A Bayesian Approach to Network Monitoring for Progressive Failure Localization

Viviana Arrigoni, *Student Member, IEEE*, Novella Bartolini, *Senior Member, IEEE*, Annalisa Massini, and Federico Trombetti, *Student Member, IEEE*

***Abstract*—Boolean Network Tomography (BNT) aims at identifying failures of internal network components by means of end-to-end monitoring paths. However, when the number of failures is not known a priori, failure identification may require a huge number of monitoring paths. We address this problem by designing a Bayesian approach that progressively selects the next path to probe on the basis of its expected information utility, conditioned on prior observations. As the complexity of the computation of posterior probabilities of node failures is exponential in the number of failed paths, we propose a polynomial-time greedy strategy which approximates these values. To consider aging of information in dynamic failure scenarios where node states can change during a monitoring period, we propose a monitoring technique based on a sliding observation window of adaptive length. By means of numerical experiments conducted on real network topologies we demonstrate the practical applicability of our approach, and the superiority of our algorithms with respect to state of the art solutions based on classic BNT as well as sequential group testing.**

***Index Terms*—Fault location, computer network management, Bayesian inference, monitoring.**

I. Introduction

BOOLEAN Network Tomography (BNT) provides a series of powerful tools to localize network failures by using end-to-end monitoring paths. However, observations of the outcome of the monitoring paths (working/failed) induce a system of Boolean equations that is commonly under-determined, hence allowing multiple solutions [1]. Exact assessment of the state of each network component is not always achievable if monitors can only be deployed on a given subset of nodes and if routing is not controllable. Moreover, when the number of possible failures is unbounded, complete identification of failed components may require an enormous number of monitoring paths and related probes [2], [3], [4] which severely limits the applicability of the approach. Nonetheless, we notice that executing the probing activity in a progressive manner, hereafter referred to as *progressive BNT*, is particularly helpful in reducing the number of required probes to assess the network state. Thanks to the incrementally obtained information, we can calculate the expected utility of monitoring any additional path, conditioned on previous observations, in terms of additional failure localization. By applying a Bayesian approach, we design a stochastic optimization problem which maximizes the expected utility over a progressive monitoring activity. However, this optimization problem is intractable because of the large state space size and the computational cost of calculating posterior probabilities, which is exponential in the number of failed paths. In order to cope with the described issues, we propose two simplified approaches. First, to cope with the exponential size of the state representation, we propose a greedy approach, called *Posterior Probability Greedy* (*PoPGreedy*), that iteratively selects the path that more likely identifies the state of the largest number of network components. We provide an approximation factor of our greedy strategy by applying known results for the optimization of functions with bounded adaptive sub-modularity ratio [5]. Second, we approximate the posterior probability of a path failure with a polynomially computable approximation metric that we refer to as *failure centrality*. The failure centrality of a node reflects the probability for a node to be broken, conditioned on the currently available observations. Exploiting this metric, we propose a polynomial time greedy approach called *Failure Centrality Greedy (FaCeGreedy)*.

We experimentally compare FaCeGreedy and PoPGreedy in terms of novel metrics specifically designed to characterize the failure detection performance in both static and dynamic failure scenarios where nodes may fail and be repaired during the monitoring activity. The experiments show that FaCeGreedy provides an excellent approximation of the exact optimization approach in negligible time. By simulations, we evidence the superiority of FaceGreedy with respect to classic BNT approaches, namely *Greedy for Coverage*, *Identifiability*, and *Distiguishability* (CG, GI, and GD) [6], as well as to *AdaptiveFinder* [7], a state of the art solution based on sequential group testing, and to *Adaptive Path Construction* (APC) [8], a routing-constrained algorithm also based on sequential group testing for link failure detection.

In summary, our original contributions are the following:

- We formulate the problem of progressive network tomography in terms of stochastic optimization and Bayesian analysis. We discuss its complexity motivating the need for polynomial heuristic approaches.
- We formulate a novel failure centrality metric to approximate the failure probability of a node, given the observation of the outcome of a given set of probing paths.

Manuscript received 27 April 2021; revised 10 February 2022; accepted 16 August 2022; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor A. Gopalan. Date of publication 2 September 2022; date of current version 18 April 2023. This work was supported by the NGIAtlantic.eu under Grant 871582. *(Corresponding author: Viviana Arrigoni.)*

The authors are with the Department of Computer Science, Sapienza University of Rome, 00185 Rome, Italy (e-mail: arrigoni@di.uniroma1.it; bartolini@di.uniroma1.it; massini@di.uniroma1.it; trombetti@di.uniroma1.it).

Digital Object Identifier 10.1109/TNET.2022.3200249

- We propose two greedy approaches, called PoPGreedy and FaCeGreedy, based on Bayesian utility maximization and give an optimality approximation for PoPGreedy. For both approaches, we also provide an extended formulation to work in dynamic failure scenarios.
- We formulate four novel metrics to quantitatively measure the capability of a monitoring algorithm to properly localize network failures and to evaluate the localization uncertainty.
- By means of simulations conducted on real network topologies, we compare FaCeGreedy and PoPGreedy to classic Boolean Tomography approaches, as well as approaches based on sequential group testing, showing that our solutions outperform the others in all performance metrics and in all the considered scenarios, including static and dynamic failures.

II. Related Work

Boolean network tomography aims at measuring performance characteristics of individual network components by means of end-to-end paths. Some works [9], [10], [11], [12] studied how to measure additive performance metrics, for instance, delay, by optimizing monitor placement and path selection.

Other early works studied best effort techniques to classify individual network components in terms of a binary state, e.g., congested or not, failed or working, [1], [11], [12], [13], [14], [15].

The work in [16], and more recently in [6], [17], [18] addressed the problem of designing monitoring paths for optimal identifiability. Other works studied the fundamental limits to failure identifiability [2], [3], all working under the assumption of a bounded number of failed components.

The work by Cheraghchi *et al.* [19], formulates the identifiability problem for a graph-based group-testing framework, where the test sets are connected topology components. With a similar goal, the authors of [20] model the tomography problem as a Markov Decision Process, and solve it with a Q-learning technique. The actions of the decision process are related to the diagnosis of the congestion state of individual links. The work in [21] also utilizes machine learning techniques based on neural networks to infer a network topology from incrementally selected paths, with the purpose to predict the performance of paths that are not directly probed.

The practical applicability of Boolean Network Tomography is hindered by two important factors. First, most existing works assume prior knowledge of an upper bound on the number of congested or failed links. This assumption is made to limit the size explosion of the set of possible failure scenarios. The majority of the existing works are not designed to consider an unbounded number of failing components. Second, the proposed approaches are designed to ensure failure identifiability more than actual failure identification. By adopting the notion of identifiability formalized in [17], the general approach of Boolean Network Tomography is to provide paths that are able to uniquely identify the state of a maximum number of nodes and links, regardless of the particular instance of failure scenario being considered.

Our study also addresses the problem of identifying the state of network components by means of end-to-end monitoring. However it distinguishes itself from previous approaches in a twofold manner. First, we do not assume any bound on the number of failures, and we focus on actual node state identification rather than on identifiability. Second, we adopt a progressive approach to path selection with the purpose of incrementally identifying the state of individual network components.

With the same goal, the AdaptiveFinder algorithm [7] by Karbasi *et al.* considers progressive monitoring of graph-based test groups. We consider this proposal as a benchmark for performance comparisons with respect to our own approach. AdaptiveFinder considers a network graph and creates arbitrary sets of connected network components to determine the next paths to test according to a progressive approach. Unlike this work, we consider testing sets which are end-to-end monitoring paths, where pairs of monitors are connected by a series of nodes that strictly follow the routing protocol in use by the considered network. Similarly, Makumoto *et al.* proposed Adaptive Path Construction (APC) [8], a group-testing routing-constrained algorithm for detecting link failures by means of BNT techniques that aims at minimizing the number of path probes. Differently from our Bayesian approach, in APC, the choice of the next path to probe follows a binary search-based idea. As we will see in Section VII our approach solves the limitation imposed by the routing algorithm and provides superior performance to AdaptiveFinder and APC, thanks to the incremental knowledge constructed through our Bayesian decision support.

Unlike all the aforementioned works, very few studies take account of dynamic changes of the node states as we do in our paper. The problem of detecting failures occurring dynamically within a network attracted attention in recent years. A large portion of the available literature focuses on specific networks (e.g., data centres [22] and Wireless Sensor Networks (WSNs) [23], [24], [25], [26]). In [27], Huang *et. al.* highlight practical issues when tomography techniques are used to infer link degradation within a network. Their approach is divided into an initial offline phase (a set of paths covering the network is selected), followed by an online phase (where monitor nodes periodically probe measurements along defined paths in order to track possible changes in the performance of the links). In [28], Johnsson *et. al.* propose a two-step algorithm to interpret and analyze the outcome of path probes in order to detect and localize failures. Differently from these works, we consider path selection to be the key part of the online phase: we not only provide a way to interpret probe outcomes, but we also show how to obtain the most informative probes.

This paper extends the work presented in [29].

III. Problem Formulation

We consider a network modeled as a graph $G = (V, E)$, and a set of monitor nodes $V_M \subseteq V$ (shortly called *monitors*). For each ordered pair (i, j) of monitors in V_M we consider a unique monitoring path whose sequence of nodes is only determined by the routing algorithm in use. We consider

TABLE I
SUMMARY OF NOTATIONS

Symbol	Description
$\hat{m}$	set of nodes traversed by path $m \in \mathcal{M}$
S_i	event *node* v_i *works*
Z_j	event *path* m_j *works*
Z_j^o	observed (tested) state of path m_j
$\mathcal{A}$	$\mathcal{A} \triangleq \{a_1, a_2, \ldots, a_{\lvert\mathcal{M}\rvert}\}$: set of monitoring decisions
$\mathcal{T} \subseteq \mathcal{M}$	set of tested monitoring paths
$\mathcal{F} \subseteq \mathcal{T}$	set of failed tested monitoring paths
$O_{\mathcal{T}}$	set of observed outcomes of paths in $\mathcal{T}$

uncontrollable routing [17], i.e., monitoring paths are determined by the routing protocol used by the network, not controllable by the monitors. Routing between monitors i and j is not necessarily symmetric, but is assumed to be static, deterministic, and known.

We shortly denote with $\hat{m}$ the set of nodes traversed by the monitoring path m. We refer to $\mathcal{M}$ as to the set of monitoring paths available for probing the network. Without loss of generality we assume that links always work, and only nodes can fail. The approach can be easily extended to also consider link failures by modelling links as virtual nodes.

By probing the paths of $\mathcal{M}$ it is possible to obtain indirect information on the state of the traversed nodes. Both working and failed paths provide helpful information for the state assessment of the network components. In particular, if a path works, then all its nodes are properly functioning, whereas a non working path must contain at least a broken component. By probing paths in a sequence, it is possible to determine the most suitable choice for the next path to probe on the basis of the information gathered so far.

We address the problem of designing a *Progressive Monitoring Policy (PMP)*, i.e., a decision policy to select the next paths to probe on the basis of prior observations, such that we can identify the state of the largest number of nodes in the minimum number of steps (number of paths). We refer to this problem as the *PMP problem*.

