Scheduling Resource-Bounded Monitoring Devices for Event Detection and Isolation in Networks

Waseem Abbas, Aron Laszka, Yevgeniy Vorobeychik, *Member, IEEE*, and Xenofon Koutsoukos, *Senior Member, IEEE*

Abstract—In networked systems, monitoring devices such as sensors are typically deployed to monitor various target locations. Targets are the points in the physical space at which events of some interest, such as random faults or attacks, can occur. Most often, monitoring devices have limited energy supplies, and they can operate for a limited duration. As a result, energy-efficient monitoring of target locations through a set of monitoring devices with limited energy supplies is a crucial problem in networked systems. In this paper, we study optimal scheduling of monitoring devices to maximize network coverage for detecting and isolating events on targets for a given network lifetime. The monitoring devices considered could remain active only for a fraction of the overall network lifetime. We formulate the problem of scheduling of monitoring devices as a graph labeling problem, which unlike other existing solutions, allows us to directly utilize the underlying network structure to explore the trade-off between coverage and network lifetime. In this direction, first we present a greedy heuristic, and then a game-theoretic solution to the graph labeling problem. The proposed setup can be used to simultaneously solve the scheduling and placement of monitoring devices, which, as our simulations illustrate, gives improved performance as compared to separately solving the placement and scheduling problems. Finally, we illustrate our results on various networks, including real-world water distribution networks and random geometric networks.

Index Terms—Scheduling, network coverage, graph labeling, potential games, dominating sets

1 Introduction

Detection and isolation of unwanted events such as faults, failures, and malicious intrusions is a fundamental concern in a variety of practical networks. For example, leakage detection in water distribution networks can reduce physical damage as well as financial losses [1]. For this purpose, monitoring devices, such as sensors, are typically deployed strategically throughout the network. Spatially distributed systems over large areas may often be monitored only by battery-powered devices, as wired deployment can be prohibitively expensive or impossible. If the power supply provided by batteries is insufficient for continuous monitoring during the intended lifetime of a system, batteries must be replaced regularly. Since the cost of battery replacement for a large number of devices can be very expensive, one of the primary design concerns for such systems is increasing the time until the batteries of sensors are depleted. At the same time, it is desired to maintain a certain level of monitoring in terms of the number of targets covered throughout the network lifetime. Here, targets are the points in the physical space at which events of interest can occur. For instance, in water distribution networks, events can be the pipe bursts, and so targets can be the water pipes, which need to be monitored through sensors such as battery operated pressure sensors.

One of the primary approaches for conserving battery power is "sleep scheduling." The idea is to have only a subset of the sensors activated at any given time, and to turn off (i.e., "sleep") the remaining ones, thereby conserving power. By activating different sets of devices one after another, the overall lifetime of a system can be substantially increased. Previous research efforts, which we discuss briefly in Section 9, have mostly focused on finding schedules that ensure complete coverage, that is, guaranteeing that every target is monitored by some device at any given moment in time (e.g., [2], [3]). However, complete coverage is a very strict requirement, which severely limits the sets of devices that may be asleep at the same time. In fact, coverage (i.e., ratio of monitored targets to the total number of targets) is a submodular function of the set of active devices in most models (e.g., [4], [5]), which roughly means that attaining complete coverage is disproportionately expensive compared to achieving reasonably good coverage. Managing energy resources of monitoring devices via their scheduling to achieve an appropriate coverage of targets is a significant issue in networks where an extended network lifetime is a critical requirement.

In this paper, we design efficient scheduling schemes for a set of monitoring devices with limited battery supplies to achieve maximum target coverage for a given network lifetime. Scheduling of such devices to achieve complete

• *The authors are with the Department of Electrical Engineering and Computer Science, Vanderbilt University, Nashville, TN 37235. Email: {waseemabbas, yevgeniy.vorobeychik, xenofon.koutsoukos}@vanderbilt.edu, laszka@berkeley.edu.*

Manuscript received 21 June 2016; revised 28 Mar. 2017; accepted 16 July 2017. Date of publication 30 July 2017; date of current version 13 Mar. 2018. (Corresponding author: Waseem Abbas.)
Recommended for acceptance by S. Roy.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TNSE.2017.2734048

network coverage is a special case of this general formulation. We model the network as a graph, in which monitoring devices could be deployed at a subset of nodes, and targets could be a subset of nodes and edges. Each monitoring device has a limited active time, and covers a subset of targets within its range during its active time. For a given network lifetime, the objective is to determine the maximum possible coverage, both in terms of the detection and isolation of (events at) targets, and a schedule of monitoring devices to obtain an optimal coverage.

In this direction the main contributions of the paper are:

1) We show that the optimal scheduling of monitoring devices is an APX-hard problem, that is, there is no polynomial-time approximation scheme (PTAS) for the problem unless P = NP.
2) We provide a graph-theoretic formulation of the scheduling problem by showing that it is equivalent to a unique graph labeling problem, which allows us to directly exploit the network structure to obtain optimal schedules.
3) To solve the graph labeling, and hence the scheduling problem, we propose two solutions; first, a greedy heuristic that runs in polynomial time, and gives near optimal solutions for many networks as we illustrate. However, in general, performance guarantees of the heuristic in terms of the optimality of the solution remain unknown. Second, we present a game-theoretic solution, in which we pose the labeling problem as a potential game. Using a well known binary log-linear learning (BLLL) algorithm to solve the potential game then ensures that in the long run, we achieve a globally optimal solution with an arbitrarily high probability.
4) Moreover, we illustrate that the game-theoretic solution allows simultaneously optimizing the placement and scheduling of monitoring devices that gives better results—as shown by the numerical results—compared to separately solving the placement and scheduling. Note that the placement problem involves selecting optimal locations to deploy a given set of monitoring devices to maximize the target coverage within networks.
5) We analyze the performance of the approach through simulations on various networks including real-world water distribution networks and random networks. For random networks, we also provide analytical results to determine the performance of random scheduling, which does not require any information about the network structure.

The rest of the paper is organized as follows: Section 2 explains our system model and defines the scheduling problem. Section 3 addresses the issue of complexity of the problem. Section 4 presents a graph labeling based formulation of the scheduling, and Section 5 proposes solutions to the graph labeling problem. Section 6 extends our approach to solve the simultaneous placement and scheduling of monitoring devices. Section 7 presents a particular case of interest of the scheduling problem, and Section 8 illustrates simulation results. Section 9 provides an overview of related work, and Section 10 concludes the paper.

2 System Model and Problem Formulation

In this section, first, we present the system model, and then we formulate the problem of optimal scheduling of resource bounded monitoring devices in networks.

(a) *Network Graph* - We model the network as an *undirected graph*,[1] $G(V, E)$, in which V is the set of nodes, and E is the set of edges given by the unordered pairs of nodes. Two nodes are adjacent if there exists an edge between them. The *neighborhood of a node* v, denoted by $N(v)$, is the set of all nodes that are adjacent to v, i.e., $N(v) = \{u : (u, v) \in E\}$, and the *neighborhood of a subset of nodes* S, denoted by $N(S)$, is $\bigcup_{v \in S} N(v)$. The *degree* of a node v, represented by $\delta(v)$, is simply $\delta(v) = |N(v)|$. A *path* is a sequence of nodes such that any two consecutive nodes in the path are adjacent, and the number of edges included in the path is the *length* of the path. Any two nodes are said to be *connected* if there exists a path between them. The *distance between connected nodes* u and v, denoted by $d(u, v)$, is the length of the shortest path between them. Similarly, the *distance between node* u *and edge* $e = (i, j)$ is $d(u, e) = \max(d(u, i), d(u, j))$. The network graph abstracts interactions among various nodes within the network.

(b) *Targets* - They are a subset of nodes and/or edges, denoted by $Y \subseteq (V \cup E)$, that could be subjected to an abnormal activity (or *event*), such as pipe failure, and therefore, need to be monitored by the monitoring devices.

(c) *Monitoring Devices* - These devices are deployed at a subset of nodes $S \subseteq V$ in the network, and they can monitor the other nodes and/or links of the network for some unusual activity, for instance, detecting link failures such as pipe burst in water networks. We consider a general model of monitoring, which is independent of the specific implementation and nature of the detection devices. We refer to any abnormal activity on a target as an *event*. A monitoring device can monitor all nodes and edges for events that lie within some pre-specified distance, referred to as the *range*, of the device. If u is the node at which a monitoring device with the range λ is deployed, then the device *covers* (monitors) all the nodes and edges in the set

$$\{v \in V : d(u, v) \leq \lambda\} \cup \{e \in E : d(u, e) \leq \lambda\}.$$

In other words, a target is *covered* if and only if it lies within the range of some monitoring device. Each device is resource-bounded in terms of the available *battery supply*, denoted by B, which means that a device can be *active* (or can be operational) for only B time duration. Furthermore, a monitoring device has only two output states—*event detected* at some target without knowing the exact location of the target, and *no event detected*.

1. Our results can also be applied to directed graphs in a straightforward way. For the ease of presentation, we consider only undirected graphs in this paper.

2.1 Network Performance Measures

We are interested in measuring the quality of monitoring of targets through a set of monitoring devices, both from the detection and isolation perspectives. In *detection*, the objective is just to detect any abnormal activity on some target irrespective of determining the exact location of it, whereas in *isolation*, the goal is to *uniquely detect* the target at which the abnormal activity occurs. Moreover, we refer to the overall lifetime of the network, i.e., duration for which monitoring of targets for detection (isolation) is considered, as the *network lifetime* T. To simplify, we divide the time into *time slots* of equal length. The battery supply B of a monitoring device could be represented by the number of time slots, say σ, in which the device could remain active. Moreover, the network lifetime T could be represented by the total number of time slots, say k, for which the detection (isolation) of targets is considered. Note that T and B represent the actual *duration* of overall network lifetime and battery lifetime of individual monitoring device respectively, whereas, k and σ, which are chosen to be positive integers, represent respectively the total *number of time slots* and the time slots for which each device could remain active.

