Computers & Industrial Engineering 189 (2024) 109928

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

Enhancing anomaly detection with adaptive node inspection in large-scale networks with binary sensors

Feiran Xu [a,b], Ramin Moghaddass [a,c,*]

[a] Department of Industrial and Systems Engineering, University of Miami, Coral Gables, FL, United States of America
[b] Department of Business Analytics, Tippie College of Business, University of Iowa, Iowa City, IA, United States of America
[c] Miami Herbert School of Business, University of Miami, Coral Gables, FL, United States of America

ARTICLE INFO

Keywords:
Anomaly detection
Node inspection
Graph analytics
Bayesian networks

ABSTRACT

Maintaining the stability and reliability of large-scale networks and graph structures is a practical challenge, particularly in sensor-intensive systems. One critical task in such networks is to identify anomalies and the origin of disturbances in a timely manner. However, successful anomaly detection requires sufficient and accurate data from sensors across the network. This work aims to develop an innovative framework to improve the accuracy of anomaly detection tasks with binary sensor data through intelligent node selection and inspection. Instead of relying only on stochastic insights obtained from network sensors, we explore how a specific small set of nodes can be inspected using a Bayesian framework so that the anomaly detection performance is improved. We demonstrate the effectiveness of the proposed model with a set of numerical experiments.

1. Introduction

Many mechanical, electrical, social, communication, and computer systems have a network/graph structure with numerous interlinked nodes that represent physical entities, devices, or sensors. These nodes are connected by edges that denote dependencies and interactions between them. Many types of anomalies and disturbances may occur across these networks during their operation. Anomalies can be defined as disturbances or any deviation from normal/nominal behavior. Examples of anomalies in systems with graph structures are intrusions in wireless networks (Hu et al., 2021), water contamination in water distribution networks (Tuptuk et al., 2021), outages and power disturbances in power distribution networks (Yuan et al., 2020), and abnormal users on the social networks (Rahman et al., 2021). Anomalies, also known as outliers, disturbances, or abnormal behaviors, can result from various events in networks and can initiated from any node and then be propagated to other nodes. Network anomalies can occur in various forms, including sustained anomalies that persist over an extended period and temporary anomalies that occur briefly, affecting either a single node or multiple nodes within the network.

Anomaly detection in networks has gained more attention in recent years due to the growing availability of smart sensors and remote devices, facilitating the continuous collection of real-time data from specific network segments or nodes. These collected sensor data serve the purpose of real-time network health monitoring and, in turn, enable the identification of anomalies and nodes affected by such anomalies. Examples of sensors used for health monitoring and anomaly detection are the set of equipment and systems, referred to as Advanced Metering Infrastructure (AMI), that assist with the intelligent health management of power distribution networks (Lazim Qaddoori & Ali, 2023). This article is motivated by the problem of outage detection and isolation in modern power distribution networks, which are sensor-intensive graph-based structures that are subject to various types of anomalies over time. Monitoring and ensuring the healthy operations of the power grid systems, and pinpointing the set of anomalous nodes is a challenging task due to the large size of power distribution networks and the complexity of the topology of such networks. Numerous algorithms and mathematical techniques have been devised to address the problem of network anomaly detection with sensor data (see for instance (Ma et al., 2021) a review of neural network-based models used for graph anomaly detection).

Despite all recent advances in graph and network anomaly detection algorithms developed for various application systems, it is known that many network-based anomaly detection methods may lead to a high false positive rate (Akoglu et al., 2015), which can be due to several challenges and complexities associated with network topology and real-world network data. In this article, we explore the impact of selectively inspecting a segment of the network, which could involve examining only a limited number of nodes, in order to enhance the effectiveness

* Corresponding author.
E-mail address: ramin@miami.edu (R. Moghaddass).

https://doi.org/10.1016/j.cie.2024.109928
Received 10 April 2023; Received in revised form 16 November 2023; Accepted 22 January 2024
Available online 24 January 2024

of existing anomaly detection models that offer probabilistic assessments of each node's condition. After obtaining an estimation of the status of each node within the network, it may be possible to boost performance and enhance accuracy by conducting physical inspections on a selected subset of nodes. These selective node inspections provide supplementary information, allowing for more confident updates to the initial node status estimations.

The main contributions made in the paper are summarized below. First, we introduce two approaches for selective node inspection: a baseline node selection algorithm and a refined node selection algorithm. These algorithms aim to improve the efficiency of anomaly detection in sensor-monitored networks. Additionally, we present two distinct inspection strategies: static and sequential. The static method involves selecting and inspecting all chosen nodes simultaneously, while the sequential method inspects nodes one by one in an order that is impacted by the outcome of the previous inspections. The proposed inspection framework offers the flexibility to serve as an additional step following any existing probabilistic anomaly detection method, provided that the output of these methods includes the probability distribution of each node being anomalous. This work focuses only on using sensor attributes at the node level and anomalies that propagate from a set of source nodes to other connected nodes given known propagation paths/rules. An example of such anomalies includes power outages and water contamination in power and water distribution networks, where these anomalies have the potential to propagate to all downstream nodes starting from the source of the anomaly. Therefore, anomalies that occur on edges and those that are propagated randomly across the networks are beyond the scope of this paper. We have additionally devised computationally efficient procedures for the refined method by leveraging vectorization and matrix operations. This implementation has led to the elimination of nested 'for' loops, resulting in reduced CPU time. In the context of Bayesian networks, the utilization of sensor data to select network nodes for inspection aligns with the foundational principles of Bayesian networks, emphasizing the incorporation of evidence and optimization of information gain. By focusing on nodes likely to reveal crucial information about the latent states of the network nodes, the model enhances the efficiency of Bayesian network inference, contributing to more informed decision-making under uncertainty. This approach reflects a valuable integration of probabilistic reasoning with practical considerations, reinforcing the model's significance within the Bayesian network framework.

The rest of this paper is organized as follows. In Section 2, we will examine the relevant literature and discuss the relationship between the existing work and this article. In Section 3, we will discuss the general problem of anomaly detection in the Bayesian context and then introduce the proposed frameworks for node selection and node inspection. Section 4 shows the performance of our proposed frameworks on a comprehensive set of numerical experiments. Our conclusions are presented in Section 5, along with a discussion on important areas for further research.

2. Related work

Anomaly detection in networks has been studied in many disciplines and application settings. Some examples can be found in network intrusion detection (Ahmed et al., 2016; Bhuyan et al., 2014), credit card fraud detection (Phua et al., 2010; Popat & Chaudhary, 2018), and road traffic anomaly detection (Kim & Cho, 2018; Kumaran et al., 2019; Radford et al., 2018). There are also plenty of survey articles available for network and graph anomaly detection models and algorithms (see for instance (Akoglu et al., 2015; Erhan et al., 2021; Ma et al., 2021)). Many of the available work on network anomaly detection relies solely on the data collected from sensors, which only stochastically provide information regarding the health of the node level rather than the entire network. Sensors play a pivotal role in anomaly detection for network systems, such as smart and connected cities environment (Parra et al., 2015), telecommunication networks (Yang et al., 2011), IoT (Cui et al., 2019), and wireless sensor networks (WSNs) (Ifzarne et al., 2021). Many types of data are generated from smart meters and sensors in large-scale networks on a regular basis, which may be useful for anomaly detection. Most methods developed for network anomaly detection in the existing literature can be categorized into the following groups: statistics-based approaches, density-based techniques, clustering-based methods, as well as machine learning and deep learning-based methodologies. (Chandola et al., 2009). It is known that many network-based anomaly detection methods lead to a high false positive rate (Akoglu et al., 2015). This characteristic, in turn, adds to the complexity of applying these approaches in real-world systems. Despite extensive research in the field of network anomaly detection, it remains an evolving area with the ongoing development of new algorithms and techniques. The goal is to define methodologies that are highly reliable and produce fewer false alarms in real-world scenarios.

One potential area of research to enhance the performance and improve the accuracy of available anomaly detection methods arises through node inspection. This process involves conducting physical inspections on a selected subset of nodes within the network. These inspections can be carried out using physical inspectors, automated drone techniques, or remote sensors for real-time data collection. Leveraging the additional information obtained from these inspections, the initial estimations of node statuses can be updated with greater certainty. This expansion of the anomaly detection challenge entails identifying the optimal set of nodes to inspect in order to maximize the accuracy of the estimations. Over the past decade, research efforts have been conducted towards the selection of cluster heads (CH) in Wireless Sensor Networks (WSNs) to reduce the network's energy consumption, a problem known to be NP-hard (Nguyen et al., 2011). Various research studies have introduced different algorithms for selecting cluster heads. These include probabilistic clustering algorithms, which involve randomly assigning cluster head roles to nodes in the WSN, effectively rotating the CH responsibilities for each node (Heinzelman et al., 2002; Shigei et al., 2010); deterministic criteria-based selection methods, which consider factors such as node distances, remaining energy, node degree, centrality, and proximity (Kang & Nguyen, 2012; Ma et al., 2010); and fuzzy-logic approaches (Lee & Cheng, 2012), among others.

Another research area closely related to the idea of node inspection is influence maximization. Influence maximization deals with the task of identifying a set of nodes capable of exerting the greatest influence over other nodes or triggering the largest cascading effect within a network (Pei et al., 2020; Song et al., 2016; Zhang et al., 2023). This problem is known to be NP-hard because the number of potential node sets grows exponentially with network size. However, in the context of anomaly detection within networks, the nodes responsible for causing the most cascading effects may not necessarily possess the most critical information. For instance, if we have a high level of confidence in our estimations regarding influential nodes, we might opt to inspect different nodes to gain a deeper understanding of the network, even if these nodes are unlikely to influence a large number of nodes. Consequently, methods developed to tackle the influence maximization problem are not directly applicable to the challenges addressed in this work.

To the best of our knowledge, no similar work has been conducted from our specific perspective: enhance the performance and boost the accuracy of identifying the source of anomalies and impacted nodes by implementing node inspections guided by inferences drawn from network sensor data regarding the anomaly status of the network. In the existing literature, node inspections are mostly used to monitor the status of facilities/equipment (Kuboki & Takata, 2019), the quality of drinking water (Brentan et al., 2021; Santos-Ruiz et al., 2022), the object or surface detection (Gronle & Osten, 2016; Trucco et al., 1997), and the indoor or outdoor surveillance (Suresh & Menon, 2023), to name a few. There are available works that focus on the optimal

Table 1
The list of notations used in the paper.

Notation	Description
Topology Parameters	
$G = (N, \mathbf{A}, \mathbf{Z}, \mathbf{P})$	Graph G with adjacency matrix $\mathbf{A}$, sensor-attribute matrix $\mathbf{Z}$, propagation matrix $\mathbf{P}$
N	The set of nodes in the graph
S	The set of sensor attributes
a_{ij}	The binary parameter representing whether there is an edge between nodes i and j
z_{is}	The binary index representing whether node i generates sensor attribute s
p_{ij}	A binary index representing whether an anomaly propagates from node i to node j
$\mathbf{A} = [a_{ij}]_{\|N\|\times\|N\|}$	The adjacency matrix (representing physical/virtual connections between nodes)
$\mathbf{Z} = [z_{is}]_{\|N\|\times\|S\|}$	The sensor-attribute matrix
$\mathbf{P} = [p_{ij}]_{\|N\|\times\|N\|}$	The anomaly propagation matrix
D_i	The set of downstream/descendant nodes of node i
D_i^c	The set of immediate children nodes of node i
U_i	The set of upstream/ancestor nodes of node i
U_i^p	The set of immediate parent nodes of node i
E_I	The set of nodes selected for inspection
Network Variables	
y_{is}	The value of sensor attribute s for node i
x_i	The binary index representing the anomaly status of node i
o_i	The binary index representing whether node i is the source of an anomaly

sensor placement in networks, which can be solved as a mixed-integer programming (MIP) (Berry et al., 2005), stochastic programming problem (Shastri & Diwekar, 2006), multi-objective optimization problem, or machine learning methods (El-Zahab et al., 2018), and heuristics and rule-based approaches. However, our work has a different purpose from all these works. The primary objective of our work is to identify a limited number of nodes for precise and meticulous inspection based on the data collected from sensors distributed throughout the networks. This will be succeeded by performing the preliminary network status evaluation to find the most likely status of the nodes in the network, with the aim of bolstering the reliability and precision of our initial estimations. The proposed method leverages the large amount of data generated by sensors in a network to identify anomalies through a Bayesian framework. Also, by utilizing vectorization and matrix operations, we proposed a computationally efficient framework, making it more applicable to real systems and large-scale graph structures.

3. Main approach

This section explains the sensor selection and inspection framework in detail, which is based on a Bayesian graph anomaly detection method. The list of notations used across the paper is provided in Table 1. Before illustrating the details of the node inspection problem and the proposed method, we first introduce the concept of network/graph anomaly detection problem through a Bayesian framework. Our work is driven by the distinctive features of Bayesian Networks (BNs), which enable us to model the topological structure of attributed networks and the probabilistic relationship between network status and sensor data in an interpretable manner. In this article, we assume that all network topology characteristics are fixed, while network variables can change over time. However, as our proposed models treat data from different time points as independent samples, we have removed the time index from all variables for mathematical convenience.

