Signaling Free Localization of Node Failures in All-Optical Networks

János Tapolcai[*], Lajos Rónyai[†], Éva Hosszu[*], Pin-Han Ho[‡], Suresh Subramaniam[§]
[*] MTA-BME Future Internet Research Group, Budapest University of Technology, {tapolcai, hosszu}@tmit.bme.hu
[†] Computer and Automation Research Institute Hungarian Academy of Sciences and BME, ronyai@sztaki.hu
[‡] Dept. of Electrical and Computer Engineering, University of Waterloo, Canada, p4ho@uwaterloo.ca
[§] Dept. of Electrical and Computer Engineering, George Washington University, suresh@gwu.edu

***Abstract*—Network-wide local unambiguous failure localization (NL-UFL) [1] has been demonstrated as an interesting scenario of monitoring trails (m-trails). It attempts to enable every node to autonomously localize any failure event in the network in a distributed and all-optical manner by inspecting a set of m-trails traversing through the node. This paper investigates the m-trail allocation problem under the NL-UFL scenario by taking each link and node failure event into consideration. Bound analysis is performed using combinatorial group testing (CGT) theory and this is followed by the introduction of a novel heuristic on general topologies. Extensive simulation is conducted to examine the proposed heuristic in terms of the required cover length and the number of m-trails to achieve NL-UFL.**

I. Introduction

Generalized Multi-Protocol Label Switching (GMPLS) has served as a building block of Internet backbone control and management. It supports automatic failure restoration mechanisms in optical networks via a suite of signaling protocols, referred to as GMPLS-based recovery. The following five recovery phases [2] are defined as a standard sequence of generic operations performed when an optical layer failure event occurs: (1) failure detection, (2) failure localization (isolation), (3) failure notification, (4) failure correlation, and (5) service restoration. Phases (1)–(3) are also referred to as *fault management*, which concerns with how the control plane acquires the failure event information; phases (4)–(5) are for the recovery of the affected working traffic from the failure event. All the phases rely on electronic signaling via cross-layer protocol operations. In general, the detection of a failure event in the transport layer will trigger the control plane for subsequent actions by way of the GMPLS signaling protocol stacks.

Optical layer fault localization has been extensively studied in the past, and it is positioned to facilitate GMPLS fault management in phases (2) and (3) so that a fast and deterministic failure localization can be achieved. Using multi-hop supervisory lightpaths, referred to as *monitoring trails* (m-trails), has been claimed as an effective approach for reducing the dependence on the upper layer control signaling mechanisms that otherwise serve as the main source of complexity in achieving fast and all-optical failure restoration [1], [3]–[13]. Each m-trail is turned into the off state if it is interrupted by a failure event (e.g., loss of light, loss of signal, or any irregularity defined in the monitoring plane), and the state changes of the interrupted m-trails are sensed at some monitors and coordinated at a network controller for the failure localization and notification tasks. Therefore, the m-trail approach is expected to serve as a complement to the existing electronic signaling approaches and enables an ultra-fast and deterministic fault management process.

Local unambiguous failure localization (L-UFL) is a recently reported development under the m-trail framework first introduced in [11]. An *L-UFL capable node* is defined as one that can determine the network failure status solely by observing the on-off status of the m-trails traversing *that* node. A distinguishing feature of the L-UFL framework is that multiple nodes can share the on-off status of a common m-trail traversing them via signal tapping. Based on L-UFL, a number of research results have been reported, including [1] that considered all nodes as L-UFL capable for single link failures, referred to as network-wide L-UFL (NL-UFL); [12] that studied multi-link *shared risk link group* (SRLG) failure localization; [13] that explored the monitoring burst (m-burst) architecture on multi-link SRLGs; and [14], [15] that further integrated the failure localization mechanism with failure restoration.

All the abovementioned studies are on failures of link(s); node failures have never been considered in the NL-UFL scenario. Since a network node bears all the functions of routing, signaling, monitoring and data/information relaying and storage, the failure of a node certainly has a tremendous impact upon network operation, particularly in the aspect of control and management in the context of all-optical networks. It is expected that the instant acquisition of node failure statuses at a remote decision node can achieve significantly better network capacity efficiency. For example the bandwidth of all the connections terminating at a failed node can be released[1] and used by some protection paths corresponding to the failure event[2].

In spite of its ultimate importance, research on node failure

The project was supported by Hungarian Academy of Sciences (MTA) OTKA grants K108947, NK105645, K77476 and TÁMOP-4.2.2/b-10/1-2010-0009. This document has been produced with the financial assistance of the European Union under the FP7 GÉANT project grant agreement number 605243 as part of the MINERVA Open Call project.

