Progressive Construction of k-identifiable Networks

Yongshuo Wan[†*], Cuiying Feng[†*], Kui Wu[‡], Jianping Wang[†]
[†] Department of Computer Science, City University of Hong Kong
[‡] Department of Computer Science, University of Victoria
E-mail: yongshwan2-c@my.cityu.edu.hk, miriam10@uvic.ca, wkui@uvic.ca, jianwang@cityu.edu.hk

***Abstract*—Since the inception of networking technology, network topology design has been a fundamental step for any interconnected system. This classical problem has diverse forms due to various design criteria. One special criterion, the ease of monitoring the network (termed as *monitorability* of the network), has recently attracted much attention in the era of Industry 4.0 when many complex private networks need to be built for new industrial services. This paper extends a quantitative measure of network *monitorability*, k-identifiability, based on which a new form of network topology design problem is formulated. We prove that this network design problem is intractable. To solve it, we systematically analyze the topological features that are helpful for reducing the complexity of network construction. Based on the analysis, we propose a dual-heuristic method that runs two heuristics in parallel and selects the better topology as the preliminary design result. Moreover, we design an integrated algorithm that reduces unnecessary edges as the final design result. We compare our dual-heuristic algorithm with the theoretical optimal solution in small-scale networks where the brute-force search is feasible. The results demonstrate the near-optimality of our method. We also illustrate the capability of our method in designing large-scale networks.**

***Index Terms*—Network Topology Design, Monitorability, Network Tomography**

I. Introduction

Network topology design is a fundamental step for building any networking system [1], [2]. Topology design is generally guided by the critical performance criteria of systems to be built over the network. For instance, in the early stage of data centers, the main criteria for data center networking are to support scalability and reconfigurability of data centers [3]. These requirements triggered tremendous effort on developing appropriate network topology for data centers, e.g., Fat-Tree [4], DCell [5], and DCube [6]. Nevertheless, the *monitorability* of networks, a term referring to the ease of monitoring the network operation, has rarely been considered as the main criterion in the network design/planning stage.

We argue that the *monitorability* of networks should be emphasized more in the early stage of system design. In recent years, artificial intelligence for IT operations (AIOps) [7], [8] has become a de-facto technique for large Internet service providers (ISPs) to automate network monitoring and fend off service disruptions. Driven by this trend, it becomes more and more imperative that network operators can easily monitor operating states of network nodes and quickly identify failures. Nevertheless, the task of network monitoring is only post-hoc, i.e., monitoring strategy is developed only *after* the network has been deployed and already in operation. This practice may severely limit the power of AIOps if the underlying network topology is ill-formed in the beginning (e.g., sparse, large diameter) and thus does not render easy monitoring (e.g., using the collected monitoring information cannot accurately identify failures). In the era of Industry 4.0, considering the *monitorability* of networks in the design stage should be a norm when people have the chance of developing private networks to accommodate new industrial services [9].

Generally speaking, directly measuring every network node incurs prohibitive measurement overhead. Network tomography [10] has been proposed to infer the internal status of a network based on affordable end-to-end measurements. Even if in-band network telemetry (INT) [11] allows individual devices to report the statistics in the data plane directly to monitoring applications running in the centralized control plane, large-scale INT is still far from reality since it needs to tackle nontrivial technical challenges such as orchestration, data aggregation, security, and high monitoring overhead [11]. As such, we follow the network tomography approach that assumes only partial network nodes could be monitors.

Given a deployed network, the *monitorability* of networks has been quantitatively captured by the concept of k-identifiable networks [12], which means network operators can always accurately localize failures if the number of simultaneous failures in the network is bounded by k. A k-identifiable network can greatly simplify network monitoring and failure localization. Nevertheless, all existing work followed the post-hoc approach that develops monitoring solutions on a deployed network. It is worthy noting that all problems [12], [13], [14] investigated in the post-hoc approach are *completely* different from the network topology design problem in this paper, in that the former focus on developing monitoring methods over an existing network while the latter aims at building a network that is easy to monitor. It is still unclear, during the network design stage, what are the best network topologies (w.r.t. the *monitorability*) that we can build within a given budget?

In order to answer the above question, we need to address three main challenges. First, the question has obviously practical meaning but does not render direct analysis. To be more specific, what is the mathematical definition of *monitorability*? Can we have different forms of *monitorability*? How can we link the budget with network topology? Second, the *monitorability* of a network clearly depends on appropriate deployment of monitors [13]. We are faced with the "chicken-and-egg" problem because the optimal deployment of monitors relies on the underlying network topology, but we do not know the final

topology yet in the design stage. Third, the network design problem usually falls in the category of hard combinatorial optimization problems, and the problem studied in this paper is no exception. Given the large solution space of network topology design, how can we discover effective heuristics to shrink the search space for finding near optimal solutions?

We systematically address the above difficulties. Regarding the first challenge, we leverage the concept of k-identifiable in [12] and use weights between distinct nodes to reflect the construction cost. For the second challenge, we derive a set of new topological properties of k-identifiable networks, which set preconditions for the locations of monitors. These topological properties also provide us with useful information for designing good heuristics to address the third challenge. Accordingly, we propose a dual-heuristic method and an integrated algorithm to solve the network construction problem. Overall, the paper makes the following contributions:

- Under the framework of Boolean network tomography [15], [16], we formally formulate the k-identifiable network design problem and prove its intractability. We derive a series of properties of k-identifiable networks that are easy to implement and can effectively reduce the solution search space with the number of monitors bounded in a small range. (Sections III and IV).
- Based on our analytical results, we identify the most useful sufficient conditions for k-identifiable networks, based on which we can design good heuristics for topology design. We first propose a dual-heuristic **DH** consisting of two heuristics: *Void-First* and *Sorted Weight (SW)-First*, with the former favoring the node-degree requirement and the latter favoring a smaller total weight (i.e., cost). We then adopt an integrated algorithm (**PCTK**), which for a given network construction instance runs **DH** in the beginning and reduces unnecessary edges to obtain the best topology as the final result. (Section V).
- We validate the efficiency and near optimality of the **DH** method. For designing small-scale networks where it is feasible to find the optimal solution with a brute-force search (in a constrained search space), the gap between the optimal solution and that from **DH**, on average, is below 7% percent. For designing large-scale networks, we use major cities across North America as the designated network locations to showcase the topology with different monitorability requirements. (Section VI).

II. Related Work

Research works can be roughly classified into two groups: the post-hoc approach and the ex-ante approach. Works in the former assume that the network has been deployed, while the works in the latter are in the network design/planning stage when the actual network has not been built.

A. Post-hoc Approach

Given an existing network, directly monitoring every element of the network has been proved infeasible due to the prohibitive measurement overhead and the so-called silent failures, i.e., the failures that cannot be automatically detected and alerted [17]. Pioneered by Duffiel [15], [16], an alternative method, named Boolean network tomography, is used for network failure localization. In a nutshell, Boolean network tomography infers the status (failed or normal) of internal network nodes/links based on end-to-end path measurement results. Since its inception, Boolean network tomography has been serving as the fundamental framework for a series of research on node/link failure detection.