3.1. A generic structure for attributed networks with node sensors

The system under study is assumed to be represented as an attributed network/graph through a Bayesian structure. The features of the directed network topology and assumptions made for the Bayesian framework-based anomaly detection are discussed in this subsection.

A. Graph Topology: A network, represented by $G = (N, \mathbf{A}, \mathbf{Z}, \mathbf{P})$, is a system made up of nodes and linked edges, where N is the set of nodes. In this article, we used the cardinality of a set (denoted by $|\ |$) to represent the number of elements in the set. Here the set of nodes can be obtained from a $|N| \times |N|$ adjacency matrix $\mathbf{A} = [a_{ij}]$, where $a_{ii} = 1$, and $a_{ij} = 1$ if there is a directed link from node i to node j, where $\{i, j\} \in N$. Each node corresponds to a unique entity, such as a sensor, device, equipment, or user, and can be a potential source of the original location of an anomaly. To indicate whether the sensor attribute j is available or collected at node n, we use a matrix $\mathbf{Z} = [z_{nj}]$, where $z_{nj} = 1$ if the sensor generates attribute j and collected at node n. Otherwise, we set $z_{nj} = 0$ to denote that the sensor that generates attribute j is not installed or when that specific data is missing at node n. The fourth argument of the network is a $|N| \times |N|$ matrix of $\mathbf{P} = [p_{ij}]$, where $p_{ij} = 1$ if an anomaly can propagate from node i to node j, which summarizes the anomalies propagation paths through the network. The propagation matrix is needed to define the sets of downstream and upstream nodes (e.g., U_i, U_i^p) concerning anomalies. For instance, $j \in D_i$ if $p_{ij} = 1$. We should point out that when anomalies are propagated to all downstream nodes, matrix $\mathbf{P}$ can be defined directly from the adjacency matrix $\mathbf{A}$. The propagation matrix $\mathbf{P}$ is given, and p_{ij} is a known binary variable in deterministic scenarios. In stochastic propagation cases, p_{ij} can represent the probability of an anomaly propagating from node i to node j, which is out of the scope of this work.

B. Monitoring Variables: In our framework, there are two sets of hidden variables to monitor the status of the network-structured system. For node n, variable x_n is defined to indicate whether node n is under the effect of an anomaly, and o_n is to reflect whether node n is the original source of an anomaly in the network. Both x_n and o_n are binary variables, are not directly observable, and can only be inferred stochastically from the available sensor data. There is a deterministic relationship between x_n and o_n, therefore, the anomaly status of all nodes, represented by $x_1, x_2 ..., x_{|N|}$ can be known if $o_1, o_2, \ldots, o_{|N|}$ are known.

C. Observable Sensor Data: There are $|S|(|S| \geq 1)$ different types of sensor attributes in our framework. The variable y_{ns} is defined to store the observation or sensor data at node n, where y_{ns} is either binary or NULL if there is no output at node n. In our work, the stochastic relationship between the sensor attributes and the status of the node where the sensor attributes are collected can be defined as follows:

$$\Pr(y_{ns} = 1|x_n = 1) = \alpha_s, \quad \Pr(y_{ns} = 1|x_n = 0) = \beta_s, \quad \forall n \in N, \ \forall s \in S. \tag{1}$$

where α_s can be considered as the true positive rate and β_s is the false positive rate for attribute s. Therefore, the larger α_s with a smaller β_s makes a stronger sensor. Further, we assume that the binary sensor features are conditionally independent given the node's binary status, therefore, the joint probability distribution of sensor observations across the network is shown as follows:

$$\Pr(y_{n1} = v_1, \ldots, y_{n|S|} = v_{|S|}|x_n = a) = \left[\prod_{s\in S} \alpha_s^{v_s}(1-\alpha_s)^{1-v_s}\right]^a \left[\prod_{s\in S} \beta_s^{v_s}(1-\beta_s)^{1-v_s}\right]^{1-a}, \forall n \in N, a \in \{0, 1\}.$$

3.2. Network anomaly detection through Bayesian networks

Our work focuses on detecting those anomalies that are not able to be observed or cannot be detected directly by processing stochastic sensor data. In order to better explain the node inspection problem, we will briefly review the anomaly detection problem through a Bayesian network in this section. The generic problem of anomaly detection for a sensor-driven network is defined below.

Problem 1 (*Anomaly Detection for Heterogeneous Networks*). Given a network-structured system with known topology G and sensor observations ($\mathbf{Y} = [y_{ns}; n \in N, s \in S]$), how to estimate the most likely status of all nodes in the network ($\boldsymbol{x} = [x_1, \dots, x_{|N|}]$) and locate the sources of potential anomalies ($\boldsymbol{o} = [o_1, \dots, o_{|N|}]$)?

This problem can be framed through a Bayesian network, where a directed acyclic graph (DAG) (shown in Fig. 1) represents the deterministic and stochastic causal relationships between sensor nodes and hidden variables. There are three stochastic variables in the Bayesian framework defined for each node in the DAG: hidden monitoring variables o_n and x_n, and observation variables/sensor outputs $\boldsymbol{y}_n = [y_{n1}, \dots, y_{n|S|}]$. We used a DAG to indicate the relationships between the parent and the children nodes, where the direction of the edges represents the propagation flow. The hidden nodes are shown in circles, while the observable sensor nodes are shown as rectangles. Two types of dependency relationships are represented by two different edges in the DAG. The solid edges stand for the deterministic causal relationships, and the dashed lines symbolize the stochastic relationships. It is assumed in our work that the topology of the network will remain the same, therefore, the propagation paths from ancestor nodes to descendant nodes are deterministic. Additionally, the solid directed edge from o_n to x_n confirms the fact that if the node is one of the sources of anomalies, the status of the node itself must be anomalous as well. Mathematically, $x_n = 1$ if $o_n = 1$. The constraints below are defined to ensure the deterministic dependencies between the variables $\boldsymbol{x}$ and $\boldsymbol{o}$, that is:

$$\Phi(G, \boldsymbol{x}, \boldsymbol{o}): \begin{cases} o_n + \sum_{m \in U_n} x_m \le (|U_n| + 1) x_n & \forall n \in N; \\ x_n \le o_n + \sum_{m \in U_n} x_m & \forall n \in N. \end{cases} \tag{2}$$

The first constraint in Eq. (2) suggests that if node n is the original source of anomalies (i.e., $o_n = 1$) and/or at least one of the upstream nodes of node n is anomalous (i.e., $\sum_{m \in U_n} x_m \ge 1$), then node n will also be anomalous (i.e., $x_n = 1$). The second constraint of Eq. (2) ensures that if node n is not the source of anomalies (i.e., $o_n = 0$), and none of its upstream nodes are anomalous (i.e., $\sum_{m \in U_n} x_m = 0$), then node n must also not be anomalous (i.e., $x_n = 0$). The outcome of the anomaly detection algorithm provides the distribution of $\Pr(o_1, o_2, \dots, o_{|N|} | \mathbf{Y}, G)$ for any values of $o_1, o_2, \dots, o_{|N|}$. Once this probability is found, we can generate K posterior MCMC samples, denoted by $\{o_1^{(k)}, o_2^{(k)}, \dots, o_{|N|}^{(k)}\}$ for $1 \le k \le K$. Clearly, the MCMC samples for variable o_n automatically give the corresponding values for x_n as well (and vice versa). The most basic way of getting these stochastic samples is from the marginal distribution of sensor outputs as follows:

$$\Pr(x_1, x_2, \dots, x_{|N|} | \boldsymbol{y}_1, \boldsymbol{y}_2, \dots, \boldsymbol{y}_{|N|}) \propto \prod_{n=1}^{|N|} \Pr(y_n | x_n) \Pr(x_1, \dots, x_{|N|}).$$

In other words, one can sample each x_n from its corresponding distribution $\Pr(y_n|x_n)$ (from Eq. (1)) and then keep it if sample $\{x_1, x_2, \dots, x_{|N|}\}$ satisfies Eq. (2). In this paper, we utilize the MCMC samples that are obtained from a stochastic sampling approach in a Bayesian framework proposed in (Xu & Moghaddass, 2023). The summary of the steps for the MCMC sampler we used in the paper is presented in Algorithm 1. The output of Algorithm 1 will be K stochastic samples for each variable, where the kth sample is denoted by $\boldsymbol{o}^{(k)} = \{o_1^{(k)}, \dots, o_{|N|}^{(k)}\}$ and $\boldsymbol{x}^{(k)} = \{x_1^{(k)}, \dots, x_{|N|}^{(k)}\}$. Samples generated from the Markov chain in Algorithm 1 can be used to estimate the unknown monitoring variables $o_1, \dots, o_{|N|}$ and $x_1, \dots, x_{|N|}$. Also, through the characterization of the stochastic behavior via the approximation of probability distributions from MCMC samples (empirical distribution estimation), we can also infer the uncertainty associated with the health status estimation of each node.

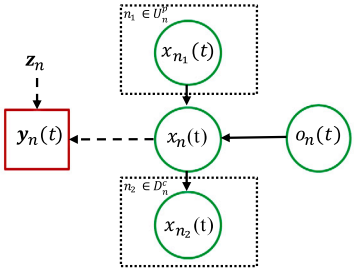

Fig. 1. A graphical model representing the dependencies between BN variables for node n and its immediate children and parents. Circle nodes represent hidden variables.

Algorithm 1 Proposed Stochastic Sampler to Estimate Network Status Variables

Input: Network Structure (G), Sensor Data ($\mathbf{Y} = [y_{ns}]_{|N| \times |S|}$), Stopping Criterion (Number of Iterations K), burn-in period (k_0)

Output: MCMC Samples $\boldsymbol{o}^{(k)}$ and $\boldsymbol{x}^{(k)}$ for $k \in \{1, ..., K\}$ denoted by $\boldsymbol{o}^{(1:k)}$ and $\boldsymbol{x}^{(1:K)}$ and estimated network status variables $\hat{\boldsymbol{o}} = \{\hat{o}_1, ..., \hat{o}_{|N|}\}$ and $\hat{\boldsymbol{x}} = \{\hat{x}_1, ..., \hat{x}_{|N|}\}$

Initialize $\boldsymbol{o}^{(0)} = \{o_1^{(0)}, ..., o_{|N|}^{(0)}\}$ from the corresponding priors (or randomly generate) and then find $\boldsymbol{x}^{(0)} = \{x_1^{(0)}, ..., x_{|N|}^{(0)}\}$ from Eq. (2).
Set $\boldsymbol{o}^{update} = \boldsymbol{o}^{(0)}$ and $\boldsymbol{x}^{update} = \boldsymbol{x}^{(0)}$.

for $k = 1 : K$ **do**
  for $n = 1 : N$ **do**
    Propose $o_n^{new} = 1 - o_n^{(k-1)}$.
    The acceptance rate is obtained as

$$\alpha = \min\left\{1, \frac{\Pr(o_n^{new}) \prod_{n' \in \{n, D_n\}} \Pr(\mathbf{Y}_{n'} | x_{n'}^{new}, \boldsymbol{z}_{n'})}{\Pr(o_n^{(k-1)}) \prod_{n' \in \{n, D_n\}} \Pr(\mathbf{Y}_{n'} | x_{n'}^{update}, \boldsymbol{z}_{n'})}\right\},$$

    where

$$x_{n'}^{new} = \min\left\{1, o_{n'}^{new} + \sum_{n'' \in U_n, n'' \ne n} o_{n''}^{update}\right\}, \quad n' \in \{n, D_n\}. \tag{3}$$

    Sample a random number u from U(0, 1).
    if $u \le \alpha$ **then**
      Accept the proposed sample and update network status variables as follows:

$$o_n^{(k)} = o_n^{new}, o_n^{update} = o_n^{new}$$
$$x_{n'}^{(k)} = x_{n'}^{new}, x_{n'}^{update} = x_{n'}^{new} \quad \forall n' \in \{n, D_n\}$$

    else
      Reject the proposed sample o_i^{new}, set

$$o_n^{(k)} = o_n^{(k-1)}, x_n^{(k)} = x_n^{(k-1)}.$$

    end if
  end for
  - Record new samples $\boldsymbol{o}^{(k)} = \{o_1^{(k)}, ..., o_{|N|}^{(k)}\}$ and $\boldsymbol{x}^{(k)} = \{x_1^{(k)}, ..., x_{|N|}^{(k)}\}$.
end for
Estimate network status variables $\hat{\boldsymbol{o}} = \{\hat{o}_1, ..., \hat{o}_{|N|}\}$ and $\hat{\boldsymbol{x}} = \{\hat{x}_1, ..., \hat{x}_{|N|}\}$ based on MCMC samples. In this paper, the estimates for each variable are obtained by calculating the mean of the MCMC samples collected after the burn-in period of length k_0, which is set by the user).