[1]It is also called stub release.
[2]We assume that if a node is down then every incident link is down.

localization, to the best of our knowledge, is a missing piece of the state-of-the-art toward a complete solution plane for all-optical failure restoration. Note that the L-UFL m-trail allocation problem under node failures cannot be analyzed by transforming the topology into a line graph[3] and reusing the reported results for link failures, because these approaches only work in the scenarios where the considered failure events affect a small number of links (see also Section IV for a comprehensive analysis).

Motivated by the above observation, this paper presents our research results on node failure localization using m-trails for achieving NL-UFL. We require every node to be able to determine *any* remote node failure by solely inspecting the on-off status of the traversing m-trails with a target of minimizing the number of m-trails deployed in the network. The paper presents a series of bound analyses based on combinatorial group testing (CGT) theory, followed by a novel heuristic scheme that can efficiently determine the required m-trails and the alarm code table (ACT) at each node for every single link and node failure event. Extensive simulation is conducted to examine the proposed heuristic in terms of cover length and the number of m-trails, which is related to the consumed wavelength channels and the required transponders corresponding to the m-trail solution; it also demonstrates the effectiveness of the proposed heuristic algorithm and the performance impact of topology diversity.

Our contributions in this paper are summarized as follows.

- Although localizing single link failures under NL-UFL was studied in [1] the developed theories and heuristic schemes cannot be used in the node failure cases because a single node failure may affect many links. We claim that this paper is the first attempt in approaching this problem and gaining insights into the performance through bounds.
- We show how the m-trail allocation problem of NL-UFL under node failures is related to the Ahlswede-Katona theory, which focuses on bounded test sets in the context of combinatorial group testing (CGT) [16], [17]. Our problem leads to a novel and quite general CGT scenario. The notion of observatories allows us to capture the characteristics of our problem, and allows us to give a new lower bound on the number of tests. Somewhat surprisingly, Shannon entropy seems to enter the picture.
- We show that the lower bound can be tight within a small factor of about 1.23 by giving a special sparse network structure with m-trails via a novel construction based on Gray codes.
- We provide a simple yet powerful heuristic that can solve the NL-UFL m-trail allocation problem under node and sparse SRLG failures on realistic network topologies.

The rest of the paper is organized as follows. Section II presents a literature review and presents the background knowledge for the research. Section III defines the m-trail problem. Section IV presents a bound analysis on the formulated problem. Section V introduces the proposed heuristic algorithm on general graphs and Section VI shows simulation results which verify the proposed heuristic algorithm. Section VII concludes the paper.

[3]In the line graph $L(G)$ each node represents an edge of G; two nodes of $L(G)$ are adjacent if and only if their corresponding edges are incident in G.

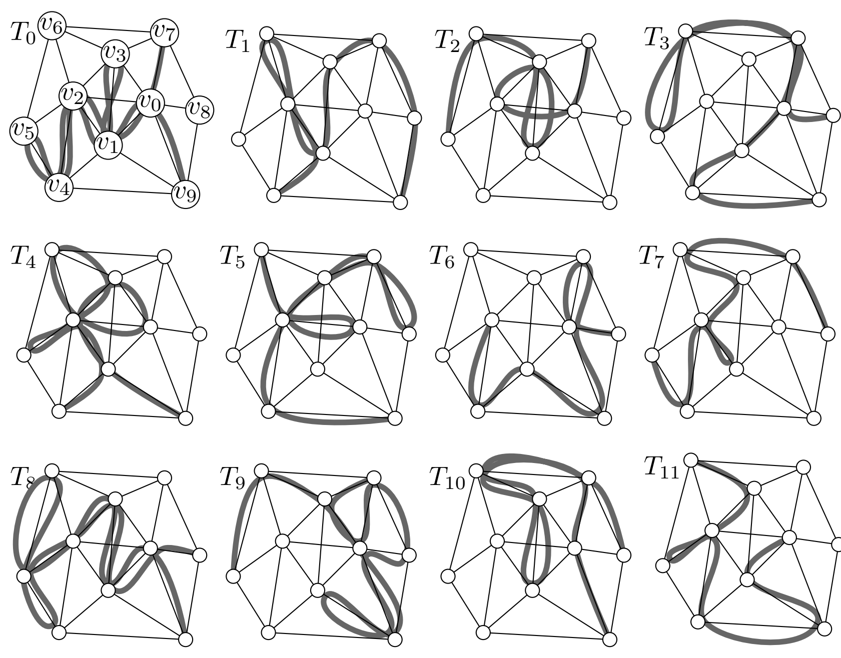
II. Background

Failure localization using multi-hop supervisory lightpaths (m-trails) has been extensively studied in the past decade [1], [3]–[9]. L-UFL [1], [10]–[12] is an interesting implementation of m-trails, aiming at signaling-free failure localization that operates purely in the optical domain. With the set of m-trails properly allocated, a node is *L-UFL capable* if the node can unambiguously identify any link failure according to locally available m-trail on-off status information.

