Service Placement for Detecting and Localizing Failures Using End-to-End Observations

Ting He[‡], Novella Bartolini[†], Hana Khamfroush[§], InJung Kim[§], Liang Ma[‡], and Tom La Porta[§]
[‡]IBM T. J. Watson Research Center, Yorktown, NY, USA. {the, maliang}@us.ibm.com
[†]Sapienza University of Rome, Rome, Italy. novella@di.uniroma1.it
[§]Pennsylvania State University, State College, PA, USA. {hkham, ikk5033, tlp}@cse.psu.edu

***Abstract*—We consider the problem of placing services in a telecommunication network in the presence of failures. In contrast to existing service placement algorithms that focus on optimizing the quality of service (QoS), we consider the performance of monitoring failures from end-to-end connection states between clients and servers, and investigate service placement algorithms that optimize the monitoring performance subject to QoS constraints. Based on novel performance measures capturing the coverage, the identifiability, and the distinguishability in monitoring failures, we formulate the service placement problem as a set of combinatorial optimizations with these measures as objective functions. In particular, we show that maximizing the distinguishability is equivalent to minimizing the uncertainty in failure localization. We prove that all these optimizations are NP-hard. However, we show that the objectives of coverage and distinguishability have a desirable property that allows them to be approximated to a constant factor by a greedy algorithm. We further show that while the identifiability objective does not have this property, it can be approximated by the maximum-distinguishability placement in the high-identifiability regime. Our evaluations based on real network topologies verify the effectiveness of the proposed algorithms in improving the monitoring performance compared with QoS-based service placement.**

I. Introduction

The rapid progress of IT technologies has revolutionized how people view telecommunication networks. As envisioned by the initiative of *Network Functions Virtualization (NFV)* [1], future networks will replace fixed network services implemented by proprietary hardware with virtualized network services implemented by software running on general-purpose IT infrastructure. This allows services to be shared efficiently across different service and network providers. A main advantage of virtualizing network services is the flexibility in their placement, i.e., a service provider implementing NFV can place the virtualized services at various locations ranging from servers in data centers to network nodes on customer premises. Proper placement of such services is thus crucial for realizing the benefit of NFV.

Meanwhile, the increased complexity of virtualized networks also implies increased risk of failures due to faults/misconfigurations at various levels. While low-level failures (e.g., failures in the physical nodes/links) can be detected by in-network mechanisms, many high-level failures (e.g., failures caused by software bugs, black holes, firewalls, and other unanticipated issues) can only be observed from end-to-end connections [2], [3]. In networks where the service provider and the network provider belong to different administration domains, end-to-end connection states are the only first-hand evidence in trouble-shooting services. Generally, states of end-to-end connections between service hosts and clients provide the most accurate measurements of the network state as it manifests in the service layer. Moreover, such measurements can be taken as a byproduct of fulfilling the service, thus saving the overhead of active probes. The challenge is that service-layer observations are coarse-grained, e.g., if a client-server connection fails, we can only infer that there is at least one failed element on the connection path (including endpoints), but not the exact number or locations of failures. With observations from a diverse set of paths resulting from monitoring many client-server connections, it becomes possible to infer the failure locations using a technique known as *(Boolean) network tomography* [4]. The capability of such failure localization is, however, limited by the set of observable paths [5], which in turn depend on the service placement.

While service placement has been extensively studied in the literature, the predominant goal has been to optimize the *quality of service (QoS)* [6]. From a network monitoring perspective, however, the best-QoS placement may not provide sufficient information to effectively localize failures. In a network prone to failures, a good service placement should achieve dual objectives: during normal operations, it should provide satisfactory QoS; in case of failures, it should allow accurate detection and localization of the failed network elements, which then helps to speed up recovery.

In this work, we unify these objectives by investigating service placement algorithms that provide the most useful observations for monitoring failures while satisfying given QoS constraints. The high-level questions we seek to answer are: (i) How do we quantify the "usefulness", in terms of measurable metrics, of a given service placement for monitoring failures? (ii) How can we efficiently select a service placement among exponentially many candidate placements to maximize this usefulness? (iii) What is the tradeoff between the QoS and

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

the monitoring performance?

A. Summary of Contributions

In this work, we revisit the service placement problem from the perspective of monitoring failures based on binary service-layer observations. Our contributions are:

1) We propose a set of performance measures that capture key aspects of failure monitoring, including the capabilities of detecting failures (*coverage*), uniquely determining node states (*identifiability*), and distinguishing between candidate sets of failure locations (*distinguishability*).

2) Using the proposed measures as objective functions, we formulate the service placement problem as a combinatorial optimization of maximizing a selected objective subject to QoS constraints. We show that the optimal placement is NP-hard to compute for all the above objectives.

3) We show that under the coverage or distinguishability objective, our problem can be cast as a submodular maximization under matroid constraints, which can be approximated to a factor of $1/2$ by a greedy algorithm. We show that although the identifiability objective is not submodular, it can be approximated by the distinguishability-based placement in the high-identifiability regime.

4) We evaluate the proposed algorithms on real network topologies, which shows that: (i) the proposed algorithms achieve significantly better monitoring performance than the best-QoS placement, and (ii) the distinguishability objective leads to a service placement with the best overall performance across all the three objectives.

Note that we only consider minimum available observations (success/failure in accessing a given service) to study the basic capability of monitoring failures using service-layer observations. These minimum observations can be augmented with other information (e.g., traceroutes and other active probes) to uniquely localize failures. In this sense, our solution minimizes the need of additional measurements.

B. Related Work

Service placement has been extensively studied based on the *facility location theory* [6], where two of the most well-known formulations are: (1) the *uncapacitated k-median* problem, which optimizes the locations for placing a fixed number of services to minimize the distance between clients and (closest) servers, and (2) the *uncapacitated facility location* problem, which jointly optimizes the number and the locations of services to minimize the combined cost of hosting and accessing services. The classic formulations have been extended in various directions, such as distributed service placement in large-scale networks [7] and iterative service placement/migration in mobile ad hoc networks [8]. The focus, however, remains on optimizing the QoS and provisioning cost.

A parallel line of works study the placement of nodes dedicated to network monitoring (called *monitors*). Under round-trip probing (e.g., ping, traceroute) where only sources of probes need to be monitors, [9] shows that the optimal monitor placement is NP-hard and proposes a greedy approximation algorithm. Under one-way probing where both sources and destinations need to be monitors, [10] and references therein propose polynomial-time algorithms to place a minimum number of monitors to uniquely localize a given number of failures. Our problem differs in that: (i) we only control one endpoint (server) of each measurement path, and (ii) the service placement must satisfy QoS constraints.

Given the states of a set of paths, there may be multiple sets of failure locations that are consistent with the path states. Most existing works handle such ambiguity by *best-effort* solutions. For example, [9], [11] assume one failure at a time, [12], [4], [2] attempt to find a minimum set of failures that can explain all the observed path states. Followup works try to improve accuracy by looking for failures that occur with higher probabilities [13] or improving the accuracy in estimating path states [3]. There is, however, lack of understanding in the fundamental capability of failure localization. Recently, [5] proposes to model this capability by the maximum number of failures that can be uniquely localized, called *maximum identifiability*. It develops tight upper/lower bounds on the maximum identifiability and polynomial-time algorithms to compute the bounds for several types of measurement paths (arbitrarily controllable, controllable but cycle-free, or uncontrollable).

We study, *for the first time*, the impact of service placement on the capability of monitoring failures based on end-to-end measurements between clients and servers. We adopt the uncontrollable path assumption in [5], but substantially generalize the performance measure in [5]: (i) while the maximum identifiability measure requires all the node states to be identifiable, our new identifiability measure captures cases where only a subset of node states are identifiable; (ii) we propose novel measures capturing two other aspects of failure monitoring, including the capability of detecting failures and the capability of reducing uncertainty in failure locations.

The rest of the paper is organized as follows: Section II formulates the service placement problem, Section III discusses the prerequisites of solving this problem, Section IV presents the hardness results, Section V presents efficient approximate solutions, Section VI evaluates the proposed solutions, Section VII discusses extensions of the solutions, and Section VIII concludes the paper.

II. Problem Formulation

A. Network Model

We model the service network as an undirected graph $\mathcal{G} = (N, L)$, where N is the set of nodes, including client nodes, (candidate) server nodes, and communication nodes in between, and L is the set of communication links. Each node is associated with a binary state: *normal* or *failed*. Here a "client node" represents an access point for end-clients in the service network, and thus its state is also of interest to the service provider. We assume links do not fail, as link failures can be modeled by the failures of logical nodes that represent the links. Given a vector of node states, the set of *all* failed nodes is called a *failure set*, denoted by F. We assume that

node states cannot be measured directly, but only indirectly via *measurement paths*. Let P be a given set of measurement paths comprising the paths between all servers and all clients interested in their services (see Section II-C), with one path per client-server pair as determined by the underlying routing protocol employed by the network. Each path $p \in P$ is represented as a *set* of nodes traversed by a client-server connection, whose state is normal if and only if all the traversed nodes (including endpoints) are in normal states. We use $P_F \subseteq P$ to denote the subset of paths affected by a failure set F (i.e., traversing at least one node in F); in particular, $P_v \subseteq P$ denotes the subset of paths traversing node v.