(a) Detection Measure - Let there be a total of m targets, and m_i be the number of targets that are covered by the monitoring devices that are active in the i^{th} time slot. We define the *average detection performance*, denoted by $\mathcal{D}$, as

$$\mathcal{D} = \frac{1}{k}\sum_{i=1}^{k}\left(\frac{m_i}{m}\right). \tag{1}$$

(b) Isolation Measure - Consider two targets x and y, and let $S(x), S(y) \subseteq S$ be the subsets of sensing devices that detect events at targets x and y respectively. If $S(x)$ is identical to $S(y)$, then we can never distinguish or isolate the event at target x from the event at target y. Thus, to isolate events at x and y, $S(x)$ must be different from $S(y)$, which simply means that there should exist at least one sensing device that gives different outputs in the case of events at x and y. In other words, a sensing device should exist that detects event at either x or y, but not both at the same time. If such a sensing device exists for x and y, we say that the *target-pair* x, y *is covered*. Now to isolate (distinguish) event at x from events at all other targets, it is necessary that all target-pairs x, y, $\forall y \neq x$ are covered. If the total number of targets is m, then for each target x, there are $\binom{m-1}{2}$ target-pairs that need to be covered to isolate event at x from events at all other targets. Considering all m targets, we have a total of $\binom{m}{2}$ target-pairs in the whole network. If all such target-pairs are covered, event at any target can be isolated. Thus, the goal is to maximize the number of target-pairs that are covered. We denote by ℓ_j the number of target-pairs that are covered in the jth time slot by the sensing devices active in the jth time slot. Then we define the *average isolation performance*, denoted by $\mathcal{I}$, as

$$\mathcal{I} = \frac{1}{k}\sum_{j=1}^{k}\left(\frac{\ell_j}{\ell}\right), \tag{2}$$

where k is the total number of time slots.

2.2 Problem Formulation

Consider a network $G(V, E)$ in which $S \subseteq V$ is the subset of nodes at which monitoring devices with ranges λ are deployed, and $Y \subseteq (V \cup E)$ are the set of targets. Each monitoring device could remain active in at most σ of the total of k time slots due to battery supply constraints. In each time slot i, let $S_i \subseteq S$ be the subset of nodes with active monitoring devices. Thus, we get a *schedule* of (active) monitoring devices as $S_1, S_2, \ldots, S_k$.

The objective is to determine the maximum average detection performance $\mathcal{D}$ (or average isolation performance $\mathcal{I}$) for a given network life time, represented by k time slots, under the battery constraints of monitoring devices, represented by σ time slots, and also a schedule of monitoring devices that achieves the maximum $\mathcal{D}$ (or $\mathcal{I}$).

It is obvious that increasing k could decrease the maximum value of $\mathcal{D}$ (or $\mathcal{I}$). So, in a way, our goal is to understand a relationship between k and $\mathcal{D}$ (or $\mathcal{I}$), and design a systematic scheme to obtain a schedule for activating monitoring devices with limited battery supplies to obtain the desired network performance. Note that the scheduling problem for a *complete coverage* of targets, in which the objective is to determine a schedule that ensures $\mathcal{D} = 1$ throughout the network life is a special case.

3 Problem Complexity

In this section, we show that the problem of finding a schedule that maximizes the average detection performance for a given network lifetime and battery supplies, as discussed in Section 2.2, is APX-hard. APX-hardness implies that (unless P = NP), there does not exist a polynomial-time algorithm that can solve the problem to within arbitrary multiplicative factor of the optimum.

In our case, for a target τ, if Q_τ represents the fraction of the total number of time slots in which an event on τ can be detected (i.e., τ is covered), then the expected value of detecting an event on an arbitrary target, denoted by $\mathcal{Q}$ is

$$\mathcal{Q} = \frac{1}{|Y|}\sum_{\tau \in Y} Q_\tau. \tag{3}$$

Note that $\mathcal{Q}$ and $\mathcal{D}$ have exactly same values for a given schedule $(S_1, S_2, \ldots, S_k)$, and therefore, they both measure the average detection performance of the schedule. We formulate finding a schedule that maximizes detection performance as the following optimization problem:

Definition (Maximum Average Detection). *Given a graph $G = (V, E)$, a set of monitoring devices $S \subseteq V$, a set of targets $Y \subseteq (V \cup E)$, range of the monitoring device λ, a network lifetime represented by k time slots, a battery supply represented by σ time slots, find a schedule $(S_1, S_2, \ldots, S_k)$ that maximizes the average detection performance $\mathcal{Q}$.*

Theorem 3.1. *The Maximum Average Detection Problem is APX-hard.*

We show APX-hardness by reducing a well-known APX-hard problem, the Maximum Cut Problem [6] to the detection problem. The Maximum Cut Problem is defined as follows:

Definition (Maximum Cut Problem). *Given a graph $G = (V, E)$, find a disjoint partition V_1, V_2 of V that maximizes the number of edges $|E(V_1, V_2)|$ between V_1 and V_2.*

Proof (Theorem 3.1). We prove APX-hardness by showing that there exists a PTAS-reduction from the Maximum Cut Problem to the Maximum Average Detection Problem. First, we define a polynomial-time mapping from an instance of the cutting problem to an instance of the detection problem:

- let the network of the Maximum Average Detection Problem be the graph of the Maximum Cut Problem;
- let the set of monitoring devices be $S = V$;
- let the set of targets be $Y = E$;
- let the range of the monitoring device be $\lambda = 1$;
- let the network lifetime be $k = 2$ time slots;
- and let the battery supply be $\sigma = 1$ time slot.

Second, we define a polynomial-time mapping from a solution (S_1, S_2) of an instance of the detection problem (i.e., a schedule) to a solution (V_1, V_2) of the corresponding instance of the cutting problem (i.e., a cut)

$$V_1 := S_1 \text{ and } V_2 := S_2. \tag{4}$$

Next, observe that if an edge is cut by (V_1, V_2), then the corresponding target is covered by both S_1 and S_2, which implies $Q_\tau = 1$. On the other hand, if an edge is not cut by (V_1, V_2), then the corresponding target is covered in only one time slot, which implies $Q_\tau = \frac{1}{2}$. Consequently, for any pair of solutions (S_1, S_2) and (V_1, V_2), we have

$$\mathcal{Q}(S_1, S_2) = \frac{1}{|E|}\left(\sum_{\tau \in E(V_1,V_2)} 1 + \sum_{\tau \notin E(V_1,V_2)} \frac{1}{2}\right) \tag{5}$$

$$= \frac{1}{|E|}\left(|E(V_1, V_2)| + \frac{1}{2}(|E| - |E(V_1, V_2)|)\right) = \frac{1}{2} + \frac{1}{2}\frac{|E(V_1, V_2)|}{|E|}. \tag{6}$$

Using the same argument, we can also show that if a schedule (S_1, S_2) is an optimal solution to the detection problem, then the cut $(V_1 = S_1, V_2 = S_2)$ is also an optimal solution to the cutting problem, and vice versa. Therefore, if a schedule (S_1, S_2) is at most $(1 - \epsilon)$ times worse than the optimal schedule, then the corresponding cut (V_1, V_2) is at most $(1 - 2\epsilon)$ times worse than the optimal cut. Consequently, there is a PTAS-reduction from the Maximum Cut Problem to the Maximum Average Detection Problem. □

As a consequence, we cannot optimally solve the maximum average detection problem in a polynomial time. Hence, we need efficient heuristics that can provide reasonably good solutions with acceptable time complexities. In this regard, it becomes crucial to maximally exploit the structure of the problem in a systematic way. To achieve this objective, we first provide a graph-theoretic formulation of the scheduling problem in the next section.

4 A Graph-Theoretic Formulation of the Scheduling Problem

In this section, using various graph-theoretic notions, we formulate the scheduling problem as a graph labeling problem. Our approach is to first obtain a *bi-partite graph*, denoted by $\mathcal{G}(\mathcal{V}, \mathcal{E})$, from a given graph. This bi-partite graph illustrates targets and the monitoring devices with given ranges covering those targets. We then formulate the scheduling problem on the original network $G(V, E)$ as a graph labeling problem on the bi-partite graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$.

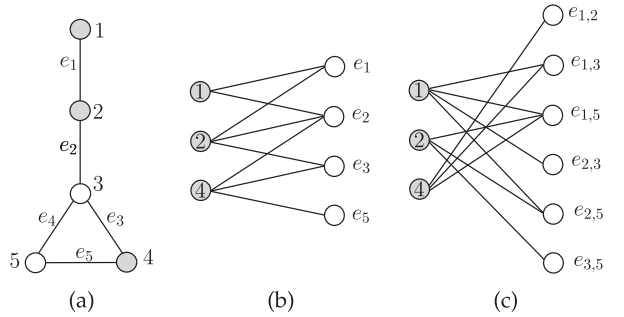


Fig. 1. (a) An example network graph $G(V, E)$. Bi-partite graph representations for (b) detection and (c) isolation.

4.1 Bi-Partite Graphs for Detection and Isolation

When scheduling of monitoring devices is required with an objective to maximize the average *detection* score $\mathcal{D}$, as described in Section 2.1, the bi-partite graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is simply obtained as follows: the vertex set $\mathcal{V}$ is the union $\mathcal{X} \cup \mathcal{Y}$, where $\mathcal{X} = S \subseteq V$ is the set of nodes corresponding to the set of monitoring devices, and $\mathcal{Y} = Y$ is the set of targets in the original network G. Moreover, each $x \in \mathcal{X}$ is adjacent to vertices in $\mathcal{Y}$ that are at most λ distance away from x in G. An example is shown in Fig. 1.

If maximizing the average *isolation* measure $\mathcal{I}$, as in Section 2.1, is the objective of scheduling, then $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is obtained as follows: As in the case of detection, the vertex set of the bi-partite graph is $\mathcal{V} = \mathcal{X} \cup \mathcal{Y}$, where $\mathcal{X} = S \subseteq V$ corresponds to the set of monitoring devices. To obtain $\mathcal{Y}$, we define a node for every pair of targets in Y. There will be $\binom{|Y|}{2}$ such nodes in $\mathcal{Y}$. As for the edge set $\mathcal{E}$ of the bi-partite graph, let $y \in \mathcal{Y}$ corresponds to the (unordered) target-pair $(\tau_1, \tau_2) \in Y$. Then, each $x \in \mathcal{X}$ is adjacent to $y \in \mathcal{Y}$ in $\mathcal{G}$ if and only if exactly one of the targets τ_1 or τ_2 is within λ distance from (the monitoring device corresponding to) x in the original network G. In other words, in the bi-partite graph $\mathcal{G}$, there will be no edge between x and y corresponding to the target-pair (τ_1, τ_2), if and only if the monitoring device x covers both targets τ_1 and τ_2 in G, or does not cover any of the targets τ_1 and τ_2. An example is illustrated in Fig. 1.

Example. Consider a graph $G(V, E)$ in Fig. 1. Let $S = \{1, 2, 4\} \subseteq V$ be the set of monitoring devices and edges in the set $Y = \{e_1, e_2, e_3, e_5\}$ be the targets. Moreover, each monitoring device has the range $\lambda = 2$. The bi-partite graphs $\mathcal{G}(\mathcal{V}, \mathcal{E})$ for the scheduling of monitoring devices to maximize the detection and isolation measures are shown in Figs. 1b and 1c respectively. The vertex set of bi-partite graphs in both cases is $\mathcal{V} = \mathcal{X} \cup \mathcal{Y}$, where $\mathcal{X} = S$. For the detection case, $\mathcal{Y} = Y$, and for the isolation case, $\mathcal{Y} = \{e_{12}, e_{13}, e_{15}, e_{23}, e_{25}, e_{35}\}$, where e_{ij} corresponds to the pair of edges (e_i, e_j) in Y. Note that an edge between $x \in \mathcal{X}$ and $e_{ij} \in \mathcal{Y}$ indicates that the monitoring device at x covers the target-pair (e_i, e_j), or in other words, can distinguish between events at e_i and e_j.