The early works [15], [16] mainly focus on the minimum set of nodes that leads to the observed performance of the given measurement paths. Nevertheless, a shortcoming of this idea is that such a minimum set of nodes may not be exactly the set of failed nodes. To better capture the status of each node (i.e., normal or failed), Ma et al. assumed that the number of simultaneous multiple failures is bounded by k and proposed a novel concept called k-identifiability [12]. They gave an upper bound of the number of k-identifiable nodes if monitors are known in advance. They also studied the optimal monitor placement to achieve the maximum number of k-identifiable nodes [18], [13]. The problem of finding the minimum number of monitors for link failure detection under a BGP-like routing policy was studied in [19]. Mukamoto et al. [20] proposed an adaptive Boolean network tomography scheme, in which measurement paths are established sequentially according to a candidate set of failure links.

Different probing mechanisms may pose different constraints on the measurement paths. For instance, Ma et al. [13] studied three different probing mechanism, *controllable arbitrary-path probing* (CAP) where monitors can measure arbitrary paths subject to connectivity, *controllable simple-path probing* (CSP) where measurement paths must be cycle-free paths between monitors, and *uncontrollable probing* (UP) where measurement paths must follow underlying routing protocol. Considering that source routing is broadly available, we in this paper assume the CSP scheme.

B. Ex-ante Approach

Ex-ante approach belongs to the large field of network topology design [21], [22], [23], [24], which covers diverse application domains with various design considerations. For instance, Fencl at al. [21] presented an interactive genetic algorithm to design network topology that satisfies a given fault tolerance requirement. In the context of network synchronization, Summers et al. [22] used the network coherence as the main design goal for network topology, where network coherence is a measure quantifying variance of states around the consensus subspace [25]. For data center networks, Akella at al. [24] proposed using *cost universality* as the network design goal for large-scale general-purpose datacenters, where *cost universality* refers to the datacenter's ability of emulating any other network that could be built at the same cost. Nevertheless, despite the huge volume of papers on network topology design, very few works have considered the *monitorability* as the main network design goal.

TABLE I
MAIN NOTATIONS

Symbol	Meaning
M, N	Set of monitors/non-monitors, respectively
$d(v)$	Degree of node v
$\mathcal{N}(v)$	v's neighbor set
F	Set of failed non-monitors
P	Set of measurement paths
P_F	Incident set of measurement paths w.r.t. F
m	Bound of the number of monitors

Fault-tolerant computing [26], [27], [28], another big research field that has a long history, is loosely related to network monitoring and failure localization. Nevertheless, the main objective of fault-tolerant computing is to develop and verify computing systems that can work correctly in the presence of faults. In most cases, the speciality of the system under study, e.g., special topology in circuits and computer architecture, should be considered.

Our work essentially belongs to the ex-ante approach. Nevertheless, we unify the post-hoc and ex-ante approaches by solving a more general network construction problem. Specifically, we assume that we might have an initial network that was poorly designed and do not meet the monitorability requirement, and our goal is to introduce new links to improve the network's monitorability. Clearly, an initial network with no links (i.e., the ex-ante approach) is a special case of the network construction problem. To the best of our knowledge, there is no previous research on designing easy-to-monitor networks over the Boolean network tomography framework.

III. Problem Formulation

A. Model and Assumptions

A network is modeled as an undirected weighted graph $\mathcal{G} =< \mathcal{V}, \mathcal{E}, \mathcal{W} >$, where $\mathcal{V}$ is the set of nodes, $\mathcal{E}$ is the set of links, and $\mathcal{W}$ is the set of weights between distinct nodes. A weight reflects the construction cost of building a link between two distinct nodes. For instance, it may be geographic distance representing the length of the cable. We assume that all links are reliable, but each node may be either *normal* or *failed*. Our goal is to identify the failed nodes.

Following the main notations listed in Table I, we introduce the following basic assumptions and terminologies in the field of network monitoring.

- *Monitors*: a special type of nodes in $\mathcal{G}$ that have self-checking mechanisms (e.g., failures can be reported automatically and get fixed as soon as possible) and monitoring capabilities (e.g., initiating and collecting measurements). Apparently, the states of monitors are always clear. The set of monitors is denoted by $M \subset \mathcal{V}$.
- *Measurement path*: a *non-loop* path that only contains two distinct monitors at its end nodes.
- *Non-monitors:* nodes in $\mathcal{G}$ that are not monitors. We denote the set of non-monitors by $N = \mathcal{V} \backslash M$. The states of non-monitors (i.e., normal or failed) are unknown and need to be determined from the results of measurement paths.
- *Measured state:* the result of a measurement path. The state of a measurement path is normal if and only if all traversed nodes are normal, and is failed if at least one node on the path fails.
- *Failure set:* the set of simultaneously failed nodes in $\mathcal{G}$. We denote the failure set by $F(\subseteq N)$.
- *Incident set:* we say that a measurement path $p \in P$ is affected by the failure set F if p includes at least one node in F. We use P_F, called the incident set of P w.r.t. F, to denote all paths in P that are affected by F.

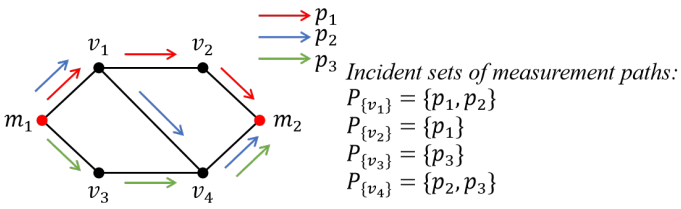


Fig. 1. An example of 1-identifiable nodes/network.

Intuitively, if we assume that all non-monitor nodes could fail *simultaneously*, no monitoring solution can infer the states of these nodes in a general network, because no information is available except that all measurement paths fail. Due to this reason, when we quantify the topological feature from the network monitoring perspective, we need to constrain the network by limiting the maximum number of *simultaneous* node failures. We borrow (with slight modification) the concept of k-identifiability in [12] for this purpose.

Definition 1. *In a network $\mathcal{G}$, a failed node $v_i \in \mathcal{V}$ is called k-identifiable when there exists a set of measurement paths P such that for any failure sets F_1 and F_2, where $F_1 \cap \{v_i\} \neq F_2 \cap \{v_i\}$ and $|F_j| \leq k$ $(j \in \{1, 2\})$, the incident sets P_{F_1} and P_{F_2} are different.*

In other words, a node v_i is k-identifiable if and only if the state of v_i could be uniquely determined when there are *at most k simultaneous* non-monitor failures in the network $\mathcal{G}$. For example, the network in Fig. 1 contains two monitors $\{m_1, m_2\}$ and four non-monitors $\{v_1, v_2, v_3, v_4\}$. There are three available measurement paths p_1, p_2, and p_3. Since each non-monitor in Fig. 1 has a unique incident set of measurement paths, we can uniquely localize any node failure if we assume that at most one non-monitor node can fail at a time in the network, i.e., every non-monitor node is 1-identifiable.

Definition 2. *For a network $\mathcal{G}$, we say $\mathcal{G}$ is k-identifiable when every non-monitor node in $\mathcal{G}$ is k-identifiable.*

B. Progressive Construction and Goal

The progressive k-identifiable network construction problem: Assume that we are given a network $\mathcal{G} =< \mathcal{V}, \mathcal{E}, \mathcal{W} >$, an integer bound of simultaneous failures k, and an integer bound of the number of monitors m. Assume that the monitors can be deployed anywhere on $\mathcal{V}$. Our goal

is to deploy a new set of links $\mathcal{E}'$ to construct a network $\mathcal{G}' =< \mathcal{V}, \mathcal{E} \bigcup \mathcal{E}', \mathcal{W} >$ such that $\mathcal{G}'$ is k-identifiable with the minimum total weight of $\mathcal{E}'$.