B. Performance Measure of Network Monitoring

Given a set of measurement paths P, we quantify the value of P in monitoring the node states as follows.

1) Coverage: A basic objective is to detect node failures from failures of measurement paths. Denote the set of *covered* nodes, i.e., nodes traversed by at least one path in P, by $\mathcal{C}(P) \triangleq \bigcup_{p\in P} p$. Then $|\mathcal{C}(P)|$ measures the number of nodes whose failures are detectable from measurements in P.

2) Identifiability: Besides detection, it is also important to localize the failures, i.e., determine the failure set F from observed path states. Generally, there can be multiple failure sets that generate the same path states, leading to ambiguity. The extent to which we can overcome such ambiguity thus measures our capability of localizing failures.

Intuitively, two failure sets can be distinguished if and only if they lead to different states for at least one measurement path, formalized as follows.

Definition 1 ([5])**.** *Given a set of measurement paths P, two failure sets F_1 and F_2 are* distinguishable *with respect to (wrt) P if $P_{F_1} \neq P_{F_2}$ (i.e., $\exists p \in P$ that fails under only one failure set) and* indistinguishable *otherwise.*

Based on this definition, one measure of the failure localization capability is the number of nodes whose states can be determined without ambiguity. To formalize this idea, we introduce the following definition.

Definition 2 ([10])**.** *Given a set of measurement paths P and a node $v \in N$, v is k-*identifiable *wrt P if for any failure sets F_1 and F_2 satisfying (1) $|F_i| \leq k$ ($i = 1, 2$) and (2) $F_1 \cap \{v\} \neq F_2 \cap \{v\}$, F_1 and F_2 are distinguishable wrt P.*

If a node v is k-identifiable wrt P, then the state of v can be uniquely determined from measurements in P as long as the total number of failures is bounded by k, because any indistinguishable failure sets of up to k nodes must imply the same state for v (i.e., both containing or excluding v). Definition 2 differs from the definition of k-identifiability in [5] in that [5] requires every two different failure sets to be distinguishable, while Definition 2 only requires failure sets that differ in v to be distinguishable. Note that k is an input parameter representing the maximum number of failures that the system is designed to handle.

Let $S_k(P)$ denote the set of all the k-identifiable nodes in the network. Its size $|S_k(P)|$ thus measures the number of nodes whose states can be *uniquely* determined.

3) Distinguishability: Another measure of the capability of failure localization is our ability to distinguish between candidate failure sets. Given a maximum number of failures k, let $\mathcal{F}_k \triangleq \{F \subseteq N : |F| \leq k\}$ denote the collection of all possible failure sets. Define $D_k(P) \triangleq \{(F,\ F') \in \mathcal{F}_k^2 :\ P_F \neq P_{F'}\}$ as the set of (unordered) pairs of failure sets in $\mathcal{F}_k$ that are distinguishable from each other. Its size $|D_k(P)|$ measures the distinguishability in failure localization.

The significance of this measure is that it is directly related to the uncertainty in localizing failures. Specifically, given a set of measurement paths P, let $\mathcal{I}_k(F;\ P) \triangleq \{F' \in \mathcal{F}_k \setminus F :\ P_{F'} = P_F\}$ be the failure sets indistinguishable from F. Then $|\mathcal{I}_k(F;\ P)|$ measures the uncertainty in localizing failures, if the true failure set is F. We have the following relationship.

Lemma 3. *The average uncertainty in failure localization, measured by $\frac{1}{|\mathcal{F}_k|}\sum_{F\in\mathcal{F}_k} |\mathcal{I}_k(F;\ P)|$, equals $\frac{2}{|\mathcal{F}_k|}(\binom{|\mathcal{F}_k|}{2} - |D_k(P)|)$.*

Proof. Define the set of (unordered) indistinguishable pairs of failure sets as $\mathcal{I}_k(P) \triangleq \{(F, F') \in \mathcal{F}_k^2 : F \neq F', P_F = P_{F'}\}$. Then each $(F,\ F') \in \mathcal{I}_k(P)$ contributes 2 to the summation $\sum_{F\in\mathcal{F}_k} |\mathcal{I}_k(F;\ P)|$, as $F \in \mathcal{I}_k(F';\ P)$ and $F' \in \mathcal{I}_k(F;\ P)$. Thus, the average uncertainty $\frac{1}{|\mathcal{F}_k|}\sum_{F\in\mathcal{F}_k} |\mathcal{I}_k(F;\ P)|$ equals

$$\frac{2}{|\mathcal{F}_k|}|\mathcal{I}_k(P)| = \frac{2}{|\mathcal{F}_k|}\left(\binom{|\mathcal{F}_k|}{2} - |D_k(P)|\right),$$

where $\binom{|\mathcal{F}_k|}{2}$ is the total number of (unordered) pairs. □

C. Monitoring-Aware Service Placement

It is clear from Section II-B that the performance of network monitoring depends on P, the set of paths connecting servers and clients, which are determined by the network topology, the locations of clients, the routing of service requests/responses, and the positioning of services. The last parameter, positioning of services, is of particular interest as it is controlled by the service placement algorithm. We now formally define the service placement problem when taking network monitoring performance into account.

Given a service network $\mathcal{G} = (N,\ L)$ and a set of services $\mathcal{S} = \{s_1, \ldots, s_{|\mathcal{S}|}\}$, the goal of service placement is to map each service $s \in \mathcal{S}$ onto a node $v \in N$ to optimize a certain performance objective under feasibility constraints. Different from traditional service placement that aims at optimizing QoS or cost, we want to maximize the capability in monitoring the network in case of failures, while satisfying given QoS constraints in normal circumstances.

Specifically, let $C_s \subseteq N$ denote the locations of clients interested in service s. We model the QoS constraints by a set of *candidate hosts* H_s, such that all clients in C_s can be served with satisfactory QoS if service s is placed at any node in H_s (see Section III-A). Given the routing of service requests and responses, each request from client c to host h traverses a set

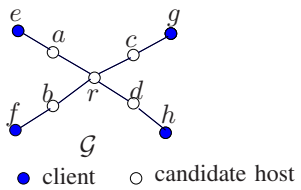


Fig. 1. Service placement example.

of nodes $p(c, h) \subseteq N$, which have to be all in normal states for the request to be served successfully. We refer to the set $p(c, h)$ as a measurement path. Let h_s denote the host for service $s \in \mathcal{S}$ and $P(C_s, h_s) \triangleq \{p(c, h_s) : c \in C_s\}$ the corresponding set of measurement paths. Then $\bigcup_{s\in\mathcal{S}} P(C_s, h_s)$ is the overall set of measurement paths when placing services according to $\mathbf{h} \triangleq (h_s)_{s\in\mathcal{S}}$.

Given a measure $f(P)$ of the monitoring performance using measurement paths P, the problem of *monitoring-aware service placement* is defined as:

$$\max f(\bigcup_{s\in\mathcal{S}} P(C_s,\ h_s)) \tag{1}$$

$$\text{s.t. } h_s \in H_s, \quad \forall s \in \mathcal{S}. \tag{2}$$

That is, we want to select a host for each service $s \in \mathcal{S}$ from its candidate hosts such that the paths between the host and the corresponding clients provide the best network monitoring performance measured by $f(P)$.

Using the measures proposed in Section II-B as concrete examples, we refer to the problem (1) as *maximum-coverage service placement (MCSP)* for $f(P) = |\mathcal{C}(P)|$, *maximum-identifiability service placement (MISP)* for $f(P) = |S_k(P)|$, and *maximum-distinguishability service placement (MDSP)* for $f(P) = |D_k(P)|$.

Example: Consider the example in Fig. 1, where there are five services, all having clients $\{e, f, g, h\}$ and candidate hosts $\{r, a, b, c, d\}$. Suppose that at most one node may fail ($k = 1$). Recall that the failed node may be a client (access point). If only considering the QoS, we should place all the services on node r as it minimizes the maximum distance to clients. This placement generates measurement paths $\{e, a, r\}$, $\{f, b, r\}$, $\{g, c, r\}$, and $\{h, d, r\}$, which cover all the nodes but only allow the identification of the state of node r, as the failures of e and a, f and b, g and c, and h and d are indistinguishable. If we place one service on each candidate host, we will have 16 additional measurement paths between each node in $\{a, b, c, d\}$ and each node in $\{e, f, g, h\}$, which not only cover all the nodes but also allow their states to be uniquely identified. This example shows that by adjusting service placement, we can monitor failures much more effectively.

III. Prerequisites

We start by considering two prerequisites of solving problem (1): how to determine the candidate hosts and how to evaluate the objective function for a given service placement.

