with their probability as an associated cost. This allows us to apply a linear transformation to convert probability to cost, where the most probable path in the original graph becomes

[1]For a non-minimal edge set $|\bar{\mathcal{E}}| > np$, the equality does not hold.

[2]There is an implicit cost to interactions, this is why we aim to minimize the number of edges in the interaction graph. All edges have the same implied cost, but the importance of an edge lies in its probability.

the shortest path in the transformed graph. To reduce our problem to a variation of the rooted p-connections problem, we apply the following three transformations to the graph $\mathscr{G}(\mathscr{V},\mathscr{E})$ to obtain $\mathscr{G}(\mathscr{V},\mathscr{E}_{\odot})$:

1) Perform a series simplification to remove observers by connecting the robots directly to the node o.
2) Reverse the direction of every edge in the graph.
3) Convert edge probability to cost by applying the following transformation: $c_{ij} = -ln(p_{ij})$.

Theorem 3: Solving (4) on $\mathscr{G}(\mathscr{V},\mathscr{E})$ is equivalent to solving the rooted p-connections problem on $\mathscr{G}(\mathscr{V},\mathscr{E}_{\odot})$.

Proof: We will show how transformations 1-3 make the problems equivalent. Step 1 removes nodes, specifically observers, which are not required to have p-disjoint paths to the node 'o'. Step 2 is needed because our problem involves finding disjoint paths from all robots to 'o', while the p-connections problem finds disjoint paths from 'o' to all robots. These two problems can be made equivalent by reversing each edge's direction. Step 3 allows us to convert probability to cost. Applying the transformation to (4) we get $-ln(\prod_{i=1}^{|\bar{\mathcal{E}}|} P(e_i)) = \sum_{i=1}^{|\bar{\mathcal{E}}|} c(e_i)$. Because of the negative sign, minimizing the sum of the cost will maximize the product of the probability [30]. And the result follows. The transformation of Figure 1a is shown in Figure 1c. ■

IV. Matroid Independence Oracles

The family of p-connections problems was optimally solved in [23] using a polyhedral description formulation. Then, it was shown that some of the problems in the family can be reduced to a matroid intersection problem using count matroids. The mathematical formulation involved is tedious and requires a deep understanding of submodular optimization and matroid theory. As such, we will present some of the results of [23] using a simplified algorithmic approach tailored specifically to our problem. The interested reader is referred to [22], [23] for a more in-depth mathematical approach. From this point on, any mention of the p-connections problem refers to our single problem of interest.

Given the graph $\mathscr{G}(\mathscr{V},\mathscr{E}_{\odot})$, we define two matroids $\mathcal{M}_1$ and $\mathcal{M}_2$, along with their oracles, who's weighted intersection results in a solution to the rooted p-connections problem. $\mathcal{M}_1(\mathscr{E}_{\odot},\mathcal{I}_1)$ is the partition matroid on the ground set $\mathscr{E}_{\odot}$, where membership to the independence set $\mathcal{I}_1$ can be established using the following oracle: Given a set $I \subseteq \mathscr{E}_{\odot}$, find the graph $G(V,I)$ induced by the edges of I. Then, check the in-degree of every vertex $v \in V$, denoted by $d_i(v)$. If the condition $d_i(v) \leq p \ \ \forall v$ is satisfied, then $I \in \mathcal{I}_1$.

$\mathcal{M}_2(\mathscr{E}_{\odot},\mathcal{I}_2)$ is the direct sum of two matroids $\mathcal{M}_2^a(\mathscr{E}_{\odot}^a,\mathcal{I}_2^a)$ and $\mathcal{M}_2^b(\mathscr{E}_{\odot}^b,\mathcal{I}_2^b)$, where $\mathscr{E}_{\odot}^a$ is the set of edges outgoing from the node o, and $\mathscr{E}_{\odot}^b$ is the set of remaining edges, i.e., $\mathscr{E}_{\odot}^b = \mathscr{E}_{\odot} - \mathscr{E}_{\odot}^a$. The first matroid, $\mathcal{M}_2^a(\mathscr{E}_{\odot}^a,\mathcal{I}_2^a)$, is a free matroid, which means that any subset of $\mathscr{E}_{\odot}^a$ belongs to the independent set. As such, we do not need an oracle to check for independence as every subset is independent.

