Robust Monitor Placement for Network Tomography in Dynamic Networks

Ting He*, Liang Ma*, Athanasios Gkelias†, Kin K. Leung†, Ananthram Swami‡, and Don Towsley§
*IBM T. J. Watson Research Center, Yorktown, NY, USA. Email: {the, maliang}@us.ibm.com
†Imperial College, London, UK. Email: {a.gkelias, kin.leung}@imperial.ac.uk
‡Army Research Laboratory, Adelphi, MD, USA. Email: ananthram.swami.civ@mail.mil
§University of Massachusetts, Amherst, MA, USA. Email: towsley@cs.umass.edu

***Abstract*—We consider the problem of placing a minimum number of monitors in a communication network to identify additive link metrics from path metrics under topology changes. The core of our solution is a suite of robust monitor placement algorithms with different performance-complexity tradeoffs that guarantee network identifiability across multiple topologies. In particular, we show that the optimal (i.e., minimum) monitor placement is the solution to a generalization of the hitting set problem, for which we provide a polynomial-time algorithm to construct the input. Although the optimal placement is NP-hard to compute in general, we identify non-trivial special cases that can be solved efficiently. We further demonstrate how the proposed algorithms can be augmented to handle unpredictable topology changes and tradeoffs between monitor cost and adaptation cost. Our evaluations on real topologies verify the effectiveness of the proposed algorithms over an optimal monitor placement algorithm designed for static networks.**

I. Introduction

Network tomography refers to the methodology of inferring internal performance metrics (e.g., link delays/losses) of a network from external metrics measured between nodes with monitoring capabilities (*monitors*). Since its introduction [1], network tomography has attracted significant interest in the research community as a promising alternative to the approach of direct measurement. Traditionally, network monitoring systems rely on diagnostic tools such as *traceroute*, *pathchar* [2], and *Network Characterization Service (NCS)* [3] to directly measure performance metrics of individual links by sending active probes, which suffers a high measurement overhead and requires cooperation (e.g., support of Internet Control Message Protocol (ICMP)) from every internal node. In contrast, tomography-based monitoring only requires cooperation of nodes employed as monitors and can utilize end-to-end performance experienced by data packets to reduce the need of active probes [4].

A major challenge in applying network tomography to network state monitoring is the lack of a unique solution. For example, consider the inference of link metrics that are *additive*, i.e., the combined metric over multiple links is the sum of individual link metrics (e.g., delays, jitters, log of packet delivery ratio). Network tomography infers such link metrics by solving a system of linear equations, where the unknown variables are the link metrics, and each measurement path provides an equation that relates the metrics of traversed links to the end-to-end measurement on this path. From linear algebra, we know that the system has a unique solution if and only if the measurement paths span the entire link space, i.e., the number of *linearly independent* paths equals the number of links. However, past experience shows that without careful design, it is frequently impossible to uniquely determine all link metrics from path measurements [5], [6], [7].

This problem, known as the *identifiability* problem, has been recognized in the literature with several solutions proposed to place monitors so as to be able to identify all link metrics [8], [9], [10], under the assumption that the network topology is fixed. While the fixed-topology assumption is valid in wired networks, applying network tomography in wireless networks faces the additional challenge that the network topology may vary dynamically at runtime due to factors such as node mobility and channel variation. While a straightforward solution is to handle the changes *reactively* by repeatedly applying static-network solutions to design a new monitor placement after each topology change, such a solution can lead to frequent reconfigurations and instability in the monitoring system. To maintain seamless monitoring of link performance in such dynamic networks, it is desirable to have a monitor placement strategy that can handle topology changes *proactively*.

In this paper, we aim at developing monitor placement algorithms that are *robust* against topology changes. In many networks, it is possible to predict network topologies under changes with certain accuracy. For example, we can predict topology changes based on models of node mobility and communication range [11], [12], [13], [14], patterns of link failures [15], or directly from frequently occurring topologies in the past. Given such prediction capabilities, we are interested in two closely-related problems: (i) At the network planning phase, how does one select locations for monitors such that the monitors can ensure identifiability across all predicted topology changes? (ii) At runtime, how does one adapt the monitor placement so that we can still achieve identifiability under unpredictable changes? In both problems, we wish to use as few monitors as possible to minimize cost.

A. Related Work

Our work belongs to a family of monitor placement algorithms for uniquely identifying link metrics from end-to-end

Research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

measurements collected at monitors. Existing solutions can be categorized based on how measurements are performed: (1) traceroute-like round-trip probing, (2) end-to-end probing.

In category (1), each monitor (aka beacon) independently computes metrics of a subset of links by sending round-trip probes to all possible destinations along given paths. Algorithms based on set covering are proposed to select the minimum set of monitors such that their monitored link sets cover all the links [16], [17]. The problem is proved to be NP-hard, and the NP-hardness persists even if monitors can control the first hop of probes [18]. In category (2), link metrics are inferred from measurements of all monitors using network tomography. Under the assumption that monitors can measure arbitrary cycles or paths (possibly) containing cycles, [8] derives the first necessary and sufficient condition on the network topology for identifying additive link metrics from end-to-end measurements. Based on this condition, an efficient monitor placement algorithm is developed. The condition is later modified by [9] to incorporate an additional constraint that measurement paths must be cycle-free, which allows the use of data packets for measurements without incurring forwarding loops. Based on the modified condition, a linear-complexity algorithm is developed to identify all link metrics using the minimum number of monitors. Interestingly, while the optimal monitor placement is NP-hard to compute under uncontrollable routing (category (1)), it can be computed efficiently under controllable routing, with or without cycles (categories (2)).

Existing work mostly assumes static network topology and routing with a few exceptions in category (1). Specifically, [16] and [17] propose variations of the set-covering-based solution to select monitors that are guaranteed to cover all active links under a limited number of link failures or route changes. For network tomography (category (2)), the problem is much more complicated as the identifiability of a given link is determined by measurements between *all* monitors and can no longer be attributed to a single monitor. We study the problem of monitor placement under the measurement model in [9] but consider arbitrary topology changes. Although it is possible to apply the algorithm in [9] to every possible topology and maintain identifiability by placing a monitor on every selected node, such a solution can require an unnecessarily large number of monitors. In this work, we aim at reducing the number of monitors by jointly considering the required monitor locations under different topologies.

A key enabler in robust monitor placement is knowledge of potential topologies after changes. Modeling and prediction of topology changes have been studied in the context of wireless ad-hoc and vehicular networks, where techniques including adaptive filtering and fluid dynamic modeling have been proposed to predict link changes [11], [12], [13], [14]. In this work, we assume the existence of such a topology predictor and focus on developing robust monitor placement algorithms based on the predicted topologies.

B. Summary of Contributions

We study robust monitor placement for inferring additive link metrics from end-to-end measurements along cycle-free paths under dynamic topology changes. Our contributions are:

1) We develop robust monitor placement algorithms that place monitors to *simultaneously* achieve identifiability for a given set of topologies, including: (i) a one-shot placement algorithm that applies an existing algorithm for static networks to an aggregate topology, (ii) an incremental placement algorithm that sequentially places monitors in each topology, and (iii) an optimal placement algorithm that jointly considers monitor requirements of different topologies by casting the problem as a generalized hitting set problem. All these algorithms guarantee identifiability with different tradeoffs between number of monitors and complexity. We also provide an algorithm to remove unnecessary monitors in any robust placement by solving the dual of the joint placement problem.

2) Although the robust monitor placement problem is NP-hard in general (due to the hardness of the hitting set problem), we identify several cases where the problem can be solved efficiently. In particular, one case corresponds to static networks with fixed topologies, explaining why the problem is solvable (by [9]) in this case.

3) Using the above solution as a building block, we present several augmentations to address practical issues including topology selection, unpredictable changes, and tradeoff between offline placement and online adaptation.

4) We evaluate the proposed solutions on realistic dynamic topologies driven by node mobility traces. Besides verifying the performance-complexity tradeoffs of various algorithms, our results show that it is possible for a small number of monitors ($10-30\%$) to maintain identifiability across hundreds of topology changes for dense networks, and our monitor placement is highly robust against error in topology prediction.

The rest of the paper is organized as follows. Section II formulates the problem of robust monitor placement. Section III reviews existing results in static networks, based on which Section IV presents a set of algorithms for robust monitor placement in dynamic networks. Since the general problem is NP-hard, Section V identifies special cases that can be solved efficiently. Section VI discusses practical challenges and solutions. Section VII evaluates the proposed solutions via simulations. Section VIII concludes the paper.

II. Problem Formulation

A. Network Models

Consider a fixed set of nodes V which form a (possibly partitioned) network with time-varying topologies. We assume that the network topology can be predicted (up to a certain accuracy) at the time the network is planned using existing topology prediction models such as [11], [12], [13], [14]. Based on the prediction, we identify a set of topologies of interest, represented by undirected graphs $\{\mathcal{G}_t : t = 1, \ldots, T\}$, where $\mathcal{G}_t = (V, L_t)$ is a graph representing the t-th topology. We note that these topologies do not need to occur sequentially in time and do not need to be exclusive. We first focus on monitor placement for given topologies and defer the selection of topologies to Section VI-A. Although we focus on link changes in this paper, node changes can also be handled by modeling them as special link changes; see Section VI-D.