A. Computing Candidate Set

The goal of computing the candidate set H_s is to ensure a minimum QoS for all clients while maximizing the flexibility for service placement. As a concrete example, we consider latency as the QoS measure. Let $d(C_s, h)$ denote the maximum distance (in hop count) between node h and any client in C_s under a given routing protocol. Then a natural way of guaranteeing QoS is to impose an upper bound on $d(C_s,\ h)$ and only view nodes satisfying the upper bound as candidate hosts. This definition of the candidate set however requires careful selection of the bound for each problem instance to guarantee feasibility. Instead, we adopt a relative QoS constraint by bounding the *QoS degradation* compared with a placement that optimizes the QoS. Let $d_{\min}(C_s) \triangleq \min_h d(C_s,\ h)$ and $d_{\max}(C_s) \triangleq \max_h d(C_s,\ h)$ be the minimum and the maximum distances between clients C_s and any possible service location. We define the *relative distance* $\widetilde{d}(C_s,\ h)$ as

$$\widetilde{d}(C_s,\ h) \triangleq \frac{d(C_s,\ h) - d_{\min}(C_s)}{d_{\max}(C_s) - d_{\min}(C_s)}, \tag{3}$$

i.e., the increase over the minimum distance normalized by the maximum possible increase. By definition, the relative distance is always bounded in $[0,\ 1]$. Given a *maximum tolerable relative distance* α_s, the candidate host set for service s is given by $H_s \triangleq \{h \in N :\ \widetilde{d}(C_s,\ h) \le \alpha_s\}$, and is guaranteed to be nonempty for any $\alpha_s \ge 0$ (as it contains at least the host achieving $d_{\min}(C_s)$). Thus, the problem of computing the candidate host set is to find the set of locations and their corresponding $d(C_s,\ h)$ such that (3) is no more than α_s.

According to the above definition, we can compute the candidate host set in two steps: (i) computing the routing distance $d(c,\ h)$ from each potential candidate host $h \in N$ to each client $c \in \bigcup_{s\in\mathcal{S}} C_s$, and (ii) for each service s, computing the worst-case distance $d(C_s,\ h) = \max_{c\in C_s} d(c,\ h)$ and then the relative distance $\widetilde{d}(C_s,\ h)$ as in (3). The candidate set H_s is then formed by all the nodes with $\widetilde{d}(C_s,\ h) \le \alpha_s$.

Complexity: For shortest path routing, step (i) can be implemented by the Dijkstra's algorithm with a complexity of $O(|L|+|N|\log|N|)$ per host, and step (ii) has a complexity of $O(|N|^2)$ per service, dominated by the computation of worst-case distances. Thus, computing the candidate set has a total complexity of $O(|N||L| + |N|^2 \log|N| + |N|^2|\mathcal{S}|)$.

Remark: The above only considers QoS constraints; see Section VII-A for how to handle capacity constraints.

B. Computing Objective Function

Another prerequisite is an efficient method to evaluate the objective function in problem (1). For coverage, we can easily compute $|\mathcal{C}(P)|$ by taking the union of all measurement paths. For distinguishability, we have to compute $|D_k(P)|$ by enumerating all pairs of failure sets and testing whether they affect the same set of measurement paths, which requires $O(|N|^{2k})$ tests, each of complexity $O(|P|)$ (assuming $k \ll |N|$).

For identifiability, we leverage the following result from [14]. Given a node $v \in N$, we define the *minimum set cover* for node v under measurement paths P, denoted by $\text{MSC}(v;\ P)$, as the minimum number of nodes other than v whose failures will disrupt all the paths in P that traverse v, i.e., it is the size of the minimum cover of path set P_v by

$\{P_w : w \in N \setminus \{v\}\}$. We can express a sufficient and a necessary condition on the k-identifiability of v in terms of MSC$(v;\ P)$ as follows.

Theorem 4 ([14])**.** *Given a set of measurement paths P, node v is k-identifiable wrt P:*

a) *if MSC$(v;\ P) \geq k+1$;*
b) *only if MSC$(v;\ P) \geq k$.*

These conditions directly lead to the following bounds on the set of k-identifiable nodes.

Corollary 5 ([14])**.** *Let $\widetilde{S}_k(P) \triangleq \{v \in N : \text{MSC}(v; P) \geq k\}$. Then $\widetilde{S}_{k+1}(P) \subseteq S_k(P) \subseteq \widetilde{S}_k(P)$.*

By Corollary 5, we can bound the identifiability objective $|S_k(P)|$ from above (below) by counting the number of nodes with MSC$(v;\ P)$ of at least k $(k+1)$. Although computing MSC$(v;\ P)$ is itself computationally hard as it requires solving the NP-complete *set covering problem (SCP)*, the above bounds allow us to borrow techniques from SCP. Specifically, the greedy algorithm for SCP[1] has an approximation ratio of $\log|P_v|+1$ [15], i.e., the size of the greedily selected set cover, denoted by GSC$(v;\ P)$, satisfies GSC$(v;\ P)/(\log|P_v|+1) \leq$ MSC$(v;\ P) \leq$ GSC$(v;\ P)$. This implies a relaxed bound of

$$|\{v \in N :\ \frac{\text{GSC}(v;\ P)}{\log|P_v|+1} \geq k+1\}| \leq |S_k(P)| \leq |\{v \in N :\ \text{GSC}(v;\ P) \geq k\}|. \quad (4)$$

Note that the lower bound is usually loose, as we have observed that GSC$(v;\ P) \approx$ MSC$(v;\ P)$ in most cases [5].

1) Special Case of $k = 1$: A case of particular interest is $k = 1$, which models single-node failure. In this case, there are only $|N|+1$ possible failure sets, i.e., $\mathcal{F}_1 = \{\emptyset\} \cup \{\{v\} : v \in N\}$ (including the case of no failure). Each node v is 1-identifiable if and only if (i) there is at least one path traversing v, and (ii) there is at least one path traversing only one of v and w for any other node $w \in N \setminus v$. We can compute $|S_1(P)|$ and $|D_1(P)|$ using the following data structure.

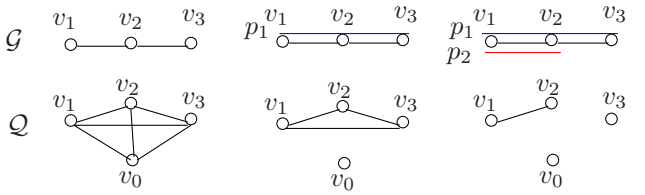
Define the *equivalence graph* $\mathcal{Q}$ as an undirected graph on the set of nodes in N plus a virtual node v_0 representing the case of no failure. There is a link between v and w in $\mathcal{Q}$ if and only if failure sets $\{v\}$ and $\{w\}$ are indistinguishable (i.e., $P_v = P_w$). By definition, $P_{v_0} = \emptyset$. A key observation is that once $\{v\}$ and $\{w\}$ become distinguishable, they remain distinguishable after adding new measurement paths. Thus, $\mathcal{Q}$ can be constructed incrementally as in Algorithm 1. Starting from a complete graph (line 1), we sequentially consider each measurement path $p \in P$ and remove the links between all pairs of nodes distinguished by p, including (i) a node traversed by p and the virtual node v_0 (line 4) and (ii) a node traversed by p and a node not traversed by p (line 6). The complexity of Algorithm 1 is $O(|N|^2|P|)$. Fig. 2 illustrates the process of constructing $\mathcal{Q}$.

[1]This algorithm iteratively selects a set from $\{P_w : w \in N \setminus \{v\}\}$ that covers the most uncovered paths in P_v, until all the paths in P_v are covered.

Algorithm 1: Construct Equivalence Graph

Input: A set of nodes N and a set of measurement paths P
Output: Equivalence graph $\mathcal{Q}$ wrt P

1 $\mathcal{Q} \leftarrow$ complete graph with vertices $\{v_0\} \cup N$
2 **for** *each path* $p \in P$ **do**
3 **for** *each node* $v \in p$ **do**
4 remove edge $(v,\ v_0)$ in $\mathcal{Q}$
5 **for** *each node* $w \in N \setminus p$ **do**
6 remove edge $(v,\ w)$ in $\mathcal{Q}$
7 return $\mathcal{Q}$



(a) initial state (b) after adding p_1 (c) after adding p_2

Fig. 2. Construct equivalence graph $\mathcal{Q}$.

By definition, $|S_1(P)|$ is the number of *isolated* nodes in Q (excluding v_0), and $|D_1(P)|$ is the number of links in the *complementary* graph of $\mathcal{Q}$ (i.e., not in $\mathcal{Q}$). In the example in Fig. 2 (c), $|S_1(P)| = 1$, and $|D_1(P)| = 5$.

IV. Hardness of Optimal Solution

Given a set of candidate hosts and a method to evaluate the objective function, it remains to find the optimal service placement among exponentially many candidate placements. We show that finding the optimal placement is hard even in the simplest case of monitoring single-node failures ($k = 1$).

Our analysis is based on a classic NP-hard problem known as the *maximum coverage problem (MCP)*. Given a ground set E and a collection of sets $\{U_i\}_{i=1}^n$ $(U_i \subseteq E)$, MCP aims at selecting a subcollection $\{U_i\}_{i\in I}$ $(|I| \leq M)$ that maximizes $|\bigcup_{i\in I} U_i|$.