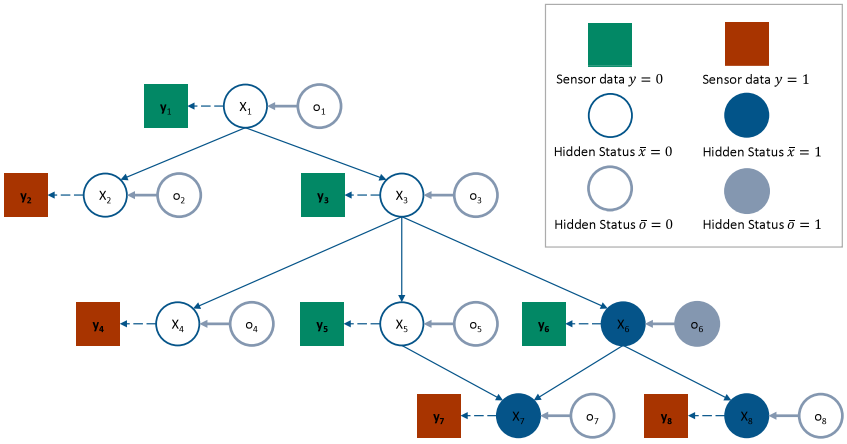

Fig. 2. A toy model representing the node selection problem, that is, which node(s) should be selected for inspection?

3.3. Node selection for inspection in network/graph structures

The outcome of Section 3.2 only provides an uncertain estimation of each node's status. In the context of anomaly detection, if can verify the status of certain nodes, we can improve our confidence in the original estimation. However, it is not feasible to inspect all nodes in a network due to budget and time limitations. Therefore, it is important to choose which nodes should be selected for inspection. In many network systems, such as power networks, we can obtain the true status (o and x values) of each node through physical inspections, such as ground-based, helicopter-based, and drone-based inspections. To formalize the node inspection problem in the context of Bayesian networks, we propose an approach that filters the MCMC samples generated from a stochastic MCMC sampler. This approach can simulate the distributions of the hidden variables x and o, and update the estimations of network nodes' status based on the results of the inspection.

There can be three scenarios for a selected node after inspection, as discussed below. (1) *Anomalous node that is the source of an anomaly*: In this case, all downstream nodes on the propagation path of this node are anomalous. Scenario (2) *Anomalous node that is not the source of an anomaly*: In this case, we can determine that an anomaly must come from one of its upstream nodes. (3) *Non-anomalous node*: We can confirm that all the upstream nodes of the inspected node should be healthy, and anomalies, if exist, can only come from downstream nodes. Algorithm 2 summarizes the steps needed to be taken to update the health status of the network after each inspection. It is assumed in the paper that multiple sources of anomalies may exist in the network. Regarding the sources of anomalies along any given path in the network, two distinct assumptions can be made. First, it can be assumed that each node is subject to, at most, one anomaly at a time. In other words, there is a maximum of one anomalous source that may exist upstream of any node. Second, an alternative assumption is that multiple anomalous sources may coexist within the same path, and each node may be impacted by multiple anomalies simultaneously. Under this assumption, both parent and child nodes can simultaneously serve as sources of anomalies. Assuming that there is only one source of anomalies within any given path in the network can lead to additional updates in the status of certain nodes within the network after inspection. For example, when the inspected node n is identified as the source of the anomaly, we can infer that all upstream nodes must be non-anomalous and there is no downstream node as the source of another anomaly. Conversely, if the inspected node n is found to be anomalous but not the source of the anomaly, we can conclude that there is only one source of anomalies located upstream of node n and there is no downstream node as the source of another anomaly. When the inspected node is found to be non-anomalous, the assumption of a single-source anomaly does not impact the network's update. In Algorithm 2, we have included these additional steps as optional updates. Algorithm 2 still covers cases where multiple sources of anomalies exist, only if there is no directed path between any pair of these anomalies. The problem of node selection is now formalized as follows:

Problem 2 (*Node Selection for Inspection Problem*)**.** For a network with set of nodes denoted by N, given $\hat{o}_1, \hat{o}_2, \ldots, \hat{o}_{|N|}$ as the estimated status of nodes $1, \ldots, |N|$ in the network, how to determine which set of m ($m \geq 1$) nodes should be inspected in order to improve the reliability and confidence of previous estimations of the network status?

Consider a simple 8-node network as depicted in Fig. 2, where the true source of the anomaly is node 3, and thus nodes 3-8 are all anomalous. Also, assume that the outcome of the anomaly detection algorithm suggests that node 6 is the source of the anomaly, which means it also identifies node 4 and node 5 as anomalous. Now, if we decide to inspect node 6, the inspection outcome will confirm that node 6 is indeed anomalous, and it will also flag nodes 7 and 8 as anomalous since they are downstream from node 6. However, this inspection alone cannot conclusively determine whether the source of the anomaly lies in node 1 or node 3, both of which are upstream nodes of node 6. On the other hand, if we choose to inspect node 3, the inspection outcome will reveal that node 3 is the source of the anomaly, and it will also mark all of its downstream nodes (4-8) as anomalous. Selecting nodes 1 or 2 for inspection does not provide crucial information regarding the location of anomalies since inspection merely confirms that these nodes are not anomalous. While selecting node 4 or node 5 for inspection may not directly pinpoint the source of anomalies, it does provide additional information about the network's status. In other words, inspection of nodes 4 or 5 would confirm that they are anomalous and suggest that the source of anomalies must be either node 1 or node 3 (note that inspection reveals that node 4 or 5 are not the source of an anomaly). Consequently, inspection leads to the determination that nodes 6, 7, and 8 are anomalous as well. It is evident that even in a relatively small network, such as the one depicted in Fig. 2, there can be many possible inspection outcomes depending on what node to be chosen for inspection. In this paper, we only focus on selecting a limited number of nodes for physical inspection, which can be fine-tuned according to specific time and cost constraints or other reliability requirements. To mathematically define the node inspection problem, we use $E_I = \{i_1, i_2, \ldots, i_m\}$ to denote the set of m nodes that are selected for an inspection, where $|E_I| = m$. Here, $\{i_1, i_2, \ldots, i_m\}$ are the node numbers selected from $\{1, \ldots, |N|\}$. The corresponding optimization problem for selecting m nodes for inspection can be defined mathematically as below:

$$\max \quad z = \Pr(o_1, o_2, \ldots, o_{|N|} | \boldsymbol{o}_{E_I}, \mathbf{Y}, G) \tag{4}$$

$$\text{s.t.} \quad \Phi(G, \boldsymbol{x}, \boldsymbol{o})$$
$$|E_I| = m$$
$$o_n \in \{0,1\},\ x_n \in \{0,1\},\ \forall n \in N,\ E_I \in N^m.$$

Now, the goal is to identify a set of m nodes, in E_I, such that knowledge of their true status will allow us to make the most accurate prediction of the status of the entire network G.

Algorithm 2 Network Status Update After a Node Inspection

Input: Network Structure (G), Inspected Nodes (E_I), Estimated Network Status Variables from Algorithm 1 and Previous Inspections $\hat{\boldsymbol{o}} = \{\hat{o}_1, ..., \hat{o}_{|N|}\}$ and $\hat{\boldsymbol{x}} = \{\hat{x}_1, ..., \hat{x}_{|N|}\}$, and True values of o_n and x_n for all $n \in E_I$

Output: Updated Network Status Variables $\hat{\boldsymbol{o}}$ and $\hat{\boldsymbol{x}}$

for $n \in E_I$ **do**
  if $o_n = 1$ and $x_n = 1$ **then**
    Set $\hat{x_d} = 1, \forall d \in D_n$
    ▷ **Optional Updates for Single-Source Anomaly Scenarios**
    Set $\hat{o}_d = 0, \forall d \in D_n$
    Set $\hat{o}_u = 0, \forall u \in U_n$
    Set $\hat{x}_u = 0, \forall u \in U_n$
  else if $o_n = 0$ and $x_n = 1$ **then**
    Set $\hat{x_d} = 1, \forall d \in D_n$
    Set $\sum_{u \in U_n} \hat{o}_u \geq 1$ (at least one upstream node of node n is the source of an anomaly)
    ▷ **Optional Updates for Single-Source Anomaly Scenarios**
    Set $\hat{o}_d = 0, \forall d \in D_n$
    Set $\sum_{u \in U_n} \hat{o}_u = 1$ (exactly one upstream node of node n is the source of an anomaly.)
  else if $x_n = 0$ **then**
    Set $\hat{o}_u = 0, \forall u \in U_n$
    Set $\hat{x_u} = 0, \forall u \in U_n$
  end if
end for

3.4. A baseline node selection method for inspection

In the remainder of this section, we will explore various approaches to determine the set of nodes for inspection in a network where we have an estimated status of its nodes based on a probabilistic anomaly detection framework. First, we will introduce a baseline approach that involves using the chain rule, which enables us to get the following:

$$\Pr(o_1, o_2, \dots, o_{|N|} | \boldsymbol{o}_{E_I}, \mathbf{Y}, G) = \frac{\Pr(o_1, o_2, \dots, o_{|N|} | \mathbf{Y}, G)}{\Pr(\boldsymbol{o}_{E_I} | \mathbf{Y}, G)}. \tag{5}$$

Therefore, it is straightforward that the set of nodes with a minimum value of $\Pr(\boldsymbol{o}_{E_I} | \mathbf{Y}, G)$ can maximize Eq. (5) if we assume that the value of $[o_1, o_2, \dots, o_{|N|}]$ is known and fixed. In other words, we will need to select m nodes for inspection with the lowest levels of certainty in their initial estimation, that is

$$E_I : \quad \underset{i_1:i_m \in N}{\text{argmin}} \quad \Pr(o_{i_1:i_m} | \mathbf{Y}, G).$$

There are $\binom{|N|}{m}$ possible combinations of the joint probabilities in the above optimization problem, which cannot be solved analytically for $m > 1$. However, from the outcome of the MCMC samplers, it is possible to estimate $\Pr(\boldsymbol{o}_{E_I} | \mathbf{Y}, G)$. For any set of the initial estimations $[\hat{o}_1, \hat{o}_2, \dots, \hat{o}_{|N|}]$ from the anomaly detection algorithm, if we select $\boldsymbol{o}_{E_I}$ as the set of inspection nodes and the inspection outcome verifies that the true $\boldsymbol{o}_{E_I} \in [\hat{o}_1, \hat{o}_2, \dots, \hat{o}_n]$, then we can maintain the initial estimation with a higher confidence level. We can also update the estimations for other nodes by filtering the samples that match the inspection outcomes of $\boldsymbol{o}_{E_I}$. However, if it turns out that the inspection outcomes o_{E_I} do not match the prior estimation, then the previous estimations $[\hat{o}_1, \hat{o}_2, \dots, \hat{o}_{|N|}]$ become infeasible. As a result, the stochastic MCMC samples for each node in the network should be filtered based on the true inspection values of $\boldsymbol{o}_{E_I}$. For cases where more than one node is to be inspected ($m > 1$), we can divide the node inspection method into static (Algorithm 3) and sequential (Algorithm 4) scenarios. The difference between these two scenarios is in the manner in which multiple nodes are selected and inspected. For the static selection, we first inspect all selected nodes and then update the network status. However, for sequential selections, we will select one node at a time, and then conduct an inspection for that node. Then based on the updated information obtained from the last inspected node, we will select the next node for inspection, and repeat this process until we meet the maximum Number of Inspection Nodes. The details of the static and sequential algorithms for the baseline model are given in Algorithms 3 and 4, respectively. The sequential model involves an additional step that occurs after inspecting each node, which involves filtering the set of MCMC samples based on the outcome of each inspection. This step is necessary as the updated set of MCMC samples can influence the selection of the next node for inspection. While this filtering step contributes to improving the effectiveness of inspection, it is computationally intensive and, consequently, more time-consuming than the static approach. We will discuss this in the numerical experiment section.