4.2 A Graph Labeling Problem and Its Equivalence to the Scheduling Problem

After obtaining the bi-partite graph $\mathcal{G}(\mathcal{V} = \mathcal{X} \cup \mathcal{Y}, \mathcal{E})$ from a given network $G(V, E)$, we can re-write the detection and isolation scores as in (1) and (2) respectively in terms of $\mathcal{G}$. Note that if $S_i \subseteq \mathcal{X}$ is the subset of active monitoring devices in the ith time slot, then for the detection (isolation), the set of targets (target-pairs) covered by S_i is simply the neighborhood of set S_i, i.e., $N(S_i) = \bigcup_{x \in S_i} N(x)$. Here, $N(x)$ is the neighborhood of node x as defined in Section 2. Hence, for a given schedule $(S_1, S_2, \ldots, S_k)$ where k is the total number of time slots, the average detection (isolation) measure is simply $(1/k)\sum_{i=1}^{k} |N(S_i)|$. Thus, given a bi-partite graph $\mathcal{G}(\mathcal{X} \cup \mathcal{Y}, \mathcal{E})$, network life in terms of k time slots, and battery supply constraint in terms of σ time slots, the problem of finding an optimal schedule that maximizes the average detection (isolation) measure as described in Section 2.2 becomes equivalent to finding a set of k subsets $\{S_1, S_2, \ldots, S_k\}$, where $S_i \subseteq \mathcal{X}$, such that

$$\max_{\{S_1,\ldots,S_k\}} \sum_{j=1}^{k} |N(S_j)|, \tag{7}$$

and each node $x \in \mathcal{X}$ is included in at most σ such subsets.

The above problem can be cast as a graph labeling problem as described below.

Graph Labeling Problem. Let $\mathcal{K} = \{1, 2, \ldots, k\}$ be the set of labels, and $\mathcal{L}$ be the set of all σ-subsets[2] of $\mathcal{K}$. Note that $|\mathcal{L}| = \binom{k}{\sigma}$. We define

$$f: \mathcal{X} \longrightarrow \mathcal{L}, \tag{8}$$

i.e., f is a set function that assigns a subset of σ labels from $\mathcal{K}$ to each $x \in \mathcal{X}$. Also, for $y \in \mathcal{Y}$, we define $F(y)$ as follows:

$$F(y) \triangleq \bigcup_{x \in N(y)} f(x). \tag{9}$$

Note that $|F(y)|$ is simply the number of distinct labels available in the neighborhood of y. The objective is to obtain an assignment of labels to the nodes in $\mathcal{X}$ (i.e., (8)) such that

$$\text{Objective:} \quad \max_{f} \sum_{y \in \mathcal{Y}} |F(y)|. \tag{10}$$

Here, the objective is to assign σ labels to each node in $\mathcal{X}$ such that the sum of the number of distinct labels available in the neighborhood of y, $\forall y \in \mathcal{Y}$, is maximized. The scheduling problem in (7) and Section 2.2, is equivalent to the graph labeling problem described above.

Proposition 4.1. *The problem of obtaining an optimal schedule that maximizes the average detection (isolation) through a set of monitoring devices with limited battery supplies that cover a set of targets (target-pairs) for a given network lifetime, which is divided into k time slots, is equivalent to the graph labeling problem as defined in Equations (8), (9), and (10).*

Proof. In the graph labeling problem, let the subset of labels assigned to the vertex x, i.e., $f(x) \in \mathcal{L}$, corresponds to the indices of time slots in which the monitoring device corresponding to x is active. Since x has at most σ distinct labels by the definition of f, the monitoring device corresponding to node x can be active in at most σ time slots. Hence, the battery supply condition that requires a monitoring device to be active in at most σ time slots, is always satisfied. Moreover, $F(y)$ indicates time slots in which the target (target-pair) $y \in \mathcal{Y}$ remains covered by some $x \in \mathcal{X}$. Then, $(1/k)\sum_{y \in \mathcal{Y}} |F(y)|$ is simply the average detection (isolation) measure. The set of vertices that have label i correspond to the monitoring devices active in the ith time slot, i.e., S_i. Thus, finding a labeling (8) that maximizes (10) is basically finding a schedule $(S_1, S_2, \ldots, S_k)$ that maximizes the average detection (isolation) measure. □

2. The cardinality of each subset is σ, where σ is some positive integer.

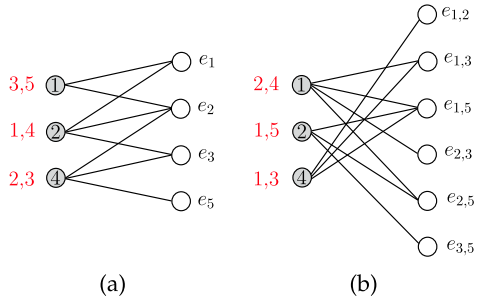


Fig. 2. Graph labelings for $\mathcal{K} = \{1, 2, \ldots, 5\}$ and $\sigma = 2$. Node labels, i.e., $f(x)$ are shown in red.

An illustration of the graph labeling for the scheduling problem is given below.

Example. In Fig. 2, instances of optimal labeling of graphs in Figs. 1b and 1c are shown for $\mathcal{K} = \{1, 2, \ldots, 5\}$ and $\sigma = 2$. Here $|\mathcal{K}| = 5$ means that the given network lifetime spans five time slots. Each node x has at most two labels, which represents that a node can be active in at most two time slots. The node labels indicate time slots in which they remain active, thus, giving us optimal schedules. Here, the optimal detection score is 0.75, which could be obtained with the schedule $S_1 = S_4 = \{2\}, S_2 = \{4\}, S_3 = \{1, 4\}, S_5 = \{1\}$. Similarly, the optimal isolation score is 0.633, which could be obtained with the schedule $S_1 = \{2, 4\}, S_2 = \{1\}, S_3 = \{4\}, S_4 = \{1\}, S_5 = \{2\}$.

5 Solutions to the Graph Labeling

In this section, first, we discuss the random assignment of labels to nodes, and then provide two improved solutions to the graph labeling problem. The first one is a simple greedy heuristic, whereas, the second solution utilizes game-theoretic concepts. The greedy heuristic runs in polynomial time, and gives a near optimal solution for many practical networks as illustrated in the next section. However, in general, the approximation ratio of the algorithm is not known. On the other hand, the game-theoretic solution guarantees probabilistic convergence to a globally optimal solution if the algorithm is run for a sufficiently large number of iterations.

The simplest way to label a graph is to randomly assign σ labels to each node from a set of k label. The scheduling thus, obtained is the *random scheduling*. As expected, the detection (localization) performance of random scheduling is far from being optimal. However, it can be useful in

applications where information regarding the network structure is not available. In fact, we can compute the detection performance $\mathcal{D}$ due to random scheduling for random geometric and Erdős-Rényi random networks as follows:

Proposition 5.1. *Let $G(V, E)$ be a random geometric graph in which each node contains a monitoring device that remains active in σ time slots that are randomly chosen from a total of k time slots, which correspond to the overall lifetime of the network. If each node in a graph is also a target, then the average detection performance of this random scheduling is*

$$\mathcal{D}(G) = 1 - \frac{(k-\sigma)}{k}\exp\left(\frac{-\sigma\lambda\pi r^2}{k}\right), \tag{11}$$

where r is the radius of the sensing footprint of node, and λ is the number of nodes per unit area.

Proof. The average detection performance is equivalent to finding the probability that an arbitrary node u is covered in an arbitrary time slot i. We observe that

$$\begin{aligned}&\Pr\,(u \text{ is not covered in the } i\text{th slot})\\ &= \Pr\begin{pmatrix}u \text{ is not active}\\ \text{in the } i\text{th slot}\end{pmatrix}\prod_{v\in N(u)}\Pr\begin{pmatrix}v \text{ is not active}\\ \text{in the } i\text{th slot}\end{pmatrix}.\end{aligned} \tag{12}$$

Here, probability that u is not active in the ith time slot is simply $(k-\sigma)/\sigma$. The second term in (12) is the probability that none of the nodes in the neighborhood of node u is active in the ith time slot. The probability of having j neighbors in $N(u)$ in a random geometric graph is given by Poisson distribution, i.e., $\frac{(\lambda\pi r^2)^j e^{-\lambda\pi r^2}}{j!}$. Thus,

$$\begin{aligned}&\prod_{v\in N(u)}\Pr\,(v \text{ is not active in the } i\text{th slot})\\ &\qquad= \sum_{j=0}^{\infty}\frac{(\lambda\pi r^2)^j e^{-\lambda\pi r^2}}{j!}\left(\frac{k-\sigma}{k}\right)^j.\end{aligned} \tag{13}$$

Inserting in (12), we get

$$\begin{aligned}&\Pr\,(u \text{ is not covered in the } i\text{th slot})\\ &= e^{-\lambda\pi r^2}\frac{(k-\sigma)}{k}\sum_{j=0}^{\infty}\frac{1}{j!}\left(\frac{\lambda\pi r^2(k-\sigma)}{k}\right)^j\\ &= e^{-\lambda\pi r^2}\frac{(k-\sigma)}{k}e^{\frac{\lambda\pi r^2(k-\sigma)}{k}} = \left(\frac{k-\sigma}{k}\right)e^{\frac{-\sigma\lambda\pi r^2}{k}}.\end{aligned} \tag{14}$$

The desired result follows directly from above. □

As above, it can be shown that in the case of Erdős-Rényi random graphs with n nodes, denoted by $G_{n,p}$, in which any two nodes are adjacent with some probability p, this random scheduling scheme results in an average detection performance given by

$$\mathcal{D}(G_{n,p}) = 1 - \frac{(k-\sigma)}{k}\exp\Big(\frac{-\sigma}{k}np\Big). \tag{15}$$

Note that in (15), it is assumed that all the nodes have monitoring devices and all the nodes need to be covered.