B. Objective of Network Tomography

Given a topology $\mathcal{G} = (V, L)$ (subscript omitted), network tomography aims at inferring the performance metrics of individual links in L from end-to-end metrics along paths that are monitored. In particular, we are interested in inferring *additive metrics* where the path metric equals summation of corresponding link metrics; examples of such metrics include delay, jitter, and log delivery rate (under the assumption of independent losses across links). Let $\mathbf{w} = (w_1, \ldots, w_{|L|})^T$ be the column vector of link metrics, and $\mathbf{c} = (c_1, \ldots, c_{|P|})^T$ be the column vector of path metrics for a set of monitored paths P. These metrics are related by the following linear system:

$$\mathbf{R}\mathbf{w} = \mathbf{c}, \tag{1}$$

where $\mathbf{R} = (R_{ij})$ is a $|P| \times |L|$ *measurement matrix* with $R_{ij} \in \{0, 1\}$ denoting whether link j is present on path i. The goal of network tomography is to invert this linear system to solve for $\mathbf{w}$ given $\mathbf{R}$ and $\mathbf{c}$. A basic requirement of network tomography is therefore the invertibility of this system, captured as follows.

Definition 1. A network $\mathcal{G}$ is *identifiable* if all its link metrics can be uniquely determined from path metrics, i.e., if the measurement matrix $\mathbf{R}$ in (1) has full column rank.

Clearly, whether $\mathcal{G}$ is identifiable depends on the set of monitored paths. We assume that we can only monitor paths that start/end at certain nodes employed as *monitors*. We further assume that monitors can control the routing of measurement packets as long as the path starts and ends at different monitors and does not contain repeated nodes, i.e., we can measure any *simple path* between monitors. Under this assumption, whether $\mathcal{G}$ is identifiable or not is completely determined by the placement of monitors. If $\mathcal{G}$ is identifiable under a given monitor placement, we say that this placement *identifies* $\mathcal{G}$. In the degenerate network of only one isolated node, we assume the network is identifiable if and only if this node is a monitor.

C. Objective of Monitor Placement

Our main objective in monitor placement is to select a smallest subset of nodes as monitors during network planning such that we can guarantee identifiability for all the topologies of interest. We will discuss later (Section VI) how the planned monitor placement can be combined with on-demand monitor placement to maintain identifiability at reduced adaptation cost.

III. Identifiability and Monitor Placement for Static Networks

We start by reviewing existing results from [9], including the condition for a monitor placement to achieve network identifiability and an efficient algorithm to satisfy this condition by placing the minimum number of monitors, all designed for a static network with a fixed topology.

A. Identifiability Condition

Under the assumption of controllable routing, the definition of identifiability (Definition 1) does not directly allow efficient testing of whether a given monitor placement achieves identifiability, as there can be exponentially many monitored paths.

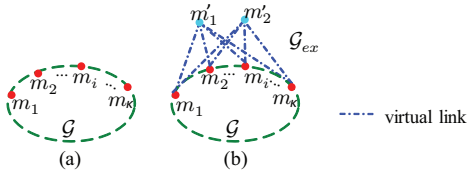


Fig. 1. (a) $\mathcal{G}$ with κ ($\kappa \geq 3$) monitors; (b) $\mathcal{G}_{ex}$ with two virtual monitors.

Fortunately, existing work [9] has established an equivalent condition in terms of the topology and the placement of monitors that can be tested efficiently. This condition is based on an *extended graph* $\mathcal{G}_{ex}$ constructed as follows: given a network $\mathcal{G}$ with κ ($\kappa \geq 3$) monitors[1], $\mathcal{G}_{ex}$ is obtained by adding two *virtual monitors* m'_1 and m'_2, and 2κ *virtual links* between each pair of virtual-actual monitors, as illustrated in Fig. 1. The identifiability condition is then expressed as a condition on the connectivity of $\mathcal{G}_{ex}$ as follows.

Theorem III.1. [9] Using κ ($\kappa \geq 3$) monitors, $\mathcal{G}$ is identifiable *if and only if* the associated extended graph $\mathcal{G}_{ex}$ is 3-vertex-connected, i.e., it remains connected after removing any two nodes.

B. Minimum Monitor Placement for Static Networks

The above result has a direct application in monitor placement. Given the identifiability condition in Theorem III.1, the objective of monitor placement is changed to placing the minimum number of monitors to ensure that the extended graph $\mathcal{G}_{ex}$ is 3-vertex-connected. The solution is an algorithm, called *Minimum Monitor Placement (MMP)*, that decomposes $\mathcal{G}$ into subgraphs with certain properties (triconnected components) and sequentially places monitors in each subgraph to satisfy the condition in Theorem III.1 [9]. We briefly review MMP for completeness.

Definition 2. A *k-connected component* of $\mathcal{G}$ is a maximal sub-graph of $\mathcal{G}$ that is either (i) k-vertex-connected, or (ii) a complete graph with up to k vertices. The case of $k = 2$ is also called a *biconnected component*, and $k = 3$ a *triconnected component*.

A biconnected component is a sub-graph connected to the rest of the graph by cut-vertices, and a triconnected component (within a biconnected component) is a sub-graph connected to the rest by 2-vertex cuts[2]. We refer to common vertices between a bi/triconnected component and its neighboring components as *separation vertices*, which are cut-vertices for a biconnected component, and cut-vertices as well as vertices in 2-vertex cuts for a triconnected component.

Algorithm 1 summarizes the steps of MMP. Given a sub-graph $\mathcal{D}$, let $V(\mathcal{D})$ denote all the nodes in $\mathcal{D}$, $S_{\mathcal{D}}$ denote the separation vertices in $\mathcal{D}$, and $M_{\mathcal{D}}$ denote the monitors placed at internal nodes (i.e., non-separation vertices) of $\mathcal{D}$. MMP processes each connected component separately (the original algorithm in [9] assumes connected graph). For each connected component, MMP first places monitors at nodes with degree 1 or 2 (line 2), where the degree of node v refers to the number

[1]It has been shown [9] that a network with more than one link needs at least three monitors to identify all link metrics.

[2]Since not all subgraphs separated by 2-vertex cuts form triconnected components according to Definition 2, an iterative procedure is proposed in [9] to fix this issue by adding virtual links between nodes in 2-vertex cuts.

Algorithm 1: Minimum Monitor Placement (MMP) [9]

input : Network topology $\mathcal{G}$
output: A subset of nodes in $\mathcal{G}$ as monitors
1 **foreach** *connected component* $\mathcal{C}_k$ *of* $\mathcal{G}$ **do**
2 choose all the nodes with degree less than 3 as monitors;
3 partition $\mathcal{C}_k$ into biconnected components $\mathcal{B}_1, \mathcal{B}_2, \ldots$;
4 **foreach** *biconnected component* $\mathcal{B}_i$ *with at least* 3 *nodes* **do**
5 partition $\mathcal{B}_i$ into triconnected components $\mathcal{T}_1, \mathcal{T}_2, \ldots$;
6 **foreach** *triconnected component* $\mathcal{T}_j$ **do**
7 compute $S_{\mathcal{T}_j}$ and $M_{\mathcal{T}_j}$;
8 **if** $|S_{\mathcal{T}_j}| + |M_{\mathcal{T}_j}| < 3$ **then**
9 randomly place $3 - |S_{\mathcal{T}_j}| - |M_{\mathcal{T}_j}|$ monitors at nodes in $V(\mathcal{T}_j) \setminus (S_{\mathcal{T}_j} \cup M_{\mathcal{T}_j})$;
10 **end**
11 **end**
12 compute $S_{\mathcal{B}_i}$ and $M_{\mathcal{B}_i}$;
13 **if** $|S_{\mathcal{B}_i}| + |M_{\mathcal{B}_i}| < 3$ **then**
14 randomly place $3 - |S_{\mathcal{B}_i}| - |M_{\mathcal{B}_i}|$ monitors at nodes in $V(\mathcal{B}_i) \setminus (S_{\mathcal{B}_i} \cup M_{\mathcal{B}_i})$;
15 **end**
16 **end**
17 compute $M_{\mathcal{C}_k}$;
18 **if** $|M_{\mathcal{C}_k}| < 3$ **then**
19 randomly place $\min(3, |V(\mathcal{C}_k)|) - |M_{\mathcal{C}_k}|$ monitors at nodes in $V(\mathcal{C}_k) \setminus M_{\mathcal{C}_k}$;
20 **end**
21 **end**

of neighbors of v. It then decomposes the component into biconnected and then triconnected components (lines 3 and 5), which can be computed in linear time using fast graph algorithms in [19] and [20]. Based on the decomposition, MMP selects monitors such that each bi/triconnected component with at least three nodes has at least three nodes that are separation vertices or monitors (lines 9 and 14). Finally, it places additional monitors if necessary to ensure that each connected component with at least three nodes has at least three monitors (line 19)[3].

MMP is computationally efficient with a complexity of $O(|V| + |L|)$ for $\mathcal{G} = (V, L)$ [9]. It is also provably optimal in the following sense.

Theorem III.2. [9] Given an arbitrary network topology $\mathcal{G}$, MMP places the minimum number of monitors to identify $\mathcal{G}$.

Key observations: MMP places two types of monitors:

- deterministically placed monitors: nodes with degree one or two have to be monitors (line 2), as otherwise their neighboring links are not measurable by simple paths;
- randomly placed monitors: it suffices to ensure at least three nodes that are either separation vertices or monitors in each bi/tri/1-connected components, but the exact monitor locations can be arbitrary (lines 9, 14, 19).