IV. Properties of k-identifiablity Pertinent to Progressive Topology Construction

A. The Hardness

A network possibly has $\sum_{i=1}^{k} \binom{|\mathcal{V}|}{i}$ different failure sets, so brute-force search for the optimal k-identifiable network is infeasible when $|\mathcal{V}|$ is large. We prove that the progressive k-identifiable network construction problem is NP-complete.

Theorem 1. *The progressive k-identifiable network construction problem is NP-complete.*

Proof. We only need to prove a special case of our problem is NP-complete, where the network is empty (i.e. only nodes are given without any link). Assume that we are given parameters $\mathcal{V}, \mathcal{W}, k$ and $m = 2$, i.e., only two monitors. Since there are only two monitors, a necessary condition for k-identifiable network is that the constructed network must be connected. Otherwise, each connected sub-network has to be monitored separately and each connected sub-networks needs at least two monitors in order to build measurement paths. This is just a necessary condition, implying that the k-identifiable network design problem is at least as hard as the problem of building a connected network with the minimum total weight. Building a connected network with the minimum total weight is equivalent to *finding the minimum spanning tree in a complete graph* consisting of node set N and link weights $\mathcal{W}$. This is because if the topology is not a tree, then at least there are two distinct nodes that have two different measurement paths, p_1 and p_2, between them in the constructed network. Then there must be a link $l \in P_1$ but $l \notin P_2$, and removing l from the network leads to a connected network that has a smaller total weight. It has been proved that finding the minimum spanning tree in a complete graph is NP-complete [29]. □

Since it is hard to solve the progressive k-identifiable network construction problem, we shift our focus to developing effective heuristic solutions based on theoretical conditions that guarantees k-identifiability. For this, we first need to break the chicken-and-egg loop: k-identifiability partly depends on the locations of monitors but the optimal (i.e., minimum number of monitors) deployment of monitors can only be determined after the network topology is known [13].

B. Breaking the Chicken-and-Egg Loop

Our idea to break this logical loop is to derive the topological properties of k-identifiable networks that set preconditions for the possible locations of monitors.

Many properties on k-identifiable networks have been proved in previous works [12], [18], [13], but not all are helpful for the purpose of the progressive k-identifiable network construction. For example, it has been proved [12] that $\mathcal{G}$ is k-identifiable if any node set $\mathcal{V}'$ ($|\mathcal{V}'| \leq k+1$) in $\mathcal{G}$ contains at most one monitor and each connected component in $\mathcal{G} - \mathcal{V}'$ (i.e., the subgraph consisting of nodes in $\mathcal{V} - \mathcal{V}'$) contains a monitor. Based on this theorem, we can translate the problem of building k-identifiable networks (with two monitors) to building a $(k+2)$-vertex-connected graph (proof omitted due to space limit and due to its less relevance of our solution proposed later). While this property is interesting, the algorithm for building a $(k+2)$-vertex-connected network may lead to a high total weight.

Therefore, we need to find new properties of k-identifiable networks that are helpful for easy construction and small costs both on total weight and number of monitors. Our investigation shows that the most suitable angle to discover the new topological properties is node degree. As such, we in the following analyze two exclusive cases.

Proofs in the rest of the paper are moved to Appendix.

C. Properties under Different Node Degrees

1) When $d(v_i) \leq k$: first we consider the case where each non-monitor node has at most k neighbors.

Theorem 2. *Assume that the node degree of a non-monitor node v_i is no larger than k, i.e., $d(v_i) \leq k$. v_i is k-identifiable if and only if the set of its neighbors includes at least two monitors.*

Theorem 2, however, might not be very useful for topology construction because it only implies the that any node with degree less than k cannot be k-identifiable unless it has monitors as its neighbor.

2) When $d(v_i) > k$: we have the following theorems:

Theorem 3. *Assume that the node degree of a non-monitor node v_i is larger than k, i.e., $d(v_i) \geq k+1$, $\mathcal{G}$ is k-identifiable if for **each** (non-monitor) node v_i, it satisfies **any** of following conditions:*

(i) ***Condition 1****: $d(v_i) \geq k+2$;*

(ii) ***Condition 2****: (otherwise, $d(v_i) = k+1$) for each $v_j \in \mathcal{N}(v_i)$, there exists at least one node v_l such that $v_l \in \mathcal{N}(v_j)$ and $v_l \notin \mathcal{N}(v_i)$.*

*In addition, the **worst-case** upper bound on the number of monitors m is* $\max\{2, \lceil \frac{2|\mathcal{V}|}{k+2} \rceil\}$.

Remark 1. *The number of monitors $m = \max\{2, \lceil \frac{2|\mathcal{V}|}{k+2} \rceil\}$ is the worst-case upper bound. As we can see in the numerical evaluation in Section VI, the actual number of needed monitors is much smaller.*

The above theorem gives the sufficient conditions for a node to be k-identifiable when its degree is larger than k, based on which we can develop an algorithm for the construction. The difference between the two conditions in Theorem 3 lies in that *Condition 1* requires more degree but less monitors while *Condition 2* could achieve lower degree on nodes with more monitors, such statement is justified by the following Theorem.

Theorem 4. *For a network $\mathcal{G}$ to achieve k-identifiable, the more nodes v_i that satisfies Condition 1 rather than Condition 2 in Theorem 3, the less monitors are needed.*

Remark 2. *Recall that for our construction goal, the number of monitors should be bounded by m. To build a k-identifiable network, if we consider satisfying* ***Condition 2*** *directly (less node degree requirement translates to a smaller total number of link, thus less total link weight), the number of monitors will increase dramatically (as indicated by Theorem 4) to surpass m. Therefore, in the construction process, we prioritize in satisfying* ***Condition 1*** *for every node. Afterwards, we delete a set of links with highest total weight as long as the consequently required number of monitors is bounded by m.*

V. Constructing k-identifiable Networks

A. Special Case: Building A Network From Scratch

To better understand our strategy in progressively constructing a k-identifiable network, we first focus on the special case of building a k-identifiable network from scratch, i.e., no link exists among a given set of nodes. For this, we need to meet three requirements by selecting a set of links $\mathcal{E}$ such that:

1) $\mathcal{E}$ comprise a network that is k-identifiable,
2) the total weight of $\mathcal{E}$ is as small as possible,
3) the number of needed monitors is within a bound.

Based on the analysis in Section IV, we note that Theorem 3 provides helpful knowledge for topology design. It not only gives the sufficient conditions for a network to be k-identifiable, but also tells us the bounds on the maximum number of required monitors. However, it is not easy to use Theorem 3 for topology design, because for an empty network it is difficult to determine each node's neighbors before the network construction is finished. Hence, we leverage the first condition in Theorem 3 and translate the design goal to selecting a set of links $\mathcal{E}$ such that:

1) any node's degree in $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$ is no smaller than $k+2$ (*degree requirement*),
2) the total weight of $\mathcal{E}$ is as small as possible (*smallest total weight requirement*).