3.5. A refined node inspection method

In this subsection, a refined search method is proposed to select nodes for inspection based on an initial estimation of network nodes. Recall Eq. (5), the denominator $\Pr(\boldsymbol{o}_{E_I} | \mathbf{Y}, G)$ is actually a part of the numerator $\Pr(o_1, o_2, \dots, o_n | \mathbf{Y}, G)$, which means that when the set of nodes for inspection changes, the estimation of the entire network changes as well. Therefore, only minimizing the denominator in Eq. (5) may not be sufficient, as the outcome is relative to the changing of the denominator and nominator. As a result, it becomes challenging to directly solve the maximization problem shown in Eq. (4). To find the optimal values of network status variables, we developed a refined search method to select the set of nodes for node inspection. In this subsection, we explain the proposed algorithm for the general case of $m \geq 1$. Similar to the method discussed in Section 3.4, the static and sequential policies for node selection would also apply here. For each node n, there are two possible scenarios, that is, anomalous or healthy (i.e., $o_n = 1$ or $o_n = 0$). In the proposed search algorithm, we estimate the value of Eq. (5) for both scenarios and set the larger value as the measure for node selection, denoted by $\gamma(n) = \max\{\gamma^1(n), \gamma^0(n)\}$, as shown below

$$\gamma(n) = \max\Bigg(\underbrace{\frac{\Pr(\hat{o}^1_{n1}, \dots, o_n = 1, .., \hat{o}^1_{n|N|} | \mathbf{Y}, G)}{\Pr(o_n = 1 | \mathbf{Y}, G)}}_{\gamma^1(n)}, \underbrace{\frac{\Pr(\hat{o}^0_{n1}, \dots, o_n = 0, \dots, \hat{o}^0_{n|N|} | \mathbf{Y}, G)}{\Pr(o_n = 0 | \mathbf{Y}, G)}}_{\gamma^0(n)} \Bigg), \tag{6}$$

where $\Pr(\hat{o}^1_{n1}, \dots, o_n = 1, .., \hat{o}^1_{n|N|} | \mathbf{Y}, G)$ and $\Pr(\hat{o}^0_{n1}, \dots, o_n = 0, \dots, \hat{o}^0_{n|N|} | \mathbf{Y}, G)$ are respectively the joint probability of the most likely status of the network given that node n is anomalous and non-anomalous, respectively. Variables $\hat{o}^1_{ni}$ and $\hat{o}^0_{ni}$ represent the conditional a posteriori estimates of o_i given that node n is in anomalous and non-anomalous modes, respectively. These variables can be estimated by their posterior means obtained from the K known MCMC samples with a burn-in period of k_0 ($k_0 \geq 0$) as follows:

$$\hat{o}^1_{ni} = \lfloor \frac{\sum_{k=k_0:K} o_i^{(k)} \times o_n^{(k)}}{\sum_{k=k_0:K} o_n^{(k)}} \rceil, \ \hat{o}^0_{ni} = \lfloor \frac{\sum_{k=k_0:K} o_i^{(k)} \times (1 - o_n^{(k)})}{\sum_{k=k_0:K} (1 - o_n^{(k)})} \rceil, \tag{7}$$

where $\lfloor \rceil$ represents the round to the nearest integer function. The denominators in Eq. (6) can be estimated by their posterior means (or

mode/median) from K MCMC samples as follows:

$$\Pr(o_n = 1|\mathbf{Y}, G) = \lfloor \frac{\sum_{k=k_0:K} o_n^{(k)}}{K - k_0 + 1} \rceil; \quad \text{and}$$
$$\Pr(o_n = 0|\mathbf{Y}, G) = 1 - \Pr(o_n = 1|\mathbf{Y}, G). \tag{8}$$

Once $\gamma(n)$ is calculated for all $n \in N$, we can add the node with a maximum value of γ to the set of inspection nodes (denoted by E_I), that is

$$E_I \longleftarrow E_I \cup \{\underset{n\in N, n\notin E_I}{\text{argmax}}\ \gamma(n)\}. \tag{9}$$

We can also establish additional criteria in the event of a tie in the variable γ, such as choosing the node with a greater number of connected nodes (either upstream or downstream). Similar to the baseline node inspection algorithm, both static and sequential scenarios can be considered for the refined method when $m > 1$. The details of the static and sequential refined node inspection methods are given in Algorithm 5 and Algorithm 6, respectively.

Algorithm 3 A Baseline Node Selection Algorithm for Inspection — Static

Input: Network Structure (G), Sensor Data ($\mathbf{Y} = [y_{nj}]_{|N|\times|S|}$), MCMC Samples $\boldsymbol{o}^{(\mathbf{1}:\boldsymbol{K})}$ and $\boldsymbol{x}^{(\mathbf{1}:\boldsymbol{K})}$, Number of Inspection Nodes (m), Propagation Matrix $\boldsymbol{P}$, and Estimated Network Status Variables $\hat{\boldsymbol{o}} = \{\hat{o}_1, ..., \hat{o}_{|N|}\}$ and $\hat{\boldsymbol{x}} = \{\hat{x}_1, ..., \hat{x}_{|N|}\}$
Output: Set E_I for Inspection and Updated Network Status Variables $\hat{\boldsymbol{o}}$ and $\hat{\boldsymbol{x}}$ After Inspection

I. Node Selection
1: Set $E_I = \{\}$
2: **while** $|E_I| < m$ **do**
3: $i \longleftarrow \underset{j\notin E_I}{\text{argmin}} \Pr(o_j|\mathbf{Y}, G)$ using the MCMC results.
4: $E_I \longleftarrow E_I \cup i$
5: **end while**

II. Inspection Step
6: Inspect and find the true values of o_n and x_n for all $n \in E_I$.

III. Update Selected Anomaly Status Variables Based on Inspection Outcomes
7: **for** $n \in E_I$ **do**
8: Run **Algorithm 2** for node n and obtain updated network status variables $\hat{o}$ and $\hat{x}$.
9: **end for**

IV. Update Estimated Network Anomaly Status Variables
10: Estimate network status variables $\hat{o}$ and $\hat{x}$ based on updated MCMC samples (see line 18 of **Algorithm 1**).

3.6. Efficient computation in the refined method

The implementation of the refined method discussed in Section 3.5 is computationally expensive, particularly because $\gamma(n)$, which needs to be calculated for all network nodes, has two terms where each term comprises elements that depend on other nodes in the network. A typical implementation of the refined method would need to sequentially compute $\gamma(n)$ and its constituent terms using two nested "for" loops. This sequential process contributes to the overall time-consuming nature of the implementation. Employing matrix operations instead of multiple "for" loops can often be a more efficient approach in many numerical libraries of programming languages (e.g., NumPy in Python, which we used for numerical experiments). Matrix operations are typically highly efficient and often parallelized to take advantage of multiple CPU cores. Additionally, they enable code vectorization, enhancing execution speed through optimized libraries like NumPy. The following Remarks are developed to elevate the efficiency of the method discussed in Section 3.5. The core concept behind these remarks is to convert the sequential computational procedures required for calculating γ for all nodes into matrix forms. This transformation enables the simultaneous computation of the necessary quantities for all nodes through the utilization of efficient matrix operations in a concise manner. For notational convenience, we discuss the remarks for $k_0 = 1$.

Algorithm 4 A Baseline Node Selection Algorithm for Inspection — Sequential

Input: Network Structure (G), Sensor Data ($\mathbf{Y} = [y_{nj}]_{|N|\times|S|}$), MCMC Samples $\boldsymbol{o}^{(\mathbf{1}:\boldsymbol{K})}$ and $\boldsymbol{x}^{(\mathbf{1}:\boldsymbol{K})}$, Number of Inspection Nodes (m), Propagation Matrix $\boldsymbol{P}$, and Estimated Network Status Variables $\hat{\boldsymbol{o}} = \{\hat{o}_1, ..., \hat{o}_{|N|}\}$ and $\hat{\boldsymbol{x}} = \{\hat{x}_1, ..., \hat{x}_{|N|}\}$
Output: Set E_I for Inspection and Updated Network Status Variables $\hat{\boldsymbol{o}}$ and $\hat{\boldsymbol{x}}$ After Inspection

1: Set $E_I = \{\}$
2: **while** $|E_I| < m$ **do**
I. Node Selection
3: $i \longleftarrow \underset{j\notin E_I}{\text{argmin}} \Pr(o_j|o_{E_I}, \mathbf{Y}, G)$ using the MCMC results.
4: $E_I \longleftarrow E_I \cup i$
II. Inspection Step (See Algorithm 3)
III. Update Selected Network Anomaly Status Variables (See Algorithm 3)
IV. Update All Network Anomaly Status Variables
5: Filter MCMC samples obtained from **Algorithm 1** based on the outcomes of I-III.
6: Estimate network status variables $\hat{o}$ and $\hat{x}$ based on updated MCMC samples (see line 18
7: of **Algorithm 1**).
8: **end while**

Remark 1. Denote the set of known MCMC samples for all nodes from iteration 1 to K (i.e., the outputs of Algorithm 1) and the unknown posterior estimates $\hat{o}_{ni}^1$ and $\hat{o}_{ni}^0$ for all n and $i \in N$ given in Eq. (7) in matrix forms as follows:

$$\mathbf{O}^{(1:K)} = \begin{bmatrix} o_1^{(1)} & \cdots & o_{|N|}^{(1)} \\ \vdots & \vdots & \vdots \\ o_1^{(k)} & \cdots & o_{|N|}^{(k)} \\ \vdots & \vdots & \vdots \\ o_1^{(K)} & \cdots & o_{|N|}^{(K)} \end{bmatrix}, \hat{\mathbf{O}}^1 = \begin{bmatrix} \hat{o}_{11}^1 & \cdots & \hat{o}_{1|N|}^1 \\ \vdots & \vdots & \vdots \\ \hat{o}_{n1}^1 & \cdots & \hat{o}_{n|N|}^1 \\ \vdots & \vdots & \vdots \\ \hat{o}_{|N|1}^1 & \cdots & \hat{o}_{|N||N|}^1 \end{bmatrix},$$
$$\hat{\mathbf{O}}^0 = \begin{bmatrix} \hat{o}_{11}^0 & \cdots & \hat{o}_{1|N|}^0 \\ \vdots & \vdots & \vdots \\ \hat{o}_{n1}^0 & \cdots & \hat{o}_{n|N|}^0 \\ \vdots & \vdots & \vdots \\ \hat{o}_{|N|1}^0 & \cdots & \hat{o}_{|N||N|}^0 \end{bmatrix}.$$

Let us also define the following square matrices based on MCMC sample matrix $\mathbf{O}^{(1:K)}$:

$$\boldsymbol{\Psi}_{|N|\times|N|}^1 = \mathbf{O}^{(1:K)^T} \times \mathbf{O}^{(1:K)}, \boldsymbol{\Psi}_{|N|\times|N|}^0 = \left[[\mathbf{1}]_{|N|\times K} - \mathbf{O}^{(1:K)}\right]^T \times \mathbf{O}^{(1:K)^T},$$
$$\mathbf{D}_{|N|\times|N|}^1 = \text{diag}(\frac{1}{\boldsymbol{\Psi}_{11}^1}, \frac{1}{\boldsymbol{\Psi}_{22}^1}, \ldots, \frac{1}{\boldsymbol{\Psi}_{|N||N|}^1}),$$
$$\mathbf{D}_{|N|\times|N|}^0 = \text{diag}(\frac{1}{K - \boldsymbol{\Psi}_{11}^1}, \frac{1}{K - \boldsymbol{\Psi}_{22}^1}, \ldots, \frac{1}{K - \boldsymbol{\Psi}_{|N||N|}^1}).$$

Here, $[\mathbf{1}]_{|N|\times K}$ is a matrix consisting of all ones, $\boldsymbol{\Psi}_{ni}^1$ denotes the ni-th elements of the square matrix $\boldsymbol{\Psi}^1$, and "diag(.)" represents a diagonal square matrix with a given set of diagonal elements. Now, the unknown

elements of matrices $\hat{\mathbf{O}}^1$ and $\hat{\mathbf{O}}^0$ can be simultaneously computed as follows:

$$\hat{\mathbf{O}}^1 = \lfloor \mathbf{D}^1 \times \mathbf{\Psi}^1 \rceil \quad \text{and} \quad \hat{\mathbf{O}}^0 = \lfloor \mathbf{D}^0 \times \mathbf{\Psi}^0 \rceil. \tag{10}$$

Proof of Remark 1. The validity of the above remark can be established by examining the element of matrices $\mathbf{\Psi}^1$ and $\mathbf{\Psi}^0$. Given that (a) the kth row of matrix $\mathbf{O}^{(1:K)}$ represents the kth sequence of MCMC samples for all nodes and (b) matrix $\mathbf{O}^{(1:K)}$ is a binary matrix, the ni-th element of matrix $\mathbf{\Psi}^1$ (e.g., $\Psi^1_{ni} = \sum_{k=1}^{K} o_n^{(k)} o_i^{(k)}$) is the numerator of $\hat{o}^1_{ni}$ in Eq. (7). Also, the nth diagonal elements of matrix $\mathbf{\Psi}^1$ (e.g., $\Psi^1_{nn} = \sum_{k=1}^{K} o_n^{(k)}$) serve as the denominator of $\hat{o}^1_{ni}$ in Eq. (7). In other words, by constructing the diagonal matrix $\mathbf{D}^1$ with the inverses of the diagonal elements of matrix $\mathbf{\Psi}^1$, denoted as $\mathbf{D}^1_{nn} = \frac{1}{\sum_{k=1}^{K} o_n^{(k)}}$, we can directly calculate the average number of samples having values 1, as needed in the denominators of $\hat{o}^1_{ni}$ in Eq. (7). Now, the multiplication of the matrix $\mathbf{D}^1$ by $\mathbf{\Psi}^1$ provides a square matrix that includes all the terms needed for the left measure in Eq. (7) for all combinations of two nodes. To mitigate numerical instability and prevent division by zero errors, we add a small epsilon value to the denominator of $\mathbf{D}^0$ and $\mathbf{D}^1$. A similar relationship can be derived for $\hat{\mathbf{O}}^0$, $\mathbf{D}^0$, and $\mathbf{\Psi}^0$ to calculate $\hat{o}^0_{ni}$ in Eq. (7). This completes the proof. □

Now, to simultaneously calculate the terms in $\gamma(n)$ (Eq. (6)) for all nodes, we introduce Remark 2.