5.1 Greedy Heuristic

The graph labeling problem closely resembles the set covering problem, since we have to 'cover' the set of targets using a set of monitoring nodes, each of which could cover a given subset of the targets. Since the straightforward greedy algorithm is known to be an efficient approximation algorithm for the set covering problem, we can expect it to perform well for the graph labeling problem also. Hence, we formulate a simple greedy heuristic for the graph labeling problem as follows (Algorithm 1): For a given labeling set $\mathcal{K}$ and σ, iteratively select a combination of a label in $\mathcal{K}$ and a source node in $\mathcal{X}$ that maximizes the sum of number of distinct labels available in the neighborhoods of all target nodes in $\mathcal{Y}$. Note that in each iteration, only a source node with less than σ labels could be selected.

Algorithm 1. Greedy Heuristic

Given: σ, $\mathcal{K} = \{1, 2, \ldots, k\}$
Initialization: $\mathcal{X}' \leftarrow \mathcal{X}$, $f(x) \leftarrow \emptyset$, $\forall x \in \mathcal{X}$
While $|\mathcal{X}'| \neq \emptyset$ **do**
  $(x, \ell) \leftarrow \arg\max_{x\in\mathcal{X}', \ell\in\mathcal{K}} \sum_{y\in\mathcal{Y}}\Big|\bigcup_{x\in N(y)} f(x)\Big|$
  $f(x) \leftarrow f(x) \cup \{\ell\}$
  If $|f(x)| = \sigma$ **do**
    $\mathcal{X}' \leftarrow \mathcal{X}' \setminus \{x\}$
  End If
End While

If n is the total number of source nodes, m be the number of target nodes, and k be the total number of labels in the labeling set, then greedy heuristic could be executed in at most $O(\sigma k n^2 m)$ time as there are $O(\sigma n)$ iterations and each iteration could take $O(knm)$ time. Greedy heuristic gives a simple strategy to solve the labeling problem, however, its approximation ratio remains unknown. Therefore, we present a game-theoretic solution by posing the labeling problem as a potential game.

5.2 Game Theoretic Solution to the Graph Labeling

Game theory concepts have been extensively employed to solve locational optimization problems, such as maximizing coverage on graphs (e.g., [7], [8]) and distributed control of multiagent systems (e.g., [9], [10]). In a particular approach, the idea is to determine a *potential function* that captures the overall global objective. The players' individual utility functions are then appropriately aligned with the global objective, such that the change in the utility of the player as a result of unilateral change in strategy equals the change in the global utility represented by the potential function. The players' strategies are then designed to ensure that local actions lead to the global objective. It turns out that this problem formulation and design can be realized using a class of non-cooperative games known as *potential games*, which are now extensively used for various distributed control optimization problems.

A finite strategic game $\Gamma(\mathcal{P}, \mathcal{A}, \mathcal{U})$ consists of a set of players $\mathcal{P} = \{1, 2, \ldots, n\}$, action space $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2 \times \cdots \times \mathcal{A}_n$ where $\mathcal{A}_x$ is a finite action set of the player $x \in \mathcal{P}$, and a set of utility functions $\mathcal{U} = \{\mathcal{U}_1, \mathcal{U}_2, \ldots, \mathcal{U}_n\}$ where $\mathcal{U}_x : \mathcal{A} \to \mathbb{R}$ is a utility function of the player x. If $a = (a_1, \ldots, a_x, \ldots, a_n) \in \mathcal{A}$ denotes the joint action profile, we let a_{-x} denote the action of players other than the player x. Using this notation, we can also represent a as (a_x, a_{-x}).

A game is a *potential game* if there exists a *potential function*, $\phi : \mathcal{A} \to \mathbb{R}$ such that the change in the utility of the

player x as a result of a unilateral deviation from an action profile (a_x, a_{-x}) to (a'_x, a_{-x}) is equal to the corresponding change in the potential function. More precisely, for every player x, $a_x, a'_x \in \mathcal{A}_x$, and $a_{-x} \in \mathcal{A}_{-x}$, we get

$$\mathcal{U}_x(a_x, a_{-x}) - \mathcal{U}_x(a'_x, a_{-x}) = \phi(a_x, a_{-x}) - \phi(a'_x, a_{-x}). \quad (16)$$

In the case of potential games, there exist algorithms, such as log-linear learning (LLL) [11], [12] and binary log-linear learning (BLLL) [13] that could be utilized to drive the players to action profiles that maximize the potential function. These algorithms embody the notion of convergence of such games to the most efficient Nash equilibrium, particularly in scenarios where utility functions are designed to ensure that the action profiles that maximize the global objective of the system coincide with the potential function maximizers [11], [13]. More precisely, in potential games, LLL and BLLL algorithms guarantee that only the joint action profiles that maximize the potential function are stochastically stable [13]. It roughly means that in the long run, we are almost certain to get a solution that is in the small neighborhood of an optimal solution as the noise parameter in the algorithm goes to zero [14]. The LLL and BLLL are in fact, *nosiy best-response* algorithms that induce a Markov chain over the action space with a unique limiting distribution that depends on the noise parameter. As the noise parameter reduces to zero, the limiting distribution has a large part of its mass over the set of potential maximizers (see e.g., [11], [13], [15] for details).

The basic idea behind these algorithms is that the noise parameter allows selecting suboptimal actions occasionally by the players. The probability of selecting a suboptimal action is dependent of the pay-off difference between the optimal and suboptimal cases. Thus, formulating the graph labeling problem as a potential game would allow us to use the above mentioned learning algorithms to find the most efficient solutions to the graph labeling problem. Thus, our objective now is to design a potential game corresponding to the labeling problem on graphs, and incorporate learning algorithms for the potential games to achieve the desired labeling.

5.2.1 A Potential Game for the Graph Labeling

We design a potential game $\Gamma(\mathcal{P}, \mathcal{A}, \mathcal{U})$ to obtain a labeling of a graph that achieves the objective in (10), thus solving the scheduling problem. In our game, the set of players is the vertex set $\mathcal{X}$ in the vertex partition $(\mathcal{V} = \mathcal{X} \cup \mathcal{Y})$ of the bipartite graph $\mathcal{G}$, i.e., $\mathcal{P} = \mathcal{X}$. For each player $x \in \mathcal{X}$, the action set $\mathcal{A}_x$ is the set of all σ-subsets of the labeling set $\mathcal{K} = \{1, \ldots, k\}$. We also need to have a potential function that captures the global objective. For this, we define S_j as the set of vertices with the label j, i.e.,

$$S_j = \{x \in \mathcal{X} : j \in f(x)\}. \quad (17)$$

A potential function is then defined as

$$\phi(a) \triangleq \sum_{j=1}^{k} \left| \bigcup_{x \in S_j} N(x) \right|. \quad (18)$$

Note that $\phi(a)$ is simply the total number of nodes in $\mathcal{Y}$ having a label $j \in \mathcal{K}$ in their neighborhoods, summed over all the labels, which is equivalent to the $\sum_{y \in \mathcal{Y}} |F(y)|$ in (10). Thus, $\phi(a)$ indeed captures the global objective. Moreover, we define the utility function of the player x as follows:

$$U_x(a_x, a_{-x}) \triangleq \sum_{j=1}^{k} a_{xj} \left| N(x) \setminus \bigcup_{z \in S_j \setminus \{x\}} N(z) \right|, \quad (19)$$

where,

$$a_{xj} = \begin{cases} 1 & \text{if } j \in a_x (= f(x)) \\ 0 & \text{otherwise.} \end{cases}$$

Note that if $y \in N(x)$, then the value of a_x to node y can be computed by counting the number of labels in a_x that are not assigned to any node in $N(y) \setminus \{x\}$. The utility of a_x is simply the sum of these values for all $y \in N(x)$. For instance, in Fig. 2a, node 1 has labels $\{3, 5\}$, which represents the action a_1. Moreover, node 1 has two neighbors, e_1 and e_2. Since node 1 is the only node in $N(e_1)$ with labels 3 and 5, the value of a_1 to node e_1 is 2. Similarly, for e_2, node 1 is the only node in $N(e_2)$ with label 5, hence, the value of a_1 to e_2 is 1. The utility of a_1 is simply the sum of these values, that is $U_1(a_1, a_{-1}) = 2 + 1 = 3$.

Next, we show that with the potential function as defined in (18), and the utility function as in (19), the game designed above is indeed a potential game.

Theorem 5.2. *$\Gamma(\mathcal{P}, \mathcal{A}, \mathcal{U})$ is a potential game if utilities are defined as in (19).*

Proof. The potential function, as defined in (18) can be written as,

$$\begin{aligned} \phi(a_x, a_{-x}) &= \sum_{j=1}^{k} \left| \bigcup_{x \in S_j} N(x) \right| \\ &= \sum_{j=1}^{k} \left(a_{xj} \left| N(x) \setminus \bigcup_{z \in S_j \setminus \{x\}} N(z) \right| + \left| \bigcup_{z \in S_j \setminus \{x\}} N(z) \right| \right) \\ &= \sum_{j=1}^{k} a_{xj} \left| N(x) \setminus \bigcup_{z \in S_j \setminus \{x\}} N(z) \right| + \sum_{j=1}^{k} \left| \bigcup_{z \in S_j \setminus \{x\}} N(z) \right| \\ &= \mathcal{U}(a_x, a_{-x}) + \sum_{j=1}^{k} \left| \bigcup_{z \in S_j \setminus \{x\}} N(z) \right|. \end{aligned} \quad (20)$$

Similarly, for $a = (a'_x, a_{-x})$, we get

$$\phi(a'_x, a_{-x}) = \mathcal{U}(a'_x, a_{-x}) + \sum_{j=1}^{k} \left| \bigcup_{z \in S_j \setminus \{x\}} N(z) \right|. \quad (21)$$

Subtracting (21) from (20) gives us the desired result, i.e.,

$$\phi(a_i, a_{-i}) - \phi(a'_i, a_{-i}) = U(a_i, a_{-i}) - U(a_i, a_{-i}).$$

□

Using the results in [13], we deduce that in our setup if players adhere to the binary log-linear learning (stated below), then the action profiles that are stochastically stable are the ones that maximize the potential function (18). In other words, in the long run, we achieve a graph labeling that maximizes the objective in (10) with arbitrarily high probability.

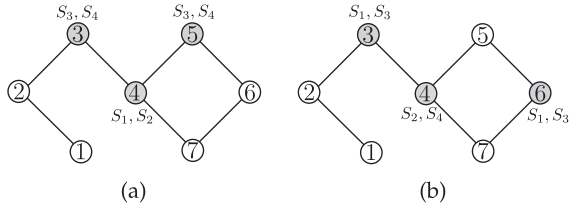


Fig. 3. (a) Optimal schedule for a given placement. (b) Optimal placement and schedule of three monitoring devices with $\lambda = 1, \sigma = 2$ for $k = 4$.