We will leverage these observations in deriving monitor placement algorithms for dynamic networks.

IV. Robust Monitor Placement for Dynamic Networks

In a dynamic network, the topology may vary over time. Let $\{\mathcal{G}_t : t = 1, \ldots, T\}$ denote the set of topologies of interest.

[3]In case of a component with only one or two nodes, all nodes are selected as monitors.

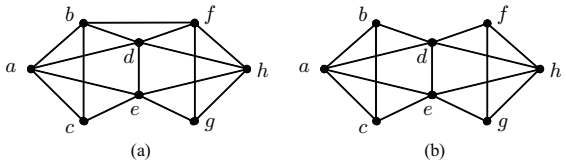


Fig. 2. Monitor placement for two topologies (the optimal monitor placement is $\{c, f, h\}$): (a) $\mathcal{G}_1$ (3-vertex-connected); (b) $\mathcal{G}_2$ (2-vertex-connected, with a 2-vertex cut $\{d, e\}$).

Based on the identifiability condition in Theorem III.1, the problem of robust monitor placement can be cast as follows: select a minimum set of nodes $M \subseteq V$ as monitors such that for each $t = 1, \ldots, T$, the extended graph $\mathcal{G}_{t,ex}$ constructed from topology $\mathcal{G}_t$ and monitors M is 3-vertex-connected.

Given a placement algorithm like MMP that guarantees identifiability for a single topology, one can guarantee identifiability for all the T topologies by applying MMP to compute a monitor placement M_t for each $\mathcal{G}_t$, and then taking the union $M = \bigcup_{t=1}^{T} M_t$ as the overall monitor placement. Such a solution, however, may select many redundant monitors as it fails to consider the common need of monitors across topologies. For example, Fig. 2 shows a sample network with two possible topologies $\mathcal{G}_1$ and $\mathcal{G}_2$. Since $\mathcal{G}_1$ in Fig. 2 (a) is 3-vertex-connected, selecting any three nodes as monitors can achieve identifiability according to Theorem III.1. In contrast, for $\mathcal{G}_2$ in Fig. 2 (b), node sets $\{a, b, c\}$ and $\{f, g, h\}$ must each have at least one monitor according to MMP, although the total number of required monitors is still three. Hence, applying MMP separately to $\mathcal{G}_1$ and $\mathcal{G}_2$ may generate the following monitor placements: $M_1 = \{a, b, c\}$ for $\mathcal{G}_1$, and $M_2 = \{c, f, h\}$ for $\mathcal{G}_2$, i.e., $|M| = |\bigcup_{t=1}^{2} M_t| = 5$. Nevertheless, M_2 itself can identify both $\mathcal{G}_1$ and $\mathcal{G}_2$, i.e., two redundant monitors are selected by MMP.

This example illustrates the need to jointly consider all topologies during monitor placement. In this regard, we outline a set of solutions that take such information into account with various tradeoffs between performance (measured by number of monitors) and complexity.

A. One-Shot Placement

Intuitively, placing monitors to identify multiple topologies is more complex than placing monitors to identify a single topology. Interestingly, we show that to identify multiple topologies, it suffices for the monitor placement to identify a single auxiliary topology referred to as the *base graph*. The base graph, denoted by $\mathcal{G}_b$, for a set of topologies $\{\mathcal{G}_t = (V, L_t) : t = 1, \ldots, T\}$ is defined as $\mathcal{G}_b := (V, \bigcap_{t=1}^{T} L_t)$, i.e., it is the maximum common subgraph of all topologies.

By definition, $\mathcal{G}_b$ is less connected than any $\mathcal{G}_t$. Meanwhile, we see from the identifiability condition in Theorem III.1 that if a network $\mathcal{G}$ is identifiable under a monitor placement M, then it remains identifiable under M after adding links (because adding links to a 3-vertex-connected graph $\mathcal{G}_{ex}$ results in a 3-vertex-connected graph). This observation immediately leads to the following result.

Lemma IV.1. If a monitor placement M achieves identifiability for the base graph $\mathcal{G}_b$ of topologies $\{\mathcal{G}_t = (V, L_t) : t = 1, \ldots, T\}$, then it achieves identifiability for each individual topology $\mathcal{G}_t$ $(t = 1, \ldots, T)$.

This result motivates a one-shot placement algorithm: first, compute the base graph $\mathcal{G}_b$ by identifying common links in all topologies, and then apply MMP to compute a monitor placement that identifies $\mathcal{G}_b$, and hence achieves identifiability for all topologies $\mathcal{G}_t$ ($t = 1, \ldots, T$).

Generally, such one-shot placement uses more than the minimum number of monitors. For example, if all $\mathcal{G}_t$ ($t = 1, \ldots, T$) are 3-vertex-connected, then by Theorem III.1, an arbitrary placement of three monitors already achieves identifiability for all $\mathcal{G}_t$; however, since the base graph may not be 3-vertex-connected, MMP may place more than three monitors in $\mathcal{G}_b$, leading to a suboptimal placement.

Complexity: The base graph can be computed in $O(T \min_t |L_t|)$ time (assuming the existence of a link in a topology can be checked in constant time), and applying MMP to the base graph takes $O(|V| + \min_t |L_t|)$ time. The overall complexity of the one-shot placement algorithm is therefore $O(|V| + T \min_t |L_t|)$.

B. Incremental Placement

A natural alternative to one-shot placement is to apply MMP to each topology. Instead of applying the original MMP, however, we use a variation that takes into account existing monitors to reduce the number of additional monitors.

Specifically, in addition to a topology $\mathcal{G}$, we modify MMP to take an additional input of M_0, the existing set of monitors. Redefine $M_{\mathcal{D}}$ to denote the internal (i.e., non-separation-vertex) monitors in a subgraph $\mathcal{D}$ among both existing and newly placed monitors. Then the modified algorithm, referred to as *incremental MMP (IMMP)*, follows the same steps as in Algorithm 1, except that it only returns the set of newly placed monitors M_a. We have the following property of IMMP.

Lemma IV.2. Given a topology $\mathcal{G}$ and existing monitors M_0, IMMP places the minimum number of additional monitors M_a such that $M_0 \cup M_a$ identifies $\mathcal{G}$.

Proof. First, $M_0 \cup M_a$ contains a (possible) monitor placement by MMP as a subset, and thus it identifies $\mathcal{G}$ since the placement by MMP identifies $\mathcal{G}$ by Theorem III.2. Moreover, for any M'_a with $|M'_a| < |M_a|$, the placement $M_0 \cup M'_a$ must violate the condition in line 8, 13, or 18 for a subgraph $\mathcal{D}$ of $\mathcal{G}$ ($\mathcal{D}$ can be a tri/bi/1-connected component). One of the following cases will occur: (i) $\mathcal{D}$ is an isolated non-monitor, (ii) $\mathcal{D}$ is a two-node graph connected by one link, where at most one node is a monitor, or (iii) $\mathcal{D}$ contains at least three nodes, out of which at most two are separation vertices or monitors. In cases (i–ii), $\mathcal{D}$ is not identifiable by definition; in case (iii), $\mathcal{G}$ does not satisfy the condition in Theorem III.1 as one can remove the separation vertices/monitors in $\mathcal{D}$ (at most two of them) to disconnect $\mathcal{D}$ from the rest of $\mathcal{G}_{ex}$. Thus, $M_0 \cup M_a$ is the minimum placement that identifies $\mathcal{G}$. □

The incremental placement algorithm based on IMMP works as follows: for each $\mathcal{G}_t = \mathcal{G}_1, \ldots, \mathcal{G}_T$ ($M_0 = \emptyset$),

1) $M_{a,t} \leftarrow$IMMP($\mathcal{G}_t$, M_{t-1});
2) $M_t \leftarrow M_{t-1} \cup M_{a,t}$.

Then M_T is guaranteed to achieve identifiability for all $\mathcal{G}_1, \ldots, \mathcal{G}_T$.

Complexity: Since IMMP has the same complexity as MMP, i.e., $O(|V| + |L_t|)$ for each $\mathcal{G}_t$, the overall complexity of the incremental placement algorithm is $O(T|V| + \sum_{t=1}^{T} |L_t|)$.

C. Joint Placement

Despite being locally optimal (Lemma IV.2), the overall placement generated by IMMP is suboptimal in general. Such suboptimality is caused by the lack of a *joint* consideration of monitor requirements of all topologies so as to satisfy them with the minimum number of monitors. Joint consideration of all monitor requirements is highly nontrivial, as each topology can have exponentially many, equally optimal monitor placements, corresponding to all combinations of possible outcomes of lines 9, 14, and 19 of Algorithm 1. The brute-force strategy of enumerating all possible placements that are optimal in individual topologies to find a minimum union is clearly inefficient. Our idea in addressing this issue is to decouple the problem into two stages: (i) constraint characterization and (ii) monitor selection.