In the following, we denote with S_v the event 'node v works', and $\bar{S}_v$ the event 'node v is broken'. If no information is available concerning the distribution of failures in the network, it is reasonable to assume that all nodes have equal *prior probability* p to be damaged, that is $P(\bar{S}_i) = p$ for all $v_i \in V$, while we denote with $P(S_i) = (1-p)$ the prior probability that v_i is working.

Classic approaches to Boolean Network Tomography adopt the concept of k-identifiability [2], [3], [17], which refers to the capability of inferring the state of individual nodes from the state of the monitoring paths. A node v is k-identifiable in $\mathcal{M}$ if any two sets of failing nodes F_1 and F_2 of size at most k, which differ at least in v (i.e., one contains v and the other does not), cause the failures of different subsets of paths in $\mathcal{M}$. The concept of k-identifiability assumes knowledge of an upper bound k to the number of occurring failures and characterizes nodes regardless of their state (failed or working) but only in terms of whether their state can be uniquely inferred by observing the outcome of the monitoring paths of $\mathcal{M}$. However, our setting is characterized by (1) absence of a bound on the number of simultaneous failures, (2) uncontrollable position of monitor nodes and (3) given routing algorithm. In such a setting, the PMP problem is particularly challenging as no node is guaranteed to be identifiable according to classic tomography.

A common practice in Network Tomography is to model the underlying topology with a graph that has one node for each network node and a virtual node for each network link (v_i, v_j). Therefore without loss of generality, we assume that links do not fail and model network links through logical nodes so that a link failure corresponds to the failure of a logical node [2]. Furthermore, we do not investigate the reason for nodes' malfunctioning and assume that the state of the nodes is binary. Faulty nodes can be heavily congested or broken nodes.

A. Bayesian Utility of Path Probing

Let Z_j be the event that path $m_j \in \mathcal{M}$ properly works, and $\bar{Z}_j$ the event that path m_j fails. Under the assumption of uniform probability of node failures, a *prior estimate* of the state probability of path $m_i \in \mathcal{M}$ is $P(Z_i) = (1-p)^{\lvert\hat{m}_i\rvert}$, and $P(\bar{Z}_i) = 1 - P(Z_i)$.

In our problem setting, the state of the network can only be observed by probing monitoring paths in a sequence of monitoring interventions. We denote by $\mathcal{A} \triangleq \{a_1, a_2, \ldots, a_{\lvert\mathcal{M}\rvert}\}$ the set of possible monitoring decisions, where decision a_i implies monitoring the network through path m_i. We denote by $\mathcal{T} \subseteq \mathcal{M}$ the already monitored paths, and by $\mathcal{F} \subseteq \mathcal{T}$ the set of failed tested paths. We denote with Z_i^o the outcome of the probing activity along path m_i, for any $m_i \in \mathcal{T}$. Knowledge of the outcome of the paths in $\mathcal{T}$ constitutes a source of information $O_{\mathcal{T}} \triangleq \{Z_j^o \mid m_j \in \mathcal{T}\}$ that can be used to produce an estimate, *a posteriori*, of the network state. We note that the outcome of any monitoring path is as informative as it contributes to the identification of the state of individual network components or decreases the size of the identification problem instance. More specifically, we observe that monitoring working or non-working paths contributes useful information for failure identification in different manners. In the following, we formulate the notion of *path probe utility*. In words, we consider the utility of path m_i equal to the number of new node states discovered by probing m_i.

- If $W^{(\mathcal{T})}$ is the set of working nodes discovered after paths $\mathcal{T}$ were tested, we observe that we can ideally prune every path $m \in \mathcal{M} \setminus \mathcal{T}$ of the already discovered working nodes. We call the pruned path $m^{(\mathcal{T})}$, i.e., $\hat{m}^{(\mathcal{T})} = \hat{m} \setminus W^{(\mathcal{T})}$ only accounts for the nodes traversed by m whose state is still unknown after paths $\mathcal{T}$ were probed.
- If by probing path m we observe that the path fails, we might be in two different scenarios: *i.* if $\lvert\hat{m}^{(\mathcal{T})}\rvert = 1$, meaning that all nodes of m except for one have been classified to be working nodes, we can claim with certainty that the only node left is failed. In this case, we are discovering one new node state. *ii.* If $\lvert\hat{m}^{(\mathcal{T})}\rvert > 1$ (i.e., $m^{(\mathcal{T})}$ traverses more than one node whose state is still undiscovered), we do not know how many and which nodes of $\hat{m}^{(\mathcal{T})}$ are failed.

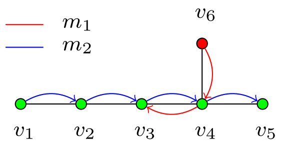


Fig. 1. Identification by exclusion.

- The utility of probing a working path m accounts for two additive terms. The first is $|\hat{m}^{(\mathcal{T})}|$. As for the second term, we observe that there might exist a path $m' \in \mathcal{F}$ (i.e., a failed tested path) such that $|\hat{m}'^{(\mathcal{T})} \setminus \hat{m}^{(\mathcal{T})}| = 1$. In this case, we are able to determine the state of the only unclassified node as failed *by exclusion*. Consider the example of Fig. 1. Assume that the monitoring activity starts by probing path m_1 first, which is found to be non-working; hence m_1 is inserted in $\mathcal{T}$. Then the monitoring activity proceeds by considering path m_2, which is properly working. Knowledge of the outcome of m_2 allows us to assess the state of the nodes v_i, with $i = 1, \dots, 5$, as working (green nodes). As a consequence, these nodes are all pruned and can be removed by all the non working paths included in $\mathcal{T}$. Due to the pruning of v_4 and v_3, the length of the already monitored path m_1 reduces to 1 in the logical representation of the network graph without pruned components, which implies $|\hat{m}_1^{(\mathcal{T})}| = 1$. Hence, $m_1^{(\mathcal{T})}$ turns to be a failing path of one only node, v_6 (in red in the figure), whose state must be failed by exclusion.
  Then the utility of probing a working path m (in the example m_2) also accounts for $|\mathcal{F}_1^{m,(\mathcal{T})}|$, where $\mathcal{F}_1^{m,(\mathcal{T})}$ denotes the set of failed paths having only one node left after pruning the nodes of m.
- Finally, we also observe that it is not useful to probe paths of which we can infer the outcome without actually testing them. We know that a path $m \notin \mathcal{T}$ is working if $\hat{m} \subseteq W^{(\mathcal{T})}$ (all its nodes have been pruned, and $\hat{m}^{(\mathcal{T})} = \emptyset$). In contrast, we know that a path m fails when it traverses at least a failed node, which happens when there is a path $m' \in \mathcal{F}$ such that $\hat{m}'^{(\mathcal{T})} \subseteq \hat{m}^{(\mathcal{T})}$.

In summary, if we make decision a_i corresponding to monitoring path m_i, the information utility is proportional to $|\hat{m}_i^{(\mathcal{T})}| + |\mathcal{F}_1^{m_i,(\mathcal{T})}|$ if the path m_i works, and to $\lfloor 1/|\hat{m}_i^{(\mathcal{T})}| \rfloor$ otherwise. We can then formulate the information utility function for each decision $a_i \in \mathcal{A}$ as follows:

$$\lambda(a_i|Z_i) = \begin{cases} |\hat{m}_i^{(\mathcal{T})}| + |\mathcal{F}_1^{m_i,(\mathcal{T})}| & \text{if } Z_i \\ \lfloor 1/|\hat{m}_i^{(\mathcal{T})}| \rfloor & \text{if } \bar{Z}_i \end{cases} \quad (1)$$

Notice that by construction we only consider paths for which $|\hat{m}_i^{(\mathcal{T})}| \neq 0$.

Correspondingly, we calculate the following *expectation of conditional utility given the observation*:

$$\mathcal{U}(a_i|O_\mathcal{T}) = \lambda(a_i|Z_i)P(Z_i|O_\mathcal{T}) + \lambda(a_i|\bar{Z}_i)P(\bar{Z}_i|O_\mathcal{T}) \quad (2)$$

As the available paths may give a different contribution to the identification task, some of them may become redundant, depending on the probing order.

IV. Stochastic Optimization of PMP

We consider a progressive decision process, in discrete time, which maximizes the number of nodes whose state is identified, i.e. the cumulative utility, according to equation 2, and which may end when one of the following conditions occurs: (1) There are no more useful paths to monitor (each of the remaining paths cannot add any information on the node states); (2)A bound on the number of probing steps has been reached.

At each step, the process makes one of the decisions in $\mathcal{A}$, whose utility depends on the outcome of the related monitoring path. The number of steps before termination is uncertain. An upper bound is given by the number of monitoring paths $|\mathcal{M}|$. We recall that we do not assume symmetric routing, i.e. the upper bound on the number of monitoring paths is given by the number of ordered pairs of monitoring nodes $|\mathcal{M}| = V_M \cdot (V_M - 1)$.

Considering the discussion made in Section III-A, we formulate the PMP problem in terms of *stochastic optimization*. At the n-th step, the state of the decision process is given by the set $O_\mathcal{T}$, which reflects the observations made until step n, provides the current knowledge of the state of network components after having probed a set of paths $\mathcal{T}$, $(n = |\mathcal{T}|)$ and determines the future action utility values, according to Equation 2.

In a static failure scenario, monitoring actions are not repeated in consecutive monitoring steps, hence we denote the actions available at step n, $\mathcal{A}(O_\mathcal{T})$, shortly as follows:

$$\mathcal{A}^{(n)} \triangleq \mathcal{A}(O_\mathcal{T}^{(n)}) = \{a_i : P(Z_i|O_\mathcal{T}) \neq 0, 1\}.$$

We look for a decision policy that maximizes the expected sum of the utilities incurred by its decisions. Let $V(O_\mathcal{T}^{(n)}, n)$ denote the expected information (utility) that will be obtained by the optimal decision policy starting from the observation $O_\mathcal{T}^{(n)}$ at step n (e.g. nodes still to be assessed). In particular, by the principle of optimality (Bellman equation) we derive:

$$\begin{aligned} V(O_\mathcal{T}^{(n)}, n) = \max_{a_i \in \mathcal{A}^{(n)}} \Big\{ & P(Z_i|O_\mathcal{T}^{(n)}) \Big(|\hat{m}_i^{(\mathcal{T})}| + |\mathcal{F}_1^{m_i,(\mathcal{T})}| \\ & + V(O_\mathcal{T}^{(n)} \cup \{Z_i\}, n+1) \Big) + P(\bar{Z}_i|O_\mathcal{T}^{(n)}) \\ & \times \Big(\Big\lfloor 1/|\hat{m}_i^{(\mathcal{T})}| \Big\rfloor + V(O_\mathcal{T}^{(n)} \cup \{\bar{Z}_i\}, n+1) \Big) \Big\}. \end{aligned}$$

While the equation suggests the use of a dynamic programming approach over a finite horizon to solve the PMP problem, we underline the following challenges: (1) The computational complexity in the calculation of the posterior probability $P(Z_i|O_\mathcal{T})$ is exponential in the number of failed paths. (2) The combinatorial size of the state space is also prohibitive for a large number of paths. These facts suggest looking for polynomial approaches to the design of efficient PMP policies and to metrics to quantitatively measure such efficiency.