Remark 3. *Since we initially do not know which node will be selected as monitor, we request any node's degree to be no smaller than $k+2$. In addition, we only use Theorem 3 as a rough estimation on the maximum number of required monitors. The actual number of monitors will be derived by the method in [13] after network topology has been constructed.*

The NP-hardness of the constructing k-identifiable problem prevents us from finding a polynomial algorithm to get the theoretical optimal solution (TOS). The *degree requirement* determines that the number of total edges must be at least higher than some number (details disclosed later in this section). The candidate solutions that satisfy the *degree requirement* may have different total weights. If we find the TOS with brute-force search, for a network with n nodes, the size of the search space will be $\sum_{t=\lceil n(k+2)/2 \rceil}^{n(n-1)/2-1} \binom{n(n-1)/2}{t}$ (i.e., the total number of possible edge combinations). Clearly, we need to find a way to reduce the search space. For this purpose, we introduce the following concept and theorem.

Definition 3 (Redundant Edge)**.** *In a graph, a redundant edge (re) for the $k+2$ degree requirement is an edge whose two end nodes' degrees are both greater or equal to $k+2$ after the removal of re.*

It is obvious that the existence of any redundant edge will add unnecessary link weight to the total weight; hence, towards the goal of constructing a k-identifiable network while minimizing the total weight, the objective network has to satisfy the following **construction criteria**:

(a) each node's degree is equal or greater than $k+2$,
(b) there is no redundant edge for the $k+2$ degree requirement.

Theorem 5. *For n nodes, to build a network satisfying the* ***construction criteria****, the* ***exact*** *number of candidate edges for the Theoretical Optimal Solution is $\lceil n(k+2)/2 \rceil$.*

Proof. The proof is divided into 2 parts (necessity and sufficiency): in the necessary part, we prove that in order to satisfy the construction criteria (a), the number of edges has to be at least $\lceil n(k+2)/2 \rceil$. In the sufficient part, we prove that the number of edges cannot exceed $\lceil n(k+2)/2 \rceil$ to satisfy the construction criteria (b). Therefore, the number of edge satisfying the construction criteria has to be exactly $\lceil n(k+2)/2 \rceil$.

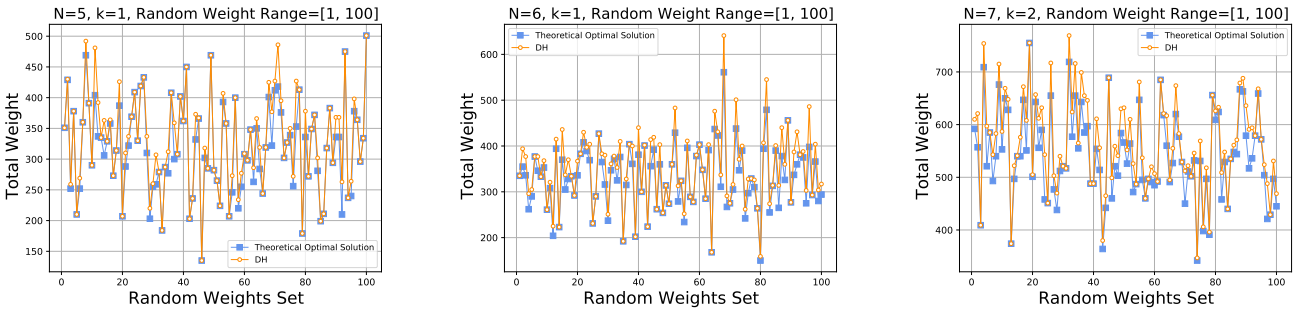
Necessity: The construction criteria (a) requires that each node's degree is at least $k+2$. Focusing on meeting criteria (a) only, we do not need to care the weight of edges. As such, we can use the follow process to obtain the minimum number of edges satisfying (a): for each node u, adding an arbitrary edge that is incident to u until it has $k+2$ neighbors. This process, no matter which node order it follows, will result in the total number of edges listed in Table II, that is, the number of edge is $\lceil n(k+2)/2 \rceil$.

TABLE II
The Total Number of Required Edges

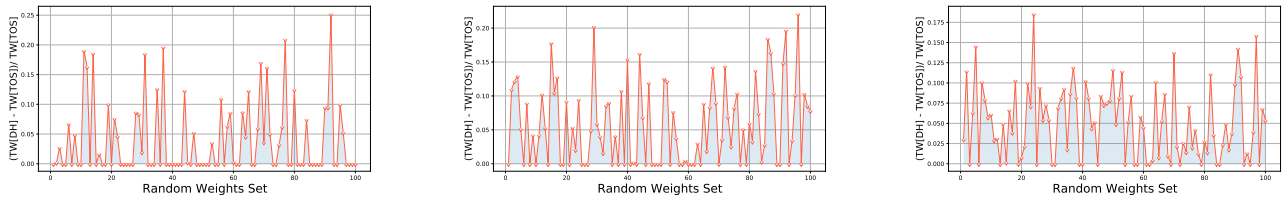
Parity of n and k $(1 \le k \le n-3)$	number of edges needed
n is odd, k is odd	$(n(k+2)+1)/2$
n is odd, k is even	$n(k+2)/2$
n is even, k is odd	$n(k+2)/2$
n is even, k is even	$n(k+2)/2$

Sufficiency: Once the construction criteria (a) is satisfied, as described in the necessary part, every node will have at least $k+2$ neighbors. To be more specific, for the case where n is odd and k is odd, $n-1$ nodes will have exactly $k+2$ neighbors and only 1 node will have $k+3$ neighbors, and for other 3 cases listed in Table II, all the n nodes will have exactly $k+2$ neighbors. Obviously, if we add any new link, no matter which end nodes it has, it will become a redundant edge. □

While Theorem 5 greatly reduces the search space for the TOS from $\sum_{t=\lceil n(k+2)/2 \rceil}^{n(n-1)/2-1} \binom{n(n-1)/2}{t}$ to $\binom{n(n-1)/2}{\lceil n(k+2)/2 \rceil}$, the complexity of searching in a space size $\binom{n(n-1)/2}{\lceil n(k+2)/2 \rceil}$ is still too high. When n becomes big, we cannot find the TOS in reasonable time. Therefore, we design heuristic methods. Imagine that we select edges step by step. From the beginning, there are n nodes without edges, thus each node's degree is 0. Our goal



(a) Total Weight (Task 1) (b) Total Weight (Task 3) (c) Total Weight (Task 6)



(e) Gap Ratio (Task 1) (f) Gap Ratio (Task 3) (g) Gap Ratio (Task 6)

Average: 0.039 Average: 0.057 Average: 0.050

Fig. 2. Performance comparison between **DH** and TOS in designing small networks.

is to select a proper number of edges to meet the $k+2$ degree requirement while keeping the total weight as small as possible. In this process, if we treat each node's degree as an initially empty box and each edge's end nodes as balls, then selecting one edge is equivalent to throwing two balls into two boxes. Once all the boxes are filled with $k+2$ balls, the degree requirement is satisfied. Therefore, a heuristic comes from the following observation: No matter which edge is selected, it equivalently provides two balls to fill the boxes (no advantage exists for quickly achieve degree requirement); however, different edges may contribute different weights to the final total weight. Naturally, an edge with a smaller weight should be favored. Based on this observation, we propose two heuristics:

- *Void-First*: in the edge selection process, any node whose degree is smaller than $k+2$ is considered as a void that needs to be filled. The n nodes are firstly arranged in order by their incident edges' weights, then are filled one by one. In other words, this heuristic focuses on satisfying the *degree requirement*, following the arranged node order.
- *Sorted weight (SW)-First*: firstly sort all the (potential) edges by their weights in increasing order. At each step, take the first remained edge (i.e., edge that has not been selected) witmh the smallest weight, if it is not a redundant edge for the current topology, put it to the solution set. Once the *degree requirement* is met, terminate.