1) Constraints on Monitor Placement: Our key insight is that all possible placements generated by MMP can be succinctly encoded into a set of constraints. Specifically, instead of randomly picking one placement as in MMP, we record the constraints on monitor placement in the form of set-integer pairs $\mathcal{F} = \{(S_i,\ k_i) :\ i = 1, 2, \ldots\}$, where each $S_i \subseteq V$ is a set of candidate monitors, and k_i is the minimum number of monitors selected from this set. For example, if MMP randomly places a monitor at any node in set S, we can represent all possible placements by a constraint $(S,\ 1)$. Given a topology $\mathcal{G}$, we can compute these constraints explicitly using an algorithm called *Feasible Monitor Placement (FMP)*. As shown in Algorithm 2, FMP follows a similar procedure as MMP, except that instead of selecting specific nodes as monitors, FMP records the constraints on where monitors should be placed such that there is one constraint for each node that must be a monitor (line 3) and each component that requires at least one monitor (lines 10, 14, 17). Here $S_{\mathcal{D}}$ again denotes the set of separation vertices in subgraph $\mathcal{D}$. We can show that the resulting constraints are both sufficient and necessary for identifying $\mathcal{G}$.

Lemma IV.3. The constraints computed by FMP for an input topology $\mathcal{G}$ are necessary and sufficient for a monitor placement to identify $\mathcal{G}$, i.e., any monitor placement $M \subseteq V$ identifies $\mathcal{G}$ if and only if M satisfies $\mathcal{F}$, i.e., $|M \cap S_i| \geq k_i$ for all $(S_i,\ k_i) \in \mathcal{F}$.

Proof. If M satisfies $\mathcal{F}$, then there exists a possible placement by MMP that is a subset of M. Hence, M must identify $\mathcal{G}$ as MMP is guaranteed to achieve identifiability by Theorem III.2. If M does not satisfy $\mathcal{F}$, then by arguments similar to the proof of Lemma IV.2, $\mathcal{G}_{ex}$ does not satisfy Theorem III.1 and thus M cannot identify $\mathcal{G}$. □

Discussion: It is possible that not all constraints are needed, e.g., if the total number of monitors required by triconnected components within their parent biconnected component exceeds the number of monitors required by the biconnected component, then we do not need a separate constraint for this biconnected component. We can avoid redundant constraints

Algorithm 2: Feasible Monitor Placement (FMP)

input : Network topology $\mathcal{G}$
output: Monitor placement constraints $\mathcal{F}$
foreach *connected component* $\mathcal{C}_k$ *of* $\mathcal{G}$ **do**
  foreach *node* v *with degree less than* 3 **do**
    add $(\{v\},\ 1)$ to $\mathcal{F}$;
  end
  partition $\mathcal{C}_k$ into biconnected components $\mathcal{B}_1, \mathcal{B}_2, \ldots$;
  foreach *biconnected component* $\mathcal{B}_i$ *with at least* 3 *nodes* **do**
    partition $\mathcal{B}_i$ into triconnected components $\mathcal{T}_1, \mathcal{T}_2, \ldots$;
    foreach *triconnected component* $\mathcal{T}_j$ **do**
      if $|S_{\mathcal{T}_j}| < 3$ **then**
        add $(V(\mathcal{T}_j) \setminus S_{\mathcal{T}_j},\ 3 - |S_{\mathcal{T}_j}|)$ to $\mathcal{F}$;
      end
    end
    if $|S_{\mathcal{B}_i}| < 3$ **then**
      add $(V(\mathcal{B}_i) \setminus S_{\mathcal{B}_i},\ 3 - |S_{\mathcal{B}_i}|)$ to $\mathcal{F}$;
    end
  end
  add $(V(\mathcal{C}_k),\ \min(3, |V(\mathcal{C}_k)|))$ to $\mathcal{F}$;
end

by counting the number of monitors for each bi/tri/1-connected component due to existing constraints (by mimicking MMP) and adding a new constraint to $\mathcal{F}$ only if the current number of monitors is not sufficient.

2) Constrained Monitor Selection: After obtaining the constraints $\mathcal{F}$ by applying FMP to each topology $\mathcal{G}_t$ $(t = 1, \ldots, T)$, we convert the problem to one of selecting a minimum subset $M \subseteq V$ such that $|M \cap S_i| \geq k_i$ for all $(S_i,\ k_i) \in \mathcal{F}$. We refer to this problem as the *minimum hitting set problem (min-HSP)* with input $(V,\ \mathcal{F})$, as it contains the classic hitting set problem (HSP) as a special case (when $k_i \equiv 1$). Because the constraints computed by FMP are necessary and sufficient for achieving identifiability (Lemma IV.3), we can obtain an optimal monitor placement by solving the corresponding min-HSP optimally, stated as follows.

Theorem IV.4. Let $\mathcal{F}$ be the overall set of constraints computed by applying FMP to each topology $\mathcal{G}_t$ $(t = 1, \ldots, T)$. Then the optimal solution to min-HSP$(V, \mathcal{F})$ yields an optimal (i.e., minimum) monitor placement for identifying $\mathcal{G}_1, \ldots, \mathcal{G}_T$.

Proof. First, the monitor placement computed by min-HSP$(V,\ \mathcal{F})$ satisfies all the constraints in $\mathcal{F}$ and thus identifies all $\mathcal{G}_1, \ldots, \mathcal{G}_T$ since the constraints in $\mathcal{F}$ are sufficient (Lemma IV.3). Moreover, by definition, any placement M involving fewer monitors than min-HSP$(V,\ \mathcal{F})$ must violate at least one constraint in $\mathcal{F}$. Suppose the violated constraint is generated by $\mathcal{G}_t$. Then M cannot identify $\mathcal{G}_t$ since the constraints are necessary (Lemma IV.3). □

The challenge is that min-HSP is NP-hard since HSP is HP-hard. In fact, the inapproximability result for HSP implies that no polynomial-time algorithm can guarantee an approximation ratio smaller than $(1 - o(1)) \log |V|$ [21]. Although given constraints $\mathcal{F}$, the problem of constrained monitor selection is a special case of min-HSP, we show that it is still NP-hard by a reduction from (general) HSP.

Theorem IV.5. The optimal monitor placement for arbitrary topologies $\mathcal{G}_t$ $(t = 1, \ldots, T)$ is NP-hard.

Proof. We prove hardness of the optimal monitor placement problem by reducing HSP to it as follows. Given an HSP with input $(U,\ \mathcal{E})$, where $\mathcal{E} = \{S_1, \ldots, S_N\} \subseteq 2^U$ is a collection of subsets of a universe U such that at least one item needs to be selected from each S_i. The goal of HSP is to satisfy this requirement by selecting a minimum subset of U. We introduce five dummy items $v_1, \ldots, v_5$, such that none of them is in U. For each $S_i \in \mathcal{E}$, we construct a topology $\mathcal{G}_i$ consisting of three cliques (i.e., complete graphs): a clique formed by items in $S_i \cup \{v_1, v_2\}$, a clique formed by items in $(U \setminus S_i) \cup \{v_1, \ldots, v_4\}$, and a clique (i.e., triangle) formed by items in $\{v_3, v_4, v_5\}$ (each item is represented by a node). It can be verified that $\mathcal{G}_i$ is a biconnected graph with three triconnected components separated by 2-vertex cuts $\{v_1, v_2\}$ and $\{v_3, v_4\}$, for which the monitor placement constraints generated by FMP are $(S_i,\ 1)$ and $(\{v_5\},\ 1)$. By Theorem IV.4, the optimal monitor placement for topologies $\{\mathcal{G}_1, \ldots, \mathcal{G}_N\}$ constructed as above is the solution to min-HSP$(U \cup \{v_1, \ldots, v_5\},\ \mathcal{F})$ for $\mathcal{F} = \{(S_1,\ 1), \ldots, (S_N,\ 1),\ (\{v_5\},\ 1)\}$. Denote the optimal placement for $\{\mathcal{G}_1, \ldots, \mathcal{G}_N\}$ by M^*. Then $M^* \setminus \{v_5\}$ is an optimal solution to HSP$(U,\ \mathcal{E})$ (note that M^* must contain v_5). Thus, the optimal monitor placement problem is at least as hard as HSP, which is NP-hard. □

In our problem, we can speed up computation by first determining nodes that must be monitors and excluding them from subsequent selection. Specifically, let V' denote all the nodes with degree less than three in *at least one* of $\mathcal{G}_1, \ldots, \mathcal{G}_T$. We know that nodes in V' have to be monitors to achieve identifiability, captured by degenerate constraints $\mathcal{F}' = \{(\{v\},\ 1) :\ v \in V'\}$. It thus suffices to focus on selecting the remaining monitors by solving a reduced min-HSP on input $(V \setminus V',\ \mathcal{F} \setminus \mathcal{F}')$.

Moreover, the greedy heuristic, known to achieve the optimal approximation ratio of $(1 + \log |V|)$ for HSP [22], can be applied to solve min-HSP efficiently[4]. For our problem, the greedy heuristic works as follows: while there are unsatisfied constraints, select the monitor that helps in satisfying the maximum number of unsatisfied constraints, i.e., given current monitors M, select v as the next monitor such that v is in the maximum number of sets among $\{S_i :\ (S_i,\ k_i) \in \mathcal{F},\ |S_i \cap M| < k_i\}$.

Complexity: Together, FMP and the greedy heuristic for min-HSP provide a joint monitor placement algorithm. FMP has the same complexity as MMP, which is $O(|V| + |L_t|)$ for each $\mathcal{G}_t$ $(t = 1, \ldots, T)$ [9]. The greedy heuristic has complexity $O(T|V|^2)$, as there are $O(T|V|)$ constraints and an $O(|V|)$-complexity update upon satisfying each constraint (to compute the number of unsatisfied constraints each candidate monitor is involved in). Thus, the overall complexity is $O(T|V|^2)$.