A. The PoPGreedy Approach

We introduce PoPGreedy (Posterior Probability Greedy), a Bayesian strategy that progressively selects the next path to probe based on the current *utility maximization rule*, and updates the overall observation for the next step (Algorithm 1).

Given a graph G representing the network topology, a set of paths $\mathcal{M}$ and a prior probability of node failure p, PoPGreedy returns the posterior probability of failure of all nodes as is obtained after probing at most K paths in $\mathcal{M}$. At each step i, the Bayesian strategy selects the action $a^{(i)} = a^*$ that maximizes the expected utility defined in Equation 2 given the current observation (line 6). Ties are broken by randomly picking a path among the ones with the same maximum utility. An action corresponds to the decision to monitor a new path m^* and to consequently update the current estimate of path failure probabilities. The sets of tested paths, failed paths and working nodes are updated accordingly. If the current path m^* fails, all actions corresponding to probing paths that are supersets of $\hat{m}^*$ are removed from $\mathcal{A}^{(i+1)}$ (line 13), as they would certainly fail. In particular, if path m^* consists of only one node after being pruned off its working nodes, then that one node is certainly broken (line 14). If m^* works, the set of actions corresponding to probing paths that are subsets of $\hat{m}^*$ are removed, as also in this case it is not useful to probe such paths, since their outcome is already known without testing them (line 17). In addition, we check whether we can localize broken nodes by exclusion (line 18). If there are no more useful paths to test (line 7), we compute the failure probability of all nodes and rank them accordingly. We store the ordered sequence in an array $\mathbf{V}_f$ and return it (line 8).

Algorithm 1 PoPGreedy

```
1:  $W^{(0)}, B^{(0)} = \emptyset$ (set of working and broken nodes)
2:  $\mathcal{T}, \mathcal{F} = \emptyset$ (set of tested and failed paths)
3:  $\mathcal{A}^{(0)} = \{a_1, \dots, a_{|\mathcal{M}|}\}$
4:  $O_{\mathcal{T}}^{(0)} = \emptyset$
5:  for $i = 0, \dots, K-1$ do
6:      $a^* \triangleq \arg\max_{a \in \mathcal{A}^{(i)}} \mathcal{U}(a|O_{\mathcal{T}}^{(i)})$ ($a^*$: select path $m^*$)
7:      if $\mathcal{U}(a^*|O_{\mathcal{T}}^{(i)}) = 0$ then
8:          return list of nodes sorted by $P(\bar{S}_v|O_{\mathcal{T}}^{(i)})$, $\boldsymbol{V}_f$
9:      $\mathcal{T}^{(i+1)} \leftarrow \mathcal{T}^{(i)} \cup \{m^*\}$
10:     ▶ test $m^*$
11:     if $m^*$ fails then
12:         $\mathcal{F}^{(i+1)} \leftarrow \mathcal{F}^{(i)} \cup \{m^*\}$, $O_{\mathcal{T}}^{(i+1)} = O_{\mathcal{T}}^{(i)} \cup (\bar{Z}^*)$
13:         $\mathcal{A}^{(i+1)} \leftarrow \mathcal{A}^{(i)} \setminus (\{a^*\} \cup \{a_j : \hat{m}^{*(\mathcal{T})} \subseteq \hat{m}_j^{(\mathcal{T})}\})$
14:         if $|\hat{m}^{*(\mathcal{T})}| = 1$ then $B^{(i+1)} \leftarrow B^{(i)} \cup \hat{m}^{*(\mathcal{T})}$
15:     else
16:         $W^{(i+1)} \leftarrow W^{(i)} \cup \{\hat{m}^*\}$, $O_{\mathcal{T}}^{(i+1)} = O_{\mathcal{T}}^{(i)} \cup (Z^*)$
17:         $\mathcal{A}^{(i+1)} \leftarrow \mathcal{A}^{(i)} \setminus (\{a^*\} \cup \{a_j : \hat{m}_j^{(\mathcal{T})} \subseteq \hat{m}^{*(\mathcal{T})})$
18:         update $B^{(i+1)}$ with nodes failed by exclusion
19:     update ranking $\boldsymbol{V}_f$
```

In Appendix A, we prove the optimality approximation of PoPGreedy.

An example of execution of PoPGreedy: We show an example of execution on the network represented in Figure 2. We assume priori probability $p = 0.1$ and let node v_9 be the only failed node in the network. Nodes v_1, v_2, v_3 and v_4 are monitors, and consider undirected paths. We consider the 6 monitoring paths shown in the figure.

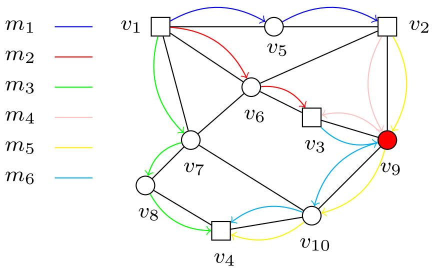


Fig. 2. Example topology with a failure in v_9.

- *Step 1:* The paths that maximize utility at the first iteration are m_3, m_5 and m_6. For the tie breaking rule we choose path m_3. The path works. Hence $\mathcal{A}^{(1)} = \mathcal{A}^{(0)} \setminus \{a_3\}$, $O^{\mathcal{T}} = \{Z_3\}$ and the set of working nodes $W = \{1,4,7,8\}$.
- *Step 2:* It results that $a_4 = \arg\max_{a \in \mathcal{A}^{(1)}} u(a|O_{\mathcal{T}})$, $u(a_4|O_{\mathcal{T}}) = 2.187$. We test m_4 and set $\mathcal{A}^{(2)} = \mathcal{A}^{(1)} \setminus \{a_4\}$. The path fails, therefore $\mathcal{F} = \{m_4\}$ and $O_{\mathcal{T}} = O_{\mathcal{T}} \cup \bar{Z}_4$.
- *Step 3:* It holds that $a_1 = \arg\max_{a \in \mathcal{A}^{(2)}} u(a|O_{\mathcal{T}})$, with $u(a_1|O_{\mathcal{T}}) = 1.1358$. We test m_1, $\mathcal{A}^{(3)} = \mathcal{A}^{(2)} \setminus \{a_1\}$. The path works, hence: $W = W \cup \hat{m}_1^{(\mathcal{T})}$ and $O_{\mathcal{T}} = O_{\mathcal{T}} \cup Z_1$. By knowing that m_4 failed while node v_2 works, we can claim with certainty that m_6 will also fail, as it steps onto all the remaining nodes of m_4. We set $\mathcal{A}^{(4)} = \mathcal{A}^{(3)} \setminus \{a_1, a_6\}$.
- *Step 4:* At this point, it results that $\mathcal{U}(a_2|O_{\mathcal{T}}) = \arg\max_{a \in \mathcal{A}^{(4)}} u(a|O_{\mathcal{T}}) = 1.278$. We test m_2, $\mathcal{A}^{(5)} = \mathcal{A}^{(4)} \setminus \{a_2\}$. The path works, hence: $W = \{1,2,3,4,5,6,7,8\}$ and $O_{\mathcal{T}} = O_{\mathcal{T}} \cup Z_2$. The utility of the non visited path m_5 is zero, therefore the execution is over. The algorithm returns the failure probabilities: $P(\bar{S}_9|O_{\mathcal{T}}) = 1$, $P(\bar{S}_{10}|O_{\mathcal{T}}) = 0.1$ and $P(\bar{S}_v|O_{\mathcal{T}}) = 0$ for all the other nodes.

We note that the algorithm leaves some uncertainty on the state of node v_{10}. However, this is due to the impossibility to obtain certain state identification for v_{10} with the available paths. In fact, all the paths traversing v_{10} fail because of the failure of node v_9.

B. Computational Complexity

Theorem 1: The computational complexity of PoPGreedy (Algorithm 1) is $O(K \cdot |\mathcal{M}| \cdot 2^{|\mathcal{F}|})$, where K is the maximum number of path probes.

Proof: At each of the $O(K)$ steps of the algorithm, expected utilities are updated (line 6). This operations requires computing $P(\bar{Z}|O_{\mathcal{T}})$ for all paths that are not tested yet:

$$P(\bar{Z}|O_{\mathcal{T}}) = P(\bar{Z} \wedge O_{\mathcal{T}})/P(O_{\mathcal{T}}). \quad (3)$$

Observe that, when computing the joint probability of the outcomes of previously tested paths, the contribution given by working paths simply results in pruning working nodes from non working paths. Therefore, the joint probabilities in equation (3) may be computed as follows:

$$P(O_{\mathcal{T}}) = P\left(\bigwedge_{m\in\mathcal{F}} \bar{Z}^{o}_{m^{(\mathcal{T})}}\right) \tag{4}$$

$$= 1 + \sum_{k=1}^{|\mathcal{F}|}\left((-1)^k \sum_{\substack{S\in 2^{\mathcal{F}}:\\ |S|=k}} (1-p)^{\left|\bigcup_{m\in S}\hat{m}^{(\mathcal{T})}\right|}\right). \tag{5}$$

The expression in the previous equation requires a number of addends that is exponential in the number of failed paths ($2^{|\mathcal{F}|}$). Computing node failure probabilities requires the same number of operations. In fact:

$$P(\bar{S}_i|O_{\mathcal{T}}) = P(\bar{S}_i \wedge O_{\mathcal{T}})/P(O_{\mathcal{T}}) = p\cdot P\Big(\bigwedge_{\substack{m_j\in\mathcal{F}\\ v_i\notin\hat{m}_j}} \bar{Z}_j\Big)/P(O_{\mathcal{T}}). \tag{6}$$

The final cost is $O(K\cdot|\mathcal{M}|\cdot 2^{|\mathcal{F}|})$. □

Even considering sporadic failures, it is hard to predict how much the exponential factor may grow. Even within the same topology, the time required for computing the joint probability $P(O_{\mathcal{T}})$ is highly dependent on where failures occur: if highly connected nodes fail, the number of failed paths may be large, which makes the computation of the failure probabilities extremely time consuming.

The above reasoning motivates the use of polynomially computable metrics to approximate the nodes' conditioned failure probabilities. In the next section, we define a polynomially computable centrality metric that captures the trend of how node failure probabilities are influenced when conditioned by iterative observations on test outcomes.

V. Failure Centrality

We hereby define the *failure centrality* of a node v given the observation $O_{\mathcal{T}}$, to approximate the value of $P(\bar{S}_v|O_{\mathcal{T}})$ (Equation 6) in the calculation of the posterior estimate of path state probabilities.

Definition 1: The failure centrality *of a node v given the observation $O_{\mathcal{T}}$ is $c(v|O_{\mathcal{T}}) = 0$ if v is traversed by some working paths in $\mathcal{T}$, it is equal to the prior probability of failure if v is not traversed by any path in $\mathcal{T}$, otherwise $c(v|O_{\mathcal{T}}) = \max\{T_1; T_2\}$, where*:

$$T_1 = \left\lceil \frac{1}{|\mathcal{F}_v|}\cdot\sum_{m\in\mathcal{F}_v}\left\lfloor\frac{1}{|\hat{m}^{(\mathcal{T})}|}\right\rfloor\right\rceil, \tag{7}$$

$$T_2 = \mathcal{P}_v + H(\lfloor\mathcal{P}_v\rfloor - 1)\cdot\left(1-\frac{\epsilon}{\mathcal{P}_v}-\mathcal{P}_v\right), \tag{8}$$

$$\mathcal{P}_v = |\mathcal{F}_v|/|\bigcup_{m\in\mathcal{F}_v}\hat{m}^{(\mathcal{T})}|, \tag{9}$$

where $\mathcal{F}_v\subseteq\mathcal{T}$ is the set of failed tested paths crossing node v. $H(x)$ is the Heaviside function ($H(x)=0$, if $x<0$, and $H(x)=1$ when $x\geq 0$). $\epsilon>0$ is a small constant.