Algorithm 2. Binary Log-Linear Learning [13]

1: **Initialization:** Pick a small $\epsilon \in \mathbb{R}_+$, an $a \in \mathcal{A}$, and total number of iterations.
2: **While** $i \leq$ number of iterations **do**
3: Pick a random node $x \in \mathcal{X}$, and a random $a'_x \in \mathcal{A}_x$.
4: Compute $P_\epsilon = \frac{\epsilon^{-U_x(a'_x, a_{-x})}}{\epsilon^{-U_x(a'_x, a_{-x})} + \epsilon^{-U_x(a_x, a_{-x})}}$.
5: Set $a_x \leftarrow a'_x$ with probability P_ϵ.
6: $i \leftarrow i + 1$
7: **End While**

Note that initially each node is assigned a set of σ labels randomly. Afterwards, in each iteration, a node is selected at random, and a set of σ labels that improve the overall labeling to attain the objective in (10), is assigned to the node with a certain probability (as in line 4 above).

6 Simultaneous Placement and Scheduling of Monitoring Devices

So far, we have considered optimal scheduling of resource bounded monitoring devices, assuming that their placement is fixed, i.e., locations at which monitoring devices are deployed are given. If $\mathcal{S}$ is the set of all such nodes at which monitoring devices could be deployed, then the *placement problem* is to select a subset $\mathcal{X} \subseteq \mathcal{S}$ with the given cardinality such that the number of covered targets (target-pairs in the case of isolation) is maximized. Typically, to maximize the coverage of targets for a given network lifetime, the placement problem is first solved, followed by the computation of efficient schedules.

However, for a given network lifetime, and a fixed number of resource bounded monitoring devices, simultaneously optimizing their placement and scheduling could further improve the average detection (isolation) measure. For instance, consider the network in Fig. 3, in which three monitoring devices with $\lambda = 1$ and $\sigma = 2$ are deployed to cover the maximum number of nodes for $k = 4$. Fixing the placement of devices at nodes $\{3, 4, 5\}$, optimal schedule (for instance, $S_1 = S_2 = \{4\}, S_3 = S_4 = \{3, 5\}$) gives $\mathcal{D} = 0.642$, whereas the maximum possible $\mathcal{D}$ under the conditions is 0.714, which could be obtained by placing the devices at nodes $\{3, 4, 6\}$ and with a schedule $S_1 = S_3 = \{3, 6\}, S_2 = S_4 = \{4\}$.

The BLLL based algorithm to schedule a set of monitoring devices with fixed locations, presented in Section 5.2, can be modified to simultaneously optimize placement as well as scheduling of such devices to maximize the average detection (isolation). This modification is presented as Algorithm 3. Fixing the number of monitoring devices $|\mathcal{X}|$, the objective is to *select $\mathcal{X} \subseteq \mathcal{S}$, and assign at most σ labels to each node from a labeling set $\mathcal{K} = \{1, 2, \ldots, k\}$ so that the average detection measure $\mathcal{D}$ (or the isolation measure $\mathcal{I}$) is maximized*. The labeling of nodes selected in $\mathcal{X}$ will then give the schedule. As previous, we can formulate this problem as a potential game, and can thus, solve the problem using the BLLL algorithm.

6.1 A Potential Game Formulation

In this case, players $\mathcal{P}$ are the monitoring devices, for which we need to find the locations - the nodes at which they are deployed; as well as schedules - time slots in which they become active. The action of each player p, denoted by a_p is the selection of $(x_p, f(x_p))$, where $x_p \in \mathcal{S}$ and $f(x_p) \in \mathcal{L}$. Note that $\mathcal{L}$ is the set of all subsets of the labeling set $\mathcal{K}$ containing σ labels (as defined in (8)). Moreover, as in (17), let S_j to be the subset of nodes (containing monitoring devices) with label j, that is

$$S_j = \{x_p \in \mathcal{S} : j \in f(x_p)\}. \tag{22}$$

Next, similar to (19), we define the utility function of the player p as

$$U_p(a_p, a_{-p}) = \sum_{j=1}^{k} a_{pj} \left| N(x_p) \setminus \bigcup_{z \in S_j \setminus \{x_p\}} N(z) \right|, \tag{23}$$

where,

$$a_{pj} = \begin{cases} 1 & \text{if } j \in f(x_p) \\ 0 & \text{otherwise.} \end{cases}$$

If f or each target node $y \in \mathcal{Y}$, we define $F(y)$ (similar to (9)) as $F(y) = \sum_{x_p \in N(y)} f(x_p)$, then our global objective is to select a subset $\mathcal{X} \subseteq \mathcal{S}$ and assign σ labels to each $x_p \in \mathcal{X}$ such that

$$\text{Objective:} \quad \max_{\mathcal{X}, f} \sum_{y \in \mathcal{Y}} |F(y)|. \tag{24}$$

A potential function that captures the above objective is

$$\phi(a) = \sum_{j=1}^{k} \left| \bigcup_{x_p \in S_j} N(x_p) \right|. \tag{25}$$

Here a represents the actions of all players, that is $a = (a_1, a_2, \ldots, a_{|\mathcal{X}|})$.

Using exactly the same argument as in the proof of Theorem 5.2, we can state the following.

Proposition 6.1. *The game described in Section 6.1 is a potential game with the utility and potential functions defined as in (23) and (25) respectively.*

Algorithm 3. Simultaneous Placement and Scheduling

1: **Initialization:** Pick a small $\epsilon \in \mathbb{R}_+$ and the number of iterations. Select randomly a subset of nodes $\mathcal{X} \subseteq \mathcal{S}$, and assign labels to nodes in $\mathcal{X}$, i.e., select $a \in \mathcal{A}$.
2: **While** $i \leq$ number of iterations **do**
3: Randomly select a node $x \in \mathcal{X}$.
4: Randomly select a node $s \in (\mathcal{S} \setminus \mathcal{X}) \cup \{x\}$, and $a_s \in \mathcal{A}_s$.
5: Compute $P_\epsilon = \frac{\epsilon^{-U_s(a_s, a_{-x})}}{\epsilon^{-U_s(a_s, a_{-x})} + \epsilon^{-U_x(a_x, a_{-x})}}$.
6: With probability P_ϵ, set $\mathcal{X} \leftarrow (\mathcal{X} \setminus \{x\}) \cup \{s\}$, and select a_s for node s.
7: $i \leftarrow i + 1$
8: **End While**

Hence, using a binary log-linear learning, we get a solution that, as the number of iteration goes to infinity, selects nodes at which monitoring devices can be placed, as well as their schedules that achieve the maximum average detection performance. We note that the placement and scheduling of monitoring devices obtained by first optimally solving the placement problem and then optimally solving the scheduling to maximize the detection performance $\mathcal{D}$, is also a solution of the problem of simultaneously placing and scheduling monitoring devices to maximize $\mathcal{D}$. As a result an optimal solution of the simultaneous placement and scheduling problem gives a detection performance that is at least as good as the detection performance obtained by separately solving the optimal placement and the optimal scheduling problems. Simulation results for the above algorithm are illustrated in Section 8.3. Using various networks, it is shown that simultaneously selecting locations and schedules of monitoring devices using Algorithm 3, gives improved average detection compared to the one obtained by solving the placement and scheduling separately.

7 Scheduling to Maximize Network Lifetime While Ensuring Complete Coverage

So far we have studied the problem of finding schedules maximizing the detection performance $\mathcal{D}$ given the battery and overall network lifetime σ and k respectively. A relevant problem of interest is to compute schedules of monitoring devices that maximize the network lifetime k for a fixed σ and $\mathcal{D} = 1$, that is schedules ensuring *complete coverage*. Considering targets to be the set of nodes (i.e., $\mathcal{Y} = \mathcal{V}$) and ranges of monitoring devices to be $\lambda = 1$, the optimal scheduling problem is very much related to finding distinct *dominating sets* in a graph, where dominating sets are defined as following.

Definition. *A dominating set is a subset of vertices in a graph $S_i \subseteq V$, such that for every $u \in V$, either $u \in S_i$, or there exists some $v \in S_i$ such that $v \in N(u)$.*

Note that the network is guaranteed to be completely covered whenever the set of nodes with active monitoring devices form a dominating set. Thus, the objective here is to compute distinct dominating sets in a given graph. Moreover, since a monitoring device can be active in at most σ time slots, it can be included in at most σ dominating sets. As a result, the scheduling problem to maximize network lifetime given σ and complete coverage constraint is equivalent to computing the maximum number of distinct dominating sets in a network graph under the condition that a node can be included in at most σ such dominating sets. Owing to a wide variety of applications, finding distinct dominating sets under various constraints has been a problem of great interest (e.g., [16], [17], [18]). There are two approaches to obtaining distinct dominating sets: *disjoint*, and *non-disjoint* dominating sets based approaches.

In the *disjoint dominating sets* based approach, the objective is to partition the vertex set $\mathcal{V}$ into a maximum number of (disjoint) subsets such that each subset in the partition is a dominating set. Such a partition is called the *maximum domatic partition (MDP)*, and the size of the partition, that is the number of disjoint dominating sets obtained, is referred to as the *domatic number*, γ. For a given σ, nodes in each dominating set can remain active for σ time slots, thus, achieving a network lifetime of k time slots given by,

$$k = \sigma\gamma. \tag{26}$$

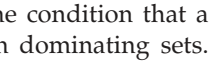
The MDP problem is known to be NP-hard [19]. Various sensor scheduling schemes based on MDP have been proposed in literature (e.g., [18], [20], [21]).

Is it possible to achieve a network lifetime better than $\sigma\gamma$? The answer is yes, that is by using a *non-disjoint dominating sets* based approach [2], [22]. In this approach, the goal is to obtain the maximum number of subsets $S_i \subset \mathcal{V}$ such that each S_i is a dominating set and each $v \in \mathcal{V}$ is included in at most σ dominating sets. Unlike MDP based approach, dominating sets obtained here do not have to be disjoint. The problem of finding the maximum number of dominating sets with a restriction on the number of times a node can be included in a dominating set is related to the notion of (k, σ)-configurations [23], [24] defined below.

Definition ((k, σ)-Configurations in Graphs). *Let σ, k be two positive integers, and $\mathcal{K} = \{1, \ldots, k\}$ be the set of labels, then (k, σ)-configuration of a graph is the assignment of σ distinct labels from the set $\mathcal{K}$ to each node in the graph such that for every $i \in \mathcal{K}$ and every node v, the label i is assigned to v or one of its neighbors.*

Note that in a (k, σ)-configuration, the set of nodes corresponding to a particular label in $\mathcal{K}$ constitutes a dominating set. For a given σ, we denote the maximum value of k for which (k, σ)-configuration exists by k^*. Consequently, the scheduling problem to maximize the network lifetime while ensuring complete coverage is equivalent to computing (k^*, σ)-configuration of the network graph. From a MDP of a graph, it is trivial to obtain a $(\sigma\gamma, \sigma)$-configuration, that is by assigning σ unique labels to each dominating set in MDP, we deduce that $k^* \geq \sigma\gamma$. In other words, non-disjoint dominating sets based approach is always at least as good as the disjoint dominating sets based approach. In fact, for many graphs $k^* > \sigma\gamma$, for instance, many *cubic graphs*[3] have $\gamma = 2$, however, all cubic graph have $k^* \geq \frac{5}{2}\sigma$ for a given σ [23]. Recently, in [24] we have extended this result to a bigger class of graphs as stated in Theorem 7.1. Here, $K_{1,6}$ is a star graph with one central node of degree six, and six end nodes each with a degree one ($K_{1,6} =$ ).