For building a k-identifiable network from scratch, we propose a dual heuristic (**DH**) method that runs *Void-First* and *SW-First* in parallel and selects the network that has the smaller total weight as the final result. The time complexity of *Void-First* and the worst-case time complexity of *SW-First* are $O(n^2 log(n))$ and $O(n^4)$, respectively, where n is the total number of input nodes. Hence the worst-case time complexity of **DH** is $O(n^4)$.

The main purpose of this section is to illustrate how we can build a k-identifiability network from scratch using the **DH** method. Since **DH** ignores the bound on the number of monitors m and assumes the worst-case upper bound of required monitors, we may have the room to further trim down the network if we are given a high value of m, which is introduced in the next section.

B. Progressively Constructing A K-identifiable Network

The **DH** algorithm for building a k-identifiable network from scratch lays a good foundation for us to develop an effective algorithm for progressively constructing a k-identifiable network from an initially ill-formed network. Formally, given an initial network $\mathcal{G} =< \mathcal{V}, \mathcal{E} >$, we are expected to construct a new set of links $\mathcal{E}'$ (i.e., $\mathcal{E}' \cap \mathcal{E} = \emptyset$) such that:

1) the new network $\mathcal{G}' =< \mathcal{V}, \mathcal{E} \cup \mathcal{E}' >$ is k-identifiable;
2) the total weight of $\mathcal{E}'$ is as small as possible;
3) the number of needed monitors is within a bound.

Our main idea for progressively constructing a k-identifiable network is to leverage the topological information of existing network and apply the **DH** algorithm to make every node's degree reach at least $k+2$ first, then place monitors in the network using existing methods. Since [13] has already developed methods for deploying monitors, we borrowed the **MNMP** algorithm in [13] for monitor deployment. After the locations of monitors are determined, there may be **unnecessary links** (criteria defined below), which need to be removed.

We follow Theorem 3 to decide the criteria for **unnecessary links**. Specifically, a link l is an unnecessary link if the network meets the following conditions after removing l:

1) any node's degree in $\mathcal{G}' =< \mathcal{V}, \mathcal{E} \cup \mathcal{E}' >$ is no smaller than $k+1$,
2) any node whose degree is $k+1$ needs to satisfy Condition 2 in Theorem 3;
3) the total weight of $\mathcal{E}'$ is as small as possible.

Based on the above analysis, we propose our progressive construction of k-identifiable networks (**PCTK**) algorithm.

The time complexity of **DH** is $O(n^4)$. The time complexity of **MNMP** is $O(|\mathcal{E} \cup \mathcal{E}'| \cdot n^3)$ [13], where $|\mathcal{E} \cup \mathcal{E}'|$ is the total number of original and newly-added edges. Hence, the worst-case time complexity of **PCTK** is $O(k^2 \cdot n^5)$, where n is the number of nodes in the network.

Algorithm 1: Progressive Construction To K-identifiable (*PCTK*)

input : $\mathcal{G}$, k, m
output: k-identifiable network $\mathcal{G}'$ or "need to increase monitor bound m"

run the **DH** algorithm over the original network $\mathcal{G}$ to generate a k-identifiable network $\mathcal{G}' =< \mathcal{V}, \mathcal{E} \cup \mathcal{E}' >$, where $\mathcal{E}'$ is the set of newly-added links;
run **MNMP** [13] on $\mathcal{G}'$ to get the number of monitors $m*$ needed for $\mathcal{G}'$;
if $m* > m$ **then**
    return *print ("need to increase monitor bound m");*
end
order the edges in $\mathcal{E}'$ in the descending order of weight;
$numOfRemoved = 0$;
for *each edge e in* $\mathcal{E}'$ **do**
    if *e is an unnecessary link* **then**
        remove e from $\mathcal{E}'$;
        $numOfRemoved = numOfRemoved + 1$;
    end
    if $numOfRemoved = \lceil (m - m^*)/2 \rceil$ **then**
        run **MNMP**[13] on $\mathcal{G}'$ to update the number of monitors $m*$
    end
    if $m* == m$ **then**
        break
    end
end
return $\mathcal{G}'$;

VI. Evaluation

In this section, we evaluate the performance of **DH**. Since there is no existing work on designing k-identifiable networks, we only compare **DH** with the Theoretical Optimal Solution (TOS) in small-scale networks where finding TOS is feasible. For large-scale networks, we visualize two example networks designed with **DH**. After a network is constructed with **DH** or TOS, we follow the idea from [18] and [13] to find the number of monitors and their locations.

A. Near optimality of *DH* in Small Networks

To our best knowledge, this is the first work addresses the k-identifiable network design problem, so there is no existing methods that can be adopted to compare with **DH**. Therefore, we show the performance of **DH** with that of TOS on small-scale construction task (the amount of network nodes is small). TOS is found by brute-force search using Theorem 5, which reduces the original search space. Note that, even the reduced search space has the time complexity $O(\binom{n(n-1)/2}{\lceil n(k+2)/2 \rceil})$, where n is to total number of nodes, so finding TOS is only feasible for small-scale networks.

We have evaluated 6 design tasks for small-scale networks with different total numbers of nodes and different k values. The parameters for the 6 tasks are listed in Table III. For each input (n, k), we compare TOS and **DH** in 100 rounds. At each round, the weights of links are integers randomly generated in range $[1, 100]$. For each round, we compute the total weight of the constructed network by TOS and the total weight of the constructed network by **DH**, respectively. We also use a metric **Gap Ratio**, calculated by $\frac{TotalWeight[DH]-TotalWeight[TOS]}{TotalWeight[TOS]}$, to quantify the difference between **DH** and TOS in each round.

TABLE III
Parameters of Small-scale Construction Task

Task No.	n	k	Random Weights
(1)	5	1	integer in [1, 100]
(2)	5	2	integer in [1, 100]
(3)	6	1	integer in [1, 100]
(4)	6	2	integer in [1, 100]
(5)	7	1	integer in [1, 100]
(6)	7	2	integer in [1, 100]

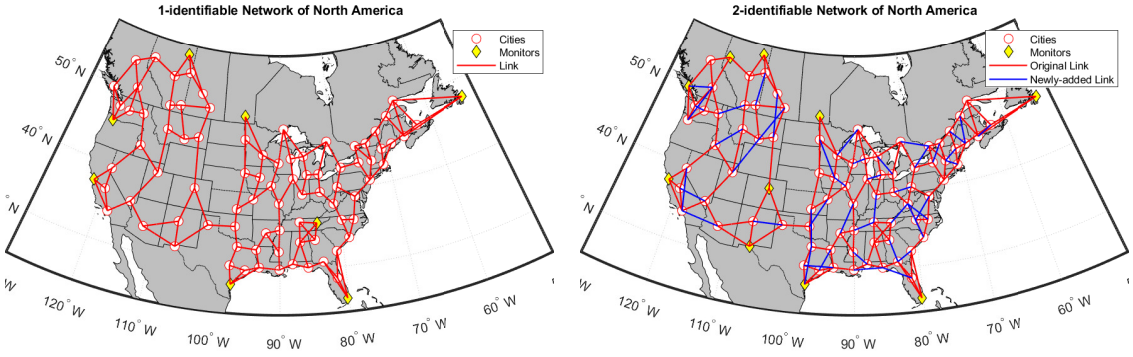
The evaluation results are listed in Table. IV. The average Gap Ratio is under 7% for all the tests. Moreover, **DH** behaves exactly as TOS in task (2). This is because this particular input $(n, k) = (5, 2)$ $(1 \leq k \leq n-3)$ meets the upper limit for $k = n-3$. Consequently, the constructed k-identifiable network must be a complete graph, and **DH** and TOS return the same topology. To save space, we only show the detailed evaluation results of three design tasks (i.e., Tasks 1, 3, 6) in Fig. 2. It is worth noting that all the above-constructed networks only need two monitors, which is smaller than the bound given by Theorem 3.