This approximation derives from the observation that a node failure probability is directly proportional to the number of paths traversing the node and inversely proportional to the length of such paths (see Equation 6), as captured by the term $\mathcal{P}_v$ in Equation 9. The term T_1 in Equation 7 reflects the fact that when a node v is traversed by a failing path m, and all the other nodes in $\hat{m}$ are known to be working, it is possible to claim with certainty that v is broken, by exclusion. The possible values of $c(v|O_{\mathcal{T}})$ span in the interval $[0,1]$ and, in analogy with probabilities, $c(v|O_{\mathcal{T}})=0$ if the failure probability of node v is 0, whereas if $c(v|O_{\mathcal{T}})=1$ if the node is broken.

In the following, we give some observations and propositions to characterize the values of the node failure centrality given the observation.

Observation 1: Firstly, observe that $T_1\in\{0,1\}$. Indeed, for any failed path it holds that $|\hat{m}^{(\mathcal{T})}|\geq 1$, therefore the maximum value of the sum in equation (7) is $|\mathcal{F}_v|$, proving that T_1 can not be greater than 1. When there is at least one path m s.t. $|\hat{m}^{(\mathcal{T})}|=1$, $T_1=1$, otherwise $T_1=0$.

Proposition 1: For all nodes v and observations $O_{\mathcal{T}}$ it holds that $0\leq T_2<1$.

Proof: While it is trivially true that $T_2\geq 0$, we prove that T_2 cannot be greater than or equal to 1. We observe that if $\mathcal{P}_v<1$, then $T_2=\mathcal{P}_v$. When $\mathcal{P}_v=1$, T_2 becomes $1-\epsilon$, while if $\mathcal{P}_v>1$, $T_2=1-\frac{\epsilon}{\mathcal{P}_v}$, that is a monotonically growing function with a horizontal asymptote in 1. □

Proposition 2: Let $v\in V$ be a node and $O_{\mathcal{T}}$ the outcome of some path probes. If $c(v|O_{\mathcal{T}})=1 \implies v$ is broken.

Proof: In Proposition 1 we prove that $T_2<1$, hence $c(v)=1 \iff T_1=1$. When there is at least a failed path m traversing v such that $|\hat{m}^{(\mathcal{T})}|=1$, the numerator num of T_1 is $0<num\leq|\mathcal{F}_v|$ and therefore $T_1=c(v|O_{\mathcal{T}})=1$. When this situation occurs, the probability of failure of node v is indeed 1, as this means that the failure of path m is only due to the failure of node v. □

Proposition 3: Let $v\in V$ be a k-identifiable node with respect to the set of paths $\mathcal{T}$, where k is the number of failures in the network, and let $O_{\mathcal{T}}$ be the outcomes of path probes on $\mathcal{T}$. If v is broken $\implies c(v|O_{\mathcal{T}})=1$.

Proof: Because v is k-identifiable, the set of paths crossing v is different from the set of paths crossing any other set of nodes of size at most k. In particular, it differs from the set of paths crossing the other $k-1$ broken nodes. Hence there must be at least one path m that passes through v and not through any other failed node, and therefore $\hat{m}^{(\mathcal{T})}=\{v\}$. We now need to prove that there is some set of observations $O_{\mathcal{T}}$ that allows for disambiguating v by finding out that indeed $\hat{m}^{(\mathcal{T})}=\{v\}$. If $|\hat{m}^{(\mathcal{T})}|=1$, then this is trivially true. By definition of k-identifiability, the failure of node v must produce different sets of failed paths than the ones resulting from simultaneous failures of v and any other node in $\hat{m}\backslash\{v\}$. As a consequence, there must be some working path passing through the nodes in $\hat{m}\backslash\{v\}$ and not through v, making it possible to verify through end-to-end monitoring measurements that $\hat{m}^{(\mathcal{T})}=\{v\}$, which results in $T_1=c(v|O_{\mathcal{T}})=1$. □

To conclude the discussion on the formulation of the centrality, we comment on the choice of term T_2 in equation (8). This formulation is motivated by the observation that node failure probabilities are directly proportional to the number of failed paths traversing a node and inversely proportional to the number of nodes $w\notin W$ being traversed by such paths. This property is satisfied by both $\mathcal{P}_v$ and $1-\frac{\epsilon}{\mathcal{P}_v}$.

Furthermore, by experimental observations, we noticed that $P(\bar{S}_v|O_\mathcal{T})$ grows steeply with the number of terms $\bar{Z}_i$ (where $v \in \hat{m}_i^{(\mathcal{T})}$) in $O_\mathcal{T}$ when $P(\bar{S}_i|O_\mathcal{T}) \ll 1$, while it slowly converges to 1 for $P(\bar{S}_v|O_\mathcal{T}) \lesssim 1$ for increasing numbers of negative tests on paths passing through v. Similarly, $T_2 = \mathcal{P}_v$ when $\mathcal{P}_v < 1$, while $T_2 = 1 - \frac{\epsilon}{\mathcal{P}_v}$ for $\mathcal{P}_v \geq 1$.

In order to tune the value of ϵ we observe that if $q^* = \max \mathcal{P}_v \ s.t. \mathcal{P}_v < 1$, then $q^* \leq \frac{d-1}{d}$, where $d = |\bigcup_{m \in \mathcal{M}_v} \hat{m}^{(\mathcal{T})}|$ and $\mathcal{M}_v := \{m \in \mathcal{M} : v \in \hat{m}\}$. Therefore, for $\epsilon < 1 - q^*$, the growing trend of T_2 would be still satisfied when $\mathcal{P}_v$ exceeds 1.

A. Centrality-Based Utility

Because of the dependencies among path failures, computing the joint probability $P(O_\mathcal{T})$ requires the computation of $2^{|\mathcal{F}|}$ addends. In order to reduce computational costs, we approximate the probability that a path works, conditioned to the set of observations $O_\mathcal{T}$, as follows:

$$\tilde{P}_c(Z_i|O_\mathcal{T}) = \prod_{v \in \hat{m}_i^{(\mathcal{T})}} (1 - c(v|O_\mathcal{T})). \tag{10}$$

Definition 2: The expected conditional utility based on failure node centrality is given by the formula:

$$\mathcal{U}_c(a_i|O_\mathcal{T}) = \lambda(a_i|Z_i)\tilde{P}_c(Z_i|O_\mathcal{T}) + \lambda(a_i|\bar{Z}_i)\tilde{P}_c(\bar{Z}_i|O_\mathcal{T}) \tag{11}$$

if $\nexists\ m' \in \mathcal{F} : \hat{m}'^{(\mathcal{T})} \subseteq \hat{m}_i^{(\mathcal{T})}$. *Otherwise* $\mathcal{U}_c(a_i|O_\mathcal{T}) = 0$. *Here,* $\lambda(a_i|Z_i)$ *is defined as in equation (1) and* $\tilde{P}_c(\bar{Z}|O_\mathcal{T}) = 1 - \tilde{P}_c(Z|O_\mathcal{T})$.

The condition that equation (11) is valid if $\nexists\ m' \in \mathcal{F} : \hat{m}'^{(\mathcal{T})} \subseteq \hat{m}_i^{(\mathcal{T})}$ serves to recognize that super-paths of failed paths are going to be failing, too. Thanks to prior observations, we can assess the state of such paths and therefore, there is no use in probing them.

B. Probing Algorithm With Centrality: FaCeGreedy

Algorithm 1 may be adapted to use the centrality metric instead of the exact conditional probability by applying the following modifications:

- Input: change p for c as initial node centrality.
- Lines 6 and 7: substitute $\mathcal{U}(a|O_\mathcal{T})$ with $\mathcal{U}_c(a|O_\mathcal{T})$.
- Line 8: replace $P(\bar{S}_v|O_\mathcal{T})$ with $c(v|O_\mathcal{T})$.

We hereby call FaCeGreedy (Failure Centrality Greedy algorithm) the Algorithm 1 with the modifications described above.

1) Computational Complexity: The computational complexity of Algorithm 1 changes when centrality and centrality-based utility (Definitions 1 and 2) are implemented instead of probability and utility (equation 2).

Theorem 2: The computational complexity of FaCeGreedy (Algorithm 1 with the changes described above) is $O(K \cdot (|V| \cdot |\mathcal{F}|^2 + |\hat{m}_{max}|))$*, where* K *is the maximum number of path probes,* V *is the set of nodes of the network,* $\mathcal{F}$ *is the set of failing tested paths and* $|\hat{m}_{max}|$ *is the maximum path length.*

Proof: The total number of tests is $O(K)$. Computing the centrality of a node v requires scrolling the failed paths and searching for possible sub-paths in order to compute $\mathcal{P}_v$ (equation (9)). This is comprehensive of computing $|\mathcal{M}_v \cap \mathcal{F}|$ (equation 7) and takes $O(|\mathcal{F}|^2)$ operations. This is done for all nodes v at each iteration. Computing the centrality-based utility of a path requires a number of operations that is linear in the number of nodes paths traverse. The overall cost of the algorithm is $O(K \cdot (|V| \cdot |\mathcal{F}|^2 + |\hat{m}_{max}|))$, where $|\hat{m}_{max}|$ is the maximum path length. □

VI. Dynamic Failures

Progressive monitoring requires a series of sequential probing decisions. During its execution, it is reasonable to assume that the state of some network components may change due to new failures, congestion, or to recovery interventions. This is especially true in large scale networks where congestions can be caused by several phenomena: routers are susceptible to large traffic loads [30] causing delays and packet drops. Some Denial of Service attacks like SYN flood, ICMP flood and DSN flood may also cause servers' unavailability. In this section, we show how the proposed algorithms can be implemented in a persistent, always-on monitoring system taking into account dynamic scenarios where nodes' states may change throughout the monitoring activity (working nodes may fail because of network congestion and high bandwidth consumption, and faulty nodes may be recovered). We do not assume any knowledge of the time required by a node to be fixed, nor of the nodes' lifetime. As a consequence, the information obtained by monitoring given paths may soon become obsolete. To take account of the network dynamics, we propose our algorithms in two dynamic variants *Dynamic PoPGreedy (DPoPGreedy)* and *Dynamic FaCeGreedy (DFaCeGreedy)*. To address the dynamic scenario, we discretize time into the intervals between path probes, and for each node v we define a probability to transition from working to failed ($p_{W \to F}$) and from failed to working ($p_{F \to W}$), at each time step. We assume that it is more likely for a broken node to be fixed rather than for a working node to fail (i.e., $p_{W \to F} < p_{F \to W}$). We ground our procedure on the observation that information gained in the past progressively expires with time. Because of the computational complexity that would result in a Bayesian analysis where probabilities are explicitly time-dependent [28], we consider the following simplified and easily computable approach: we define a *sliding observation window* considering only a set of the recently probed paths to account for information obsolescence. We assume that the width of the window ℓ_w is large enough to make it possible to cover the entire network with path probes. The window slides progressively: at each time step, the least recently probed path exits the window, and a new path enters and is probed. DPoPGreedy and DFaCeGreedy work similarly to their static counterparts by using the only paths belonging to the sliding observation window, until a *contradiction* is detected. A contradiction inside a window occurs when the joint probability of the last ℓ_w path probe outcomes is 0. This could happen either because a path traversing supposedly working nodes fails, or because a super-path of supposedly failed path works. When this happens, it means that there has been at least