For the coverage objective, we show that the corresponding service placement problem, MCSP, is a generalization of MCP and is therefore NP-hard.

Proposition 6. *MCSP is NP-hard.*

Proof. Denote by $\mathcal{N}_{s,h} \triangleq \bigcup_{c\in C_s} p(c,\ h)$ the set of nodes covered by paths between clients of service s and a candidate host h. MCSP selects one set $\mathcal{N}_{s,h_s} \in \{\mathcal{N}_{s,h} : h \in H_s\}$ for each $s \in \mathcal{S}$ to maximize $|\bigcup_{s\in\mathcal{S}} \mathcal{N}_{s,h_s}|$. In the special case of $C_s \equiv C$ and $H_s \equiv H$ for all $s \in \mathcal{S}$, it reduces to the classic MCP that selects $|\mathcal{S}|$ sets from $\{\mathcal{N}_h : h \in H\}$ $(\mathcal{N}_h \triangleq \mathcal{N}_{s,h}$ for any $s)$ to maximize coverage. The result follows from the NP-hardness of MCP. □

For the identifiability objective, intuitively, MISP for $k = 1$ wants to maximize the number of nodes that are *uniquely covered*, i.e., traversed by a non-empty and unique set of paths. We show that this problem is harder than MCP and therefore NP-hard.

Proposition 7. *MISP is NP-hard even if $k = 1$.*

Proof. We prove the result by reducing MCP to MISP with $k = 1$. The result then follows from the NP-hardness of MCP.

Consider an MCP with a ground set E, a collection of sets $\{U_i\}_{i=1}^n$ ($U_i \subseteq E$), and a maximum number of selected sets M. Without loss of generality, assume $|U_i| > 1$ ($i = 1, \ldots, n$), as otherwise we can add a virtual element $e_0 \notin E$ to all U_i without changing the optimal solution. We construct an instance of MISP as follows. For each U_i, define a set of paths $\mathcal{U}_i \triangleq \{\{e, v_i\} : e \in U_i\}$, where $v_i \notin E$. Viewing $(U_i, \{v_i\})$ as the client-host set of a service, where each client $e \in U_i$ is directly connected to the host v_i (i.e., $p(e, v_i) = \{e, v_i\}$), we form an MISP that aims at selecting $I \subseteq \{1, \ldots, n\}$ ($|I| \leq M$) to maximize the number of 1-identifiable nodes in $E \cup \{v_i\}_{i=1}^n$ wrt paths in $\bigcup_{i \in I} \mathcal{U}_i$.

Given a solution I, each node $v \in U_i \cup \{v_i\}$ ($i \in I$) is 1-identifiable, because: (i) if $v = v_i$, it is distinguished from any node covered by $\mathcal{U}_j$ ($j \neq i$) by a path $\{e, v_i\}$ for $e \in U_i$, and from any node $e \in U_i$ by a path $\{e', v_i\}$ for $e' \in U_i \setminus \{e\}$ (e' must exist since $|U_i| > 1$); (ii) if $v \in U_i$, it is distinguished from any node other than v_i by the path $\{v, v_i\}$, and from v_i by a path $\{e, v_i\}$ for $e \in U_i \setminus \{v\}$. Thus, the number of 1-identifiable nodes equals $|\bigcup_{i \in I} U_i| + M$ (it suffices to consider $|I| = M$). Therefore, an optimal solution to the MISP with $k = 1$ gives an optimal solution to the MCP. □

Using similar arguments, we show that the distinguishability objective is also hard to optimize.

Proposition 8. *MDSP is NP-hard even if $k = 1$.*

Proof. We reduce MCP to MDSP with $k = 1$. Construct a service placement problem from the input of an MCP as in the proof of Proposition 7. MDSP aims at selecting $I \subseteq \{1, \ldots, n\}$ ($|I| \leq M$) to maximize the number of pairs of nodes in $E \cup \{v_i\}_{i=0}^n$ that are distinguishable from each other (v_0 is a virtual node not on any path).

Given a solution I, let $\mathcal{C}(I) = \bigcup_{i \in I} U_i \cup \{v_i\}_{i \in I}$ denote the set of covered nodes and $\mathcal{N} = E \cup \{v_i\}_{i=1}^n$ the set of all the nodes excluding v_0. Let $c(I) = |\mathcal{C}(I)|$ and $\mu = |\mathcal{N}|$. From the proof of Proposition 7, we know that every covered node is 1-identifiable and thus distinguishable from any other node. Then the number of distinguishable pairs equals

$$c(I) + \frac{1}{2} c(I)(c(I) - 1) + c(I)(\mu - c(I)), \tag{5}$$

where the first term corresponds to pairs between $\mathcal{C}(I)$ and $\{v_0\}$, the second term to pairs in $\mathcal{C}(I)$, and the third term to pairs between $\mathcal{C}(I)$ and $\mathcal{N} \setminus \mathcal{C}(I)$. Since (5) is increasing in $c(I)$ on $[0, \mu]$, the optimal I for MDSP that maximizes (5) also maximizes $c(I)$, and is hence the optimal solution for the MCP (note that $c(I) = |\bigcup_{i \in I} U_i| + M$ for $|I| = M$). □

V. Efficient Approximation

The NP-hardness of the optimal solution motivates us to seek for efficient suboptimal solutions with guaranteed performance. To this end, we identify properties of the objective functions that allow for easy approximation.

A. Service Placement as Matroid Optimization

We introduce the following concepts from combinatorial optimization.

Definition 9 (Matroid [16])**.** *A matroid $\mathbb{M}$ is a pair (E, I), where E is a finite ground set and $I \subseteq 2^E$ a non-empty collection of subsets of E, with the following properties:*

- *$\forall A \subset B \subseteq E$, if $B \in I$, then $A \in I$;*
- *$\forall A, B \in I$ with $|B| > |A|$, $\exists x \in B \setminus A$ such that $A \cup \{x\} \in I$.*

Definition 10 (Monotone submodular function [16])**.** *Given a finite ground set E and a function $f : 2^E \to \mathbb{R}$,*

- *f is monotone if $\forall A \subset B \subseteq E$, $f(A) \leq f(B)$;*
- *f is submodular if $\forall A \subset B \subseteq E$ and $e \in E \setminus B$, $f(A \cup \{e\}) - f(A) \geq f(B \cup \{e\}) - f(B)$.*

The significance of these definitions is that if our objective function is monotone submodular and our constraint forms a matroid, then we can apply techniques from combinatorial optimization with a known approximation guarantee. In particular, a greedy algorithm that iteratively selects an element, subject to the constraint, that maximizes the objective function achieves a near-optimal[2] approximation ratio of $1/2$.

Theorem 11 ([18])**.** *Consider the problem of maximizing a set function $f : 2^E \to \mathbb{R}$ over a collection $I \subseteq 2^E$ of sets. Let f^* denote the optimal value and f^g the value achieved by the greedy algorithm. If $\mathbb{M} = (E, I)$ is a matroid and f is monotone and submodular, then $f^g \geq f^*/2$.*

We will show that our service placement problem can be cast as maximizing a set function under matroid constraints.

1) Matroid Constraints: In our problem, the "ground set" is the collection of all path sets corresponding to feasible service placements $\{P(C_s, h) : s \in \mathcal{S}, h \in H_s\}$, and each "element" is a set of measurement paths $P(C_s, h)$ between all the clients C_s of a given service and a candidate host h. Each service placement effectively selects $|\mathcal{S}|$ elements from the ground set, under the constraint that at most one element is selected from each $\{P(C_s, h) : h \in H_s\}$ ($s \in \mathcal{S}$). This is known as a special type of matroid called the *partition matroid*.

2) Set Objective Function: Meanwhile, we can also write our objective function (1) as a function of the selected elements: $\widetilde{f}(\{P(C_s, h_s) : s \in \mathcal{S}\}) \triangleq f(\bigcup_{s \in \mathcal{S}} P(C_s, h_s))$. Strictly speaking, to apply Theorem 11, we have to verify monotonicity and submodularity of $\widetilde{f}(\cdot)$. Nevertheless, we show that it suffices to verify these properties for $f(\cdot)$.