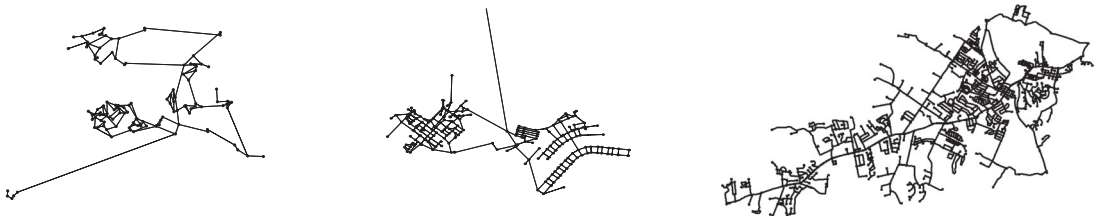
Theorem 7.1 ([24]). *Let G be a graph such that*

- *G has a minimum degree at least two,*
- *no subgraph of G is isomorphic to $K_{1,6}$, and*
- *$G \neq \{$*  *$\}$;*

then G has an (k, σ)-configuration with $k = \lfloor \frac{5\sigma}{2} \rfloor$.

The above result is particularly useful as proximity graphs (e.g., random geometric graphs), which are often used to model the limited range communication in networks such as wireless sensor networks, are always $K_{1,6}$-free. As a result, if we consider proximity networks modeled by the graphs in Theorem 7.1, and consider B as the time duration for which each monitoring device (placed at each node) can remain active, then it is always possible to compute schedules

3. Graphs in which each vertex has a degree three.



(a) Water network 1 (b) Water network 2 (c) Water network 3

Fig. 4. Layouts of three water networks considered.

through which complete coverage of targets (nodes) is ensured for at least $\lfloor \frac{5B}{2} \rfloor$ time duration.

8 Numerical Results

In this section, we present numerical results for the greedy and BLLL based algorithms on urban water distribution networks and random geometric networks.

8.1 Scheduling Monitoring Devices in Water Distribution Networks

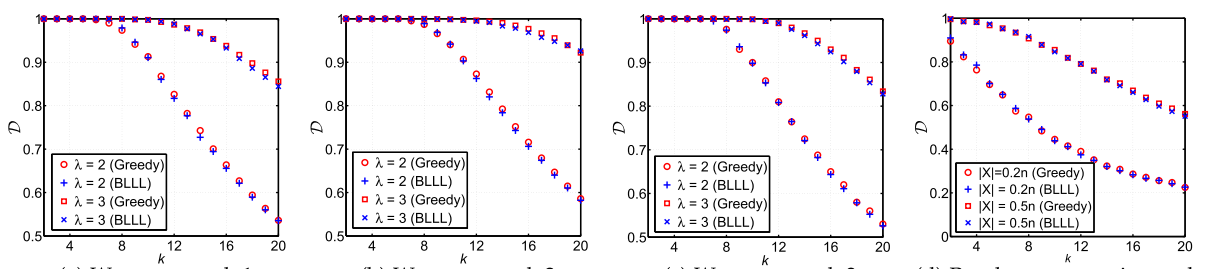
Water distribution networks can be modeled as undirected graphs in which edges represent the pipes and nodes represent the junctions (e.g., [25]). To detect pipe bursts and leakages, pressure sensors are deployed at junctions, which could sense the pressure transient generated as a result of pipe burst within a certain distance (range) from the sensor. The distance threshold based model has been used in water networks in the context of sensor placement problems, e.g., [26], [27]. The pressure sensors are battery operated devices with limited battery lifetime. Thus, to operate these sensors for an extended period of time, they need to be scheduled. Here, we simulate scheduling algorithms, including simple greedy and BLLL based algorithm for the efficient scheduling of monitoring devices, which are pressure sensors in this case, to obtain high values of $\mathcal{D}$ in three water distribution networks of various sizes. The details of these networks, referred to as the *Water Network 1, Water Network 2*, and *Water Network 3*, are as follows:

Water Network 1 [28], [29], often used as a benchmark network in the context of sensor placement problems for water quality, has 126 nodes, 168 pipes, one reservoir, one pump, and two storage tanks. Water network 2 [30] is a grid system in Kentucky with 366 pipes, 270 nodes, three tanks, and five pumps. Water network 3 [30] is primarily a loop system in Kentucky with four tanks, two pumps, 1,156 pipes, and 962 nodes. The layouts of all three networks are illustrated in Fig. 4. For all the networks, we consider that the sensors are deployed at the junctions as source nodes $\mathcal{X}$ (monitoring devices), and the set of pipes, which are edges in the corresponding network graph, as targets $\mathcal{Y}$. Moreover, for each sensing device, we assume $\sigma = 2$, and compute $\mathcal{D}$ for a network lifetime, given by k time slots, using greedy and BLLL algorithms. For each BLLL instance, we perform 25,000 iterations by selecting ϵ to be 0.015. The plots of $\mathcal{D}$ as a function of k for various ranges of sensing devices (as defined in Section 2) are given in Fig. 5.

We can see that both greedy and BLLL gives approximately same results. However, BLLL has an advantage over the greedy algorithm as it allows to simulatneously solve the placement as well as scheduling problem (as discussed in Section 6), which gives improved $\mathcal{D}$ compared to individually solving the placement problem and the scheduling problem. Moreover, if BLLL is run for sufficiently large number of iterations, the algorithm achieves an optimal solution with a very high probability. Similar plots can be obtained for the scheduling of monitoring devices to maximize the average isolation measure $\mathcal{I}$ by first obtaining the appropriate network representation as outlined in Section 4.1. In Figs. 6 and 7, we illustrate the performance of BLLL algorithm for all three water networks by plotting $\mathcal{D}$ as a function of iterations. In Fig. 6, we consider various values of k and observe that after a sufficient number of iterations, the algorithm maintains optimal values with high probability. In Fig. 7, we see a similar behavior for various values of σ.

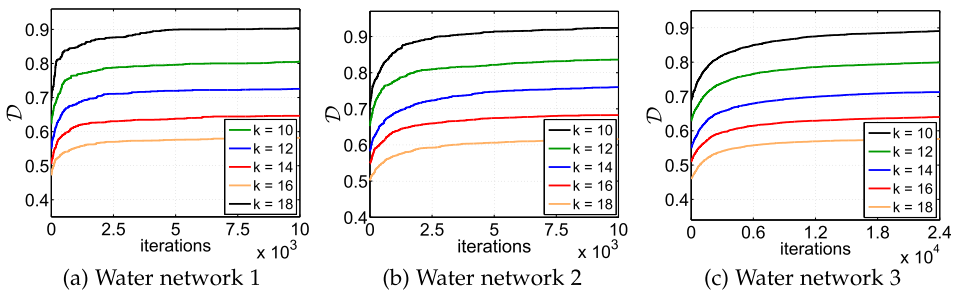
8.2 Scheduling Monitoring Devices in Random Geometric Networks

Random geometric networks are a form of spatial networks in which nodes are deployed uniformly at random in a certain area. An edge exists between two nodes if the euclidean distance between them is at most r, which is often referred to as the *radius of the sensing footprint*. Owing to a wide variety of applications in various domains, these networks have



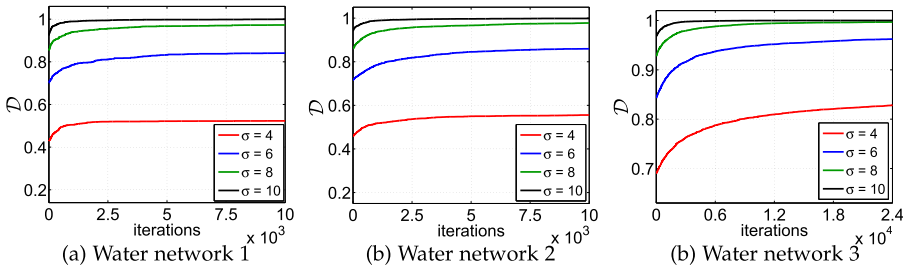
(a) Water network 1 (b) Water network 2 (c) Water network 3 (d) Random geometric graphs

Fig. 5. Plots of $\mathcal{D}$ as a function of network lifetime k for scheduling on water networks and random geometric networks, assuming that each monitoring device has a battery lifetime of $\sigma = 2$ time slots.



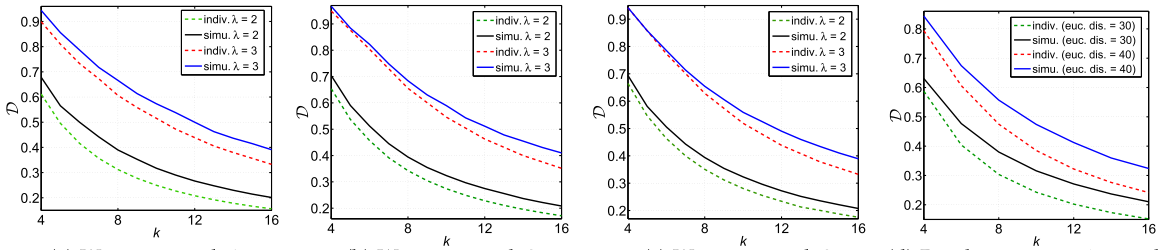
(a) Water network 1 (b) Water network 2 (c) Water network 3

Fig. 6. Plots of $\mathcal{D}$ as a function of (BLLL) iterations with $\sigma = 2$ and various k.



(a) Water network 1 (b) Water network 2 (b) Water network 3

Fig. 7. Plots of $\mathcal{D}$ as a function of (BLLL) iterations with $k = 20$ and various σ.