TABLE II

METRICS FOR EVALUATING STATIC HEURISTICS. $\boldsymbol{V}_f$ IS THE VECTOR RETURNED BY ALGORITHM 1, SORTED BY NODE FAILURE PROBABILITIES; $k = |F|$ IS THE NUMBER OF ACTUALLY FAILED NODES; $\boldsymbol{V}_f[1:k]$ ARE THE FIRST k ELEMENT OF $\boldsymbol{V}_f$; i IS THE INDEX OF A FAILED NODE APPEARING IN $\boldsymbol{V}_f$

Metrics	**Value**	**Meaning**
accuracy over working nodes, a_w	$a_w \triangleq \lvert W_h\rvert / \lvert W_{\mathcal{M}}\rvert \in [0,1]$	Fraction of nodes classified as working by a heuristics h (W_h) over nodes classified as working by probing all available paths $\mathcal{M}$ ($W_{\mathcal{M}}$)
accuracy over broken nodes, a_B	$a_B \triangleq \lvert B_h\rvert / \lvert B_{\mathcal{M}}\rvert \in [0,1]$	Fraction of nodes classified as failed by a heuristics h (B_h) over nodes classified as failed by probing all available paths $\mathcal{M}$ ($B_{\mathcal{M}}$)
first ranking, R_1	$R_1 \triangleq \frac{\lvert \boldsymbol{V}_f[1:k] \cap F\rvert}{k} \in [0,1]$	Counts how many broken nodes appear in the highest k positions in the ranking $\boldsymbol{V}_f$
second ranking, R_2	$R_2 \triangleq \frac{k}{n-i+1} \in [0,1]$	Counts how many nodes must be inspected before all k broken nodes are found

a change in at least one node state. To handle this event, we locate the most recent path that causes the contradiction. Then we remove it from the window, together with all the older observations, as the information they provide is considered obsolete. If the methods reach convergence the algorithms continue probing paths with zero utility, with tie breaking rules aimed at ensuring network coverage and prompt anomaly detection.

VII. EXPERIMENTAL RESULTS

In the following, we provide a performance evaluation of both the variants PoPGreedy and FaCeGreedy of our approach against state of the art solutions for classic BNT and sequential graph-based group testing. In the experiments, we assume cycle-free routing between monitor nodes. Our evaluation considers the metrics described in Section VII-A. If not explicitly stated otherwise, initial failure probability and centrality are set to 0.1.

A. Metrics

We consider the output of any of the probing algorithms in terms of the probability associated with each node failure. We compare the performance of the heuristics with respect to the results that would be obtained by using all the monitoring paths. For the static scenario, we use the metrics that we introduced in [29], namely *accuracy over working nodes*, a_W, *accuracy over broken nodes*, a_B, and the two ranking metrics, R_1 and R_2. They are described in Table II. In addition to these metrics, we also consider the number of probes required to reach convergence and the execution time.

For evaluating DPoPGreedy and DFaCeGreedy, we use metrics that capture the ability to detect node state changes and metrics that measure the reliability of the classification results step by step. For the first category, we compute the percentage of detected node state changes in both ways ($W \rightarrow F$ and $F \rightarrow W$) and the time for detection in terms of time steps. For assessing the classification reliability at each time, we use precision and recall:

$$P = \frac{tp}{tp + fp}, \qquad R = \frac{tp}{tp + fn}$$

where tp (true positive) is the number of correctly classified nodes; fp (false positive) is the number of nodes erroneously classified either as working or as failed; fn (false negative) is the sum of the number of real working nodes that are not classified as working, and of the number of real failed nodes that are not classified as failed. Note that we do not produce a binary classification, as at each step, the state of some nodes may be uncertain. For this reason, the binary classification that is usually evaluated by recall and precision is calibrated on 'correctly' and 'incorrectly/ambiguously classified' nodes in our case. Furthermore, observe that a false positive may only exist in the dynamic scenario (and not in the static one) and occurs in the interval between a state change of an already classified node and the detection of such state change (i.e., when a contradiction is verified). Note that the recall is similar to R_1, except that R_1 evaluates probabilistic outcomes instead of binary classifications.

B. Benchmark Solutions

To validate our approach, we compare it with existing solutions based on classical BNT as well as approaches based on progressive graph-constrained group testing. For the first set of benchmarks, we consider the *greedy for coverage*, *greedy for identifiability* and *greedy for distinguishability* (GC, GI, GD) heuristics defined in [6]. At each iteration, the next path to probe among the available input paths is chosen as the one that maximizes network coverage/identifiability/distinguishability, respectively. When the greedy procedures meet some stopping criteria, node failure probabilities $P(\bar{S}_v|O_{\mathcal{T}})$ are computed, and the outcome is evaluated in terms of the metrics described in Section VII-A.

In addition, we compare our methods to the adaptive, graph-constrained group testing algorithm introduced in [7], called AdaptiveFinder, (AF). The goal of AdaptiveFinder is to detect the set of defective items (nodes) in a graph with the least number of probes. We highlight that unlike our algorithms, AF is simply graph-constrained, as opposed to routing-constrained. This implies that, with a single path, AF can monitor portions of the network topology that would otherwise require multiple paths from our routing-constrained solutions. For instance, AF can use a unique path to monitor a tree-shaped sub-graph. AF can also monitor individual nodes with direct inspection. These activities would require more path-probing decisions from our algorithms. As a consequence, AF is apparently more flexible and powerful in classifying the state of the network components, but it achieves these capabilities at the expense of requirements that are not realistic in traditional communication network settings. By working on a wider solution space, AF is not susceptible to lack of identifiability, which means that, when run until convergence, AF always manages to assess the state of all the nodes without any uncertainty.

For this reason, when evaluating the accuracy of AF, we consider the ground truth as a term of comparison, i.e., $a_W = W_{AF}/W$ and $a_B = B_{AF}/B$, where W and B are the set of the truly working and failed nodes, respectively. The solution obtained by AF is proved to be a $(2+\epsilon)$-approximation of the optimal solution, [7]. Nevertheless, it is not possible to make perfect parallelism between the optimality approximation of PoPGreedy and AF, as they move in different action and solution spaces. We also stress that AF assumes an unrealistic and expensive capability of node detection because it requires all nodes of the network to be provided with a monitoring system software and fully controllable routing. Finally, we also compare our results with the Adaptive Path Construction (APC) algorithm, [8]. Similarly to PoPGreedy and FaCeGreedy, APC assesses the state of the network by means of end-to-end monitoring paths whose routing is given and uncontrollable. APC acts in two phases. In the first, it implements a greedy for coverage algorithm. In the second, it follows an adaptive approach: at each step it selects the path whose number of still unclassified nodes (i.e., nodes whose state -working/failed - is unknown, called *candidate nodes*) is closer to a half of the overall number of candidate nodes. APC outputs the set of the failed nodes and of the candidate nodes. In PoPGreedy and FaceCeGreedy, these sets correspond to the set of nodes whose failing probability/centrality is 1 and to the set of nodes whose failing probability/centrality lies in $(0, 1)$, respectively. In order to compare APC in terms of the metrics introduced in Section VII-A, we evaluate the failure probability of the nodes in the candidate set considering the outcome of the paths selected by APC, and we set the failure probability to 1 for the nodes in the identified set, and to 0 for all remaining nodes.

C. Tests

We conduct experiments on two different networks: an internet network in Europe, BICS [31], and a fiber network topology in Minnesota [32]. Table III detail features of the two topologies (left) and networks' features (right), taking into account monitor-to-monitor path choices. We use the smaller network, BICS, for running a thorough study of the behaviour of our algorithms and benchmarks, before extending our conclusions to the case of the larger network, Minnesota (MN). Experiments on real life dense networks will be considered in future work. In the experiments, the set of candidate monitors is chosen randomly, with several paths between the same monitor pairs, to ensure broad network coverage.

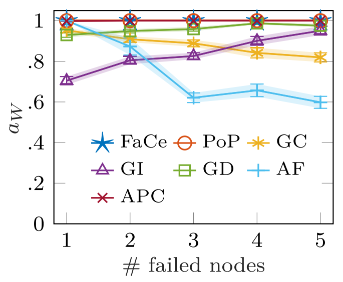
1) Experiments on BICS Network: Figures 3-4 are related to the BICS network. All curves are averaged on 20 experiments and show the value of the metrics defined in Section VII-A on PoPGreedy, FaCeGreedy and all benchmarks. Shades/bars depict standard deviations. In the experimental configurations shown in Figure 3, all the algorithms stop when they either reach convergence or a maximum number of iterations K, whichever the earliest. In this experiment, K is set to the number of path probes needed by PoPGreedy to converge. Figure 3 shows how the classification metrics discussed in Section VII-A, as well as the processing times and the average number of tested paths, change for a growing number of failed nodes (from 1 to 5 failures), distributed with uniform probability. Notice that FaCeGreedy and GC always reach convergence before PoPGreedy (Fig. 3f), but while GC has

TABLE III

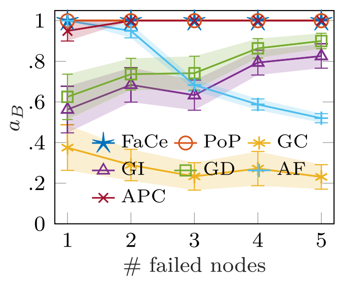
On the Left: Experimental Settings. δ = Node Degree, $n_{\delta=1}$ = Number of Dangling Nodes (Degree 1). On the Right: Path Characteristics. V^C is the Set of Covered Nodes; E^C is the Set of Covered Links; $\delta_i^{\mathcal{M}}$ = Number of Paths in $\mathcal{M}$ Traversing Node v_i

	BICS	MN
$\lvert V\rvert$	33	681
$\lvert E\rvert$	48	921
δ_{min}	1	1
δ_{max}	8	13
δ_{avg}	3	2.7
diameter	9	29
$n_{\delta=1}$	5	134

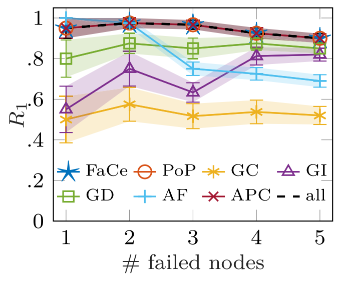
	BICS	MN
$\lvert V^C\rvert$	33	450
$\lvert E^C\rvert$	43	610
$\delta^{\mathcal{M}}_{min}$	1	1
$\delta^{\mathcal{M}}_{max}$	29	631
$\delta^{\mathcal{M}}_{avg}$	9.9	60.7
longest path	9	27
$\lvert V_M\rvert$	10	62
$\lvert\mathcal{M}\rvert$	55	1996