Lemma 12. *If two functions $\widetilde{f} : 2^{2^E} \to \mathbb{R}$ and $f : 2^E \to \mathbb{R}$ satisfy $\widetilde{f}(\mathcal{A}) = f(\bigcup_{A \in \mathcal{A}} A)$, then $\widetilde{f}$ is monotone submodular if and only if f is monotone submodular.*

Proof. First, suppose $\widetilde{f}$ is monotone submodular. For any $A \subset B \subseteq E$, $f(A) = \widetilde{f}(\{A\}) \leq \widetilde{f}(\{A, B \setminus A\}) = f(B)$, implying monotonicity of f. For any $A \subset B \subseteq E$ and $e \in E \setminus B$,

[2]The best approximation ratio is $(1 - 1/e)$, achieved by a more complicated algorithm based on continuous relaxation and rounding [17].

we have $f(A\cup\{e\})-f(A)=\widetilde{f}(\{A,\,\{e\}\})-\widetilde{f}(\{A\})$, and $f(B\cup\{e\})-f(B)=\widetilde{f}(\{A,\,B\setminus A,\,\{e\}\})-\widetilde{f}(\{A,\,B\setminus A\})$. By submodularity of $\widetilde{f}$, we have $\widetilde{f}(\{A,\,\{e\}\})-\widetilde{f}(\{A\})\geq \widetilde{f}(\{A,\,B\setminus A,\,\{e\}\})-\widetilde{f}(\{A,\,B\setminus A\})$, i.e., $f(A\cup\{e\})-f(A)\geq f(B\cup\{e\})-f(B)$, implying submodularity of f.

Next, suppose f is monotone submodular. For any $\mathcal{A}\subset\mathcal{B}\subseteq 2^E$, we have $\bigcup_{A\in\mathcal{A}}A\subseteq\bigcup_{A\in\mathcal{B}}A$. Since f is monotone, $\widetilde{f}(\mathcal{A})=f(\bigcup_{A\in\mathcal{A}}A)\leq f(\bigcup_{A\in\mathcal{B}}A)=\widetilde{f}(\mathcal{B})$, implying monotonicity of $\widetilde{f}$. To prove submodularity of $\widetilde{f}$, we use an equivalent definition that f is submodular if for every $A,\,B\subseteq E$, $f(A)+f(B)\geq f(A\cup B)+f(A\cap B)$ [16]. For any $\mathcal{A},\,\mathcal{B}\subseteq 2^E$, we have $(\bigcup_{A\in\mathcal{A}}A)\cup(\bigcup_{B\in\mathcal{B}}B)=\bigcup_{A\in\mathcal{A}\cup\mathcal{B}}A$, and $(\bigcup_{A\in\mathcal{A}}A)\cap(\bigcup_{B\in\mathcal{B}}B)=\bigcup_{A\in\mathcal{A},B\in\mathcal{B}}(A\cap B)\supseteq\bigcup_{A\in\mathcal{A}\cap\mathcal{B}}A$. Since f is monotone and submodular,

$$f(\bigcup_{A\in\mathcal{A}}A)+f(\bigcup_{B\in\mathcal{B}}B)\geq f(\bigcup_{A\in\mathcal{A}\cup\mathcal{B}}A)+f(\bigcup_{A\in\mathcal{A},B\in\mathcal{B}}(A\cap B))$$
$$\geq f(\bigcup_{A\in\mathcal{A}\cup\mathcal{B}}A)+f(\bigcup_{A\in\mathcal{A}\cap\mathcal{B}}A),\qquad(6)$$

where (6) is by monotonicity of f. This implies $\widetilde{f}(\mathcal{A})+\widetilde{f}(\mathcal{B})\geq\widetilde{f}(\mathcal{A}\cup\mathcal{B})+\widetilde{f}(\mathcal{A}\cap\mathcal{B})$, proving submodularity of $\widetilde{f}$. □

3) Greedy Heuristic: Due to Theorem 11, if our objective function is monotone and submodular, then the greedy heuristic is guaranteed to achieve good performance. Given an objective function $f(P)$, the algorithm for greedy service placement is presented in Algorithm 2. Specifically, let $\mathcal{S}_u$ denote the set of unplaced services, and P the set of measurement paths generated by placed services, initialized in Lines 1–2 respectively. Lines 3–7 iteratively place one service at a time, where in each iteration, line 4 evaluates the objective function for each unplaced service and its candidate host, and then selects the placement that maximizes the objective function. The iteration stops when all the $|\mathcal{S}|$ services are placed.

Algorithm 2: Greedy Service Placement

Input: A set of services $\mathcal{S}$, a family of candidate service locations $\{H_s:\,s\in\mathcal{S}\}$, a path set $P(C_s,\,h)$ for each $s\in\mathcal{S}$ and $h\in H_s$, and an objective function $f(P)$
Output: A service placement $\mathbf{h}=(h_s)_{s\in\mathcal{S}}$

1 $\mathcal{S}_u\leftarrow\mathcal{S}$
2 $P\leftarrow\emptyset$
3 **for** *iteration* $1,\ldots,|\mathcal{S}|$ **do**
4 $\quad(s^*,\,h^*)=\arg\max_{s\in\mathcal{S}_u,h\in H_s}f(P\cup P(C_s,\,h))$
5 $\quad h_{s^*}=h^*$
6 $\quad\mathcal{S}_u=\mathcal{S}_u\setminus\{s^*\}$
7 $\quad P=P\cup P(C_{s^*},\,h^*)$
8 return $\mathbf{h}$

Remark: Algorithm 2 is a generic greedy algorithm that works for any objective function $f(P)$. It remains to verify monotonicity and submodularity of the proposed objective functions to guarantee its performance.

B. Coverage Maximization

It is easy to verify that the coverage objective $|\mathcal{C}(P)|$ has the desired property.

Lemma 13. *Function $|\mathcal{C}(P)|$ is a monotone submodular function of P.*

Proof. The monotonicity is easy to see. To show submodularity, consider two path sets $P_1\subset P_2$, and a path $p\notin P_2$. Adding p increases the number of covered nodes by $|p\setminus\bigcup_{p'\in P_1}p'|$ for P_1, and $|p\setminus\bigcup_{p'\in P_2}p'|$ for P_2. Clearly, $|p\setminus\bigcup_{p'\in P_1}p'|\geq|p\setminus\bigcup_{p'\in P_2}p'|$, satisfying submodularity. □

Lemma 13 and Theorem 11 imply that when applied to coverage maximization, the greedy heuristic achieves guaranteed approximation as follows.

Corollary 14. *Algorithm 2 for $f(P)=|\mathcal{C}(P)|$ achieves $1/2$-approximation of the optimal solution to MCSP.*

That is, the number of nodes covered by client-host paths under the service placement computed by Algorithm 2 for $f(P)=|\mathcal{C}(P)|$ is at least half of the maximum number of nodes that can be covered.

C. Identifiability Maximization

For the identifiability objective $|S_k(P)|$, we have the following result.

Proposition 15. *Function $|S_k(P)|$ is monotone but not submodular in P.*

Proof. For monotonicity, we show that a node that is k-identifiable wrt P is also k-identifiable wrt $P\cup\{p\}$, where p is a measurement path not in P. This is because v being k-identifiable wrt P implies that any failure sets F_1 and F_2 with $|F_i|\leq k$ $(i=1,\,2)$ and $F_1\cap\{v\}\neq F_2\cap\{v\}$ must be distinguishable by at least one path $p'\in P$. Since $p'\in P\cup\{p\}$, F_1 and F_2 must also be distinguishable wrt $P\cup\{p\}$, and thus v is k-identifiable wrt $P\cup\{p\}$.

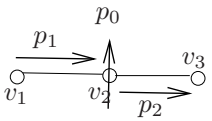


Fig. 3. Example: $|S_1(P)|$ is not submodular.

For submodularity, we show by a counterexample that even $|S_1(P)|$ is not submodular. For example, as illustrated in Fig. 3, there are three possible measurement paths[3] $p_0=\{v_2\}$, $p_1=\{v_1,\,v_2\}$, and $p_2=\{v_2,\,v_3\}$. The sets of 1-identifiable nodes when measuring various paths are $S_1(\{p_0\})=\{v_2\}$, $S_1(\{p_1\})=\emptyset$, $S_1(\{p_0,\,p_1\})=\{v_1,\,v_2\}$, $S_1(\{p_1,\,p_2\})=\{v_1,\,v_2,\,v_3\}$, and $S_1(\{p_0,\,p_1,\,p_2\})=\{v_1,\,v_2,\,v_3\}$. The increase in $|S_1(\cdot)|$ by selecting p_0 on top of an existing path set P equals $|S_1(\{p_0\})|-|S_1(\emptyset)|=1$ for $P=\emptyset$, $|S_1(\{p_0,\,p_1\})|-|S_1(\{p_1\})|=2$ for $P=\{p_1\}$, and $|S_1(\{p_0,\,p_1,\,p_2\})|-|S_1(\{p_1,\,p_2\})|=0$ for $P=\{p_1,\,p_2\}$, violating the requirement of submodularity. □

Since $|S_k(P)|$ can be approximated (upper-bounded) by $|\widetilde{S}_k(P)|$ according to Corollary 5, we also analyze properties of this bound.

[3]Note that a degenerate path containing a single node is possible when a service is co-located with a client.