$\mathcal{M}_2^b(\mathscr{E}_{\odot}^b,\mathcal{I}_2^b)$ is a count matroid defined on the graph $\mathscr{G}(\mathscr{V}^b,\mathscr{E}_{\odot}^b)$, where $\mathscr{V}^b$ is the set of vertices excluding o, i.e., $\mathscr{V}^b = \mathscr{V} - o$. The oracle takes as input the graph $\mathscr{G}(\mathscr{V}^b,\mathscr{E}_{\odot}^b)$, an independent set $I \in \mathcal{I}_2^b$, and an edge $e_{sz} \in \mathscr{E}_{\odot}^b - I$ from the ground set not already in the independent set. It then returns whether or not $I' = I + e_{sz}$ belongs to the independent set, i.e., if $I' \in \mathcal{I}_2^b$. As such, starting with the empty set, $\emptyset \in \mathcal{I}_2^b$, and recursively checking the independence oracle for every added edge, one can determine whether a subset of edges belongs to the independent set. The oracle begins by constructing an undirected bipartite graph $G(V',V'';E)$ using the following rules:

1) For all $v \in \mathscr{V}^b$, create a node $v' \in V'$ and a node $v'' \in V''$. Then connect v' and v'' using a single edge.
2) For every edge in the independent set I, $e_{ij} \in I$, add an edge between nodes $i' \in V'$ and $j'' \in V''$.

Next, for every node in $G(V',V'';E)$, we calculate an upper bound on its degree, denoted by $U(v)$, using:

1) $U(s') = 1$ 3) $U(v') = 1 \ \ \forall v' \in V' - s'$
2) $U(z'') = 0$ 4) $U(v'') = p \ \ \forall v'' \in V'' - z''$

We remind the reader that s and z are the source and terminal nodes of the edge e_{sz}. If the undirected graph $G(V',V'';E)$ can be orientated (directed) such that the in-degree of every node, $d_i(v)$, does not exceed the upper bound function, $d_i(v) \leq U(v) \ \forall v \in V',V''$, then the set I' is independent. To check if such an orientation is possible, apply the orientation Algorithm 1 [23]. To apply step 6, add a node q to the graph, connect all the bad nodes from the set Z_0 to q, then check for every node $v \notin Z_0$, if there exists a path from v to q. If such a path exists, add v to the set Z. To apply step 15, use the path from v to q computed in step 6 and reverse the direction of all the edges along this path. In step 16, discard the added node q and repeat from step 2.

Algorithm 1: Orientation Algorithm

```
1  Start with an arbitrary orientation;
2  Find the set of bad nodes Z_0 having d_i(v) > U(v);
3  if Z_0 = ∅ then
4  |  Orientation exists, Return;
5  else
6  |  Find the set of nodes Z from which Z_0 is reachable;
7  end
8  if Z = ∅ then
9  |  Orientation does not exist, Return;
10 else
11 |  Find a node v ∈ Z satisfying d_i(v) < U(v);
12 |  if No such v exists then
13 |  |  Orientation does not exist, Return;
14 |  else
15 |  |  Reorient all edges along the path from v to Z;
16 |  |  Repeat from step 2;
17 |  end
18 end
```

V. Weighted Matroid Intersection Algorithm

To find the optimal solution for the rooted p-connections problem, and as a result to optimization (4), we need to find a minimum weight common independent set between $\mathcal{M}_1(\mathscr{E}_{\odot},\mathcal{I}_1)$ and $\mathcal{M}_2(\mathscr{E}_{\odot},\mathcal{I}_2)$. A weighted matroid intersection algorithm (WMIA) [24], [31] takes as input two