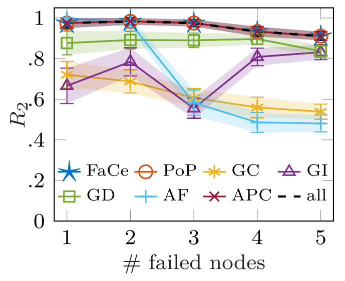
(a) Accuracy over working nodes, a_W



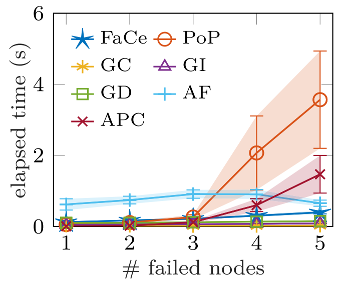
(b) Accuracy over broken nodes, a_B



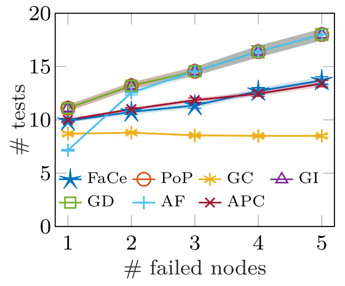
(c) R_1



(d) R_2

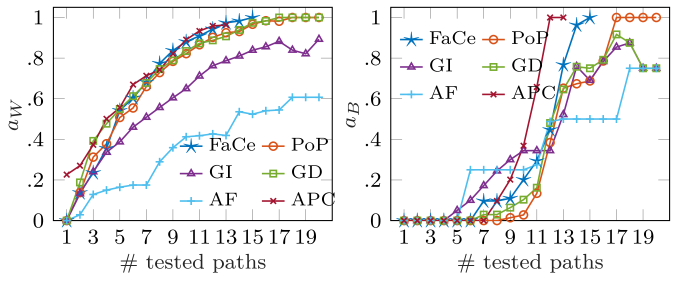


(e) Elapsed time

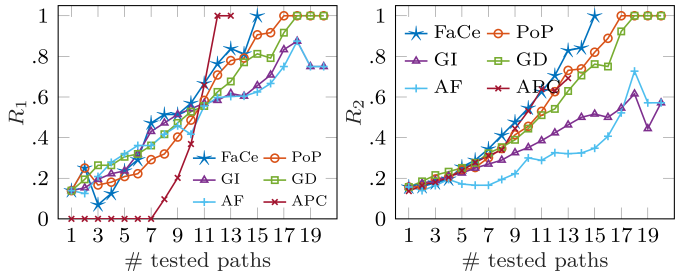


(f) Number of tests

Fig. 3. Tests on BICS network. Bounded.

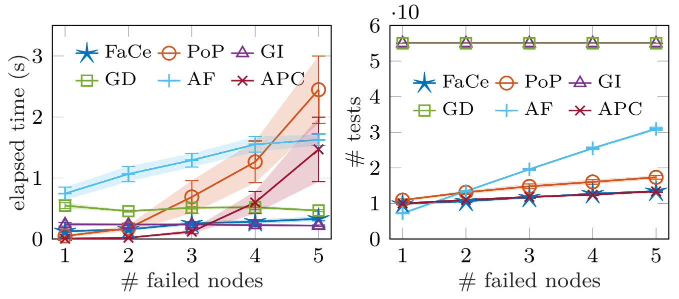


(a) Accuracy over working nodes, a_W (b) Accuracy over broken nodes, a_B



(c) R_1 (d) R_2

Fig. 4. Metrics evolution iteration-wise (BICS network with 4 failed nodes). Bounded.



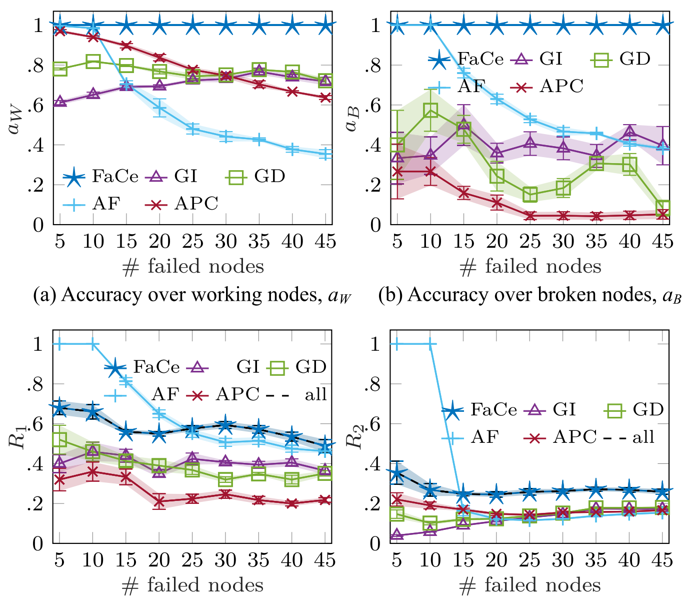
(a) Elapsed time (b) Number of tests

Fig. 5. Tests on BICS network. Unbounded.

poor, non-improvable performance in terms of node classification, FaCeGreedy, together with PoPGreedy, achieve the same performance that would be obtained by probing all the available monitoring paths, $\mathcal{M}$. For readability, we omit GC from the plots of the following experiments since they do not show any qualitative difference in the behavior of GC with respect to those of Figure 3. GI and GD probe the same number of paths, K, as does PoPGreedy (set as an upper bound for this experiment) because they are unable to reach their respective convergence criterion with fewer paths. In fact, to converge, both GI and GD would require an enormous amount of paths, as they aim at ensuring complete failure identifiability and distinguishability for any failure scenario, regardless of the progressively available information. AF manages to converge with a very small number of paths only when a single failure occurs in the network. For higher numbers of failures, AF requires many more steps to converge, as is shown in the next set of experiments. In contrast, APC performs similarly to FaCeGreedy. This is because the number of failures considered in this experimental scenario is small, and the initial coverage phase implemented by APC helps with the detection of many working nodes. In Figure 4, we analyse the growth of the evaluation metrics step-wise when four failures occur in the network, and the number of tests is bounded by the number of tests used by PoPGreedy to converge. We can observe that within the same experiments, a_W and a_B have a monotone growing trend (Figures 4a and 4b), whereas R_1 and R_2 may oscillate as a consequence of the fact that during intermediate probes, working nodes may gain a high failure probability that goes abruptly to 0 when a working path traverse them (Figures 4c and 4d). We can also observe that a_W curves are concave and grow steeply with the very first path tests. a_B instead has a convex trend, and in the first steps it might be 0. This is because it takes a number of tests before a node can be classified as failed, whereas working paths give immediate node classification.

Figure 5 considers the scenario where all the algorithms are allowed to use all the monitoring paths or run until they reach convergence according to their respective criteria. The latter condition does not hold for AF since it is not limited to moving along given paths between monitors. AdaptiveFinder requires a greater number of tests than those used by PopGreedy and FaCeGreedy to converge (see Figure 5b), while GI and GD always probe all available paths. When we do not impose constraints on the maximum number of paths to probe, AF converges to the ground truth, i.e., it correctly classifies all nodes. We stress that this is due to its possibility to monitor single nodes directly and to its freedom to route through the network without the restriction of moving along given paths. Once again, PoPGreedy and FaCeGreedy test small portions of the available monitoring paths to reach convergence (Figure 5b). This also holds for APC in this failure scenario. As expected, the average elapsed time required by PoPGreedy considerably increases with the number of failed nodes, even on a small network (see Figures 3e and 5a). The high variance is due to its exponential dependence on the number of failed paths, which amplifies the discrepancy between the situations in which central or non central nodes fail. Because of its computational complexity, we do not include PoPGreedy in the next experiments conducted on the larger graph of the Minnesota network.

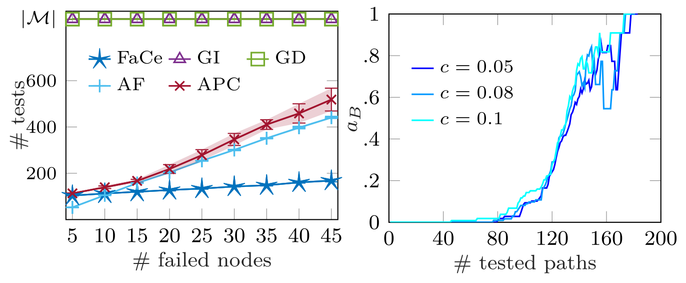
2) *Minnesota:* Figures 6 and 7 show our experiments on the Minnesota network. In Figure 6, tests are run until convergence or until a maximum number of tests K has been reached, whichever occurs earlier. In this case, the bound K is given by the number of path probes needed by FaCeGreedy to converge. In contrast, the experiments of Figure 7 are run until convergence or until all available paths are probed. Also in this setting, GI and GD need to test all available paths and are still unable to converge because of their inability to take account of the progressively available information which can be obtained by probing the paths in a sequence. In fact, in Figure 6, GI and GD use the same number of paths as FaCeGreedy but with very poor classification performance, whereas for the unbounded tests in Figure 7, they reach the same performance as FaCeGreedy by probing all available paths. On the other hand, FaCeGreedy is able to obtain full network information



(a) Accuracy over working nodes, a_W (b) Accuracy over broken nodes, a_B

(c) R_1 (d) R_2

Fig. 6. Tests on the minnesota topology. Bounded.

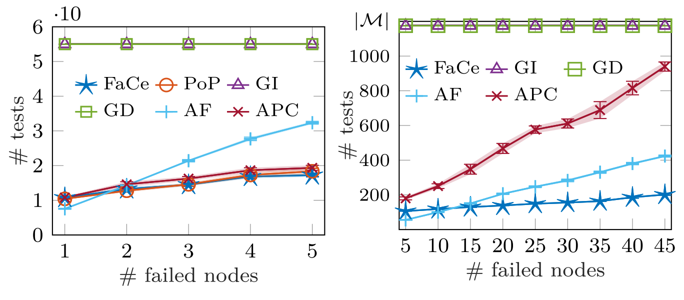


(a) Number of tests (b) Accuracy over broken nodes, a_B

Fig. 7. Tests on the minnesota topology. Unbounded.

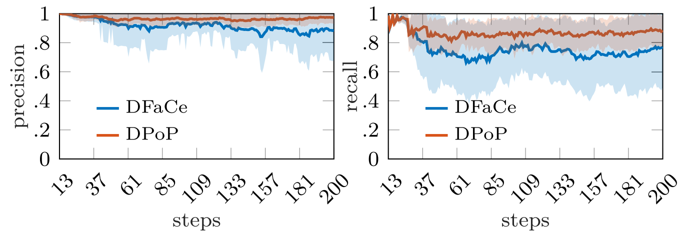
by converging with less than 9% of all the available paths. Once again, in this configuration, in the unbounded case of Figure 6, AF is able to correctly detect all the failures within the maximum number of tests K only when the failure set is very small. Despite the good performance of APC in the BICS network, when APC is applied to a larger network and when many failures occur, it uses many more paths to reach convergence than FaCeGreedy (Figure 7a) and performs poorly in the bounded tests (Figure 6). Indeed, performing a greedy for coverage (as APC does in its preliminary phase) is a convenient way to discover many working nodes only when very few failures occur. Similarly, the approach implemented in APC and summarized in Section VII-B is beneficial only when the failed nodes do not lay on the same paths and are few.