(a) Water network 1 (b) Water network 2 (c) Water network 3 (d) Random geometric graphs

Fig. 8. Plots of $\mathcal{D}$ as a function of k to compare the performance of simultaneously optimizing placement and schedules using Algorithm 3 with the case of individually optimizing the placement problem and the scheduling problem.

been extensively studied, such as in the modeling of wireless sensor networks. For our simulations, we consider a network with 100 nodes, deployed uniformly at random over an area of 10×10 unit^2, and $r = 2$. The set of targets here is the set of all nodes. Moreover, a certain fraction of nodes (either 20 or 50 percent) are selected at random as source nodes. A monitoring device has a battery lifetime of at most $\sigma = 2$ time slots, and can monitor targets that are at a euclidean distance of at most 2 units from it,[4] that is the radius of sensing footprint is 2 units. In Fig. 5, $\mathcal{D}$ as functions of k are illustrated using greedy and BLLL algorithms. Each point on the plots is an average of fifty randomly generated graph instances. In Fig. 6, the convergence of BLLL algorithm is shown for some instances of random geometric graphs with 100 nodes, out of which 20 randomly selected nodes contain monitoring devices.

8.3 Simultaneous Placement and Scheduling of Monitoring Devices

We illustrate the Algorithm 3 for the simultaneous placement and scheduling of monitoring devices for all three water networks and the random geometric graphs here. For the water networks, we consider that monitoring devices can be placed at twenty percent of the nodes, which need to be selected. Each monitoring device has a range λ, and we separately consider two cases of $\lambda = 2$ and $\lambda = 3$. The set of pipes (or edges in the corresponding network graph) are the targets that need to be covered by these devices. We simulate two scenarios; in the first case we use Algorithm 3 to simultaneously select the nodes and schedules for the monitoring devices; in the second scenario, we first solve the placement problem by selecting nodes $\mathcal{X} \subset V$ that maximize the number of edges that are at most distance λ from some node in $\mathcal{X}$, and then solving the scheduling problem using Algorithm 2. We note here that the placement problems, in this context, are typically solved using some variant of the minimum set cover problem, or the maximum coverage problem in case the number of monitoring devices is fixed (e.g., [4], [5], [31]). Since the number of devices is fixed here, and the targets to be covered are edges, we use the maximum coverage problem to place (a given number of) monitoring devices at nodes that maximize the number of edges that are at most λ distance from at least one of the selected nodes. Since maximum coverage problem is NP-hard, we solve it using a greedy heuristic, which gives the best approximation ratio, $(1 - 1/e)$[32].

The results are illustrated in Fig. 8. It can be seen that Algorithm 3 (simultaneously solving placement and

4. In terms of the (graph) distances as defined in Section 2, the range of each monitoring device is $\lambda = 1$, as the euclidean distance of at most 2 between two nodes u and v implies $d(u, v) = 1$.

scheduling) always gives higher average detection $\mathcal{D}$. For the random geometric graphs, we simulate instances of 500 nodes deployed at random in an area of 500×500 unit^2, out of which 100 could contain monitoring devices capable of covering nodes within a euclidean distance of 30 units in one case, and 40 units in the other. The targets here are nodes, and the objective is to maximize the average detection for a given network lifetime. As with the water networks, average detection is improved if placement and scheduling problems are solved simultaneously using Algorithm 3 as compared to optimizing placement and scheduling separately. In all cases, the battery lifetime of each monitoring device is assumed to be $\sigma = 2$ time slots.

9 Related Work

Mechanisms for detecting link and node failures based on system dynamics have been studied extensively in the literature. For example, Dhal et al. consider the detection of link failures in a network synchronization process from noisy measurements at a single network component [33]. As another example, Rahimian and Preciado propose a methodology to detect and isolate link failures in a weighted and directed network of identical multi-input multi-output LTI systems, based on the output responses of a subset of nodes [34]. However, since we are interested in the placement and scheduling of these devices instead of the specific detection mechanisms, our model abstracts away the specific mechanisms. In other words, our model assumes that monitoring devices are available to us, which can detect link and node failures based on some detection mechanism, such as the ones presented in the above papers.

One of the earliest efforts to conserve battery power through scheduling sensor devices is the work of Slijepcevic and Potkonjak [35]. In [35], the authors consider the problem of maximizing lifetime while preserving complete coverage of an area, which they formulate as the Set K-Cover Problem. To solve this problem, they introduce a heuristic for finding mutually exclusive sets of sensors such that each set completely covers the monitored area. In a follow-up work, Abrams et al. introduce three approximation algorithms for a variation of the Set K-Cover Problem [36]. Later, Deshpande et al. study several generalizations of the Set K-Cover Problem, and develop an approximation algorithm based on a reduction to Max K-Cut [37].

Besides the Set K-Cover Problem, researchers have studied various other formulations of the scheduling problem. Moscibroda and Wattenhofer consider disjoint dominating-set based clustering in sensor networks [20]. The authors study the problem of maximizing the lifetime of a sensor network, and provide approximation algorithms for multiple variations of the problem. Cardei et al. study schedules that consist of non-disjoint sets of sensors and continuously monitor all targets [2]. They model the solution as the maximum set covers problem, and propose two heuristics based on linear programming and a greedy approach. Koushanfar et al. consider the problem of scheduling sensor devices such that the values of sleeping devices can always be recovered from the measurements of active devices within a given error bound [38]. The authors first introduce a polynomial-time isotonic regression for recovering the values of sleeping devices, and building on this regression, they then formulate the scheduling problem as domatic partitioning problem, which they solve using an ILP solver.

Our approach is most related to the work of Wang et al., who study the trade-off between maximizing lifetime and minimizing "coverage breach," that is, minimizing the total amount of time that each target is not covered by any of the sensors [22], [39]. The authors propose organizing the sensors into non-disjoint sets, and introduce an algorithm based on linear programming as well as a greedy heuristic. In a follow-up work, Rossi et al. propose an exact approach based on a column-generation algorithm for solving the scheduling problem, and they also derive a heuristic from their approach [40]. However, graph-theoretic formulation proposed in this paper allows us to directly exploit the network structure to obtain optimal schedule for a given network lifetime maximizing the detection or identification of targets. Moreover, game theory based solution could be used to simultaneously solve the placement problem and the scheduling problem, which gives improved overall performance compared to the case in which the placement and scheduling problems are solved separately.

A few research efforts have considered simultaneous placement and scheduling. Krause et al. study simultaneous placement and scheduling of sensor devices for monitoring spatial phenomena, such as road traffic [41]. The authors assume that for any set of active sensors, the resulting "sensing quality" is given by a submodular function, and they aim to maximize the worst-case sensing quality. In the case of network monitoring, compared to this approach, our approach has the advantage of considering and taking advantage of the network topology. Türkoğulları et al. consider the problem of maximizing lifetime through sink placement, scheduling, and determining sensor-to-sink flow paths under energy, coverage, and budget constraints [42]. To solve this problem, they propose a mixed-integer linear programming model as well as a heuristic, which is more scalable but lacks performance guarantees. However, these approaches are constrained in the sense that a solution must monitor every target with a given quality in every single time step. Our approach, on the other hand, has more flexible constraints, which can result in much longer lifetime.

A number of studies have focused on the placement of sensor nodes, without considering sleep scheduling. Younis and Akkaya have surveyed earlier literature on node placement, including the placement of sensor nodes [43]. Krause et al. consider the problem of deploying sensors for detecting malicious contaminations in large-scale water-distribution networks [5]. Based on the submodularity of realistic objective functions, the authors design scalable placement algorithms with provable performance guarantees. Furthermore, they show that their method can be extended to multicriteria optimization and adversarial objectives. Hart and Murray provide a survey of sensor placement strategies for water-distribution networks [44]. Finally, besides scheduling, researchers have also studied other similar approaches for conserving battery power. For example, Zhao et al. consider selective collaboration of sensors in order to minimize communication, which increases the longevity of networks of battery-powered sensors [45].

10 Conclusion

We studied the problem of scheduling resource bounded monitoring devices in networks to maximize the detection and isolation of failure events for a given network lifetime. We showed that the scheduling problem is equivalent to a graph labeling problem, which allowed direct exploitation of the network structure to obtain optimal schedules. To solve the graph labeling problem, we presented a game-theoretic solution. We also showed that the detection (isolation) performance of monitoring devices deployed within the network was better when the placement and scheduling problems for these devices were solved simultaneously compared to the case in which the optimal placement of these devices was solved first followed by the computation of optimal schedules. Our graph labeling formulation and game-theoretic solution allowed us to simultaneously solve placement and scheduling problems. We demonstrated results for various networks including water distribution and random networks. The graph labeling problem presented here could be useful in solving resource allocation problems in other domains such as multi-agent and multi-robot systems. Moreover, the proposed approach could be effective in characterizing and comparing network topologies in terms of the coverage performance, especially when resource-constraint monitoring devices are utilized.

Acknowledgments

This work was supported in part by the US National Science Foundation (CNS-1238959, CNS-1640624), Air Force Research Laboratory (FA 8750-14-2-0180), Army Research Office (W911NF-16-1-0069), and by the Office of Naval Research (N00014-15-1-2621).

References

[1] M. Farley and S. Trow, *Losses in Water Distribution Networks: A Practitioner's Guide to Assessment, Monitoring and Control*. London, U.K.: The International Water Association (IWA) Publishing, 2003.

[2] M. Cardei, M. T. Thai, Y. Li, and W. Wu, "Energy-efficient target coverage in wireless sensor networks," in *Proc. 24th Annu. Joint Conf. IEEE Comput. Commun. Soc.*, 2005, pp. 1976–1984.

[3] B. Wang, "Coverage problems in sensor networks: A survey," *ACM Comput. Surveys*, vol. 43, no. 4, pp. 32:1–32:53, 2011.

[4] L. S. Perelman, W. Abbas, X. Koutsoukos, and S. Amin, "Sensor placement for fault location identification in water networks: A minimum test cover approach," *Automatica*, vol. 72, pp. 166–176, 2016.

[5] A. Krause, J. Leskovec, C. Guestrin, J. VanBriesen, and C. Faloutsos, "Efficient sensor placement optimization for securing large water distribution networks," *J. Water Resources Planning Manage.*, vol. 134, no. 6, pp. 516–526, 2008.

[6] C. Papadimitriou and M. Yannakakis, "Optimization, approximation, and complexity classes," *J. Comput. and Syst. Sci.*, vol. 43, no. 3, pp. 425–440, 1991.

[7] A. Y. Yazicioglu, M. Egerstedt, and J. S. Shamma, "A game theoretic approach to distributed coverage of graphs by heterogeneous mobile agents," in *Proc. FAC Workshop Distrib. Estimation Control Netw. Syst.*, 2013, vol. 4, pp. 309–315.

[8] M. Zhu and S. Martínez, "Distributed coverage games for energy-aware mobile sensor networks," *SIAM J. Control Optimization*, vol. 51, no. 1, pp. 1–27, 2013.

[9] G. Arslan, J. R. Marden, and J. S. Shamma, "Autonomous vehicle-target assignment: A game-theoretical formulation," *J. Dyn. Syst. Meas. Control*, vol. 129, no. 5, pp. 584–596, 2007.