Remark 2. Denote $\hat{\mathbf{O}}^1_n$ and $\hat{\mathbf{O}}^0_n$ as the nth row of matrices $\hat{\mathbf{O}}^1$ and $\hat{\mathbf{O}}^0$, respectively. Also, denote $\mathbf{O}^{(1:K)}_k$ as the kth row of matrix $\mathbf{O}^{(1:K)}$. We can compare each row of matrix $\hat{\mathbf{O}}^1$ and $\hat{\mathbf{O}}^0$ with matrix $\mathbf{O}^{(1:K)}$ and compute the number of times each sample $\hat{\mathbf{O}}^1_n$ and $\hat{\mathbf{O}}^0_n$ is repeated in the MCMC samples. Then, by dividing those counts for node n by the number of times the variable $o_n^{(k)}$ is 1 and 0 in the MCMC samples, we can simultaneously compute the two terms in Eq. (6) for all nodes as follows:

$$\begin{bmatrix} \gamma^1(1) \\ \vdots \\ \gamma^1(n) \\ \vdots \\ \gamma^1(|N|) \end{bmatrix} = \mathbf{D}^1 \times \begin{bmatrix} \sum_{k=1}^{K} \mathbf{I}\{\hat{\mathbf{O}}^1_1 = \mathbf{O}^{(1:K)}_k\} \\ \vdots \\ \sum_{k=1}^{K} \mathbf{I}\{\hat{\mathbf{O}}^1_n = \mathbf{O}^{(1:K)}_k\} \\ \vdots \\ \sum_{k=1}^{K} \mathbf{I}\{\hat{\mathbf{O}}^1_{|N|} = \mathbf{O}^{(1:K)}_k\} \end{bmatrix},$$

$$\begin{bmatrix} \gamma^0(1) \\ \vdots \\ \gamma^0(n) \\ \vdots \\ \gamma^0(|N|) \end{bmatrix} = \mathbf{D}^0 \times \begin{bmatrix} \sum_{k=1}^{K} \mathbf{I}\{\hat{\mathbf{O}}^0_1 = \mathbf{O}^{(1:K)}_k\} \\ \vdots \\ \sum_{k=1}^{K} \mathbf{I}\{\hat{\mathbf{O}}^0_n = \mathbf{O}^{(1:K)}_k\} \\ \vdots \\ \sum_{k=1}^{K} \mathbf{I}\{\hat{\mathbf{O}}^0_{|N|} = \mathbf{O}^{(1:K)}_k\} \end{bmatrix}. \tag{11}$$

Here, $\mathbf{I}\{.\}$ is an indicator function that returns 1 if the condition inside $\{.\}$ is true and 0 otherwise. For example, $\mathbf{I}\{\hat{\mathbf{O}}^1_n = \mathbf{O}^{(1:K)}_k\}$ is 1 if all the elements in the nth row of matrix $\hat{\mathbf{O}}^1_n$ match the elements in the kth row of matrix $\mathbf{O}^{(1:K)}_k$. The sum (i.e., $\sum_{k=1}^{K} \mathbf{I}\{\hat{\mathbf{O}}^1_n = \mathbf{O}^{(1:K)}_k\}$) adds up as the number of times this condition is true for all K MCMC samples. Now by dividing the sum by the total number of samples where $o_n = 1$, we can calculate $\gamma^1(n)$ in Eq. (6). Similar computation applies to calculate $\gamma^0(n)$. Now, it can be noticed that the entire computation of $\gamma(n)$ for all nodes in the network can be streamlined into a series of matrix multiplications (i.e., Eq. (10)) and row-wise matrix alignment processes (i.e., Eq. (11)). These operations are computationally efficient and can be readily implemented in various programming languages, such as Python using NumPy, as utilized in this article. We should also point out that since most of the elements of the MCMC sample matrix $\mathbf{O}^{1:K}$ are zero, algorithms available for sparse matrix multiplication can take advantage of this sparsity to reduce the number of operations required, leading to faster computations. Please refer to Section 4.4 and Table 7 to see how the proposed framework can be efficiently implemented on different network topologies within a reasonable timeframe.

Algorithm 5 A Refined Node Selection Algorithm for Inspection — Static

Input: Network Structure (G), Sensor Data ($\mathbf{Y} = [y_{nj}]_{|N|\times|S|}$), MCMC Samples in matrix form $\mathbf{O}^{(1:K)}$ and $\boldsymbol{X}^{(1:K)}$, Number of Inspection Nodes (m), Propagation Matrix $\boldsymbol{P}$, and Estimated Network Status Variables $\hat{o} = \{\hat{o}_1, ..., \hat{o}_{|N|}\}$ and $\hat{\boldsymbol{x}} = \{\hat{x}_1, ..., \hat{x}_{|N|}\}$

Output: Set E_I for Inspection and Updated Network Status Variables in matrix form After Inspection

I. Node Selection

1: Set $E_I = \{\}$

2: Compute $\hat{\mathbf{O}}^1$ and $\hat{\mathbf{O}}^0$ from Eq. (10)

3: Compute $\left[\gamma^1(1) \cdots \gamma^1(n) \cdots \gamma^1(|N|)\right]^T$ and $\left[\gamma^0(1) \cdots \gamma^0(n) \cdots \gamma^0(|N|)\right]^T$ from Eq. (11).

4: **while** $|E_I| < m$ **do**

5: $\quad i \longleftarrow \underset{n\in N, n\notin E_I}{\operatorname{argmax}} \gamma(n)$.

6: $\quad E_I \longleftarrow E_I \cup i$

7: **end while**

II. Inspection Step (See Algorithm 3)

III. Update Selected Network Anomaly Status Variables (See Algorithm 3)

IV. Update All Network Anomaly Status Variables (See Algorithm 3)

4. Numerical experiments

A comprehensive set of randomly generated graphs with different network configurations is used to evaluate the performance of the proposed methods in the paper. Our objective is to demonstrate the effectiveness of node inspection, compare variations among these methods, and delve into their functionality across various network setups. These setups encompass factors such as the number of sensor attributes, sensor power levels, and the total number of nodes. The inputs for the entire network node selection phase are the stochastic samples that are the outcomes of the MCMC sampler (Algorithm 1). We also compare our results with two simplistic random node inspection policies, illustrating the benefits of employing the proposed methods. Lastly, we conduct a numerical assessment of the computational complexity of each method.

4.1. Experiment setup

To evaluate the effectiveness of the proposed inspection methods with respect to the size of the network (number of nodes), the number of sensor attributes, the power of sensors, the topology of networks, and the number of nodes selected for inspection, we generated a comprehensive set of random graphs, including three types of widely-used directed acyclic graphs (DAG): (i) regular DAG (a non-tree DAG where every node has about the same degree), (ii) tree DAG (a DAG that follows a tree structure with no nodes having more than one parent), and (iii) Watts Strogatz (WS) DAG (a non-tree DAG that interpolates between the regular and the Erdos–Renyi graph, which is a non-tree graph that connects two nodes based on the edge probability or the

Algorithm 6 A Refinded Node Selection Algorithm for Inspection — Sequential

Input: Network Structure (G), Sensor Data ($\mathbf{Y} = [y_{nj}]_{|N|\times|S|}$), MCMC Samples $\boldsymbol{o}^{(\mathbf{1}:K)}$ and $\mathbf{X}^{(1:K)}$, Number of Inspection Nodes (m), Propagation Matrix $\boldsymbol{P}$, and Estimated Network Status Variables $\hat{\boldsymbol{o}} = \{\hat{o}_1, ..., \hat{o}_{|N|}\}$ and $\hat{\boldsymbol{x}} = \{\hat{x}_1, ..., \hat{x}_{|N|}\}$

Output: Set E_I for Inspection and Updated Network Status Variables $\hat{\boldsymbol{o}}$ and $\hat{\boldsymbol{x}}$ After Inspection

1: **Node Selection**
2: Set $E_I = \{\}$
3: **while** $|E_I| < m$ **do**
4: Compute $\hat{\mathbf{O}}^1$ and $\hat{\mathbf{O}}^0$ from Eq. (10)
5: Compute $\left[\gamma^1(1)\cdots\gamma^1(n)\cdots\gamma^1(|N|)\right]^T$ and $\left[\gamma^0(1)\cdots\gamma^0(n)\cdots\gamma^0(|N|)\right]^T$ from Eq. (11).
6: $i \longleftarrow \underset{n\in N, n\notin E_I}{\operatorname{argmax}}\ \gamma(n)$.
7: $E_I \longleftarrow E_I \cup i$.

II. Inspection Step (See Algorithm 3)

III. Update Selected Network Anomaly Status Variables (See Algorithm 3)

IV. Update All Network Anomaly Status Variables

8: Filter MCMC samples obtained from **Algorithm 1** based on the outcomes of I-III.
9: Estimate network status variables $\hat{\boldsymbol{o}}$ and $\hat{\boldsymbol{x}}$ based on updated MCMC samples (see line 18
10: of **Algorithm 1**).
11: **end while**

edges selected uniformly at random). For each type of topology, we considered four configurations of sensors, representing different levels of sensor power, strong sensors $\{\alpha = 0.9, \beta = 0.1\}$, moderate sensors $\{\alpha = 0.8, \beta = 0.2\}$, mild sensors $\{\alpha = 0.7, \beta = 0.3\}$, and weak sensors $\{\alpha = 0.6, \beta = 0.4\}$, where a higher α and a lower β makes a stronger and more reliable sensor. In addition, we will discuss the effectiveness of the node inspection methods with respect to the size of the networks. To do that, we consider three scenarios (i.e., $|N| \in \{500, 1000, 2000\}$) of the number of sensors for each type and configuration of the network. When interpreting the results for the Watts Strogatz DAGs, the readers should note that the graph simulator (igraph) used to generate random graphs generated networks with a slightly smaller size, specifically $|N| \in \{495, 991, 1973\}$.

4.2. Main observations

For each combination of the network topology, number of nodes, power of sensors, and number of sensors, we applied the baseline node inspection model and refined node inspection model, in both static and sequential manners. To evaluate the performance of these models, we calculated key metrics, including true positives or TPs (correctly identified anomalous nodes), true negatives or TNs (correctly identified non-anomalous nodes), false positives or FPs (incorrectly identified anomalous nodes), and false negatives or FNs (missed anomalies). These metrics were based on the estimated status of all nodes within the network. Additionally, we tracked whether the source of an anomaly could be successfully detected. To ensure robust results, each experiment was repeated 100 times. This approach allowed us to obtain more reliable and statistically significant outcomes by averaging the results across multiple runs. The rest of this section presents the results of the node inspection experiments and provides comparisons among different methods and network configurations.

4.2.1. Performance evaluation metrics

Our experiments effectively assessed the performance of both the baseline and refined inspection models, confirming their efficacy in improving anomaly detection. We report three widely recognized performance metrics to evaluate the results: the F1-score, the overlap coefficient, and the source detection accuracy (SDA). The F1-score, a standard comprehensive metric in binary classification, offers a balanced view of precision and recall. It measures the accuracy of detecting network anomalies while considering the potential trade-offs between precision and recall, with precision representing the percentage of correctly identified anomalies and recall indicating the percentage of anomalies identified out of the total number of nodes in each network. A higher F1-score indicates better overall performance in terms of anomaly detection accuracy. The formula used to calculate F1-score is as follows:

$$\text{F1-Score} = \frac{2\times TP}{2\times TP + FP + FN}.$$

The overlap coefficient, which calculates the degree of overlap between the actual anomalies and those correctly identified by the model, serves as a valuable measure of anomaly detection efficiency in networks. A higher overlap coefficient (closer to 1) is an indicator of better performance in anomaly detection. This measure is calculated by dividing the number of true positives by the sum of true positives, false negatives, and false positives, as shown below

$$\text{Overlap Coefficient} = \frac{TP}{TP + FP + FN}.$$

Neither the F1-score nor the overlap coefficient can effectively evaluate the performance of a model in terms of correctly identifying the source of anomalies. To address this, we utilize the source detection accuracy (SDA) metric, which measures the model's ability to accurately pinpoint the source or origin of anomalies within a network. The SDA can be defined as the ratio of the number of correctly identified source nodes to the total number of source nodes as shown below

$$\text{SDA} = \frac{\text{Number of Correctly Identified Anomalous Source Nodes}}{\text{Total Number of Anomalous Source Nodes}}.$$

By evaluating these metrics, we can gain a fair and comprehensive understanding of the improvements in anomaly detection that result from conducting node inspections. To better illustrate the potential outcome of the proposed models, we first provide a simple example below. In Fig. 3, a 495-node Watts Strogatz network with 14 anomalous nodes is shown. Prior to conducting physical inspections, a total of 34 nodes were estimated to be anomalous, resulting in an anomaly detection accuracy of 96%. However, this accuracy was accompanied by a high false positive rate (FPR) of 56%, primarily due to the identification of 19 false positive nodes. Following the implementation of the refined node selection model, which involved inspecting only two nodes ($m = 2$) sequentially, the accuracy of the anomaly detection task surged to 100%. This remarkable improvement occurred because both of the inspected nodes corresponded to previously estimated false positive nodes. As a result, the inspection outcome confirmed the non-anomalous status of these nodes and their upstream nodes, leading to the removal of all false positive nodes. In this example, the inspection model intelligently selected two non-anomalous nodes that happened to be the parents of the true source node and had been initially misclassified as anomalous. This serves as a compelling demonstration of the effectiveness of the inspection model. By selecting and inspecting two closely positioned nodes in proximity to the true source of anomalies, the accuracy of network anomaly detection significantly improved, showcasing the potential impact of intelligent node selection on anomaly detection outcomes.