D. Refinement of Placement

Due to the hardness of the optimal solution, the computed monitor placement generally contains more than the minimum number of monitors. A natural question is therefore whether we can refine this placement by removing a subset of monitors without losing identifiability. As shown below, the problem of

[4]Note that the log-approximation may not hold for min-HSP.

Algorithm 3: Redundant Monitor Discovery (RMD)

input : Network topology $\mathcal{G}$, monitor placement M that identifies $\mathcal{G}$
output: Monitor removal constraints $\mathcal{R}$

1 **foreach** *connected component* $\mathcal{C}_k$ *of* $\mathcal{G}$ **do**
2 **foreach** *node* v *with degree less than* 3 **do**
3 add $(\{v\},\ 0)$ to $\mathcal{R}$;
4 **end**
5 partition $\mathcal{C}_k$ into biconnected components $\mathcal{B}_1, \mathcal{B}_2, \ldots$;
6 **foreach** *biconnected component* $\mathcal{B}_i$ *with at least* 3 *nodes* **do**
7 partition $\mathcal{B}_i$ into triconnected components $\mathcal{T}_1, \mathcal{T}_2, \ldots$;
8 **foreach** *triconnected component* $\mathcal{T}_j$ **do**
9 **if** $|S_{\mathcal{T}_j}| < 3$ **then**
10 add $(M_{\mathcal{T}_j},\ |M_{\mathcal{T}_j}| - 3 + |S_{\mathcal{T}_j}|)$ to $\mathcal{R}$;
11 **end**
12 **end**
13 **if** $|S_{\mathcal{B}_i}| < 3$ **then**
14 add $(M_{\mathcal{B}_i},\ |M_{\mathcal{B}_i}| - 3 + |S_{\mathcal{B}_i}|)$ to $\mathcal{R}$;
15 **end**
16 **end**
17 add $(M_{\mathcal{C}_k},\ |M_{\mathcal{C}_k}| - \min(3, |V(\mathcal{C}_k)|))$ to $\mathcal{R}$;
18 **end**

(redundant) monitor removal can also be decoupled into two stages.

1) Constraints on Monitor Removal: The problem of characterizing constraints on monitor removal is similar to that of characterizing constraints on monitor selection (Section IV-C1). Since the constraints computed by FMP are necessary and sufficient, we can follow a similar procedure to compute the complement constraints for removing monitors.

The algorithm, referred to as *Redundant Monitor Discovery (RMD)*, is summarized in Algorithm 3, where $M_{\mathcal{D}}$ denotes the set of internal monitors in a subgraph $\mathcal{D}$; note that in contrast to MMP where internal monitors vary during monitor placement, here $M_{\mathcal{D}}$ is fixed for a given $\mathcal{D}$, as it is based on a given monitor placement. RMD follows similar steps as FMP, except that it computes constraints on monitor removal (lines 3, 10, 14, 17). Each constraint is also represented as a set-integer pair (S_i, k_i), but now it means that no more than k_i monitors from S_i can be removed to maintain identifiability.

It is easy to verify that starting from a monitor placement that identifies $\mathcal{G}$, satisfying the constraints computed by RMD in removing monitors is necessary and sufficient for the remaining monitors to satisfy the constraints computed by FMP. This duality in combination with Lemma IV.3 immediately yields the following result.

Lemma IV.6. Given a monitor placement M that identifies $\mathcal{G}$, RMD computes the necessary and sufficient constraints for monitor removal, i.e., for any subset of monitors $M' \subset M$, $M \setminus M'$ identifies $\mathcal{G}$ if and only if M' satisfies $\mathcal{R}$, i.e., $|M' \cap S_i| \leq k_i$ for all $(S_i,\ k_i) \in \mathcal{R}$.

Discussion: As in FMP, it is also possible that not all constraints computed by RMD are needed to ensure identifiability. For example, if the parent biconnected component of a triconnected component contains fewer redundant monitors, then we do not need a separate constraint for this triconnected component. We can simplify the constraints by removing redundant constraints before adding each new constraint, i.e., when adding a constraint $(S,\ k)$, all existing constraints $(S',\ k')$ with $S' \subseteq S$ and $k' \geq k$ can be removed.

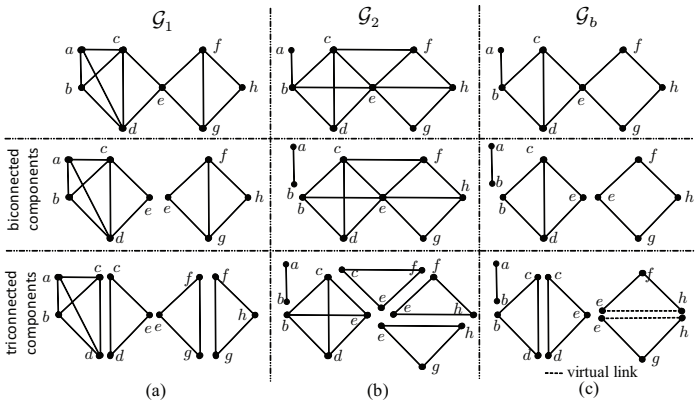


Fig. 3. Example of placement algorithms (an optimal monitor placement is $\{a, d, g, h\}$): (a) $\mathcal{G}_1$ and its decomposition; (b) $\mathcal{G}_2$ and its decomposition; (c) base graph $\mathcal{G}_b$ of $\mathcal{G}_1$ and $\mathcal{G}_2$ and the decomposition of $\mathcal{G}_b$.

2) Constrained Monitor Removal: Given an initial placement M_0 known to achieve identifiability for all $\mathcal{G}_t$ ($t = 1, \ldots, T$) and the constraints $\mathcal{R}$ obtained by applying RMD to each $\mathcal{G}_t$, the problem of selecting the maximum redundant monitors to remove is converted to a problem of selecting a maximum subset $M' \subset M_0$ such that $|M' \cap S_i| \leq k_i$ for all $(S_i, k_i) \in \mathcal{R}$. We refer to this problem as the *maximum hitting set problem (max-HSP)* with input $(M_0,\ \mathcal{R})$. The duality between minimum monitor selection and maximum redundant monitor removal implies an alternative way of computing the optimal placement as follows.

Corollary IV.7. Let $\mathcal{R}$ be the overall set of constraints computed by applying RMD to each of $\mathcal{G}_1, \ldots, \mathcal{G}_T$ with an initial placement $M_0 = V$. Let M' be the optimal solution to max-HSP$(V, \mathcal{R})$. Then $V \backslash M'$ is an optimal monitor placement for identifying $\mathcal{G}_1, \ldots, \mathcal{G}_T$.

Unfortunately, max-HSP is again NP-hard. In fact, it is exactly as hard as min-HSP because solving a min-HSP for input $(V,\ \{(S_i,\ k_i) :\ i = 1, 2, \ldots\})$ is equivalent to solving a max-HSP for input $(V,\ \{(S_i,\ |S_i| - k_i) :\ i = 1, 2, \ldots\})$. We can apply a greedy heuristic similar to the one used in Section IV-C2: while there are redundant monitors (i.e., monitors such that removing any one of them does not violate any constraint), remove the redundant monitor that is involved in the minimum number of constraints.

Complexity: Although max-HSP has the same complexity as min-HSP on the same input, the actual complexity of monitor removal depends on the size of the initial placement, which can be much smaller than $|V|$. Specifically, RMD has the same complexity as FMP, which is $O(T|V| + \sum_t |L_t|)$ for processing $\mathcal{G}_1, \ldots, \mathcal{G}_T$. The greedy heuristic for max-HSP has complexity $O(T|V| \cdot |M_0|)$ to select from a size-$|M_0|$ set under $O(T|V|)$ constraints. The overall complexity of refining an initial placement M_0 using RMD and greedy max-HSP is therefore $O(T|V| \cdot |M_0| + \sum_t |L_t|)$.

Example: Fig. 3 illustrates an example of applying different algorithms to identify $\mathcal{G}_1$ and $\mathcal{G}_2$. Using the one-shot placement, we first obtain the base graph $\mathcal{G}_b$ of $\mathcal{G}_1$ and $\mathcal{G}_2$ as shown in Fig. 3 (c) and then apply MMP to $\mathcal{G}_b$, which

yields a monitor placement $\{a, d, f, h, g\}$. In comparison, the incremental placement first applies MMP to $\mathcal{G}_1$, which selects $\{b, c, f, h\}$ as monitors. Based on these monitors, it then checks if further monitor placement is required in $\mathcal{G}_2$. Since nodes a and g (degree less than three in $\mathcal{G}_2$) must be monitors, the incremental placement finally selects monitors $\{b, c, f, h, a, g\}$. Meanwhile, it can be verified using Theorem III.1 that monitor placement $\{a, d, g, h\}$ identifies both $\mathcal{G}_1$ and $\mathcal{G}_2$ and is thus an optimal placement (since $\mathcal{G}_1$ already requires 4 monitors); therefore, both one-shot and incremental placements select redundant monitors. In contrast, the joint placement first computes the placement constraints: $(\{a, b\}, 1)$, $(\{a, b, c, d\}, 2)$, $(\{h\}, 1)$, and $(\{f, h, g\}, 2)$ for $\mathcal{G}_1$, and $(\{a\}, 1)$, $(\{g\}, 1)$, and $(\{c, d, e, f, g, h\}, 2)$ for $\mathcal{G}_2$. It then applies the greedy heuristic to solve the min-HSP problem under these constraints, which yields an optimal monitor placement $\{a, g, h, d\}$. Following similar procedures, we can show that the refined placement also generates the same optimal monitor placement.