[10] I. Menache and A. Ozdaglar, "Network games: Theory, models, and dynamics," *Synthesis Lectures Commun. Netw.*, vol. 4, no. 1, pp. 1–159, 2011.

[11] L. E. Blume, "The statistical mechanics of strategic interaction," *Games Econ. Behavior*, vol. 5, no. 3, pp. 387–424, 1993.

[12] W. A. Brock and S. N. Durlauf, "Discrete choice with social interactions," *Rev. Econ. Studies*, vol. 68, no. 2, pp. 235–260, 2001.

[13] J. Marden and J. Shamma, "Revisiting log-linear learning: Asynchrony, completeness, and pay-off based implementation," *Games Econ. Behavior*, vol. 75, no. 2, pp. 788–808, 2012.

[14] H. P. Young, "The evolution of conventions," *Econometrica: J. Econometric Soc.*, vol. 61, pp. 57–84, 1993.

[15] A. Y. Yazicioglu, M. Egerstedt, and J. S. Shamma, "Communication-free distributed coverage for networked systems," *IEEE Trans. Control Netw. Syst.*, 2016, doi: 10.1109/TCNS.2016.2518083.

[16] N. Ahn and S. Park, "A new mathematical formulation and a heuristic for the maximum disjoint set covers problem to improve the lifetime of the wireless sensor network," *Ad Hoc Sensor Wireless Netw.*, vol. 13, no. 3/4, pp. 209–225, 2011.

[17] S. Henna and T. Erlebach, "Approximating maximum disjoint coverage in wireless sensor networks," in *Ad-Hoc, Mobile, and Wireless Network*. Berlin, Germany: Springer, 2013, pp. 148–159.

[18] S. V. Pemmaraju and I. A. Pirwani, "Energy conservation via domatic partitions," in *Proc. 7th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2006, pp. 143–154.

[19] U. Feige, M. M. Halldórsson, G. Kortsarz, and A. Srinivasan,, "Approximating thedomatic number," *SIAM J. Comput.*, vol. 32, no. 1, pp. 172–195, 2002.

[20] T. Moscibroda and R. Wattenhofer, "Maximizing the lifetime of dominating sets," in *Proc. 19th IEEE Int. Symp. Parallel Distrib. Process.*, 2005, Art. no. 8.

[21] J. Yu, Q. Zhang, D. Yu, C. Chen, and G. Wang, "Domatic partition in homogeneous wireless sensor networks," *J. Netw. Comput. Appl.*, vol. 37, pp. 186–193, 2014.

[22] C. Wang, M. T. Thai, Y. Li, F. Wang, and W. Wu, "Optimization scheme for sensor coverage scheduling with bandwidth constraints," *Optimization Lett.*, vol. 3, no. 1, pp. 63–75, 2009.

[23] S. Fujita, M. Yamashita, and T. Kameda, "A study on r-configurations—a resource assignment problem on graphs," *SIAM J. Discrete Mathematics*, vol. 13, no. 2, pp. 227–254, 2000.

[24] W. Abbas, M. Egerstedt, C.-H. Liu, R. Thomas, and P. Whalen, "Deploying robots with two sensors in $k_{1,6}$–free graphs," *J. Graph Theory*, vol. 82, pp. 236–252, 2016.

[25] A. Whittle, M. Allen, A. Preis, and M. Iqbal, "Sensor networks for monitoring and control of water distribution systems," in *Proc. 6th Int. Conf. Structural Health Monitoring Intell. Infrastructure*, 2013, https://dspace.mit.edu/openaccess-disseminate/1721.1/92764

[26] A. Deshpande, S. Sarma, K. Youcef-Toumi, and S. Mekid, "Optimal coverage of an infrastructure network using sensors with distance-decaying sensing quality," *Automatica*, vol. 49, no. 11, pp. 3351–3358, 2013.

[27] W. Abbas, L. S. Perelman, S. Amin, and X. Koutsoukos, "An efficient approach to fault identification in urban water networks using multi-level sensing," in *Proc. 2nd ACM Int. Conf. Embedded Syst. Energy-Efficient Built Environments*, 2015, pp. 147–156.

[28] A. Ostfeld, et al., "The battle of the water sensor networks: A design challenge for engineers and algorithms," *J. Water Resources Planning Manage.*, vol. 134, no. 6, pp. 556–568, 2008.

[29] Centre of Water Systems, University of Exeter. [Online]. Available: http://emps.exeter.ac.uk/engineering/research/cws/downloads/benchmarks/, Accessed on: Apr. 18, 2016.

[30] M. D. Jolly, A. D. Lothes, L. Sebastian Bryson, and L. Ormsbee, "Research database of water distribution system models," *J. Water Resources Planning Manage.*, vol. 140, no. 4, pp. 410–416, 2013.

[31] M. Krysander and E. Frisk, "Sensor placement for fault diagnosis," *IEEE Trans. Syst. Man Cybern. Part A: Syst. Humans*, vol. 38, no. 6, pp. 1398–1410, Nov. 2008.

[32] S. Khuller, A. Moss, and J. S. Naor, "The budgeted maximum coverage problem," *Inf. Process. Lett.*, vol. 70, no. 1, pp. 39–45, 1999.

[33] R. Dhal, J. A. Torres, and S. Roy, "Detecting link failures in complex network processes using remote monitoring," *Physica A: Statist. Mech. Appl.*, vol. 437, pp. 36–54, 2015.

[34] M. A. Rahimian and V. M. Preciado, "Detection and isolation of failures in directed networks of LTI systems," *IEEE Trans. Control Netw. Syst.*, vol. 2, no. 2, pp. 183–192, Jun. 2015.

[35] S. Slijepcevic and M. Potkonjak, "Power efficient organization of wireless sensor networks," in *Proc. IEEE Int. Conf. Commun.*, 2001, pp. 472–476.

[36] Z. Abrams, A. Goel, and S. Plotkin, "Set k-cover algorithms for energy efficient monitoring in wireless sensor networks," in *Proc. 3rd Int. Symp. Inf. Process. Sensor Netw.*, 2004, pp. 424–432.

[37] A. Deshpande, S. Khuller, A. Malekian, and M. Toossi, "Energy efficient monitoring in sensor networks," *Algorithmica*, vol. 59, no. 1, pp. 94–114, 2011.
[38] F. Koushanfar, N. Taft, and M. Potkonjak, "Sleeping coordination for comprehensive sensing using isotonic regression and domatic partitions," in *Proc. 25th IEEE Int. Conf. Comput. Commun.*, 2006, pp. 1–13.
[39] C. Wang, M. T. Thai, Y. Li, F. Wang, and W. Wu, "Minimum coverage breach and maximum network lifetime in wireless sensor networks," in *Proc. IEEE Global Telecommun. Conf.*, 2007, pp. 1118–1123.
[40] A. Rossi, A. Singh, and M. Sevaux, "Column generation algorithm for sensor coverage scheduling under bandwidth constraints," *Netw.*, vol. 60, no. 3, pp. 141–154, 2012.
[41] A. Krause, R. Rajagopal, A. Gupta, and C. Guestrin, "Simultaneous optimization of sensor placements and balanced schedules," *IEEE Trans. Autom. Control*, vol. 56, no. 10, pp. 2390–2405, Oct. 2011.
[42] Y. B. Türkoğulları, N. Aras, İ. K. Altınel, and C. Ersoy, "An efficient heuristic for placement, scheduling and routing in wireless sensor networks," *Ad Hoc Netw.*, vol. 8, no. 6, pp. 654–667, 2010.
[43] M. Younis and K. Akkaya, "Strategies and techniques for node placement in wireless sensor networks: A survey," *Ad Hoc Netw.*, vol. 6, no. 4, pp. 621–655, 2008.
[44] W. E. Hart and R. Murray, "Review of sensor placement strategies for contamination warning systems in drinking water distribution systems," *J. Water Resources Planning Manage.*, vol. 136, no. 6, pp. 611–619, 2010.
[45] F. Zhao, J. Shin, and J. Reich, "Information-driven dynamic sensor collaboration," *IEEE Signal Process. Mag.*, vol. 19, no. 2, pp. 61–72, Mar. 2002.

Waseem Abbas received the MSc and PhD degrees both in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, Georgia, in 2010 and 2013, respectively. He is a postdoctoral research scholar in the Department of Electrical Engineering and Computer Science, Vanderbilt University, Nashville, Tennessee. He was a Fulbright scholar from 2009 till 2013. His research interests include the area of network control systems, graph-theoretic methods for large networked systems, and resilience of cyber-physical systems.

Aron Laszka received the MSc and PhD degrees in computer science and engineering from the Budapest University of Technology and Economics, in 2011 and 2014, respectively. He is a research assistant professor in the Department of Electrical Engineering and Computer Science, Vanderbilt University. Previously, he was a postdoctoral scholar with the University of California, Berkeley. Between 2014 and 2015, he was a postdoctoral research scholar with Vanderbilt University. In 2013, he was a visiting research scholar with the Pennsylvania State University. His research work focuses on the security and resilience of cyber-physical systems, the economics of security, and game-theoretic modeling of security problems.

Yevgeniy Vorobeychik is an assistant professor of computer science and biomedical informatics with Vanderbilt University. Previously, he was a principal member of technical staff at Sandia National Laboratories. Between 2008 and 2010, he was a post-doctoral research associate with the University of Pennsylvania Computer and Information Science Department. His work focuses on game theoretic modeling of security and privacy, algorithmic and behavioral game theory and incentive design, optimization, complex systems, epidemic control, network economics, and machine learning. His research has been supported by the US National Science Foundation, the National Institutes of Health, the Department of Energy, and the Department of Defense. He was nominated for the 2008 ACM Doctoral Dissertation Award and received honorable mention for the 2008 IFAAMAS Distinguished Dissertation Award. He is a member of the IEEE.

Xenofon Koutsoukos received the PhD degree in electrical engineering from the University of Notre Dame, Notre Dame, Indiana, in 2000. He is a professor in the Department of Electrical Engineering and Computer Science and a senior research scientist in the Institute for Software Integrated Systems (ISIS), Vanderbilt University, Nashville, Tennessee. He was a member of research staff at the Xerox Palo Alto Research Center (PARC) (2000-2002), working in the embedded collaborative computing area. His research work is in the area of cyber-physical systems with emphasis on formal methods, data-driven methods, distributed algorithms, security and resilience, diagnosis and fault tolerance, and adaptive resource management. He was the recipient of the NSF Career Award in 2004, the Excellence in Teaching Award in 2009 from the Vanderbilt University School of Engineering, and the 2011 NASA Aeronautics Research Mission Directorate (ARMD) Associate Administrator (AA) Award in Technology and Innovation. He is a senior member of the IEEE.