4.2.2. Comparison between the baseline and refined node selection models

In Tables 2–3, we provide a summary of the outcomes obtained from our numerical experiments on 1000-node networks, with a single sensor attribute collected at each node. We considered both randomly

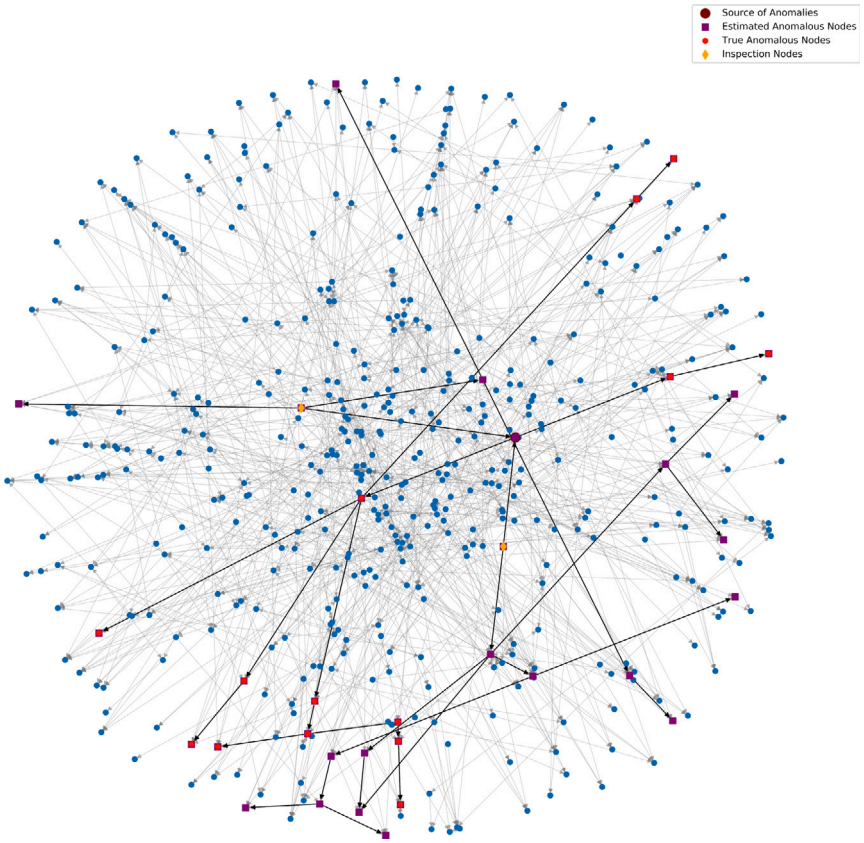

Fig. 3. An example of a Watts Strogatz network, where the mahogany node indicates the source of anomalies, small red nodes are the true impacted nodes, purple square nodes are the previously estimated anomalous nodes prior to inspection, and yellow diamond nodes are the two nodes selected for sequential inspection.

Table 2
Performance metrics for tree and regular DAGs ($|N| = 1000$, $|S| = 1$).

Topology			Tree			Regular		
Sensor Properties								
α	β	Before/After	F1-score	Overlap Coefficient	SDA	F1-score	Overlap Coefficient	SDA
0.6	0.4	Before Inspection	0.7113	0.5520	0.1200	NA	0	0
		$m = 1$ Baseline	0.8442	0.7304	0.4200	0.1075	0.0568	0.0400
		$m = 1$ Refined	0.7843	0.6452	0.3100	0.1089	0.0576	0.0400
0.7	0.3	Before Inspection	0.9213	0.8540	0.6300	0.1209	0.0643	0.0400
		$m = 1$ Baseline	0.9708	0.9432	0.8500	0.2625	0.1511	0.1000
		$m = 1$ Refined	0.9402	0.8871	0.7100	0.2985	0.1754	0.1000
0.8	0.2	Before Inspection	0.9880	0.9764	0.8400	0.2854	0.1665	0.1100
		$m = 1$ Baseline	0.9973	0.9946	0.9500	0.4177	0.2640	0.1414
		$m = 1$ Refined	0.9994	0.9988	0.9600	0.5664	0.3951	0.3000
0.9	0.1	Before Inspection	0.9999	0.9998	0.9900	0.6489	0.4803	0.4100
		$m = 1$ Baseline	0.9999	0.9998	0.9900	0.7356	0.5818	0.3579
		$m = 1$ Refined	0.9999	0.9998	0.9900	0.8696	0.7693	0.6400

generated tree and regular DAGs. For these experiments, we applied both the baseline and refined node selection methods in a static manner considering only one node for inspection (i.e., $m = 1$). As shown in Tables 2–3, both the baseline node selection and the refined node selection models for inspection exhibited a substantial improvement in the accuracy of detecting anomalies and their sources before inspection across all experiment configurations with inspecting only one node. Similar results were obtained for larger networks, and larger numbers of sensor attributes (as can be seen in Tables 3–6). This underscores the effectiveness of intelligent inspection in enhancing the performance of the anomaly detection task. The findings presented in Tables 2–3 also reveal that the refined node selection model for inspection outperforms the baseline version in the case of Regular and WS networks. However, for tree networks with weaker sensors, the results show a slightly better performance for the baseline node selection method. Interestingly, the refined model proves to be equally effective or superior to the baseline model for tree networks equipped with stronger sensors. To facilitate a direct comparison between the baseline and the refined node selection methods, Table 3 highlights the differences between these two approaches on the Watts Strogatz networks and two network sizes of 1000 and 2000. The table clearly demonstrates that the refined method surpasses the baseline method in performance. Consequently, we can employ the baseline method to enhance anomaly detection in simpler networks, such as tree networks.

Table 3
Performance Metrics for Watts Strogatz DAGs ($|S| = 1$).

Number of Nodes			$\|N\| = 1{,}000$			$\|N\| = 2{,}000$		
Sensor Properties								
α	β	Before/After	F1-score	Overlap Coefficient	SDA	F1-score	Overlap Coefficient	SDA
0.6	0.4	Before Inspection	0.2939	0.1723	0.0200	NA	0	0
		$m = 1$ Baseline	0.3238	0.1932	0.0300	0.0525	0.0270	0
		$m = 1$ Refined	0.3170	0.1883	0.0200	0.2096	0.1170	0.0300
0.7	0.3	Before Inspection	0.5709	0.3994	0.1000	0.3450	0.2085	0.0800
		$m = 1$ Baseline	0.6368	0.4672	0.1500	0.4043	0.2533	0.0700
		$m = 1$ Refined	0.6836	0.5193	0.2000	0.5765	0.4050	0.1700
0.8	0.2	Before Inspection	0.7173	0.5592	0.1800	0.6240	0.4535	0.2000
		$m = 1$ Baseline	0.7323	0.5777	0.1939	0.6965	0.5344	0.2020
		$m = 1$ Refined	0.8174	0.6911	0.3200	0.7789	0.6379	0.3000
0.9	0.1	Before Inspection	0.8060	0.6750	0.3500	0.7494	0.5992	0.3300
		$m = 1$ Baseline	0.8529	0.7435	0.3636	0.8010	0.6681	0.3125
		$m = 1$ Refined	0.9086	0.8325	0.5600	0.8640	0.7605	0.4000

However, for more intricate networks, the refined method emerges as the superior choice for achieving improved anomaly detection results.

4.2.3. Comparison between static and sequential inspection strategies

To effectively compare the static and sequential node selection for inspection methods, it is necessary to conduct experiments with more than one inspected node (i.e., $m > 1$). This is because both static and sequential methods yield equivalent results when $m = 1$. Table 4 is provided to compare the performance of the baseline node inspection model with the refined node inspection model in both static and sequential manners for $m = 2$ for 3 types of networks and four sensor power configurations. It can be seen that across nearly all experiments, the sequential node selection policy outperforms the static node selection policy. However, the extent of improvement is less significant when applied to tree and Regular networks, especially in scenarios where sensors possess higher capabilities (e.g., when $\alpha = 0.9$ and $\beta = 0.1$). Similar trends have been observed for networks of different sizes and with a greater number of sensor attributes. However, due to space constraints, detailed results for these cases are not provided. In conclusion, it is evident that employing the node inspection policy in a sequential manner enhances the accurate identification of anomalous nodes and their sources.

4.2.4. Effectiveness of increasing inspection efforts

One practical limitation of implementing inspections is the time and cost involved. Therefore, it is desirable to minimize the number of nodes subjected to inspection. By comparing the results in Tables 2–3 with Table 4, we can see that increasing the number of inspection nodes can have a positive impact on the task of anomaly detection for most cases with some variations among different network setups. To investigate whether increasing the number of inspected nodes contributes to improvements in anomaly detection and to assess the extent of this improvement, we compared the cases with 1, 2, and 5 nodes for inspection in the baseline node selection model, and 1 and 2 nodes for inspection in the refined node selection model. The results consistently indicated that augmenting the number of nodes subjected to inspection enhances the performance of anomaly detection across most scenarios. However, it is important for the model user to carefully consider the cost–benefit trade-offs, resource constraints, and the expected impact on detection accuracy associated with conducting additional inspections and determine the optimal number of nodes to select for inspection based on the specific context and constraints. Regular monitoring, experiments on historical data, and expert input can help guide this decision.

4.2.5. Sensitivity analysis with respect to the number of sensor attributes

In Table 5, we present the results of our proposed node inspection methods for 1000-node networks with five sensor attributes ($|S| = 5$) across three network topology types. Comparing this table with Table 4 where $|S| = 1$ shows that increasing the number of sensor attributes leads to improved performance in the original anomaly detection task. Although the benefits of node inspection diminish as the number of sensors per node increases, both the baseline and refined node inspection models consistently enhance anomaly detection performance. For networks with five sensors at each node and strong sensor performance (e.g., when $\alpha = 0.9$ and $\beta = 0.1$), anomaly detection accuracy approaches nearly 100%. Consequently, the improvement from pre-inspection to node inspection decreases, and the difference between static and sequential node selection approaches becomes less significant. These findings suggest that the choice of node selection approach has a limited impact on performance when sensor attributes are abundant and perform strongly. However, node selection becomes critical for networks with fewer sensor attributes and weaker sensor performance. It is worth noting that similar trends were observed across networks of different sizes, although detailed results for those cases are not presented due to space constraints.

4.2.6. Sensitivity analysis with respect to the power of sensors

The results presented so far in Tables 2–5 consistently reveal that as sensor reliability improves, characterized by higher α values and lower β values, the overall performance of the original anomaly detection methods also sees significant enhancements. This trend suggests that with more reliable sensors, there is less room for improvement through node inspections, regardless of the selection approach employed. For instance, taking Table 5 as an example, when dealing with weak sensors ($\alpha = 0.6, \beta = 0.4$), the Source Detection Accuracy (SDA) demonstrates substantial improvement from the no-inspection policy, with an approximate increase ratio of 300%. As the sensors become moderately reliable ($\alpha = 0.7, \beta = 0.3$), the rate of improvement remains notable but decreases to approximately 43%. Finally, with strong sensors ($\alpha = 0.9, \beta = 0.1$), the improvement is limited, and the performance of the anomaly detection system is already close to achieving optimal accuracy. In conclusion, these results emphasize the sensitivity of anomaly detection performance to sensor reliability. As sensors become more powerful, the potential gains from node inspections diminish. Therefore, the choice of node inspection strategy becomes particularly critical when dealing with networks featuring less reliable sensors, where inspections can have a substantial impact on anomaly detection performance.

Table 4
Comparison between static vs. sequential policies ($|N| = 1000, |S| = 1, m = 2$).