Proposition 16. *Function $|\widetilde{S}_k(P)|$ is monotone in P but not submodular for $k > 1$.*

Proof. We prove monotonicity of $|\widetilde{S}_k(P)|$ by showing that MSC$(v;\ P)$ is monotone non-decreasing in P. By definition, MSC$(v;\ P)$ is the minimum number of nodes other than v to remove in order to disconnect all paths in P that traverse v. Denote all paths in P that traverse v by $g_v(P)$. After adding a path p to P, it is easy to see that $g_v(P) \subseteq g_v(P \cup \{p\})$. By definition, there exists a set of MSC$(v;\ P\cup\{p\})$ nodes (not including v) such that removing all of them disconnects all paths in $g_v(P \cup \{p\})$ and hence $g_v(P)$. Therefore, the minimum number of nodes to remove to disconnect all paths in $g_v(P)$, defined as MSC$(v;\ P)$, is no larger than MSC$(v;\ P \cup \{p\})$.

For a general $k > 1$, $|\widetilde{S}_k(P)|$ is not submodular. Consider the example in Fig. 3 and $k = 2$. We have that $|\widetilde{S}_2(\emptyset)| = |\widetilde{S}_2(\{p_1\})| = |\widetilde{S}_2(\{p_2\})| = 0$, and $|\widetilde{S}_2(\{p_1,\ p_2\})| = 1$. Thus, $|\widetilde{S}_2(\{p_1\})| - |\widetilde{S}_2(\emptyset)| = 0$, but $|\widetilde{S}_2(\{p_1, p_2\})| - |\widetilde{S}_2(\{p_2\})| = 1$, violating the requirement of submodularity. □

Due to the lack of submodularity, applying Algorithm 2 for $f(P) = |S_k(P)|$ (or $f(P) = |\widetilde{S}_k(P)|$) may not give guaranteed approximation for MISP.

Remark: Note that for $k = 1$, $|\widetilde{S}_1(P)|$ is reduced to the coverage objective $|\mathcal{C}(P)|$ which is known to be submodular.

D. Distinguishability Maximization

We show that the distinguishability objective $|D_k(P)|$ also has the desired property.

Lemma 17. *Function $|D_k(P)|$ is a monotone submodular function of P.*

Proof. Consider any path sets $P \subset P'$. For any pair of failure sets $(F_1,\ F_2) \in D_k(P)$, there must be a path $p \in P$ traversing only one of F_1 and F_2. Since $p \in P'$, F_1 and F_2 must be distinguishable wrt P', i.e., $(F_1,\ F_2) \in D_k(P')$. Thus, $D_k(P) \subseteq D_k(P')$.

Given a path $p \notin P'$, define the set of pairs of failure sets that are distinguished by p as $d(p) \triangleq \{(F_1,\ F_2) :$ p traverses only one of F_1 and $F_2\}$. By definition, we have $|D_k(P \cup \{p\})| - |D_k(P)| = |d(p) \setminus D_k(P)|$. Since $D_k(P) \subseteq D_k(P')$, we have $|d(p) \setminus D_k(P')| \leq |d(p) \setminus D_k(P)|$, and thus $|D_k(P'\cup\{p\})| - |D_k(P')| \leq |D_k(P\cup\{p\})| - |D_k(P)|$, proving the submodularity of $|D_k(P)|$. □

By Lemma 17 and Theorem 11, we can apply Algorithm 2 for $f(P) = |D_k(P)|$ to solve MDSP with guaranteed approximation.

Corollary 18. *Algorithm 2 for $f(P) = |D_k(P)|$ achieves $1/2$-approximation of the optimal solution to MDSP.*

1) Efficient Implementation: For general $k > 1$, even the greedy algorithm can be computationally expensive as evaluating $|D_k(P)|$ requires enumeration of all pairs of failure sets. For $k = 1$, however, we can utilize the equivalence graph $\mathcal{Q}$ introduced in Section III-B1 to simplify the computation by reusing computation in previous iterations. Specifically, since $\mathcal{Q}$ can be constructed incrementally as seen from Algorithm 1, we can maintain $\mathcal{Q}$ for paths selected in previous iterations, and compute the objective $f(P\cup P(C_s, h)) = |D_1(P\cup P(C_s, h))|$ (line 4 in Algorithm 2) by (hypothetically) updating $\mathcal{Q}$ for paths in $P(C_s,\ h)$ using steps 3-6 in Algorithm 1, and counting the number of missing links in the updated graph. The placement that removes the maximum number of links in $\mathcal{Q}$ will be selected.

2) Performance in Identifiability: For $k = 1$, we can show that the maximum-distinguishability placement gives guaranteed approximation to the maximum identifiability.

Theorem 19. *Let σ_0 and σ^* denote the numbers of non-1-identifiable nodes (i.e., $|N \setminus S_1(P)|$) under two placements that maximize $|D_1(P)|$ and $|S_1(P)|$, respectively. Then $\sigma_0 \leq \min((\sigma^*+1)\sigma^*,\ |N|)$ and $\sigma^* \geq (\sqrt{1+4\sigma_0} - 1)/2$.*

Proof. We use the fact that indistinguishable pairs correspond to links in the equivalence graph $\mathcal{Q}$, and non-1-identifiable nodes correspond to non-isolated nodes in $\mathcal{Q}$ (other than v_0).

Let n be the number of non-isolated nodes in $\mathcal{Q}$ and σ the number of non-1-identifiable nodes. Then $n - 1 \leq \sigma \leq \min(n,\ |N|)$. If $\mathcal{Q}$ has l links, then n is bounded by $\nu(l) \leq n \leq \min(2l,\ |N| + 1)$, where $\nu(l) \triangleq \min\{m :\ \binom{m}{2} \geq l\}$ is the minimum number of nodes in an l-link graph. Note that $2l \leq \nu(l)(\nu(l) - 1)$, and $\nu(l) \geq (1 + \sqrt{1 + 8l})/2$.

Let l_0 and l^* denote the numbers of links in the equivalence graph under maximum-$|D(P)|$ and maximum-$|S_1(P)|$ placements, respectively. By definition, $l_0 \leq l^*$. Then

$$\begin{aligned}\sigma_0 &\leq \min(2l_0,\ |N|) \leq \min(2l^*,\ |N|)\\ &\leq \min(\nu(l^*)(\nu(l^*) - 1),\ |N|) \leq \min(\sigma^*(\sigma^* + 1),\ |N|).\end{aligned} \tag{7}$$

Moreover, since $\nu(l)$ is monotone increasing in l,

$$\begin{aligned}\sigma^* &\geq \nu(l^*) - 1 \geq \nu(l_0) - 1\\ &\geq \frac{1 + \sqrt{1 + 8l_0}}{2} - 1 \geq \frac{\sqrt{1 + 4\sigma_0} - 1}{2}.\end{aligned} \tag{8}$$

□

That is, if the maximum-identifiability placement identifies the states for all but σ^* nodes, then the maximum-distinguishability placement identifies the states of all but at most $O(\sigma^{*2})$ nodes; if the maximum-distinguishability placement fails to identify the states of σ_0 nodes, then the maximum-identifiability placement fails to identify the states of at least $O(\sqrt{\sigma_0})$ nodes. In the high-identifiability regime (i.e., $\sigma^* \ll |N|$), the maximum-distinguishability placement approximates the maximum identifiability.

Remark: Theorem 19 can be generalized to any $k \geq 1$. Let us call a failure set F ($|F| \leq k$) *k-identifiable* wrt P if P_F is unique in $\mathcal{F}_k$. That is, if the set of failed nodes is F, then the failures can be uniquely localized as no other failure set will generate the same set of failed paths. By arguments similar to Theorem 19, we can show that the number of non-k-identifiable failure sets under the maximum-

distinguishability placement is at most a quadratic factor away from the minimum number of non-k-identifiable failure sets.

VI. Performance Evaluation

We evaluate the performance of our proposed heuristics: *greedy coverage maximization (GC)*, *greedy identifiability maximization (GI)*, and *greedy distinguishability maximization (GD)*, against two baseline solutions: 1) *best-QoS placement (QoS)*, where each service is placed at a node that minimizes the maximum distance to its clients, and 2) *random placement (RD)* under QoS constraints, where each service is placed at a node randomly selected from its candidate hosts H_s. When feasible, we also evaluate the optimal placement computed by *brute-force search (BF)*[4]. We consider single-node failures ($k = 1$), and evaluate each algorithm by the three performance measures proposed in Section II-B.

A. Simulation Setting

We consider Rocketfuel ISP topologies, where each node represents a *Point of Presence (POP)*, i.e., a set of co-located backbone and access routers as defined in [19]. We select three topologies: Abovenet, Tiscali, and AT&T, to represent small/medium/large networks. The characteristics of each network are presented in Table I.

TABLE I
Characteristics of the network

ISP	#nodes	#links	#dangling nodes
Abovenet	22	80	2
Tiscali	51	129	13
AT&T	108	141	78

In each network, we consider the dangling nodes (nodes with degree one) as candidate clients; in the case of Abovenet, we randomly choose 6 other nodes as candidate clients due to the small number of dangling nodes. We fix the number of clients per service at 3 and set the number of services to 3 for Abovenet, 4 for Tiscali, and 7 for AT&T. Clients for each service are selected in a round-robin fashion among candidate clients. All services have the same QoS threshold $\alpha_s \equiv \alpha$, which is varied in $[0, 1]$ to evaluate the tradeoff between QoS and monitoring performance. Fig 4 shows the (box plot of) number of candidate hosts as a function of α. As α increases, the QoS constraint is relaxed and the number of candidate hosts increases. If $\alpha = 1$, all the nodes become candidate hosts. Even if $\alpha = 0$, there may still be multiple candidate hosts, all minimizing the maximum distance to clients.