matroids, namely $\mathcal{M}_1(\mathscr{E}_\odot, \mathcal{I}_1)$ and $\mathcal{M}_2(\mathscr{E}_\odot, \mathcal{I}_2)$, a common independent set of maximum weight $\mathcal{I}_{max} \in \mathcal{I}_1 \wedge \mathcal{I}_2$, and a weight/cost function $c_{ij}(e_{ij}) : \mathscr{E}_\odot \rightarrow \mathbb{R}$. The algorithm returns a common independent set $\mathcal{I}'_{max}$ of maximum weight satisfying $|\mathcal{I}'_{max}| = |\mathcal{I}_{max}| + 1$, if it exists. Then, starting with $\mathcal{I}_{max} \in \mathcal{I}_1 \wedge \mathcal{I}_2 = \emptyset$, and recursively applying the WMIA algorithm, we can build a maximum weight common independent set of desired cardinality, namely $|\mathcal{I}_{max}| = np$, as required by the optimization problem (4). Note, however, that we require a common independent set of minimum weight. So, we reverse the sign of the cost of every edge in the ground set $\mathscr{E}_\odot$ such that $c_{ij} = -c_{ij}$. The new ground set is denoted by $\mathscr{E}_\odot^-$. Finding a common independent set of maximum weight in $\mathscr{E}_\odot^-$ is equivalent to finding a common independent set of minimum weight in $\mathscr{E}_\odot$.

The WMIA algorithm goes as follows:

1) Construct a digraph $D_{M_1M_2}$:
   a) The vertex set of $D_{M_1M_2}$ is $\mathscr{E}_\odot^-$
   b) Let $y \in \mathcal{I}_{max}$ and let $x \in \mathscr{E}_\odot^- - \mathcal{I}_{max}$ then
      i) e_{yx} is an arc in $D_{M_1M_2}$ iff $\mathcal{I}_{max} - y + x \in \mathcal{I}_1$
      ii) e_{xy} is an arc in $D_{M_1M_2}$ iff $\mathcal{I}_{max} - y + x \in \mathcal{I}_2$
2) Find the sets X_1, X_2 such that:
   a) $X_1 = \{x \in \mathscr{E}_\odot^- - \mathcal{I}_{max} \mid \mathcal{I}_{max} \vee \{x\} \in \mathcal{I}_1\}$
   b) $X_2 = \{x \in \mathscr{E}_\odot^- - \mathcal{I}_{max} \mid \mathcal{I}_{max} \vee \{x\} \in \mathcal{I}_2\}$
3) Set length/cost $l(x)$ of all the nodes of $D_{M_1M_2}$ using:
   a) $l(x) = c(x) \quad if\ x \in \mathcal{I}_{max}$
   b) $l(x) = -c(x) \quad if\ x \notin \mathcal{I}_{max}$
4) Find the shortest path, denoted by sp, between X_1 and X_2, if several paths of same length/cost exist, choose the paths with the least amount of edges.
   a) If sp exists, then $\mathcal{I}'_{max} = (\mathcal{I}_{max} \Delta sp)$
   b) Else $\mathcal{I}_{max}$ is the largest common independent set

A solution exists if we can find an $\mathcal{I}_{max} \mid |\mathcal{I}_{max}| = np$. In steps 1 and 2, the oracles of section IV are applied to establish membership in the independent set. Oracle checks can be computed in parallel to improve performance. Matroid $\mathcal{M}_1$ has only one oracle and its application is straight forward. However, for matroid $\mathcal{M}_2$, we apply the oracle of $\mathcal{M}_2^a(\mathscr{E}_\odot^{a,-}, \mathcal{I}_2^a)$ if $x \in \mathscr{E}_\odot^{a,-}$, and the oracle of $\mathcal{M}_2^b(\mathscr{E}_\odot^{b,-}, \mathcal{I}_2^b)$ if $x \in \mathscr{E}_\odot^{b,-}$. Since, $\mathcal{M}_2^a$ is a free matroid, then $I + x \in \mathcal{I}_2$ is always true whenever $x \in \mathscr{E}_\odot^{a,-}$. Then, we only need to check the oracle if $x \in \mathscr{E}_\odot^{b,-}$. After the WMIA terminates, the optimal graph $\bar{\mathscr{G}}(\mathscr{V}, \bar{\mathscr{E}})$ is obtained from the graph induced by the edges in the optimal set I_{max}. Note that all the transformations previously applied have to be reversed. An example is going from Figure 1d to 1e.