V. Optimality Conditions for Robust Monitor Placement

We have seen from Section IV-C that, unlike the monitor placement problem in a static network, optimal monitor placement in a dynamic network is generally hard to compute even with knowledge of topologies. This motivates us to identify conditions under which the problem can be solved optimally. In this section, we investigate several such conditions in the order of increasing generality and identify the optimal solution in each case[5].

A. Optimality Condition for One-Shot Placement

An easy lower bound on the minimum number of monitors needed by a robust placement is the maximum number of monitors placed by MMP over all topologies. Therefore, if the number of monitors needed to identify the base graph $\mathcal{G}_b$ matches the lower bound, the one-shot placement based on $\mathcal{G}_b$ is guaranteed to be optimal.

Theorem V.1. If $\exists t \in \{1, \ldots, T\}$ such that the number of monitors placed by MMP in $\mathcal{G}_t$ equals the number of monitors placed by MMP in $\mathcal{G}_b$, then the one-shot placement is optimal.

Remark: Intuitively, this condition is satisfied when the link sets of different topologies have a (roughly) nested structure. A special case of this condition is when $\mathcal{G}_b = \mathcal{G}_t$ for some t, i.e., $\mathcal{G}_t$ is a subgraph of all other topologies $\mathcal{G}_{t'}$ for $t' \in \{1, \ldots, T\} \setminus t$.

B. Optimality Condition for Incremental Placement

Generally, the performance of incremental placement is sensitive to the order of applying IMMP to different topologies. In a special case where all topologies are 3-vertex-connected, however, the order no longer matters, and the incrementally computed placement is guaranteed to be optimal.

[5]Note that the conditions are sufficient but not necessary, i.e., it is possible for a given solution to be optimal when the corresponding optimality condition does not hold.

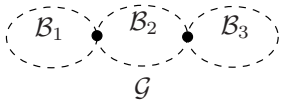


Fig. 4. A network with three biconnected (also triconnected) components $\mathcal{B}_i$ ($i = 1, \ldots, 3$).

Theorem V.2. If $\mathcal{G}_1, \ldots, \mathcal{G}_T$ are all 3-vertex-connected, then incremental placement is optimal regardless of the order of processing the topologies.

Proof. By MMP (Algorithm 1), we know that we need three and only three monitors to identify a 3-vertex-connected topology, and the three monitors can be placed arbitrarily. Therefore, incremental placement achieves the minimum number of monitors, and its performance is invariant to the order of processing the topologies. □

Remark: Although under this condition, incremental placement degenerates into MMP (monitor placement for any $\mathcal{G}_t$ identifies all the other topologies), it is different from one-shot placement. In particular, the base graph of multiple 3-vertex-connected graphs may not be 3-vertex-connected, and thus one-shot placement may need more than three monitors.

C. Optimality Condition for Joint Placement

As explained in Section IV-C2, the difficulty in solving the monitor placement problem optimally is caused by the hardness of min-HSP. Therefore, any condition that allows min-HSP to be solved optimally in polynomial time leads to an optimality condition for a polynomial-time joint placement algorithm. To this end, we establish a condition for optimally solving min-HSP that generalizes a well-known condition for solving HSP and present a polynomial-time algorithm to provide an optimal solution when this condition is satisfied.

It is well known that HSP is equivalent to the *set covering problem (SCP)*. Naturally, min-HSP can also be represented by a variation of SCP, which we call the *multi-set covering problem (MSCP)*. MSCP differs from SCP in that instead of requiring each item to be covered at least once, it can require each item to be covered multiple times, and the frequency of coverage can vary for different items. Given an input $(U, \mathcal{E})$ for min-HSP, where $\mathcal{E} = \{(S_i,\, k_i) : S_i \subseteq U\}$ means that at least k_i items must be selected from S_i, we can construct an input $(\tilde{U},\, \tilde{\mathcal{E}})$ for MSCP, where $\tilde{U} = \mathcal{E}$ specifies each "item" S_i and its minimum frequency of coverage k_i, and $\tilde{\mathcal{E}} = \{\mathcal{S}_e : e \in U\}$ specifies the "sets" used to cover S_i's ($\mathcal{S}_e := \{S_i : e \in S_i\}$). The min-HSP is equivalent to MSCP that selects the minimum number of sets $\mathcal{E}' \subseteq \tilde{\mathcal{E}}$ to cover each S_i at least k_i times. We will work on MSCP for the ease of presentation.

Although the general SCP is NP-hard, it is polynomial-time solvable when the input satisfies a condition known as the *consecutive ones property (C1P)* [23]. In words, a SCP has C1P if there is a permutation of the items such that each set covers a set of consecutive items. It can be shown that MSCP is also polynomial-time solvable under C1P[6]. This condition is, however, too strong for our problem. For example, for the network in Fig. 4, the monitor placement constraints computed by FMP has the form $\{(B_1,\, 2),\, (B_2,\, 1),\, (B_3,\, 2),\, (V,\, 3)\}$, where B_i denotes the set of internal nodes in component $\mathcal{B}_i$

[6]This is because under C1P, the constraint matrix for the ILP representation of MSCP is totally unimodular, and thus the LP relaxation gives an integral solution which is optimal [23].

($i = 1, \ldots, 3$) and V the entire node set. Since $\{B_1, V\}$, $\{B_2, V\}$, and $\{B_3, V\}$ all contain common nodes, there is no permutation of $\{B_1, B_2, B_3, V\}$ such that the sets containing a given node are always consecutive, i.e., the corresponding MSCP violates C1P. However, we know that the optimal monitor placement is polynomial-time solvable (by MMP).

Below, we give a more general condition for solving MSCP. Our condition is motivated by a root cause of complexity in MSCP: when the sets overlap arbitrarily, we can use multiple sets to cover a given item and the optimal decision depends on how other items are covered. However, for an item for which the covering sets are nested, i.e., each set is a subset of another, then the optimal decision is easy: selecting sets with the largest cardinality is always optimal as it helps to maximize the coverage of other items. The idea inspires the following condition.

Theorem V.3. If the items can be represented by nodes in a rooted tree such that each set covers a set of consecutive nodes along a leaf-to-root path in the tree, then MSCP (and the corresponding min-HSP) is polynomial-time solvable.

We prove this result by constructing an algorithm that solves MSCP optimally under the above condition. Let $\mathcal{I} = \{(e, k_e) : e \in U\}$ denote the items and their required frequency of coverage, and $\mathcal{E} = \{S_i : S_i \subseteq U\}$ the sets. The algorithm, referred to as *Leaf-based Greedy Cover (LGC)*, works as follows: while $\exists$ a non-sufficiently covered item (i.e., e such that the number of selected sets covering e is less than k_e),

1) for an arbitrary, non-sufficiently covered leaf item[7] e, select the set S that covers the most non-sufficiently covered items among sets covering e;
2) update the remaining coverage required by each item and the remaining sets.

The idea of the proof is to show that under the condition in Theorem V.3, LGC is optimal.

Proof. We prove the optimality of LGC by induction on the number of items $|U|$ to cover. If $|U| = 1$, then there is only one way to cover $e \in U$ and LGC is trivially optimal.

Assume that LGC is optimal for $|U| < n$. For $|U| = n$, let e be the leaf item selected by LGC and $S_1, \ldots, S_{k_e}$ be the k_e largest sets covering e (k_e is the coverage frequency required by e). Then LGC gives a solution that is the union of $\{S_1, \ldots, S_{k_e}\}$ and the LGC solution on a reduced input $(\mathcal{I}' := \{(e', \max(0, k_{e'} - t_{e'})) : e' \in U \backslash e\}, \mathcal{E} \backslash \{S_1, \ldots, S_{k_e}\})$, where $t_{e'}$ is the number of times an item e' is covered by $\{S_1, \ldots, S_{k_e}\}$. Suppose $\exists$ a feasible solution X that has a smaller size than the solution by LGC. Let $S'_1, \ldots, S'_{k_e}$ be the k_e largest sets covering e in X. Then the following statements hold: (i) $X \setminus \{S'_1, \ldots, S'_{k_e}\}$ is a feasible cover of $\mathcal{I}'$, and (ii) $X \setminus \{S'_1, \ldots, S'_{k_e}\}$ does not contain any of $S_1, \ldots, S_{k_e}$. The first statement holds because by the condition in Theorem V.3, sets covering e must be nested; this implies that $\tau_{e'} \leq t_{e'}$ for $\tau_{e'}$ being the number of times an item e' is covered by $\{S'_1, \ldots, S'_{k_e}\}$, and hence $X \setminus \{S'_1, \ldots, S'_{k_e}\}$ covers e' at least $\max(0, k_{e'} - t_{e'})$ times. The second statement holds because if any of $S_1, \ldots, S_{k_e}$ is in X, it must be one of the k_e largest sets covering e among sets in X and excluded from $X \setminus \{S'_1, \ldots, S'_{k_e}\}$. Together, the statements imply that $X \setminus \{S'_1, \ldots, S'_{k_e}\}$ is a feasible solution to MSCP on input $(\mathcal{I}', \mathcal{E} \backslash \{S_1, \ldots, S_{k_e}\})$, but has a smaller size than the solution by LGC, contradicting the induction assumption that LGC is optimal for $|U| < n$. □

[7]Precisely, e is such that e is not sufficiently covered, but all items below e in the rooted tree are sufficiently covered.