B. Simulation Results

For each network, we plot the number of covered nodes, the number of 1-identifiable nodes, and the number of distinguishable node pairs (see Section III-B1) as functions of α; see Fig. 5-7. We have the following observations:

Monitoring-QoS tradeoff: Since increasing α enlarges the set of candidate hosts (Fig. 4), all the algorithms that explore the entire candidate set (BF, GC, GI, GD, RD) achieve improved performance due to the increased diversity of measurement paths. The only exception is the best-QoS placement (QoS). As it (deterministically) minimizes server-client distance, it does not benefit from the enlarged candidate set.

Comparison with optimal: Comparing the heuristics with the optimal placement computed by BF (Fig. 5) shows that the greedy heuristic designed for a given performance measure performs close to the optimal wrt this measure (GC for coverage, GI for identifiability, GD for distinguishability). We only perform this comparison for the smallest network (Abovenet) due to the complexity of BF.

Comparison with baseline: The baseline solutions (QoS, RD) perform significantly worse than our best-performing heuristic in all the three measures, especially as α increases. In particular, the best-QoS placement (QoS) has the worst monitoring performance due to the lack of diversity in measurement paths. This result highlights the need to strategically exploit the flexibility in service placement.

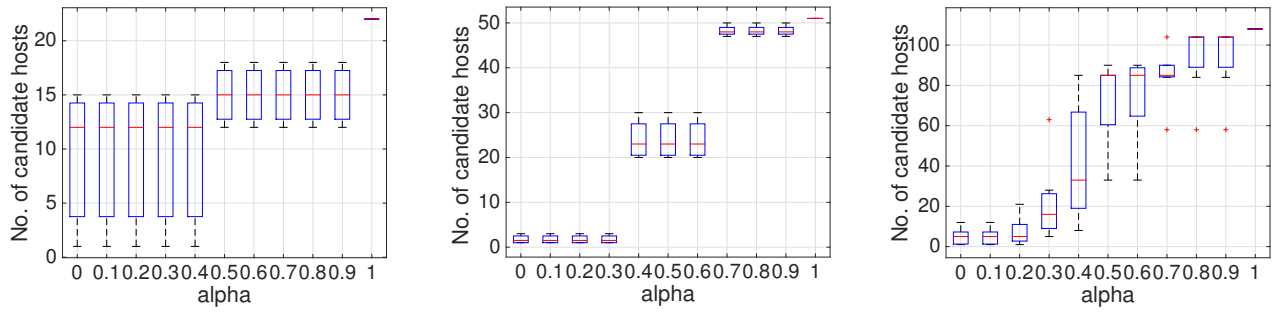
Comparison between proposed algorithms: Although each of the proposed heuristics (GC, GI, GD) targets at one performance measure, their overall performance shows the following trend: (1) the coverage-based heuristic (GC) performs well in coverage and distinguishability but not so well in identifiability; (2) the identifiability-based heuristic (GI) performs well in identifiability but poorly (even worse than RD) in coverage and distinguishability; (3) the distinguishability-based heuristic (GD) performs well in all the three measures.

To better understand the impact of service placement on the precision of failure localization, we evaluate the *degree of uncertainty* for each node, defined as the degree of this node in the equivalence graph $\mathcal{Q}$ (see Section III-B1). For a covered node, this is the number of other nodes covered by the same set of paths; for an uncovered node, this is the total number of uncovered nodes. Intuitively, this measures how precisely we can localize a failure once it is detected. For instance, if a node with degree of uncertainty n fails, we can narrow down the failure location to a set of $n + 1$ nodes, each traversed by the same set of paths that have all failed. A degree of uncertainty of zero means that the node is 1-identifiable. Fig. 8 shows the distribution of the degree of uncertainty over all the nodes in $\mathcal{Q}$ (including v_0) for AT&T and $\alpha = 0.6$ under different service placements. The y-axis represents the fraction of nodes with a given degree of uncertainty. We see that the distribution has two spikes, one around zero and the other around the number of uncovered nodes. Further investigation shows that the first spike corresponds to covered nodes and the second to uncovered nodes, implying that if the failed node is used by at least one service, we can either uniquely localize the failure or narrow down its location to a few nodes. The observations are similar for other settings.

VII. Extensions

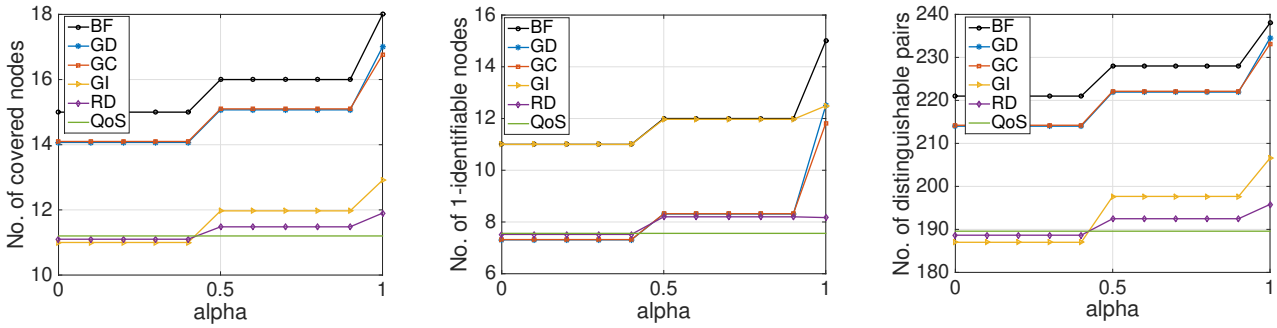
Our basic formulation (1) can be extended to incorporate further constraints and considerations in service placement.

[4]Note that the optimal placement is computed separately for each performance measure.



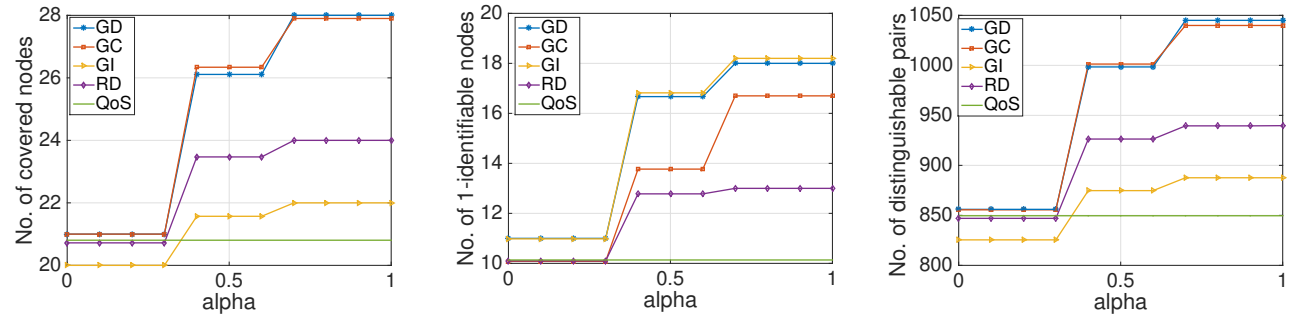
(a) (b) (c)

Fig. 4. Box plot for the number of candidate hosts for different topologies: (a) Abovenet with 3 services, (b) Tiscali with 4 services, and (c) AT&T with 7 services.



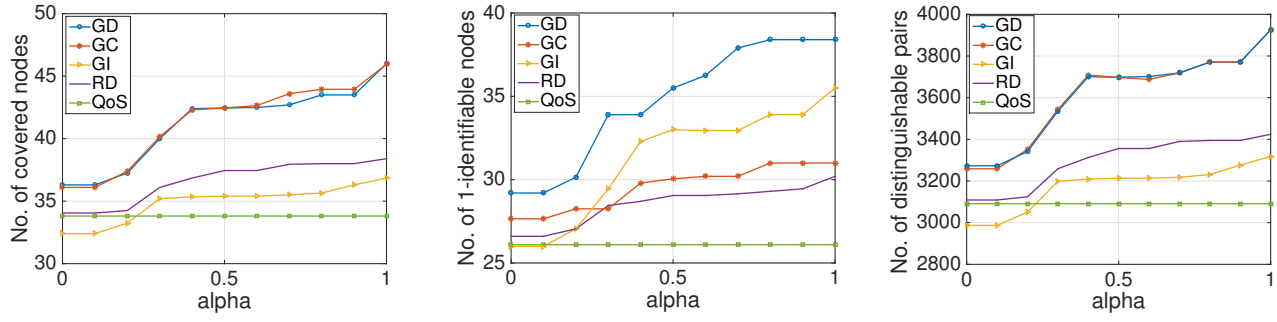
(a) (b) (c)

Fig. 5. Comparison between different heuristics and the optimal solution (brute-force) for Abovenet in terms of (a) coverage, (b) identifiability, (c) distinguishability.