Together with the aforementioned metrics, we also study how different choices of prior centrality values (c) may affect the performance of FaCeGreedy in terms of a_B. Figure 7b depicts how the accuracy of detection of broken nodes changes at each iteration of FaCeGreedy for $c = 0.05, 0.08, 0.1$. For each experiment, 35 failed nodes (8% of the total number of



(a) BICS (b) Minnesota

Fig. 8. Number of tests required by all heuristics to converge in the geographically distributed failure scenario.



(a) Precision (b) Recall

Fig. 9. Precision and recall of DFaCeGreedy and DPoPGreedy.

TABLE IV

Average Percentages of Detected State Changes and Average Time for Change Detection, and Their Standard Deviation

	$\%_{F\to W}$	$\%_{W\to F}$	$t_{F\to W}$	$t_{W\to F}$
PoP	82.3 ($\pm$15.6)	89.7 ($\pm$9.1)	6.6 ($\pm$2.4)	12.7 ($\pm$6.3)
FaCe	56.7 ($\pm$21.5)	89.7 ($\pm$13.6)	14.0 ($\pm$8.7)	14.5 ($\pm$7.6)

covered nodes) are generated. Despite curves varying throughout intermediate iterations, and despite small differences in the final number of tested paths, FaCeGreedy is able to reach maximal accuracy (i.e., $a_B = 1$) also for under and over estimated choices of c (that is, $c = 0.05$ and $c = 0.1$), proving its consistency and robustness against potentially wrong settings of the prior probability or centrality of a node.

3) Geographically-Distributed Failures: Differently from the previous tests, which considered a uniform failure distribution, we now let failures occur more likely in the proximity of specific geographic spots to simulate realistic failure scenarios, including blackouts and natural disasters. We choose a geographic point in the network to represent the epicenter of a catastrophic event and let all nodes fail according to a normal bi-variate geographic distribution around the selected point. In Figure 8, we show the number of tests required by all heuristics to converge. The figure highlights that PoPGreedy and FaceGreedy have consistent performance in this failure scenario, whereas APC is more sensitive to the concentration of failed nodes. GI, GD and AF perform similarly to previous tests (Figures 5b and 7a).

4) Experiments on Dynamic Failures: Figure 9 shows the average precision and recall of DPoPGreedy and DFaCeGreedy on the BICS network. Once again, shades represent standard deviations. We run the algorithms for 200 steps, and we consider the window size ℓ_w to be 12 to ensure coverage within each window slot. Each working node may fail with probability $p_{W\to F} = 0.1$. Failed nodes remain failed for a fixed number of steps, and then they may be fixed with probability $p_{F\to W} = 0.7$ at each successive step. We compute the evaluation metrics at each time step, from step 13 to step 200. In Table IV, we show the percentage of detected node state changes and the average time for change detection. The experiments are averaged on 50 runs, and standard deviations are provided between parenthesis.

VIII. Conclusion

Boolean Network Tomography (BNT) provides the design of end-to-end monitoring paths to ensure network failure localization. However, when the number of concurrent failures is unknown, BNT techniques hit the snag of the huge dimension and intractability of the solution space. In this paper, we propose a progressive approach to failure localization in the challenging scenario where an unknown and unbounded number of failures may occur. A set of monitoring paths is probed in a progressive manner, and decisions on which path to probe are made on the basis of a Bayesian approach which optimizes the expected value of the failure related information that can be obtained by incrementally monitoring new paths. To tackle the complexity of calculating posterior failure probabilities at each monitoring step, we propose a *failure centrality metric*, computable in polynomial time, which reflects the likelihood of a node to be the site of a failure. We use such a metric to guide decision making and provide a conclusive assessment of the state of network components. We extend the study to an online scenario where node states change dynamically throughout the experimental period. The dynamic extensions of our algorithms are based on a sliding observation window technique which adaptively considers information obsolescence. The experiments show that our approach outperforms state of the art solutions based on classic Boolean Network Tomography as well as approaches based on progressive group testing in all the considered scenarios.

Appendix
Optimality approximation

In this section, we provide the formulation of the approximation with respect to the optimal solution of the policy implemented by PoPGreedy. We refer the interested readers to [33] for a more detailed analysis with examples and detailed proofs. It is well-known that greedy algorithms for solving deterministic optimization problems whose objective functions have properties of monotonicity and submodularity provide a solution that is a constant approximation of the optimal solution [34]. More recently, the concept of *adaptive* monotonicity and submodularity, originally introduced in [35] and lately revised in [36], extended such properties to the context of *stochastic* optimization problems. In such problems, the function to be maximized depends on a set of observations $O_{\mathcal{T}}$ on the state of the elements of the ground set (in our case, the state of paths $\mathcal{M}$). The problem envisioned in this paper falls into the definition of stochastic maximization problem and can be formulated as follows:

$$\max_{\mathcal{A}^{(\mathcal{T})}\subseteq\mathcal{A}} \lambda(\mathcal{A}^{(\mathcal{T})}|O_{\mathcal{T}}) \quad s.t.\ |\mathcal{T}| \le K \tag{12}$$

where $\lambda(\mathcal{A}^{(\mathcal{T})}|O_{\mathcal{T}})$ is the utility of the set of actions $\mathcal{A}^{(\mathcal{T})} = \{a^{(1)}, \ldots, a^{(|\mathcal{T}|)}\}$. In the context of stochastic maximization problems, at each step, greedy policies choose the action that maximizes the *expected value* of the utility, whose actual value is known with certainty only after tests take place. Notice that PoPGreedy in Algorithm 1 follows the Adaptive Greedy Algorithm structure shown in [35]. To evaluate the optimality approximation of PoPGreedy, we use the results recently proven in [5], according to which it is possible to study the approximation of the solution obtained by a greedy policy with respect to the optimal one by bounding the *adaptive submodularity ratio* $\gamma_{O_{\mathcal{T}},k}(\lambda,p)$ of its utility, with a scalar $\alpha \in (0,1]$. The resulting approximation is:

$$\lambda_{avg}(\pi^G) \ge \left(1 - \exp\left(-\frac{\alpha K}{h}\right)\right)\lambda_{avg}(\pi^*) \tag{13}$$

where $\lambda_{avg}(\pi^G)$ and $\lambda_{avg}(\pi^*)$ are the average quantity of information gained by the greedy and the optimal policies π^G and π^*, respectively. The parameters K and h are the constraint to the maximum number of tests and the height of the decision tree of the optimal policy π^*, respectively.

1) Bounds of the Adaptive Submodularity Ratio: The goal of this section is to exhibit a scalar $\alpha > 0$ such that

$$\gamma_{O_{\mathcal{T}},k}(\lambda,p) \triangleq \frac{\sum_{m\in\mathcal{M}} P(m \in \mathcal{T}^{\pi})\Delta(a|O_{\mathcal{T}})}{\Delta(\pi|O_{\mathcal{T}})} \ge \alpha. \tag{14}$$

The adaptive submodularity ratio is upper-bounded by 1 and it is equal to 1 if and only if λ is adaptive submodular. The term $\Delta(*|O_{\mathcal{T}})$ is called *conditional expected marginal benefit (of an action or of a policy)* and was introduced in [35]. We report its formal definition hereunder:

Definition 3: Let $Y \subset X$, $|X| < \infty$, and let $x \in X \setminus Y$. The conditional expected marginal benefit *of x with respect to a function f, having observed O_Y is*:

$$\Delta(x|O_Y) := \mathbb{E}[f(Y \cup \{x\}, O_Y) - f(Y, O_Y)]. \tag{15}$$

It is easy to prove that the definition of conditional expected marginal benefit corresponds to the definition of expected utility given in Equation 2, $\Delta(a|O_{\mathcal{T}}) \equiv \mathcal{U}(a|O_{\mathcal{T}})$. In particular, in our scenario the ground set X is the set of all possible actions $\mathcal{A}$ on paths $\mathcal{M}$, and the state of a path is either normal or defective (or equivalently, 0 or 1).

The inequality 14 can be equivalently expressed as follows [5]:

$$\sum_{m\in\mathcal{M}} P(m \in \mathcal{T}^{\pi}) d_a(\alpha) \ge 0. \tag{16}$$

where $d_a(\alpha) := \mathcal{U}(a|O_{\mathcal{T}}) - \alpha\mathcal{U}(a|O_{\mathcal{T}'})$; $O_{\mathcal{T}'}$ is the set of observations such that the next path chosen by policy π is m, and $\mathcal{T} \subset \mathcal{T}'$. To prove the previous relation, we want to show that for every $a \in \mathcal{A}$ (i.e., for every action corresponding to probing path m such that $m \notin \mathcal{T}'$) it holds that

$d_a(\alpha) = \mathcal{U}(a|O_{\mathcal{T}}) - \alpha\mathcal{U}(a|O_{\mathcal{T}'}) \geq 0$. Among all path choices, we just need to study the contribution of those such that $d_a(1) < 0$. Therefore, we exclude all actions corresponding to probing the following sets of paths: *i.* already tested paths; *ii.* Paths m such that $P(Z|O_{\mathcal{T}}) = 0$ or 1 or such that $P(Z|O_{\mathcal{T}'}) = 0$ or 1; *iii.* paths m such that $\mathcal{U}(a|O_{\mathcal{T}}) \geq \mathcal{U}(a|O_{\mathcal{T}'})$. For all these, it holds that $d_a(1) \geq 0$.

We study the maximum difference (i.e., the maximum value of $|d_a(\alpha)|$ with $d_a(\alpha) < 0$) that may occur between $\mathcal{U}(a|O_{\mathcal{T}})$ and $\mathcal{U}(a|O_{\mathcal{T}'})$ for all other paths. To do so, we study the smallest non-zero value of $\mathcal{U}(a|O_{\mathcal{T}})$, Δ_{min}, and the greatest value of $\mathcal{U}(a|O_{\mathcal{T}'})$, Δ'_{max}. By choosing $\alpha = \frac{\Delta_{min}}{\Delta'_{max}}$, Equation 16 is always satisfied.

a) Smallest value of $\mathcal{U}(a|O_{\mathcal{T}}) \equiv \Delta_{min}$: By means of algebraic and analytic steps detailed in [33], we show that the minimum value of $\mathcal{U}(a|O_{\mathcal{T}})$ strictly depends on the minimum value of $P(Z|O_{\mathcal{T}})$ such that $P(Z|O_{\mathcal{T}}) > 0$, to which we refer to as P_{min}, and prove that:

$$P_{min} \geq \left[1 - \frac{p}{1 + (1-p)(p^{\partial_{max}} - 1)}\right]^{|\hat{m}_{max}|} \equiv LB(P_{min}),$$

where ∂_{max} is the maximum number of failing paths traversing a single node and $|\hat{m}_{max}|$ is the length of the longest path. Consequently:

$$\Delta_{min} \geq (|\hat{m}^{(\mathcal{T})}| + |\mathcal{F}_1^{m,(\mathcal{T})}|) \cdot LB(P_{min}). \tag{17}$$

b) Greatest value of $\mathcal{U}(a|O_{\mathcal{T}'}) \equiv \Delta'_{max}$: The greatest value of $\mathcal{U}(a|O_{\mathcal{T}'})$ with $P(Z|O_{\mathcal{T}'}) < 1$ results in two occasions: when path m does not intersect any failed paths; when it is possible to localize failed nodes traversed by failing paths intersecting m, and none of these nodes is traversed by path m. In these cases, $\mathcal{U}(a|O_{\mathcal{T}'}) = |\hat{m}^{(\mathcal{T}')}|(1-p)^{|\hat{m}^{(\mathcal{T}')}|} + \lfloor 1/|\hat{m}^{(\mathcal{T}')}| \rceil$. This function has a global maximum in $-\lfloor \frac{1}{\ln(1-p)} \rceil$, where $n = \lfloor x \rceil$ is the nearest integer of x. It follows that $\Delta'_{max} = -\lfloor \frac{1}{\ln(1-p)} \rceil (1-p)^{-\lfloor \frac{1}{\ln(1-p)} \rceil}$.

c) Solution approximation: By choosing $\alpha = \frac{\Delta_{min}}{\Delta'_{max}}$ as discussed in the previous sections, it holds that for all paths m and observations $O_{\mathcal{T}}$ and $O_{\mathcal{T}'}$, $d_a(\alpha) \geq 0$, implying soundness of Equation 14.