Translated back to the joint placement problem, the condition in Theorem V.3 requires that the node sets computed by FMP be representable by nodes in a rooted tree such that only sets on the same leaf-to-root path may overlap. One sufficient condition of having such arrangement is as follows.

Corollary V.4. Let $\mathcal{F} = \{(S_i, k_i), i = 1, 2, \ldots\}$ be the constraints computed by FMP. If for any $i \neq j$, S_i and S_j are either disjoint or nested (one is a subset of the other), then the constrained monitor selection problem (Section IV-C2) can be solved optimally in polynomial time.

Proof. We arrange S_i's into a rooted tree by connecting each S_i to the smallest S_j containing S_i via a directed link; the entire node set V serves as the root. It is easy to verify that since any two sets are either nested or disjoint, sets containing a common item are always nested and thus always on a leaf-to-root path. The result then follows from Theorem V.3. □

Remark: A special case satisfying the condition in Corollary V.4 is for placing monitors in a single topology, where S_i's are internal nodes in tri/bi/1-connected components of the topology. When the topology changes, this condition holds if there is only splitting or merging of components. In this sense, Corollary V.4 provides an explanation on why computing the optimal monitor placement for a single topology is easy (solved by MMP) but computing that for multiple topologies is generally hard.

Example: In Fig. 2, the monitor placement constraint is $\mathcal{F} = \{(\{a, b, c\}, 1), (\{f, g, h\}, 1), (\{a, b, c, d, e, f, g, h\}, 3)\}$, which satisfies the condition in Corollary V.4; therefore, the corresponding monitor placement is polynomial time solvable.

VI. Handling Practical Challenges

So far, we have focused on selecting monitors based on a given set of topologies that the network will have during its lifetime. In practice, a robust monitoring system has to address additional challenges. For example, how do we determine the set of topologies to plan for? What if the planned monitor placement fails to identify every link at runtime? What if it requires too many monitors? What if nodes also change? In this section, we discuss the challenges and the possible solutions for each of these issues.

A. Topology Selection

Robustness of monitor placement depends on the accuracy of topology prediction. While it is out of our scope to study specific topology prediction mechanisms, it is of interest to understand how to best utilize the results of a given predictor.

Let $\{\mathcal{G}_t : t = 1, \ldots, N\}$ denote all possible topologies during a time window of interest and τ_t the expected fraction of time for the topology to equal $\mathcal{G}_t$. Assume that $\tau_1 \geq \tau_2 \geq \ldots$.

Given a parameter τ denoting the required fraction of time that the network needs to be identifiable, the *topology selection problem* is to select a subset of topologies $\{\mathcal{G}_i : i \in I\}$ for $I \subseteq \{1, \ldots, N\}$ that can be identified with a minimum number of monitors, subject to the constraint that $\sum_{i \in I} \tau_i \geq \tau$.

The above optimization problem is clearly hard to solve as evaluating the objective of a given solution (i.e., the minimum number of monitors to identify given topologies) is already NP-hard as shown in Section IV-C2. A straightforward alternative is to select the top-K frequently-occurring topologies $\{\mathcal{G}_1, \ldots, \mathcal{G}_K\}$ for the minimum K such that $\sum_{t=1}^{K} \tau_t \geq \tau$. Such a selection, however, ignores the structural difference in these topologies and may require more than the minimum number of monitors. Observing that a monitor placement capable of identifying a given topology can also identify a denser topology (Section IV-A), we can aggregate the topologies by computing, for each $\mathcal{G}_t$, the sum frequency $\tilde{\tau}_t$ of all topologies containing $\mathcal{G}_t$ as subgraph, which gives a tighter lower bound (than τ_t) on the expected fraction of time of achieving identifiability, once we place monitors to identify $\mathcal{G}_t$. We can then select topologies in descending order of $\tilde{\tau}_t$.

B. On-demand Monitor Placement

In case of unpredictable topology changes, monitors placed during network planning may not be able to identify all links. One way to ensure identifiability in this case is to employ additional nodes as *temporary* monitors to identify the topology at hand. Temporary monitors only participate in taking measurements but not in other functions such as storage or processing of measurements; to distinguish from temporary monitors, we refer to monitors deployed during network planning as *persistent* monitors. Given the current topology $\mathcal{G}_t$ and the persistent monitors M_0 (placed by a robust monitor placement algorithm in Section IV), we can apply IMMP to select temporary monitors to identify all links in $\mathcal{G}_t$, and Lemma IV.2 guarantees that the number of temporary monitors is minimized.

C. Bounded Number of Monitors

So far, we have required that the placed monitors must identify all (planned) topologies, which may require a large number of monitors when the topologies are substantially different. In such cases, it can be beneficial to relax the objective such that we only place a small number of persistent monitors during network planning, and employ temporary monitors as needed to identify topologies at runtime. Since it incurs a cost to change the status (monitor/non-monitor) of a node, such relaxation faces an inherent tradeoff between the number of monitors and the number of changes in node status.

One way of controlling such tradeoff is to specify a bound m on the number of persistent monitors. Given m, the goal of monitor placement during network planning is changed to selecting a set M_0 ($|M_0| \leq m$) of persistent monitors such that the number of temporary monitors needed at runtime is minimized. Given topologies $\{\mathcal{G}_t : t = 1, \ldots, T\}$, the number of temporary monitors needed to identify any topology $\mathcal{G}_t$ is upper-bounded by the number of additional monitors needed to simultaneously identify all these topologies. To minimize this upper bound, it suffices to ensure that M_0 is a subset of the optimal robust monitor placement for $\{\mathcal{G}_t : t = 1, \ldots, T\}$. This observation suggests that any algorithm for robust monitor placement (see Section IV) can be used for bounded monitor placement by enforcing early termination after selecting m monitors.

TABLE I
DYNAMIC TOPOLOGIES FOR TAXI NETWORK (86 NODES)

range (m)	#topology changes	avg #links	avg #components
500	479	95.1	31.3
1000	479	334.3	6.3
1500	479	694.8	2.5
2000	479	1106.5	1.5
2500	479	1528.3	1.1
3000	479	1934.0	1.0
3500	479	2286.8	1.0

D. Changes of Nodes

We have limited topology changes to addition/removal of links, while in practice there may also be arrival/departure of nodes. We can handle arrivals by always selecting the newly arrived nodes as monitors; we can handle departures by treating each departure as removal of all the links incident to the departed node. It is easy to verify that both approaches guarantee identifiability under node changes; we leave further optimization to future work.

VII. PERFORMANCE EVALUATION

We evaluate the proposed solutions on dynamic topologies extracted from mobility traces. We use two datasets: (1) taxi cab traces from San Francisco[8], from which we select traces of 86 nodes over a 8-hour period with location updates roughly every minute; (2) mobility traces generated by the Network Science Research Laboratory at the US Army Research Laboratory [24], which contain traces of 90 nodes belonging to 7 groups during a 400-second tactical operation with location updates every second[9]. Dataset (1) represents independent node mobility, and dataset (2) represents grouped node mobility. We study both datasets to understand the impact of mobility patterns on monitor placement.

We extract dynamic topologies from each trace by specifying a communication range and connecting two nodes by a link whenever they are within the range. See Tables I and II for a summary of extracted topologies under different ranges. We see that both networks experience hundreds of topology changes throughout their lifetime. As expected, the network becomes denser (with a larger number of links) and better connected (with a smaller number of connected components) as the range increases.

Comparison of algorithms: We first compare the performance of the proposed robust monitor placement algorithms in terms of the number of monitors (note that all algorithms guarantee identifiability). Here joint placement and refined placement use greedy heuristics to solve the associated

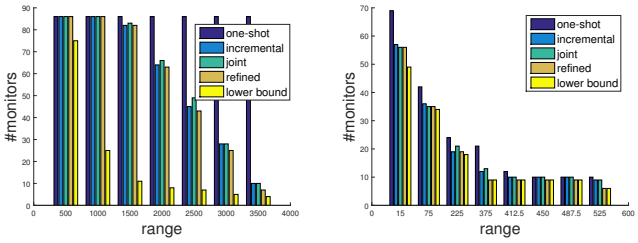
[8]Traces are available at: http://crawdad.org/epfl/mobility/.