VI. Heuristic Approach

The optimal solution to (4) is guaranteed to be found in polynomial time, since all the oracle algorithms and the WMIA run in polynomial time [23], [31]. However, MRSs are dynamic and the topology $\mathscr{G}(\mathscr{V}, \mathscr{E})$ may change much faster than the rate at which our problem can be solved optimally. To this end, we develop two heuristics to solve (4) that sacrifice optimality, but run quickly and efficiently. In theory, greedily solving a matroid intersection is not guaranteed to converge [24]. However, empirically, we have seen that the greedy algorithm does converge in the majority of cases. Thus we will augment the greedy algorithm with another heuristic to guarantee convergence. The second heuristic utilizes Suurballe's algorithm [32] and can either compliment the greedy algorithm or be used as a stand alone heuristic. The greedy algorithm converges to the optimal solution more often, while only using the heuristic based on Suurballe is faster. The greedy algorithm takes as input $\mathscr{G}(\mathscr{V}, \mathscr{E}_\odot)$ and is given in Algorithm 2. If the greedy algorithm converges, the solution is recovered by looking at the graph induced by the edges in I^g_{max}, then reversing the transformations previously applied.

Algorithm 2: Greedy Algorithm

```
Set I^g_max = ∅ ;
while |I^g_max| < np do
    Find the set E_test = 𝒮⊙^- − I^g_max;
    Find the maximum cost edge e_max = max(E_test);
    if e_max ≠ ∅ then
        if I^g_max + e_max ∈ (I_1 & I_2) then
            I^g_max = I^g_max + e_max, go back to step 2;
        else
            E_test = E_test − e_max, go back to step 4;
        end
    else
        Greedy did not converge, run Algorithm 3;
    end
end
```

If the greedy algorithm fails to converge, we will use a heuristic based on Suurballe's algorithm to find a solution to (4), if it exists. Suurbaalle's algorithm [32] is a special case of the minimum cost flow algorithm and is used to find the shortest set of p-disjoint paths between any two nodes in the graph. This is done by starting with the shortest path, then recursively augmenting the shortest set of disjoint paths, one path at a time. So, starting with the original graph $\mathscr{G}(\mathscr{V}, \mathscr{E})$, only apply transformation 3 from Section III to get $\mathscr{G}(\mathscr{V}, \mathscr{E}_h)$, effectively converting probability to cost. The heuristic goes as follow: pick a robot, find the shortest set of p-disjoint paths from this robot to 'o' ($\mathcal{H}^p_{io,min}$) using Algorithm 3, update the cost of the edges in the graph according to $c_{ij} = 0\ if\ e_{ij} \in \mathcal{H}^p_{io,min}$ (This step is necessary for minimality since it encourages Algorithm 3 to reuse edges already used in previous disjoint paths whenever possible). Repeat until you have found a set of p-disjoint paths from every robot to 'o'. Combine the edges from all the sets of disjoint paths and reverse transformation 3 to form the solution. By the nature of its construction, this heuristic is guaranteed to find a solution if it exists. However, we cannot guarantee that the number of edges is always minimum, even if empirically we have seen that non-minimal cases are rare.

Algorithm 3 takes as input a set of k-disjoint paths, $\mathcal{H}^k_{st,min}$ (s: source, t: target), of minimum total length along with the graph $\mathscr{G}(\mathscr{V}, \mathscr{E}_h)$, and outputs a set of $(k+1)$-disjoint paths of minimum total length, if it exists. Steps 1-7 apply