WS Networks				Baseline Model			Refined Model		
Sensor Properties									
α	β	Before/After		F1-score	Overlap Coefficient	SDA	F1-score	Overlap Coefficient	SDA
0.6	0.4	Before Inspection		0.2939	0.1723	0.0200	0.2939	0.1723	0.0200
		$m = 2$	Static	0.3250	0.1940	0.0404	0.3256	0.1944	0.0300
			Sequential	0.3211	0.1913	0.0400	0.3235	0.1930	0.0300
0.7	0.3	Before Inspection		0.5709	0.3994	0.1000	0.5709	0.3994	0.1000
		$m = 2$	Static	0.6521	0.4838	0.1500	0.7111	0.5518	0.1919
			Sequential	0.6598	0.4923	0.2000	0.7119	0.5526	0.1939
0.8	0.2	Before Inspection		0.7173	0.5592	0.1800	0.7173	0.5592	0.1800
		$m = 2$	Static	0.7728	0.6297	0.2268	0.7991	0.6654	0.2929
			Sequential	0.7675	0.6227	0.2268	0.8075	0.6772	0.3131
0.9	0.1	Before Inspection		0.8060	0.6750	0.3500	0.8060	0.6750	0.3500
		$m = 2$	Static	0.8657	0.7632	0.4271	0.9019	0.8214	0.5000
			Sequential	0.8766	0.7803	0.4700	0.9002	0.8185	0.5051
Regular Networks				Baseline Model			Refined Model		
Sensor Properties									
α	β	Before/After		F1-score	Overlap Coefficient	SDA	F1-score	Overlap Coefficient	SDA
0.6	0.4	Before Inspection		NA	0	0	NA	0	0
		$m = 2$	Static	0.1694	0.0926	0.0700	0.1694	0.0926	0.0700
			Sequential	0.1406	0.0756	0.0600	0.1694	0.0926	0.0700
0.7	0.3	Before Inspection		0.1209	0.0643	0.0400	0.1209	0.0643	0.0400
		$m = 2$	Static	0.3489	0.2113	0.1414	0.3456	0.2089	0.1600
			Sequential	0.3368	0.2025	0.1400	0.3456	0.2089	0.1600
0.8	0.2	Before Inspection		0.2854	0.1665	0.1100	0.2854	0.1665	0.1100
		$m = 2$	Static	0.4664	0.3041	0.2188	0.5806	0.4090	0.2828
			Sequential	0.4501	0.2904	0.1919	0.5882	0.4166	0.2857
0.9	0.1	Before Inspection		0.6489	0.4803	0.4100	0.6489	0.4803	0.4100
		$m = 2$	Static	0.7916	0.6551	0.4783	0.8647	0.7617	0.5859
			Sequential	0.7829	0.6432	0.4947	0.8631	0.7592	0.5816
Tree Networks				Baseline Model			Refined Model		
Sensor Properties									
α	β	Before/After		F1-score	Overlap Coefficient	SDA	F1-score	Overlap Coefficient	SDA
0.6	0.4	Before Inspection		0.7113	0.5520	0.1200	0.7113	0.5520	0.1200
		$m = 2$	Static	0.8575	0.7506	0.4600	0.8238	0.7004	0.4100
			Sequential	0.8581	0.7515	0.4700	0.8238	0.7004	0.4100
0.7	0.3	Before Inspection		0.9213	0.8540	0.6300	0.9213	0.8540	0.6300
		$m = 2$	Static	0.9708	0.9432	0.8500	0.9549	0.9136	0.7800
			Sequential	0.9708	0.9432	0.8500	0.9572	0.9178	0.7900
0.8	0.2	Before Inspection		0.9880	0.9764	0.8400	0.9880	0.9764	0.8400
		$m = 2$	Static	0.9972	0.9945	0.9596	0.9994	0.9988	0.9600
			Sequential	0.9973	0.9946	0.9500	0.9994	0.9988	0.9600
0.9	0.1	Before Inspection		0.9999	0.9998	0.9900	0.9999	0.9998	0.9900
		$m = 2$	Static	0.9999	0.9998	0.9900	0.9999	0.9998	0.9900
			Sequential	0.9999	0.9998	0.9900	0.9999	0.9998	0.9900

4.3. Comparison with random inspection policies

In all previous experiments, we discussed the performance of the models with respect to no inspection. In this section, we discuss how the proposed node inspection models can provide significantly better results than random policies for inspection. We considered 2 random policies as follows:

Random Inspection Policy I: In this random inspection policy, nodes are selected for inspection using a uniform discrete distribution, which means that all nodes (excluding leaf nodes) have an equal probability of being chosen for inspection. This approach ensures that the selection process is random and unbiased, with each eligible node having an equal opportunity to be inspected.

Random Inspection Policy II: Several extensions of Random Inspection Policy I may be considered, where nodes do not necessarily have an equal likelihood of being selected for inspection. One such example involves weighing each node based on the number of downstream or upstream nodes. In this article, after exploring various possible random policies, we adopt a policy in which nodes are chosen based on rankings determined by the proportions of estimated anomalous downstream nodes. In essence, between two nodes, the one with a higher percentage of estimated anomalous nodes is selected for inspection. Unlike the Random Policy Inspection I, this model benefits from utilizing the outcomes of the anomaly detection framework prior to inspection. Thus, it represents a hybrid policy that combines randomness with information derived from sensor data. It is worth noting that numerous similar policies could be considered, and we included only this policy due to the page limit and as it performed slightly better than comparable policies in our numerical experiments. In the event of a tie between two nodes with the same proportion of estimated anomalous downstream

Table 5
Performance metrics in 1000-Node networks when $|S| = 5$.

WS Networks				Baseline Model			Refined Model		
Sensor Properties									
α	β	Before/After		F1-score	Overlap Coefficient	SDA	F1-score	Overlap Coefficient	SDA
0.6	0.4	Before Inspection		0.6295	0.4593	0.14	0.6295	0.4593	0.14
		$m = 2$	Static	0.6639	0.4969	0.22	0.6601	0.4927	0.3448
			Sequential	0.6807	0.516	0.27	0.6601	0.4927	0.3448
0.7	0.3	Before Inspection		0.7957	0.6608	0.34	0.7957	0.6608	0.34
		$m = 2$	Static	0.837	0.7197	0.4124	0.8085	0.6785	0.4103
			Sequential	0.8408	0.7253	0.4592	0.8370	0.7197	0.4783
0.8	0.2	Before Inspection		0.9102	0.8352	0.65	0.9102	0.8352	0.65
		$m = 2$	Static	0.9340	0.8761	0.7053	0.9632	0.9291	0.7925
			Sequential	0.9377	0.8827	0.7041	0.9521	0.9087	0.7937
0.9	0.1	Before Inspection		0.9535	0.9111	0.8	0.9535	0.9111	0.8
		$m = 2$	Static	0.9339	0.9682	0.8970	0.9663	0.9348	0.8182
			Sequential	0.9860	0.9723	0.9149	0.9753	0.9518	0.8519
Regular Networks				Baseline Model			Refined Model		
Sensor Properties									
Alpha	Beta	Before/After		F1-score	Overlap Coefficient	SDA	F1-score	Overlap Coefficient	SDA
0.6	0.4	Before Inspection		0.1440	0.0776	0.0400	0.1440	0.0776	0.0400
		$m = 2$	Static	0.4198	0.2657	0.1546	0.4014	0.2511	0.1753
			Sequential	0.3848	0.2382	0.1600	0.4227	0.2680	0.1856
0.7	0.3	Before Inspection		0.6325	0.4625	0.3400	0.6325	0.4625	0.3400
		$m = 2$	Static	0.7582	0.6106	0.4149	0.7918	0.6553	0.5051
			Sequential	0.7617	0.6151	0.4421	0.7945	0.6591	0.5102
0.8	0.2	Before Inspection		0.9217	0.8547	0.7300	0.9217	0.8547	0.7300
		$m = 2$	Static	0.9561	0.9158	0.8125	0.9662	0.9345	0.8469
			Sequential	0.9641	0.9307	0.8469	0.9629	0.9285	0.8454
0.9	0.1	Before Inspection		0.9854	0.9711	0.9400	0.9854	0.9711	0.9400
		$m = 2$	Static	0.9943	0.9887	0.9697	0.9972	0.9943	0.9697
			Sequential	0.9938	0.9876	0.9600	0.9972	0.9943	0.9697
Tree Networks				Baseline Model			Refined Model		
Sensor Properties									
Alpha	Beta	Before/After		F1-score	Overlap Coefficient	SDA	F1-score	Overlap Coefficient	SDA
0.6	0.4	Before Inspection		0.9256	0.8615	0.6600	0.9256	0.8615	0.6600
		$m = 2$	Static	0.9700	0.9418	0.8500	0.9541	0.9122	0.7800
			Sequential	0.9700	0.9418	0.8500	0.9541	0.9122	0.7800
0.7	0.3	Before Inspection		0.9988	0.9976	0.9400	0.9988	0.9976	0.9400
		$m = 2$	Static	0.9990	0.9980	0.9600	0.9990	0.9980	0.9600
			Sequential	0.9990	0.9980	0.9600	0.9990	0.9980	0.9600
0.8	0.2	Before Inspection		0.9968	0.9936	0.7200	0.9968	0.9936	0.7200
		$m = 2$	Static	0.9987	0.9974	0.8800	0.9986	0.9972	0.8700
			Sequential	0.9987	0.9974	0.8800	0.9987	0.9974	0.8800
0.9	0.1	Before Inspection		0.9998	0.9996	0.9700	0.9998	0.9996	0.9700
		$m = 2$	Static	0.9998	0.9996	0.9800	0.9998	0.9996	0.9800
			Sequential	0.9998	0.9996	0.9800	0.9998	0.9996	0.9800

nodes, our policy selects the one with a greater number of downstream nodes.

For both of the above random selection policies, two key considerations should be highlighted. First, it is clear that selecting leaf nodes (nodes with no child nodes) for inspection in anomaly detection can be highly inefficient due to their limited network influence and information. Moreover, it is important to recognize that anomalies typically propagate through interconnected nodes before affecting distant parts of the network. Consequently, single-node anomalies are often less frequent and less critical than multi-node anomalies. To avoid selecting nodes with no downstream nodes, we have implemented a strategy to exclude leaf nodes from the inspection process for both random inspection policies. This consideration is particularly valuable in systems such as power distribution networks, where inspecting a single customer node yields limited information. Instead, nodes with more connections and downstream nodes are prioritized for inspection, offering a more meaningful assessment of network status and anomalies. Secondly, it is important to note that the distinction between static and sequential inspection policies applies to both of the random policies mentioned above. We conducted an additional set of experiments, considering various network configurations and setup parameters. While we provide results for specific settings with $|N| = 1,000$, $|S| = 1$, and $m = 2$ due to space limitations, we explored three different network topologies and two sensor types. We applied the proposed and random inspection policies in both static and sequential manners. The results presented in Table 6 demonstrate that both random inspection policies have the potential to slightly improve the performance of the anomaly detection model. In almost all cases, performance metrics show improvement after the application of random inspections. However, these random inspection policies cannot outperform the proposed models. In other

Table 6
Comparison between the proposed models and random inspection policies. ($|N| = 1000$, $|S| = 1$, $m = 2$).

Type	α	β	Method	Static Policy			Sequential Policy		
				F1-Score	Overlap Coefficient	SDA	F1-score	Overlap Coefficient	SDA
WS Networks	0.6	0.4	Before Inspection	0.294	0.172	0.020	0.294	0.172	0.020
			Random Inspection I	0.295	0.173	0.020	0.292	0.171	0.020
			Random Inspection II	0.299	0.176	0.020	0.294	0.172	0.020
			Baseline Model	0.325	0.194	0.040	0.321	0.191	0.040
			Refined Model	0.326	0.194	0.030	0.324	0.193	0.030
	0.9	0.1	Before Inspection	0.806	0.675	0.350	0.806	0.675	0.350
			Random Inspection I	0.822	0.697	0.320	0.825	0.703	0.330
			Random Inspection II	0.840	0.724	0.354	0.832	0.712	0.350
			Baseline Model	0.866	0.763	0.427	0.877	0.780	0.470
			Refined Model	0.902	0.821	0.500	0.900	0.818	0.505
Regular Networks	0.6	0.4	Before Inspection	0.000	0.000	0.000	0.000	0.000	0.000
			Random Inspection I	0.000	0.000	0.000	0.000	0.000	0.000
			Random Inspection II	0.000	0.000	0.000	0.000	0.000	0.000
			Baseline Model	0.169	0.093	0.070	0.141	0.076	0.060
			Refined Model	0.169	0.093	0.070	0.169	0.093	0.070
	0.9	0.1	Before Inspection	0.649	0.480	0.410	0.000	0.000	0.000
			Random Inspection I	0.674	0.508	0.430	0.651	0.483	0.400
			Random Inspection II	0.616	0.445	0.320	0.613	0.442	0.350
			Baseline Model	0.792	0.655	0.478	0.783	0.643	0.495
			Refined Model	0.865	0.762	0.586	0.863	0.759	0.582
Tree Networks	0.6	0.4	Before Inspection	0.711	0.552	0.120	0.711	0.552	0.120
			Random Inspection I	0.711	0.552	0.120	0.711	0.552	0.120
			Random Inspection II	0.711	0.552	0.120	0.711	0.552	0.120
			Baseline Model	0.858	0.751	0.460	0.858	0.752	0.470
			Refined Model	0.824	0.700	0.410	0.824	0.700	0.410
	0.9	0.1	Before Inspection	1.000	1.000	0.990	1.000	1.000	0.990
			Random Inspection I	1.000	1.000	0.990	1.000	1.000	0.990
			Random Inspection II	1.000	1.000	1.000	1.000	1.000	1.000
			Baseline Model	1.000	1.000	0.990	1.000	1.000	0.990
			Refined Model	1.000	1.000	0.990	1.000	1.000	0.990

words, the performance measures corresponding to the baseline and refined models in both static and sequential scenarios are better than random inspection policies. As discussed earlier, it becomes evident that the impact of inspection policies (whether random or proposed) diminishes as sensors become more powerful and network structures become less complex (e.g., tree structures). Similar results were observed across various values of $|N|$, $|S|$, and sensor parameters α and β, although detailed results are not included due to space constraints.