TABLE IV
DH Performance Gap to TOS (TW: total weight)

Task No.	Max Gap Ratio	Ave Gap Ratio	Counts of Equal TW
(1)	0.252	0.039	57 (out of 100)
(2)	0.0	0.0	100 (out of 100)
(3)	0.221	0.057	35 (out of 100)
(4)	0.266	0.040	47 (out of 100)
(5)	0.327	0.069	25 (out of 100)
(6)	0.186	0.050	23 (out of 100)

B. *DH* for Large Network Design

To demonstrate the constructed topology by **DH** on a large scale, we extract 91 Points of Presence (PoP) (including the longitude and latitude of the PoP locations) from *the Internet Topology Zoo* [30]. These PoP nodes are mainly located in major cities across North America. With these nodes' coordinates, the Euclidean distance between every pair



(a) $k = 1$ (b) $k = 2$

Fig. 3. Topology of k-identifiable network that covers 91 cities in North America. (a) Assuming that we need to build the network from scratch, we use **DH** to construct 1-identifiable network (the total number of links= 142, the number of monitors= 8). (b) Assuming that the topology of (a) is given, we use **PCTK** to construct the 2-identifiable network (the number of new links = 45, the number of monitors=10).

of nodes (u, v) is treated as the weight of the potential link. By feeding **DH** with these inputs, we finish two topology designs with $k = 1$ and $k = 2$, whose results are displayed in Fig. 3. **DH** is implemented with Python 3.7 and executed on a laptop (Quad-Core Intel Core i7, 1.7 GHz, MEM: 8 GB, macOS Catalina Version 10.15.7). For the network design task at this scale, the running times of constructing 1-identifiable and 2-identifiable networks are 0.12 seconds and 0.11 seconds, respectively, indicating that **DH** can be used to design large networks efficiently and effectively. Note that the constructed 1-identifiable network only needs 8 monitors and the constructed 2-identifiable network only needs 7 monitors, which are *significantly* smaller than the bound given by Theorem 3.

C. *PCTK for Large Network Construction*

To approach the optimal solution of progressive k-identifiable network construction, two main constraints must be followed: a) the number of monitors are within an expected bound; b) the total weight of newly-added links should be as small as possible. Between the two limitations, the former one is on the top of the list of considerations in our problem. As we have mentioned in Remark 2, the number of monitors that we are expected to use reaches minimum when each node in the network satisfies Condition 1 in Theorem 3. To check it, in Fig. 3(a), we make each node satisfying Condition 1 and the expected number of monitors is 7. However, following the idea that each node only needs to hold Condition 2, almost 60 monitors are deployed in the network, which we do not expect to see. The result is not shown in our paper since this method is not suitable for our problem.

Consider the case where the bound m is the minimum number of monitors that make the network k-identifiable (i.e., the degree of each node is at least $k+2$), we only need to run **DH** (the first step of **PCTK**), which is already evaluated in the former part. Then if there are still available monitors, the rest steps will be put into effect. To show the topology constructed by **PCTK** more intuitively, we use the result of Fig. 3(a) as a given network topology. Then we let $k = 2$ and $m = 10$. The new topology constructed by **PCTK** is shown in Fig. 3(b). As we can see seen, under the bound of 10 monitors, 45 edges are newly added (marked by blue lines). It is worth noting that if we remove one of new edges randomly and re-run **MNMP**, the expected number of monitors will exceed 10.

VII. Conclusion and Future Work

Designing a network topology that facilitates network and service management is critical for any networked system. As networks become more and more complex nowadays, troubleshooting network problems becomes more and more labour intensive. Accordingly, the monitorability of networks should be one of the core design principles for network planning. This paper initiated such a study under the framework of Boolean network tomography. By formulating, analyzing, and solving the k-identifiable network design problem, this paper is a crucial step towards building easy-to-monitor networks.

Our proposed **DH** solution is the first attempt at designing k-identifiable networks, and it is not perfect in various ways. While **DH** empirically shows good performance, it does not have a theoretical guarantee on the approximation ratio. In addition, it cannot directly answer some closely related design questions, e.g., can a k-identifiable network be built if we put constraints on the locations of monitors within a given area and the physical links within a subset of all possible links? Answering all these questions is nontrivial and is left as our future endeavour.

Acknowledgement

This work was partially supported by Hong Kong Research Grant Council under GRF 11216618 and the Natural Sciences and Engineering Research Council of Canada (NSERC) under Discovery Grant RGPIN-2018-03896.

References

[1] D. S. Johnson, J. K. Lenstra, and A. R. Kan, "The complexity of the network design problem," *Networks*, vol. 8, no. 4, pp. 279–285, 1978.
[2] M. Minoux, "Networks synthesis and optimum network design problems: Models, solution methods and applications," *Networks*, vol. 19, no. 3, pp. 313–360, 1989.

[3] Y. Liu, J. K. Muppala, M. Veeraraghavan, D. Lin, and M. Hamdi, *Data center networks: Topologies, architectures and fault-tolerance characteristics*. Springer Science & Business Media, 2013.

[4] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," *ACM SIGCOMM computer communication review*, vol. 38, no. 4, pp. 63–74, 2008.

[5] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S. Lu, "Dcell: a scalable and fault-tolerant network structure for data centers," in *Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, 2008, pp. 75–86.

[6] D. Guo, C. Li, J. Wu, and X. Zhou, "Dcube: A family of network structures for containerized data centers using dual-port servers," *Computer communications*, vol. 53, pp. 13–25, 2014.

[7] Y. Dang, Q. Lin, and P. Huang, "Aiops: real-world challenges and research innovations," in *2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*. IEEE, 2019, pp. 4–5.

[8] Y. Li, Z. M. Jiang, H. Li, A. E. Hassan, C. He, R. Huang, Z. Zeng, M. Wang, and P. Chen, "Predicting node failures in an ultra-large-scale cloud computing platform: an aiops solution," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 29, no. 2, pp. 1–24, 2020.

[9] M. Wollschlaeger, T. Sauter, and J. Jasperneite, "The future of industrial communication: Automation networks in the era of the internet of things and industry 4.0," *IEEE industrial electronics magazine*, vol. 11, no. 1, pp. 17–27, 2017.

[10] Y. Vardi, "Network tomography: Estimating source-destination traffic intensities from link data," *Journal of the American statistical association*, vol. 91, no. 433, pp. 365–377, 1996.

[11] L. Tan, W. Su, W. Zhang, J. Lv, Z. Zhang, J. Miao, X. Liu, and N. Li, "In-band network telemetry: A survey," *Computer Networks*, vol. 186, p. 107763, 2021.