a graph transformation, where in step 2 every node v that shows up along a path in $\mathcal{H}^k_{st,min}$, except $\{s,t\}$, is split into v_{in} and v_{out}, v_{in} is connected to v_{out} by a directed auxiliary edge having zero weight, all incoming edges to v are redirected to v_{in}, and all outgoing edges of v are redirected to v_{out}. In steps 5 and 6 make sure to also apply the transformations to the auxiliary edges. In Step 8, we use the Bellman-Ford algorithm [33] which allows for negative weights in the graph to find the shortest path sp. In step 12, we first need to remove any auxiliary edges from sp by joining v_{in} and v_{out} back to v. Next, we need to compare the edges in sp and $\mathcal{H}^k_{st,min}$ to perform an edge canceling step. More specifically, for every $e_{ij} \in \mathcal{H}^k_{st,min}$ we check if sp has an e_{ji} edge. The edges e_{ij} and e_{ji} cancel each other out and are removed from their respective sets. Finally, combining the remaining edges from the sets sp and $\mathcal{H}^k_{st,min}$ forms the new set of disjoint paths $\mathcal{H}^{k+1}_{st,min}$.

Algorithm 3: Suurballe's Algorithm

1 **forall** v *induced by* $e_{ij} \in \mathcal{H}^k_{st,min}$ - $\{s,t\}$ **do**
2 | Split v into v_{in} and v_{out}, and redirect edges;
3 **end**
4 **forall** $e_{ij} \in \mathcal{H}^k_{st,min}$ **do**
5 | Reverse the cost of the edge: c_{ij} becomes $-c_{ij}$;
6 | Invert the direction of the edge: e_{ij} becomes e_{ji};
7 **end**
8 Find sp in the modified graph between s and t;
9 **if** $sp = \emptyset$ **then**
10 | No set of disjoint paths of size $k+1$ exists, Return;
11 **else**
12 | Reverse node splitting in sp and cancel opposite edges;
13 | Find the set $\mathcal{H}^{k+1}_{st,min}$;
14 **end**

VII. Simulations

In this section, we evaluate the performance of the WMIA, greedy algorithm, and Suurball's algorithm for finding a solution to (4). To model edge probabilities, we adopt the exponential model of interaction [34] given by $p_{ij} = e^{\frac{-\|d_{ij}\|^2}{2\Lambda^2}}$, if $\|d_{ij}\| \leq R$, where $d_{ij} \in \mathbb{R}$ is the distance between robots i and j, $R \in \mathbb{R}_{\geq 0}$ is the interaction radius, and $\Lambda \in \mathbb{R}_{\geq 0}$ is a parameter describing how quickly interaction quality degrades. Data is generated for Monte Carlo (MC) simulations using two different methods. First, we generate 10000 random MRS topologies, denoted by M_1, with a random number of robots and observers satisfying $n \leq 10$, $m \leq n$, and $p = m$ using the method described in [12]. We restrict the number of robots to 10 because the WMIA is not efficiently scalable for MC beyond this point. Second, we adopt the environmental monitoring task, denoted by M_2, from [35] where robots start with a random initial configuration and spread in the environment to measure an environmental process. We fix the number of robots to $n = 7$, the number of observers to $m = 3$, and seek minimal security from $p = 3$ attackers. Then, we run 100 random initial configurations for 100 time steps to generate 10000 random MRS topologies. Due to space limitation, this section we will only include the statistical performance of our algorithms. The algorithms' application to MRS tasks is included in the attached video. The results are summarized in Table I.

TABLE I: Statistics of Monte Carlo Simulations

		WMIA	**Greedy**	**Suurballe**
M1	Avg. Execution time (s)	89.83	0.1868	0.0236
	Solution is Optimal (%)	100	97.0	72.5
	Avg. Percent Error (%)	0	0.58	5.14
	Avg. Adj. Per. Error (%)	0	19.7	19.1
M2	Avg. Execution time (s)	20.00	0.225	0.0155
	Solution is Optimal (%)	100	46.8	14.2
	Avg. Percent Error (%)	0	13.1	29.8
	Avg. Adj. Per. Error (%)	0	24.7	34.8