[9]The anonymized traces used for this evaluation are available at: https://www.dropbox.com/s/bkhdifos9wzjwln/trace_90node_401second.txt?dl=0

TABLE II
DYNAMIC TOPOLOGIES FOR TACTICAL NETWORK (90 NODES)

range (m)	#topology changes	avg #links	avg #components
15	399	325.9	17.3
75	293	539.9	10.4
225	196	1027.7	4.2
375	387	1256.1	2.0
450	380	1607.5	1.5
525	399	2191.1	1.1

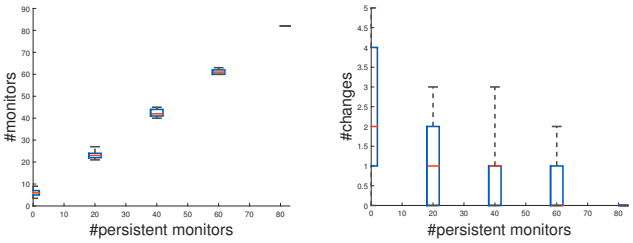
min/max-HSP, and refined placement uses the result of one-shot placement as the initial set of monitors. We also evaluate a lower bound on the number of monitors by computing the maximum number of monitors placed by MMP in any single topology. Fig. 5 (a) shows the result for the taxi network and Fig. 5 (b) shows the result for the tactical network. As expected, the one-shot placement algorithm uses more monitors than the other algorithms, although it is much faster (omitted due to space). The refined placement algorithm uses the fewest monitors at the cost of a longer computation time (omitted). Comparing results for the two networks, we see that the taxi network has very different topologies during its lifetime and thus requires a large number of monitors to maintain identifiability, while the tactical network contains subnets (of nodes in the same group) with relative stable topologies and thus requires fewer monitors. As the range for the tactical network increases (Fig. 5 (b)), the refined placement becomes optimal (achieving the lower bound) and even the one-shot placement is near-optimal.



(a) taxi (b) tactical

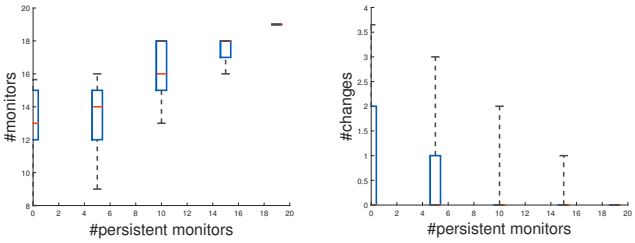
Fig. 5. Comparing different robust placement algorithms under varying ranges.

Varying number of persistent monitors: One way to further reduce the number of monitors is to limit the number of persistent monitors placed during network planning and place additional temporary monitors as needed at runtime. Fig. 6 shows the results for the taxi network under various bounds on the number of persistent monitors, where the persistent monitors are randomly selected from monitors placed by the refined placement algorithm. To evaluate the cost of (re)configure temporary monitors, we count the number of changes in node status, where a change occurs when a non-monitor is selected as a temporary monitor or a temporary monitor becomes a non-monitor. Fig. 6 (a) shows the median (red bar), 25/75-th percentile (box), and 5/95-th percentile (whisker) of the number of (both persistent and temporary) monitors over all topologies, and Fig. 6 (b) shows the corresponding values for the number of node status changes. The results show a clear tradeoff between the number of monitors and the number of changes, controlled by the number of persistent monitors deployed during network planning. Similar observations are made for the tactical network; see Fig. 7.



(a) #monitors (b) #changes

Fig. 6. Bounding number of persistent monitors (taxi network, range $= 1500$ meters).



(a) #monitors (b) #changes

Fig. 7. Bounding number of persistent monitors (tactical network, range $= 225$ meters).

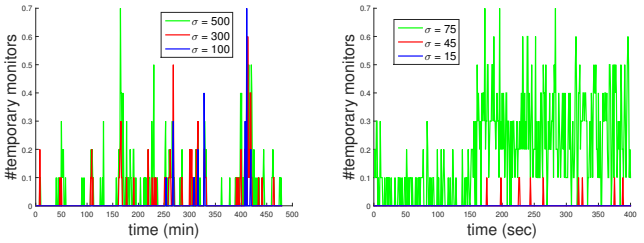
Impact of prediction error: So far we have assumed accurate knowledge of topologies. It remains to evaluate how well our placement algorithms perform when input topologies contain errors. To evaluate this, we add i.i.d. zero-mean Gaussian noise with variance σ^2 to node locations (x/y-coordinates) and treat the resulting topologies as the new ground truth; the process is then repeated for multiple Monte Carlo runs. Meanwhile, the topologies computed from the original trace are provided to the monitor placement algorithm (refined placement) as predicted topologies[10]. We evaluate the robustness of the resulting placement by: (i) testing whether the placement achieves identifiability for each newly generated topology, and (ii) if not, computing the number of temporary monitors needed to achieve identifiability. The result in Fig. 8 shows that our monitor placement is highly robust to prediction error. Specifically, our placement achieves identifiability for 0.95 fraction of time for the taxi network (Fig. 8 (a)), and 0.82 fraction of time for the tactical network (Fig. 8 (b)), even if the error in node location[11] is as large as $1/3$ of the communication range. For the rest of the time, we only need to add a couple of temporary monitors to achieve identifiability.

VIII. CONCLUSION

We have studied the problem of placing monitors to identify additive link metrics from end-to-end measurements in the

[10]This is equivalent to treating the original topologies as ground truth and the modified topologies as estimates.

[11]The value of σ is basically the root-mean-squared error (RMSE) of the maximum likelihood estimate of node locations.



(a) taxi (range = 1500 meters) (b) tactical (range = 225 meters)

Fig. 8. Performance under prediction error (10 Monte Carlo runs).

face of topology changes. Unlike existing monitor placement algorithms that consider a single topology, our algorithms provide robust monitor placements that simultaneously identify multiple topologies. By casting the problem as a generalized hitting set problem, we show that the optimal placement is NP-hard to compute, and identify special cases where the problem can be solved efficiently by the proposed algorithms. We further extend the solution to include on-demand monitor placement to handle unpredictable changes with controllable tradeoff between monitor cost and adaptation cost. Our evaluations on topologies driven by real mobility traces verify the effectiveness and robustness of the proposed solution.

REFERENCES

[1] M. Coates, A. O. Hero, R. Nowak, and B. Yu, "Internet tomography," *IEEE Signal Processing Magzine*, vol. 19, pp. 47–65, 2002.

[2] A. B. Downey, "Using pathchar to estimate internet link characteristics," in *IEEE SIGCOMM*, 1999.

[3] G. Jin, G. Yang, B. Crowley, and D. Agarwal, "Network characterization service (NCS)," in *IEEE HPDC*, 2001.

[4] E. Lawrence and G. Michailidis, "Network tomography: A review and recent developments," *Frontiers in Statistics*, vol. 54, 2006.

[5] O. Gurewitz and M. Sidi, "Estimating one-way delays from cyclic-path delay measurements," in *IEEE INFOCOM*, 2001.

[6] Y. Chen, D. Bindel, and R. H. Katz, "An algebraic approach to practical and scalable overlay network monitoring," in *ACM SIGCOMM*, 2004.

[7] A. Chen, J. Cao, and T. Bu, "Network Tomography: Identifiability and Fourier domain estimation," in *IEEE INFOCOM*, 2007.

[8] A. Gopalan and S. Ramasubramanian, "On identifying additive link metrics using linearly independent cycles and paths," *IEEE/ACM Transactions on Networking*, vol. 20, no. 3, 2012.

[9] L. Ma, T. He, K. K. Leung, A. Swami, and D. Towsley, "Inferring link metrics from end-to-end path measurements: Identifiability and monitor placement," *IEEE/ACM Transactions on Networking*, vol. 22, no. 4, pp. 1351–1368, June 2014.

[10] L. Ma, T. He, K. K. Leung, D. Towsley, and A. Swami, "Efficient identification of additive link metrics via network tomography," in *IEEE ICDCS*, 2013.

[11] M. Zhao and W. Wang, "Analyzing topology dynamics in ad hoc networks using a smooth mobility model," in *IEEE WCNC*, 2007.

[12] K. H. Wang and L. Baochun, "Group mobility and partition prediction in wireless ad-hoc networks," in *IEEE ICC*, 2002.

[13] S. M. Mousavi, Y. R. Rabiee, M. Moshref, and A. Dabirmoghaddam, "Mobility aware distributed topology control in mobile ad-hoc networks with model based adaptive mobility prediction," in *WiMOB*, 2007.

[14] I. W. Ho, K. K. Leung, and J. W. Polak, "Stochastic model and connectivity dynamics for vanets in signalized road systems," *IEEE/ACM Transactions on Networking*, vol. 19, pp. 195–208, 2011.

[15] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C.-N. Chuah, and C. Diot, "Characterization of failures in an IP backbone," in *IEEE INFOCOM*, 2004.

[16] Y. Bejerano and R. Rastogi, "Robust monitoring of link delays and faults in IP networks," in *IEEE INFOCOM*, 2003.

[17] R. Kumar and J. Kaur, "Practical beacon placement for link monitoring using network tomography," *IEEE JSAC*, vol. 24, 2006.

[18] J. D. Horton and A. Lopez-Ortiz, "On the number of distributed measurement points for network tomography," in *ACM IMC*, 2003.

[19] R. Tarjan, "Depth-first search and linear graph algorithms," *SIAM Journal on Computing*, vol. 1, pp. 146–160, 1972.

[20] J. E. Hopcroft and R. E. Tarjan, "Dividing a graph into triconnected components," *SIAM Journal on Computing*, vol. 2, pp. 135–158, 1973.

[21] I. Dinur and D. Steurer, "Analytical approach to parallel repetition," in *ACM STOC*, 2014.

[22] V. Chvatal, "A greedy heuristic for the set-covering problem," *Mathematics of Operations Research*, vol. 4, pp. 233–235, 1979.

[23] N. Rug and A. Schobel, "Set covering with almost consecutive ones property," *Discrete Optimization*, vol. 1, pp. 215–228, November 2004.

[24] "The Network Science Research Laboratory," US Army Research Laboratory. [Online]. Available: https://www.arl.army.mil/www/default.cfm?page=2485