[12] L. Ma, T. He, A. Swami, D. Towsley, K. K. Leung, and J. Lowe, "Node failure localization via network tomography," in *Proceedings of the 2014 Conference on Internet Measurement Conference*, ser. IMC '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 195–208.

[13] L. Ma, T. He, A. Swami, D. Towsley, and K. K. Leung, "On optimal monitor placement for localizing node failures via network tomography," *Performance Evaluation*, vol. 91, pp. 16–37, 2015, special Issue: Performance 2015.

[14] N. Bartolini, T. He, and H. Khamfroush, "Fundamental limits of failure identifiability by boolean network tomography," in *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, 2017, pp. 1–9.

[15] N. Duffield, "Simple network performance tomography," in *Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '03. New York, NY, USA: Association for Computing Machinery, 2003, p. 210–215.

[16] N. Duffield, "Network tomography of binary network performance characteristics," *IEEE Transactions on Information Theory*, vol. 52, no. 12, pp. 5373–5388, 2006.

[17] R. R. Kompella, J. Yates, A. Greenberg, and A. C. Snoeren, "Detection and localization of network black holes," in *IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications*, 2007, pp. 2180–2188.

[18] L. Ma, T. He, K. K. Leung, A. Swami, and D. Towsley, "Monitor placement for maximal identifiability in network tomography," in *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, 2014, pp. 1447–1455.

[19] J. D. Horton and A. López-Ortiz, "On the number of distributed measurement points for network tomography," in *Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*, 2003, pp. 204–209.

[20] M. Mukamoto, T. Matsuda, S. Hara, K. Takizawa, F. Ono, and R. Miura, "Adaptive boolean network tomography for link failure detection," in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2015, pp. 646–651.

[21] T. Fencl, P. Burget, and J. Bilek, "Network topology design," *Control Engineering Practice*, vol. 19, no. 11, pp. 1287–1296, 2011.

[22] T. Summers, I. Shames, J. Lygeros, and F. Dörfler, "Topology design for optimal network coherence," in *2015 European Control Conference (ECC)*. IEEE, 2015, pp. 575–580.

[23] N. Kamiyama and D. Satoh, "Network topology design using analytic hierarchy process," in *2008 IEEE International Conference on Communications*. IEEE, 2008, pp. 2048–2054.

[24] A. Akella, T. Benson, B. Chandrasekaran, C. Huang, B. Maggs, and D. Maltz, "A universal approach to data center network design," in *Proceedings of the 2015 International Conference on Distributed Computing and Networking*, 2015, pp. 1–10.

[25] B. Bamieh, M. R. Jovanovic, P. Mitra, and S. Patterson, "Coherence in large-scale networks: Dimension-dependent limitations of local feedback," *IEEE Transactions on Automatic Control*, vol. 57, no. 9, pp. 2235–2249, 2012.

[26] P. A. Lee and T. Anderson, "Fault tolerance," in *Fault Tolerance*. Springer, 1990, pp. 51–77.

[27] D. K. Pradhan, *Fault-tolerant computing: Theory and technique, Volume I*. Prentice Hall Inc., Old Tappan, NJ, 1986.

[28] V. P. Nelson, "Fault-tolerant computing: Fundamental concepts," *Computer*, vol. 23, no. 7, pp. 19–25, 1990.

[29] M. R. Garey and D. S. Johnson, *Computers and intractability*. freeman San Francisco, 1979, vol. 174.

[30] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The internet topology zoo," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 9, pp. 1765–1775, 2011.

Appendix

For ease of explanation, we use $\rightarrow$ to denote direct neighbour on a path. This notation does not mean that the measurement paths are unidirectional. All links are bidirectional in our study.

A. Proof of Theorem 2

Proof. Denote $d(v_i) = d$ and $\mathcal{N}(v_i) = \{v_1, v_2, \ldots, v_d\}$ $(d \leq k)$, where $\mathcal{N}(v_i)$ is the set of its neighbors.

Sufficiency: Support that $\mathcal{N}(v_i)$ includes two distinct monitors v_a and v_b $(a \neq b,\ a, b \in \{1, \ldots, d\})$. Then there exists one measurement path $p_i = v_a \rightarrow v_i \rightarrow v_b$. For any two failure set F_1 and F_2, where $v_i \in F_1$, $v_i \notin F_2$, and $|F_j| \leq k$ $(j \in \{1, 2\})$, we denote the incident measurement path sets by P_{F_1} and P_{F_2}, respectively. Then P_{F_1} contains p_i while P_{F_2} does not contain p_i. According to Definition 1, v_i is k-identifiable.

Necessity: Support that v_i is k-identifiable. There exist three cases as follows.

Case 1 (If $\mathcal{N}(v_i)$ includes no monitors): we fix two failure sets $F_1 = \{v_i, v_1, v_2, \ldots, v_{d-1}\}$ and $F_2 = \{v_1, v_2, \ldots, v_{d-1}\}$, i.e., v_i is contained in F_1 but not in F_2. Obviously, $|F_j| \leq k$, $j \in \{1, 2\}$ since $|F_1| = d, |F_2| = d - 1$. Then we have $P_{F_1} = P_{F_2} \cup P_{\{v_i\}}$. However, for any measurement path p_i that includes v_i, p_i can be represented by $p_i = \cdots \rightarrow v_a \rightarrow v_i \rightarrow v_b \ldots$, where $v_a, v_b \in \mathcal{N}(v_i)$. This means, $F_2 = \mathcal{N}(v_i) - \{v_d\}$ will contain either v_a or v_b. That is, $p_i \in P_{F_2}$. Since p_i is an arbitrary measurement path passing v_i, $P_{\{v_i\}} \subseteq P_{F_2}$. Thus, $P_{F_1} = P_{F_2}$, which is a contradiction to k-identifiability.

Case 2 (If $\mathcal{N}(v_i)$ includes only one monitor): without loss of generality, we denote the monitor by v_d. Select two failure sets $F_1 = \{v_i, v_1, v_2, \ldots, v_{d-1}\}$ and $F_2 = \{v_1, v_2, \ldots, v_{d-1}\}$. We have a contradiction using the same argument as in Case 1.

Case 3 (If $\mathcal{N}(v_i)$ includes at least two monitors): According to Sufficiency, if v_i has two monitors as neighbors, then v_i is k-identifiable.

Necessity is proved since Case 3 is the only possible case that k-identifiability holds. □

B. Proof of Theorem 3

Proof. For convenience, we use the symbol α to denote the percentage of monitors.

Given a network $\mathcal{G}$ where each node in $\mathcal{V}$ satisfies **Theorem 3**, based on **Definitions 1 and 2**, the network $\mathcal{G}$ is k-identifiable if for any failure set F ($|F| \leq k$), F satisfies **either** of the conditions above:

i) each node $v_i \in \mathcal{V} - F$ is passed by at least one measurement path p_i that does not contain any node in F;
ii) $|F| = k$ and each node failure in F can be localized in a sub-network of $\mathcal{G}$.

First, we prove that there exists a plan for monitor placement such that for any non-monitor node v_i and any v_i's non-monitor neighbor v_j, it holds: (a) $\mathcal{N}(v_i)$ and $\mathcal{N}(v_j)$ contain at least one monitor m_i and m_j, respectively, (b) $m_i \neq m_j$. To see why, for any failure set F, $\mathcal{N}(v_i)$ contains at least one node v_a that does not belong to F, because each node v_i has at least $k+1$ neighbors. Assume that v_a is a monitor. If $\mathcal{N}(v_a)$ also contains another monitor. Then we can remove the monitor from v_a to v_i and check the rest nodes.