(a) (b) (c)

Fig. 6. Comparison between different heuristics for Tiscali in terms of (a) coverage, (b) identifiability, (c) distinguishability.



(a) (b) (c)

Fig. 7. Comparison between different heuristics for AT&T in terms of (a) coverage, (b) identifiability, (c) distinguishability.

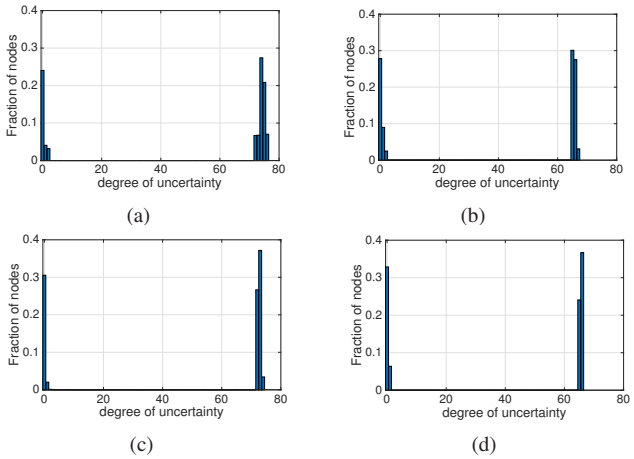


Fig. 8. Distribution of the degree of uncertainty for (a) QoS, (b) GC, (c) GI, (d) GD.

A. Handling Capacity Constraints

Although we have only considered QoS constraints (see Section III-A), our formulation can be easily extended to incorporate further constraints. Specifically, limitations on where a service can be placed due to software/hardware/security requirements can be readily incorporated by the candidate sets and all our previous results hold. Capacity constraints, however, require further consideration.

Consider node capacity constraints; link capacity constraints can be handled similarly. We can model these constraints by

$$\sum_{s\in\mathcal{S}} r_s \mathbf{1}_{h_s=h} \le R_h, \quad \forall h \in \bigcup_{s\in\mathcal{S}} H_s, \tag{9}$$

where r_s denotes the resource consumed by service s, R_h the total resource at host h, and $\mathbf{1}_{h_s=h}$ an indicator that service s is hosted by h. These constraints cause *correlation* between the placements of different services, which can no longer be modeled as a matroid as in Section V-A1. Nevertheless, the new constraints can be modeled by a generalization of matroid.

Definition 20 (p-independence system [17])**.** *A pair $(E,\ I)$ $(I \subseteq 2^E)$ is a p-independence system for an integer $p \ge 1$ if for all $A \in I$ and $e \in E$, $\exists B \subseteq A$ such that $|B| \le p$ and $(A \setminus B) \cup \{e\} \in I$.*

It is known that greedy heuristic achieves a constant-factor approximation for such constraints.

Theorem 21 ([18])**.** *For maximizing a monotone submodular function under a p-independence constraint, the greedy algorithm achieves a $1/(p+1)$-approximation to the optimal.*

Our constraints (2, 9) form a p-independence system with $p = \lceil r_{\max}/r_{\min} \rceil + 1$ ($r_{\max} \triangleq \max_s r_s$, $r_{\min} \triangleq \min_s r_s$), because to place a service s onto a host h, we need to remove at most $\lceil r_{\max}/r_{\min} \rceil$ services from h and s from its original host (if any). By Theorem 21 and Lemmas 13 and 17, we know that Algorithm 2, adapted for constraints (9), achieves a constant-factor approximation for maximizing coverage/distinguishability. We also observe that the approximation ratio decreases with $r_{\max}/r_{\min}$, with the best ratio of $1/3$ achieved for services with identical resource consumptions.

B. Handling Nodes of Interest

Our original objectives in Section II-B assume that we are interested in monitoring all the nodes. In practice, we may be interested in only a subset of nodes, e.g., nodes potentially used by any of the critical services.

Let $N_I \subseteq N$ denote the nodes of interest. We can generalize our objectives based on N_I. For coverage/identifiability, we can easily define the corresponding measure wrt N_I as the number of covered/k-identifiable nodes in N_I. For distinguishability, we define a failure set F as of interest if $F \cap N_I \neq \emptyset$. Let $\mathcal{F}_k$ denote all the failure sets with up to k failures and $\mathcal{F}_k^{(I)} \subseteq \mathcal{F}_k$ the failure sets of interest. The uncertainty in determining a randomly selected failure set $F \in \mathcal{F}_k^{(I)}$, measured by the average number of failure sets indistinguishable from F, is proportional to the number of indistinguishable pairs between failure sets in $\mathcal{F}_k^{(I)}$ and failure sets in $\mathcal{F}_k$. Its complement, $|\{(F,\ F') :\ F \in \mathcal{F}_k^{(I)},\ F' \in \mathcal{F}_k,\ P_F \neq P_{F'}\}|$, thus measures the distinguishability wrt N_I.

By similar arguments as in Lemmas 13 and 17, it can be shown that the generalized coverage/distinguishability objectives are still monotone submodular, and thus the greedy algorithm achieves $1/2$-approximation for these objectives.

VIII. Conclusion

We consider monitoring-aware service placement, which places services within QoS constraints such that in the face of failures, node states can be most accurately determined from the states of end-to-end connections between clients and servers. Measuring performance by the coverage, the identifiability, and the distinguishability in monitoring failures, we cast the problem as a set of combinatorial optimizations, each maximizing one performance measure. We show that although the optimal placement is NP-hard, the greedy heuristic achieves $1/2$-approximation for maximizing the coverage or distinguishability. Our evaluations based on real network topologies show that the proposed algorithms can significantly improve monitoring performance over an algorithm that only considers QoS, and the distinguishability-based placement achieves the best overall performance.

References

[1] "Network functions virtualisation," ETSI White Paper, 2013. [Online]. Available: https://portal.etsi.org/NFV/NFV_White_Paper.pdf

[2] R. R. Kompella, J. Yates, A. G. Greenberg, and A. C. Snoeren, "Detection and localization of network black holes," in *IEEE INFOCOM*, 2007.

[3] I. Cunha, R. Teixeira, N. Feamster, and C. Diot, "Measurement methods for fast and accurate blackhole identification with binary tomography," in *ACM IMC*, November 2009.

[4] N. Duffield, "Network tomography of binary network performance characteristics," *IEEE Transactions on Information Theory*, vol. 52, pp. 5373–5388, 2006.

[5] L. Ma, T. He, A. Swami, D. Towsley, K. K. Leung, and J. Lowe, "Node Failure Localization via Network Tomography," in *ACM IMC*, 2014.

[6] G. Wittenburg and J. Schiller, *Service Placement in Ad Hoc Networks*. Springer Briefs in Computer Science, 2012.

[7] N. Laoutaris, G. Smaragdakis, and D. Oikonomou, "Distributed Placement of Service Facilities in Large-Scale Networks," in *INFOCOM*, 2007.

[8] G. Wittenburg and J. Schiller, "A Survey of Current Directions in Service Placement in Mobile Ad-hoc Networks," in *PerCom*, 2008.

[9] Y. Bejerano and R. Rastogi, "Robust monitoring of link delays and faults in IP networks," in *IEEE INFOCOM*, 2003.
[10] L. Ma, T. He, A. Swami, D. Towsley, and K. Leung, "On optimal monitor placement for localizing node failures via network tomography," *Elsevier Performance Evaluation*, vol. 91, pp. 16–37, September 2015.
[11] J. D. Horton and A. López-Ortiz, "On the number of distributed measurement points for network tomography," in *ACM IMC*, 2003.
[12] N. Duffield, "Simple network performance tomography," in *ACM IMC*, 2003.
[13] H. Nguyen and P. Thiran, "The boolean solution to the congested IP link location problem: Theory and practice," in *IEEE INFOCOM*, 2007.
[14] L. Ma, T. He, A. Swami, D. Towsley, and K. Leung, "Network Capability in Localizing Node Failures via End-to-end Path Measurements," arXiv, 2015. [Online]. Available: http://arxiv.org/abs/1509.06333
[15] V. Chvatal, "A greedy heuristic for the set-covering problem," *Mathematics of Operations Research*, vol. 4, pp. 233–235, 1979.
[16] J. Lee, *A First Course in Combinatorial Optimization*. Cambridge University Press, 2004.
[17] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák, "Maximizing a Submodular Set Function Subject to a Matroid Constraint," in *IPCO*, 2007.
[18] M. Fisher, G. Nemhauser, and L. Wolsey, "An Analysis of Approximations for Maximizing Submodular Set Functions – II," *Math. Prog. Study*, vol. 8, pp. 73–87, 1978.
[19] N. Spring, R. Mahajan, and D. Wetheral, "Measuring isp topologies with rocketfuel," in *ACM SIGCOMM*, August 2002.