Proposition 4: If π^G is the policy representing the adaptive greedy algorithm using K steps, and $\lambda : 2^{\mathcal{M}} \times O_{\mathcal{M}} \rightarrow \mathbb{R}_{\geq 0}$ is the utility function defined in Equation 12, then:

$$\lambda_{avg}(\pi^G) \geq \left(1 - \exp\left(-\frac{\alpha K}{k}\right)\right) \lambda_{avg}(\pi^*)$$

where π^* is the optimal policy, k is the number of steps that π^* takes to reach convergence and $\alpha = \frac{\Delta_{min}}{\Delta'_{max}}$.

Proof: The statement is a direct consequence of the following facts: *i.* λ is adaptive monotone. *ii.* Theorem 1 in [5], reported in Equation 13. *iii.* PoPGreedy is an Adaptive Greedy Algorithm. □

Notice that α is dependent on controllable parameters ∂_{max} and $|\hat{m}_{max}|$ that do not depend on the network topology but only on the routing paths choice.

References

[1] H. X. Nguyen and P. Thiran, "The Boolean solution to the congested IP link location problem: Theory and practice," in *Proc. IEEE INFOCOM 26th Int. Conf. Comput. Commun.*, May 2007, pp. 2117–2125.

[2] N. Bartolini, T. He, V. Arrigoni, A. Massini, F. Trombetti, and H. Khamfroush, "On fundamental bounds on failure identifiability by Boolean network tomography," *IEEE/ACM Trans. Netw.*, vol. 28, no. 2, pp. 588–601, Apr. 2020.

[3] N. Bartolini, T. He, and H. Khamfroush, "Fundamental limits of failure identifiability by Boolean network tomography," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, May 2017, pp. 1–9.

[4] V. Arrigoni, N. Bartolini, and A. Massini, "Topology agnostic bounds on minimum requirements for network failure identification," *IEEE Access*, vol. 9, pp. 6076–6086, 2021.

[5] K. Fujii and S. Sakaue, "Beyond adaptive submodularity: Approximation guarantees of greedy policy with adaptive submodularity ratio," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 2042–2051.

[6] T. He, N. Bartolini, H. Khamfroush, I. Kim, L. Ma, and T. L. Porta, "Service placement for detecting and localizing failures using end-to-end observations," in *Proc. IEEE 36th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jun. 2016, pp. 560–569.

[7] A. Karbasi and M. Zadimoghaddam, "Sequential group testing with graph constraints," in *Proc. IEEE Inf. Theory Workshop*, Sep. 2012, pp. 292–296.

[8] M. Mukamoto, T. Matsuda, S. Hara, K. Takizawa, F. Ono, and R. Miura, "Adaptive Boolean network tomography for link failure detection," in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage. (IM)*, May 2015, pp. 646–651.

[9] S. Tati, S. Silvestri, T. He, and T. L. Porta, "Robust network tomography in the presence of failures," in *Proc. IEEE 34th Int. Conf. Distrib. Comput. Syst.*, Jun. 2014, pp. 481–492.

[10] W. Ren and W. Dong, "Robust network tomography: K-Identifiability and monitor assignment," in *Proc. IEEE INFOCOM 35th Annu. IEEE Int. Conf. Comput. Commun.*, Apr. 2016, pp. 1–9.

[11] T. He, A. Gkelias, L. Ma, K. K. Leung, A. Swami, and D. Towsley, "Robust and efficient monitor placement for network tomography in dynamic networks," *IEEE/ACM Trans. Netw.*, vol. 25, no. 3, pp. 1732–1745, Jun. 2017.

[12] H. Li, Y. Gao, W. Dong, and C. Chen, "Taming both predictable and unpredictable link failures for network tomography," *IEEE/ACM Trans. Netw.*, vol. 26, no. 3, pp. 1460–1473, Jun. 2018.

[13] N. Duffield, "Simple network performance tomography," in *Proc. ACM SIGCOMM Conf. Internet Meas. (IMC)*, 2003, pp. 210–215.

[14] N. Duffield, "Network tomography of binary network performance characteristics," *IEEE Trans. Inf. Theory*, vol. 52, no. 12, pp. 5373–5388, Dec. 2006.

[15] R. R. Kompella, J. Yates, A. Greenberg, and A. C. Snoeren, "Detection and localization of network black holes," in *Proc. IEEE INFOCOM 26th IEEE Int. Conf. Comput. Commun.*, May 2007, pp. 2180–2188.

[16] Y. Bejerano and R. Rastogi, "Robust monitoring of link delays and faults in IP networks," in *Proc. IEEE INFOCOM 22nd Annu. Joint Conf. IEEE Comput. Commun. Societies*, Mar. 2003, pp. 134–144.

[17] L. Ma, T. He, A. Swami, D. Towsley, K. K. Leung, and J. Lowe, "Node failure localization via network tomography," in *Proc. Conf. Internet Meas. Conf.*, Nov. 2014, pp. 195–208.

[18] L. Ma, T. He, A. Swami, D. Towsley, and K. K. Leung, "Network capability in localizing node failures via end-to-end path measurements," *IEEE/ACM Trans. Netw.*, vol. 25, no. 1, pp. 434–450, Feb. 2017.

[19] M. Cheraghchi, A. Karbasi, S. Mohajer, and V. Saligrama, "Graph-constrained group testing," *IEEE Trans. Inf. Theory*, vol. 58, no. 1, pp. 248–262, 2012.

[20] S. Pan, P. Li, D. Zeng, S. Guo, and G. Hu, "A Q—Learning based framework for congested link identification," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 9668–9678, Dec. 2019.

[21] L. Ma, Z. Zhang, and M. Srivatsa, "Neural network tomography," 2020, *arXiv:2001.02942*.

[22] B. Arzani *et al.*, "007: Democratically finding the cause of packet drops," in *Proc. 15th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, 2018, pp. 419–435.

[23] A. R. M. Kamal, C. J. Bleakley, and S. Dobson, "Failure detection in wireless sensor networks: A sequence-based dynamic approach," *ACM Trans. Sensor Netw.*, vol. 10, no. 2, pp. 1–29, Jan. 2014.

[24] T. Muhammed and R. A. Shaikh, "An analysis of fault detection strategies in wireless sensor networks," *J. Netw. Comput. Appl.*, vol. 78, pp. 267–287, Jan. 2017.

[25] R. R. Swain, P. M. Khilar, and S. K. Bhoi, "Heterogeneous fault diagnosis for wireless sensor networks," *Ad Hoc Netw.*, vol. 69, pp. 15–37, Feb. 2018.

[26] A. Roy, P. Kar, S. Misra, and M. S. Obaidat, "D3: Distributed approach for the detection of dumb nodes in wireless sensor networks," *Int. J. Commun. Syst.*, vol. 30, no. 1, Jan. 2017, Art. no. e2913.

[27] Y. Huang, N. Feamster, and R. Teixeira, "Practical issues with using network tomography for fault diagnosis," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 5, pp. 53–58, Oct. 2008.

[28] A. Johnsson, C. Meirosu, and C. Flinta, "Online network performance degradation localization using probabilistic inference and change detection," in *Proc. IEEE Netw. Oper. Manage. Symp. (NOMS)*, May 2014, pp. 1–8.

[29] V. Arrigoni, N. Bartolini, A. Massini, and F. Trombetti, "Failure localization through progressive network tomography," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, May 2021, pp. 1–10.

[30] N. Hohn, K. Papagiannaki, and D. Veitch, "Capturing router congestion and delay," *IEEE/ACM Trans. Netw.*, vol. 17, no. 3, pp. 789–802, Jun. 2009.

[31] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The internet topology zoo," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 9, pp. 1765–1775, Sep. 2011.

[32] *Aurora Fiber Optic Networks*. Accessed: Nov. 26, 2019. [Online]. Available: http://tomography.di.uniroma1.it/topologies

[33] V. Arrigoni, N. Bartolini, A. Massini, and F. Trombetti, "Static and dynamic failure localization through progressive network tomography," 2021, *arXiv:2103.17221*.

[34] G. L. Nemhauser and L. A. Wolsey, "Best algorithms for approximating the maximum of a submodular set function," *Math. Oper. Res.*, vol. 3, no. 3, pp. 177–188, 1978.

[35] D. Golovin and A. Krause, "Adaptive submodularity: Theory and applications in active learning and stochastic optimization," *J. Artif. Intell. Res.*, vol. 42, pp. 427–486, Nov. 2011.

[36] H. Esfandiari, A. Karbasi, and V. Mirrokni, "Adaptivity in adaptive submodularity," 2019, *arXiv:1911.03620*.

Viviana Arrigoni (Student Member, IEEE) received the Ph.D. degree in computer science from the Sapienza University of Rome. She is currently a Post-Doctoral Researcher with the Sapienza University of Rome. Her research interests comprise network tomography, computational linear algebra, HPC, and information theory.

Novella Bartolini (Senior Member, IEEE) received the degree (Hons.) and the Ph.D. degree in computer engineering from the Sapienza University of Rome, Italy, in 1997 and 2001, respectively. She was a Visiting Professor with Penn State University from 2014 to 2017. She is currently a Full Professor with the Sapienza University of Rome. Her research interests lie in the area of wireless networks and network management. She was the program chair and a program committee member of several international conferences. She also serves on the Editorial Board of IEEE/ACM TRANSACTIONS ON NETWORKING.

Annalisa Massini received the degree in mathematics and the Ph.D. degree in computer science from the Sapienza University of Rome, Italy, in 1989 and 1993, respectively. Since 2001, she has been an Associate Professor with the Department of Computer Science, Sapienza University of Rome. Her research interests include hybrid systems, sensor networks, and networks topologies.

Federico Trombetti (Student Member, IEEE) received the B.Sc. and M.Sc. degrees (Hons.) from the Sapienza University of Rome in 2017 and 2019, respectively, where he is currently pursuing the Ph.D. degree in computer science with the Department of Computer Science. He also works on Boolean network tomography and network performance optimization. He also acts as a reviewer for several leading conferences and journals.

Open Access funding provided by 'Università degli Studi di Roma "La Sapienza"' within the CRUI CARE Agreement