To accurately represent performance, we did not take into account any topologies with no feasible solutions since these cases are trivial to detect. From the first metric in Table I, notice that the WMIA takes significantly longer to execute making it unsuitable for online updates. One can make the case that the greedy algorithm takes significantly longer as compared to Suurballe's algorithm. However, the greedy approach is still fast enough to justify using it for online updates. Additionally, the second metric shows that the greedy algorithm converges to the optimal solution much more often as compared to Suurballe's algorithm. Notice the significant difference in the percentage of convergence to the optimal solution between M_1 and M_2. Although this may seem like a discrepancy, it is due to how edge probabilities are distributed. In M_1, the edge probabilities are randomly generated, and thus the probability of each edge varies significantly within the topology which makes greedily identifying the best edges an easy task. In M_2, edge probabilities depend on robot positions, which are evenly spread in the environment to maximize the efficiency of monitoring. Since the distance between robots is similar, so are the edge probabilities in the topology. This makes identifying the best edges greedily a harder task. Finally, looking at the third and forth metrics of Table I, we can see that both heuristics perform well with the greedy algorithm outperforming Suurballe. Note that average percentage error takes into account all samples, while average adjusted percentage error only takes into account the samples that did not converge to the optimal solution.

VIII. Conclusions and Future Work

In this paper, we have successfully formulated and solved the combinatorial optimization problem that seeks to minimize the number of edges in an MRS's interaction graph while maximizing the topological metric of security. Both optimal and heuristic approaches were presented each with its own advantages. Finally we showed the validity of our results through Monte Carlo simulations. Directions for future work include optimally augmenting the minimal graph with edges to reach a desired threshold of security. Additionally, one can look at the problem of designing a motion controller that maximizes the metric of security.

REFERENCES

[1] C. E. Shannon, "Communication in the presence of noise," *Proceedings of the IRE*, vol. 37, no. 1, pp. 10–21, 1949.

[2] F. Pasqualetti, F. Dorfler, and F. Bullo, "Control-theoretic methods for cyberphysical security: Geometric principles for optimal cross-layer resilient control systems," *IEEE Control Systems*, vol. 35, no. 1, pp. 110–127, 2015.

[3] R. W. Beard and T. W. McLain, "Multiple uav cooperative search under collision avoidance and limited range communication constraints," in *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475)*, vol. 1. IEEE, 2003, pp. 25–30.

[4] S. Amin, A. A. Cárdenas, and S. Sastry, "Safe and secure networked control systems under denial-of-service attacks." in *HSCC*, vol. 5469. Springer, 2009, pp. 31–45.

[5] L. Xie, Y. Mo, and B. Sinopoli, "False data injection attacks in electricity markets," in *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on*. IEEE, 2010, pp. 226–231.

[6] Y. Mo and B. Sinopoli, "Secure control against replay attacks," in *Communication, Control, and Computing, 2009. Allerton 2009. 47th Annual Allerton Conference on*. IEEE, 2009, pp. 911–918.

[7] R. S. Smith, "A decoupled feedback structure for covertly appropriating networked control systems," *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 90–95, 2011.

[8] S. Weerakkody, X. Liu, S. H. Son, and B. Sinopoli, "A graph-theoretic characterization of perfect attackability for secure design of distributed control systems," *IEEE Transactions on Control of Network Systems*, vol. 4, no. 1, pp. 60–70, 2017.

[9] F. Pasqualetti, F. Dörfler, and F. Bullo, "Attack detection and identification in cyber-physical systems," *IEEE transactions on automatic control*, vol. 58, no. 11, pp. 2715–2729, 2013.

[10] F. Pasqualetti, A. Bicchi, and F. Bullo, "Consensus computation in unreliable networks: A system theoretic approach," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 90–104, 2012.

[11] R. Wehbe and R. K. Williams, "Probabilistic graph security for networked multi-robot systems," in *IEEE International Conference on Robotics and Automation*. IEEE, 2018.

[12] R. Wehbe and R. Williams, "A deep learning approach for probabilistic security in multi-robot teams," *IEEE Robotics and Automation Letters*, 2019.

[13] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 4, no. 3, pp. 382–401, 1982.

[14] H. J. LeBlanc, H. Zhang, X. Koutsoukos, and S. Sundaram, "Resilient asymptotic consensus in robust networks," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 4, pp. 766–781, 2013.