4.4. CPU time comparisons for inspection models

The random inspection policies are evidently computationally less expensive, especially in the case of the Random Inspection Policy I where only m nodes need to be selected from the set of $|N|$ (excluding non-leaf) nodes. However, in random inspection policy II, we need to calculate the proportion of downstream anomalous nodes for all nodes, which results in complexity linearly increasing with the number of nodes in the network. For the baseline model in a static scenario, since only one measure is calculated for each node, the complexity depends on the number of nodes ($|N|$). However, for the sequential policy, a measure needs to be calculated sequentially for each node, which increases the complexity to depend on $|N| \times m$. The refined model without the utilization of matrix operations includes 2 nested **for** loops over network nodes to calculate $\gamma(n)$, making the complexity to depend on $|N|^2$. With introducing efficient matrix operations as shown in Section 3.6, we expect the CPU time needed for Algorithms 5 and 6 to be very low and not much longer than the baseline model. To numerically evaluate the computational time, we recorded the average CPU time required for each method to determine the status of the nodes after inspection for 3 types of networks with different sizes as shown in Table 7. All experiments in this section were conducted on a cloud-based platform Google Colab Pro with CPU only (Intel(R) Xeon(R) CPU @ 2.20 GHz, 12 GM RAM). Further improvements in speed may be attainable with access to more advanced computing resources. The results summarized in Table 7 confirm our expectations regarding CPU time. Specifically, the refined model and its sequential scenarios take slightly longer than random inspection models while random inspection policy I is the quickest. Overall, all experiments were completed within a very reasonable timeframe, enhancing the applicability of the findings in real-world scenarios. Users of our paper should consider high-performance computing resources as well as the trade-off between accuracy improvement and CPU time and decide whether inspection is feasible and whether inspecting more nodes is beneficial. For instance, in power distribution networks, utility operators need to respond swiftly to detect the sources of outages and assess impacted customers because sustained outages can lead to substantial financial losses and societal consequences. Consequently, dispatching repair crews to numerous locations is not a viable option, and the final decision should be based on a very small set of inspection nodes.

5. Concluding remarks

In the realm of monitoring complex systems structured as graphs, such as power distribution networks, where many sensors collect stochastic data about nodes' health conditions, we have explored an alternative to relying solely on stochastic insights derived from anomaly detection models. Our objective was to investigate whether targeted inspection of a small subset of nodes could enhance anomaly detection performance. In this paper, we introduced two node selection methods for inspection, taking into account both static and sequential approaches. These methods have the potential to significantly enhance anomaly detection, thus improving the overall monitoring of network status. We formulated the anomaly detection problem using Bayesian networks and utilized samples from a stochastic sampler for node selection and inspection. Our numerical results confirmed that this approach effectively boosts anomaly detection accuracy by focusing on intelligent inspection efforts

Table 7
CPU time (in seconds) comparison between random and proposed inspection models.

Network Size	Inspection Scenario	WS Networks				Regular Networks				Tree Networks			
		Random I	Random II	Baseline	Refined	Random I	Random II	Baseline	Refined	Random I	Random II	Baseline	Refined
$\|N\| = 500$	Static	0.02	0.63	0.02	0.42	0.02	0.71	0.02	0.44	0.02	0.35	0.02	0.43
	Sequential	0.02	0.62	0.02	0.72	0.02	0.60	0.02	0.73	0.02	0.36	0.02	0.76
$\|N\| = 1000$	Static	0.02	1.17	0.02	1.24	0.02	1.13	0.02	1.31	0.02	0.63	0.03	1.36
	Sequential	0.02	1.16	0.03	2.19	0.02	1.18	0.02	2.36	0.02	0.64	0.02	2.42
$\|N\| = 2000$	Static	0.03	2.29	0.04	5.56	0.03	2.32	0.03	5.43	0.03	1.32	0.02	5.61
	Sequential	0.02	2.55	0.03	9.92	0.02	2.46	0.03	9.78	0.03	1.30	0.03	9.98

on just a few critical nodes. The current work has limitations as it solely focuses on deterministic anomaly propagation paths while disregarding potential time dependencies within anomalies and sensor data. In future research, we plan to explore the dynamic nature of anomalies and sensor data, incorporating more intricate anomaly scenarios where propagation sets are either stochastic or not predefined. Additionally, we aim to investigate how the findings from this study can be extended to guide the optimal placement of new sensors within the network. This strategic sensor placement will enable anomaly detection models to make more effective use of new sensor data, leading to improved accuracy and quicker detection of anomalies. Another area for future work is to extend the proposed models by defining new criteria for node selection and efficiently executing them using matrix operations similar to those proposed in this paper.

CRediT authorship contribution statement

Feiran Xu: Conceptualization, Software, Validation, Visualization, Writing – original draft. **Ramin Moghaddass:** Conceptualization, Methodology, Project administration, Supervision, Writing – original draft, Writing – review & editing.

Data availability

Data will be made available on request.

Acknowledgments

This work is supported by the National Science Foundation Grant Number 1846975.

References

Ahmed, M., Mahmood, A. N., & Hu, J. (2016). A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, *60*, 19–31.

Akoglu, L., Tong, H., & Koutra, D. (2015). Graph based anomaly detection and description: A survey. *Data Mining and Knowledge Discovery*, *29*(3), 626–688.

Berry, J. W., Fleischer, L., Hart, W. E., Phillips, C. A., & Watson, J.-P. (2005). Sensor placement in municipal water networks. *Journal of Water Resources Planning and Management*, *131*(3), 237–243.

Bhuyan, M. H., Bhattacharyya, D. K., & Kalita, J. K. (2014). Network anomaly detection: Methods, systems and tools. *IEEE Communications Surveys & Tutorials*, *16*(1), 303–336.

Brentan, B., Carpitella, S., Barros, D., Meirelles, G., Certa, A., & Izquierdo, J. (2021). Water quality sensor placement: A multi-objective and multi-criteria approach. *Water Resources Management*, *35*, 225–241.

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, *41*(3), 1–58.

Cui, Y., Bao, J., Wang, J., Zhang, Q., & Jiang, X. (2019). Spatio-temporal correlation based anomaly detection and identification method for IoT sensors. In *2019 International conference on control, automation and information sciences* (pp. 1–6).

El-Zahab, S., Abdelkader, E. M., & Zayed, T. (2018). An accelerometer-based leak detection system. *Mechanical Systems and Signal Processing*, *108*, 276–291.

Erhan, L., Ndubuaku, M., Mauro, M. D., Song, W., Chen, M., Fortino, G., Bagdasar, O., & Liotta, A. (2021). Smart anomaly detection in sensor systems: A multi-perspective review. *Information Fusion*, *67*, 64–79.

Gronle, M., & Osten, W. (2016). View and sensor planning for multi-sensor surface inspection. *Surface Topography: Metrology and Properties*, *4*(2), Article 024009.

Heinzelman, W. B., Chandrakasan, A. P., & Balakrishnan, H. (2002). An application-specific protocol architecture for wireless microsensor networks. *IEEE Transactions on Wireless Communications*, *1*(4), 660–670.

Hu, N., Tian, Z., Lu, H., Du, X., & Guizani, M. (2021). A multiple-kernel clustering based intrusion detection scheme for 5G and IoT networks. *International Journal of Machine Learning and Cybernetics*, *12*.

Ifzarne, S., Tabbaa, H., Hafidi, I., & Lamghari, N. (2021). Anomaly detection using machine learning techniques in wireless sensor networks. *Journal of Physics: Conference Series*, *1743*(1), Article 012021.

Kang, S. H., & Nguyen, T. (2012). Distance based thresholds for cluster head selection in wireless sensor networks. *IEEE Communications Letters*, *16*(9), 1396–1399.

Kim, T.-Y., & Cho, S.-B. (2018). Web traffic anomaly detection using C-LSTM neural networks. *Expert Systems with Applications*, *106*, 66–76.

Kuboki, N., & Takata, S. (2019). Selecting the optimum inspection method for preventive maintenance. *Procedia CIRP*, *80*, 512–517.

Kumaran, S. K., Dogra, D. P., & Roy, P. P. (2019). Anomaly detection in road traffic using visual surveillance: A survey. arXiv preprint arXiv:1901.08292.

Lazim Qaddoori, S., & Ali, Q. I. (2023). An embedded and intelligent anomaly power consumption detection system based on smart metering. *IET Wireless Sensor Systems*, *13*(2), 75–90.

Lee, J.-S., & Cheng, W.-L. (2012). Fuzzy-logic-based clustering approach for wireless sensor networks using energy predication. *IEEE Sensors Journal*, *12*(9), 2891–2897.

Ma, Y., Guo, Y., Tian, X., & Ghanem, M. (2010). Distributed clustering-based aggregation algorithm for spatial correlated sensor networks. *IEEE Sensors Journal*, *11*(3), 641–648.

Ma, X., Wu, J., Xue, S., Yang, J., Zhou, C., Sheng, Q. Z., Xiong, H., & Akoglu, L. (2021). A comprehensive survey on graph anomaly detection with deep learning. *IEEE Transactions on Knowledge and Data Engineering*.

Nguyen, D., Minet, P., Kunz, T., & Lamont, L. (2011). New findings on the complexity of cluster head selection algorithms. In *2011 IEEE international symposium on a world of wireless, mobile and multimedia networks* (pp. 1–10).

Parra, L., Sendra, S., Lloret, J., & Bosch, I. (2015). Development of a conductivity sensor for monitoring groundwater resources to optimize water management in smart city environments. *Sensors*, *15*(9), 20990–21015.

Pei, S., Wang, J., Morone, F., & Makse, H. A. (2020). Influencer identification in dynamical complex systems. *Journal of Complex Networks*, *8*(2), cnz029.

Phua, C., Lee, V., Smith, K., & Gayler, R. (2010). A comprehensive survey of data mining-based fraud detection research. arXiv preprint arXiv:1009.6119.

Popat, R. R., & Chaudhary, J. (2018). A survey on credit card fraud detection using machine learning. In *2018 2nd International conference on trends in electronics and informatics* (pp. 1120–1125).

Radford, B. J., Apolonio, L. M., Trias, A. J., & Simpson, J. A. (2018). Network traffic anomaly detection using recurrent neural networks. arXiv preprint arXiv:1803.10769.

Rahman, M. S., Halder, S., Uddin, M. A., & Acharjee, U. K. (2021). An efficient hybrid system for anomaly detection in social networks. *Cybersecurity*, *4*, 10.

Santos-Ruiz, I., López-Estrada, F.-R., Puig, V., Valencia-Palomo, G., & Hernández, H.-R. (2022). Pressure sensor placement for leak localization in water distribution networks using information theory. *Sensors*, *22*(2), 443.

Shastri, Y., & Diwekar, U. (2006). Sensor placement in water networks: A stochastic programming approach. *Journal of Water Resources Planning and Management*, *132*(3), 192–203.

Shigei, N., Morishita, H., & Miyajima, H. (2010). Energy efficient clustering communication based on number of neighbors for wireless sensor networks. In *International multi-conference on engineers and computer scientists*. (IMECS), Hong Kong.

Song, C., Hsu, W., & Lee, M. L. (2016). Targeted influence maximization in social networks. In *Proceedings of the 25th ACM international on conference on information and knowledge management* (pp. 1683–1692).

Suresh, M. S. S., & Menon, V. (2023). A generic and scalable approach to maximize coverage in diverse indoor and outdoor multicamera surveillance scenarios. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *53*(2), 1172–1182.

Trucco, E., Umasuthan, M., Wallace, A., & Roberto, V. (1997). Model-based planning of optimal sensor placements for inspection. *IEEE Transactions on Robotics and Automation*, *13*(2), 182–194.

Tuptuk, N., Hazell, P., Watson, J., & Hailes, S. (2021). A systematic review of the state of cyber-security in water systems. *Water*, *13*(1).

Xu, F., & Moghaddass, R. (2023). A scalable bayesian framework for large-scale sensor-driven network anomaly detection. *IISE Transactions*, *1*, 445–462.

Yang, Q., Barria, J. A., & Green, T. C. (2011). Communication infrastructures for distributed control of power distribution networks. *IEEE Transactions on Industrial Informatics*, *7*(2), 316–327.

Yuan, Y., Dehghanpour, K., Bu, F., & Wang, Z. (2020). Outage detection in partially observable distribution systems using smart meters and generative adversarial networks. *IEEE Transactions on Smart Grid*, *11*(6), 5418–5430.

Zhang, R., Wang, X., & Pei, S. (2023). Targeted influence maximization in complex networks. *Physica D: Nonlinear Phenomena*, *446*, Article 133677.