After the monitor placement, $\mathcal{G}$ is k-identifiable for the following reasons:

- for any failure set F and one non-monitor node $v_i \in \mathcal{V}-F$, if there exists one of v_i's neighbors v_j such that $v_j \notin F$, then v_i is passed by one measurement path $p_i = m_i \rightarrow v_i \rightarrow v_j \rightarrow m_j$, where m_i/m_j is a monitor included in $\mathcal{N}(v_i)/\mathcal{N}(v_j)$ ($m_i \neq m_j$), respectively.
- if such v_j does not exist, then we can localize the node failure set $F = \mathcal{N}(v_i)$.

Then, to prove Theorem 3, we only need to design a plan of monitor placement satisfying conditions (a) and (b) above and meeting the requirement that the percentage of monitors α is no larger than $\frac{2}{k+2}$. The monitor placement plan is as follows.

Initially, we set an empty node set $\mathcal{V}' = \emptyset$.

Step 1: Select any node $v_i \in \mathcal{V} - \mathcal{V}'$. Let v_i be a monitor. Then each node in $\mathcal{N}(v_i)$ has one monitor as its neighbor.

Step 2: Let $\mathcal{G}-\mathcal{V}'$ denote a subgraph of $\mathcal{G}$ by removing nodes in $\mathcal{V}'$ from $\mathcal{G}$. If $\mathcal{G} - \mathcal{V}'$ is a complete graph, we randomly select one neighbor of v_i as another monitor. Then we have $\alpha \leq \frac{2}{k+2}$. Update $\mathcal{V}' \leftarrow \mathcal{V}' \cup (\mathcal{G} - \mathcal{V}')$. If $\mathcal{G} - \mathcal{V}'$ is not a complete graph and there exists at least one neighbor $v_j \in \mathcal{N}(v_i)$ such that $\mathcal{N}(v_j) - \{v_i\} \subset \mathcal{N}(v_i)$, we change the monitor placement from v_i to v_j. For convenience, we use symbol v_a to denote the latest monitor (v_i or v_j).

Step 3: For v_a, if there exists at least one node that is not v_a's neighbor, there are three cases as follows:

- **Case 1** (If there exists only one node v_b that $\mathcal{N}(v_b) \subseteq \mathcal{N}(v_a)$): We let v_b be a monitor. Then for $\mathcal{N}(v_a) + \{v_a, v_b\}$, $\alpha \leq \frac{2}{k+3}$. Update $\mathcal{V}' \leftarrow \mathcal{V}' \cup \mathcal{N}(v_a) \cup \{v_a, v_b\}$.
- **Case 2** (If there exist at least two nodes v_l ($l \in \{1, 2, \dots\}$) that $\mathcal{N}(v_l) \subseteq \mathcal{N}(v_a)$): We denote such two nodes by v_1 and v_2.
  - **Case 2.1** If the set of neighbors $\mathcal{N}(v_a)$ satisfies that $|\mathcal{N}(v_a)| \leq 2k+1$, then there exists one node v_c that is a common neighbor of nodes v_a, v_1 and v_2 simultaneously. Let v_c become a monitor. Then for $\mathcal{N}(v_a) + \{v_a, v_1, v_2\}$, $\alpha \leq \frac{2}{k+4}$. Update $\mathcal{V}' \leftarrow \mathcal{V}' \cup \mathcal{N}(v_a) \cup \{v_a, v_1, v_2\}$.
  - **Case 2.2** If the set of neighbors $\mathcal{N}(v_a)$ satisfies that $\mathcal{N}(v_a) \geq 2k+2$. Follow **Case 2.1** if v_1 and v_2 have a common neighbor. Otherwise, we select v_1 and v_2 as monitors. For $\mathcal{N}(v_a) + \{v_a, v_1, v_2\}$, $\alpha \leq \frac{3}{2k+5}$. Update $\mathcal{V}' \leftarrow \mathcal{V}' \cup \mathcal{N}(v_a) \cup \{v_a, v_1, v_2\}$.
- **Case 3** Otherwise, update $\mathcal{V}' \leftarrow \mathcal{V}' \cup \mathcal{N}(v_a) \cup \{v_a\}$.

Step 4: Exclude all nodes that have been covered by any one of the above cases. Iterate Steps 1-3 for the remaining nodes in $\mathcal{G}$ until no node is left, i.e., until $\mathcal{V}' = \mathcal{V}$. □

Note that the above monitor placement plan is only for deriving the **worst-case** upper bound on the number of monitors in **Theorem 3**. In practice, we actually do not need so many monitors as demonstrated in Section VI.

C. Proof of Theorem 4

Proof. According to **Definition 1**, any non-monitor node v_i is k-identifiable if and only if for any two node failure sets F_1 and F_2 ($v_i \in F_1, v_i \notin F_2, |F_1| \leq k, |F_2| \leq k$), there always exists one measurement path p_i containing at least one node in F_1/F_2 but none node in F_2/F_1. Then, we give analysis of two cases, respectively:

Case 1: for any non-monitor node v_i that satisfies **Condition 1** (i.e., $d(v_i) \geq k+2$), we denote the set of v_i's neighbors as $\mathcal{N}(v_i) = \{v_1, v_2, \dots, v_{k+2}\}$. For any possible node failure set F that does not contain v_i (even $F \subset \mathcal{N}(v_i)$), $\mathcal{N}(v_i)$ still contains at least two normal nodes. Without loss of generality, we assume v_1 and v_2 are normal. Then we have a normal path segment $v_1 \rightarrow v_i \rightarrow v_2$, which means nodes whose degree not smaller than $k+2$ does not require any monitor as its neighbor.

Case 2: for any non-monitor node v_i that satisfies **Condition 2** (i.e., $d(v_i) = k+1$):

- if v_i's neighbors contains no monitor. We denote the set of v_i's neighbors as $\mathcal{N}(v_i) = \{v_1, v_2, \dots, v_{k+1}\}$. Consider a scenario where there exist only two nodes v_a and v_b such that $v_a \in \mathcal{N}(v_1), v_b \in \mathcal{N}(v_2), v_a, v_b \notin \mathcal{N}(v_i)$. Support two node failure sets $F_1 = \{v_i, v_3, \dots, v_{k+1}\}$ and $F_2 = \{v_2, v_3, \dots, v_{k+1}\}$. All paths, which pass v_i (or v_2) but none node in $\{v_3, \dots, v_{k+1}\}$, must contains the path segment $v_a \rightarrow v_1 \rightarrow v_i \rightarrow v_2 \rightarrow v_b$. This contradicts the concept of k-identifiability.
- if v_i's neighbors contains only one monitor. We consider another scenario where for each neighbor v_j ($j \in \{1, 2, \dots, k+1\}$), there exists only one node v_l such that $v_l \in \mathcal{N}(v_j), v_l \notin \mathcal{N}(v_i)$. The proof is the same as the former scenario.
- if v_i's neighbors contains at least two monitors. Only in this way, v_i always satisfies k-identifiability under any network topology.

Note that both counterexamples mentioned in **Case 2** are networks with specific topologies and it does not imply that nodes that satisfies **Condition 2** must require at least two monitors as neighbors. □