[15] D. Saldana, A. Prorok, S. Sundaram, M. F. Campos, and V. Kumar, "Resilient consensus for time-varying networks of dynamic agents," in *2017 American Control Conference (ACC)*. IEEE, 2017, pp. 252–258.

[16] R.-H. Jan, F.-J. Hwang, and S.-T. Chen, "Topological optimization of a communication network subject to a reliability constraint," *IEEE Transactions on Reliability*, vol. 42, no. 1, pp. 63–70, 1993.

[17] Y. Chopra, B. Sohi, R. Tiwari, and K. Aggarwal, "Nework topology for maximizing the terminal reliability in a computer communication network," *Microelectronics reliability*, vol. 24, no. 5, pp. 911–913, 1984.

[18] K. Watcharasitthiwat and P. Wardkein, "Reliability optimization of topology communication network design using an improved ant colony optimization," *Computers & Electrical Engineering*, vol. 35, no. 5, pp. 730–747, 2009.

[19] S. Kar and J. M. Moura, "Sensor networks with random links: Topology design for distributed consensus," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 3315–3326, 2008.

[20] ——, "Consensus based detection in sensor networks: Topology optimization under practical constraints," *Proc. 1st Intl. Wrkshp. Inform. Theory Sensor Networks*, vol. 13, p. 31, 2007.

[21] S. Zhou, M.-Y. Wu, and W. Shu, "Finding optimal placements for mobile sensors: wireless sensor network topology adjustment," in *Proceedings of the IEEE 6th Circuits and Systems Symposium on Emerging Technologies: Frontiers of Mobile and Wireless Communication (IEEE Cat. No. 04EX710)*, vol. 2. IEEE, 2004, pp. 529–532.

[22] A. Frank and É. Tardos, "An application of submodular flows," *Linear algebra and its applications*, vol. 114, pp. 329–348, 1989.

[23] A. Frank, "Rooted k-connections in digraphs," *Discrete Applied Mathematics*, vol. 157, no. 6, pp. 1242–1254, 2009.

[24] A. Schrijver, *Combinatorial optimization: polyhedra and efficiency*. Springer Science & Business Media, 2003, vol. 24.

[25] J. Liu and R. K. Williams, "Submodular optimization for coupled task allocation and intermittent deployment problems," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3169–3176, 2019.

[26] R. K. Williams, A. Gasparri, and G. Ulivi, "Decentralized matroid optimization for topology constraints in multi-robot allocation problems," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 293–300.

[27] R. J. Wilson, *Introduction to Graph Theory*. New York, NY, USA: John Wiley & Sons, Inc., 1986.

[28] S. B. Akers, "Binary decision diagrams," *IEEE Transactions on computers*, no. 6, pp. 509–516, 1978.

[29] R. Wehbe and R. K. Williams, "Approximate probabilistic security for networked multi-robot systems," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 1997–2003.

[30] M. M. Asadi, H. Mahboubi, A. G. Aghdam, and S. Blouin, "Connectivity measures for random directed graphs with applications to underwater sensor networks," in *Electrical and Computer Engineering (CCECE), 2015 IEEE 28th Canadian Conference on*. IEEE, 2015, pp. 208–212.

[31] E. L. Lawler, "Matroid intersection algorithms," *Mathematical programming*, vol. 9, no. 1, pp. 31–56, 1975.

[32] J. Suurballe, "Disjoint paths in a network," *Networks*, vol. 4, no. 2, pp. 125–145, 1974.

[33] R. Bellman, "On a routing problem," *Quarterly of applied mathematics*, vol. 16, no. 1, pp. 87–90, 1958.

[34] P. Yang, R. A. Freeman, G. J. Gordon, K. M. Lynch, S. S. Srinivasa, and R. Sukthankar, "Decentralized estimation and control of graph connectivity for mobile sensor networks," *Automatica*, vol. 46, no. 2, pp. 390–396, 2010.

[35] R. K. Williams, A. Gasparri, G. Ulivi, and G. S. Sukhatme, "Generalized topology control for nonholonomic teams with discontinuous interactions," *IEEE Transactions on Robotics*, vol. 33, no. 4, pp. 994–1001, 2017.