[10] studied how to determine one or more monitoring locations (MLs) in the network in order to collaboratively identify the failed SRLGs according to the alarms collected by the MLs. When only a single ML is required, the ML is L-UFL capable. [11] extended [10] by exploring the scenario where not only the terminating node but also an intermediate node of an m-trail can obtain its on-off status via optical signal tapping. The study allocated m-trails which enable a given set of nodes as L-UFL capable via an integer linear program and discovered the fact that the total length of the m-trails scales very well with the number of L-UFL capable nodes, mostly due to the sharing of on-off statuses among the nodes traversed by a common m-trail. Motivated by the result, similar ideas were explored in [12] and [1]. The former introduced a heuristic approach for achieving L-UFL of a small set of MLs under multi-link failures, while the latter investigated the scenario that all the nodes are made to be L-UFL capable for any single link failure. An efficient heuristic was developed for allocating m-trails in the shape of a spanning tree via link code swapping. [1] defines this scenario as Network-wide L-UFL (NL-UFL).

To the best of our knowledge there is no research reported on node failures, which are the main focus of this paper. Fig. 1 shows an example of NL-UFL for any single-link and node failures using 12 m-trails, $T_0, \ldots, T_{11}$, in the SmallNet topology. Each node can achieve single-link or -node L-UFL by inspecting the locally available on-off statuses of the traversing m-trails. For example, node v_1 maintains an alarm code table (ACT) on the columns $T_0, \ldots, T_4, T_6 \ldots, T_{11}$ of the table on Fig. 1, where the on-off status of these m-trails form an alarm code of 12 bits which uniquely identifies each possible link or node failure event. If node v_1 finds that T_1 becomes suddenly off while all the remaining m-trails are still on, link (v_8, v_9) is considered down and can be localized as defined in the corresponding row of the ACT. Note that this localization is achieved at node v_1 by observing only the m-trails traversing v_1. The reader can convince himself that every node can localize any single link or node failure using only the on-off statuses of m-trails passing through that node.

The above example raises an interesting question that is investigated in the rest of the paper: how should the m-trails be



Failure	T_0	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9	T_{10}	T_{11}
(v_2, v_3)	0	0	1	0	1	1	0	1	1	0	0	1
(v_1, v_3)	1	1	1	0	0	0	0	0	1	0	1	0
(v_1, v_2)	1	1	0	0	1	0	0	1	0	0	0	0
(v_0, v_3)	0	0	0	0	1	0	0	0	0	1	0	0
(v_0, v_2)	0	0	1	0	1	1	0	0	0	0	0	0
(v_0, v_1)	1	0	0	1	0	0	0	0	1	0	0	1
(v_7, v_3)	0	1	0	0	0	1	0	0	0	1	0	0
(v_7, v_0)	1	0	1	1	0	0	1	0	0	1	1	0
(v_6, v_3)	0	0	1	0	1	0	0	1	0	1	1	1
(v_6, v_2)	0	1	0	0	1	1	0	0	0	0	0	0
(v_6, v_7)	0	0	0	1	0	0	0	1	0	0	1	0
(v_5, v_2)	0	0	0	0	1	0	0	0	1	0	0	1
(v_5, v_6)	0	0	1	1	0	0	0	0	1	1	0	0
(v_4, v_2)	1	0	0	0	0	1	1	1	0	0	0	1
(v_4, v_1)	0	1	0	1	1	0	1	0	0	0	0	0
(v_4, v_5)	1	0	0	0	0	0	0	1	1	0	0	0
(v_9, v_1)	0	0	0	0	1	0	1	0	0	1	0	1
(v_9, v_0)	1	0	0	0	0	0	1	0	1	1	1	0
(v_9, v_4)	0	0	0	1	0	1	0	0	0	0	0	1
(v_8, v_0)	0	0	0	1	0	0	1	0	1	1	0	0
(v_8, v_7)	0	1	0	0	0	1	0	1	0	1	1	0
(v_8, v_9)	0	1	0	0	0	0	0	0	0	0	0	0
v_9	1	1	0	1	1	1	1	0	1	1	1	1
v_8	0	1	0	1	0	1	1	1	1	1	1	0
v_7	1	1	1	1	0	1	1	1	0	1	1	0
v_6	0	1	1	1	1	1	0	1	1	1	1	1
v_5	1	0	1	1	1	0	0	1	1	1	0	1
v_4	1	1	0	1	1	1	1	1	1	0	0	1
v_3	1	1	1	0	1	1	0	1	1	1	1	1
v_2	1	1	1	0	1	1	1	1	1	0	0	1
v_1	1	1	1	1	1	0	1	1	1	1	1	1
v_0	1	0	1	1	1	1	1	0	1	1	1	1

Fig. 1. An NL-UFL m-trail solution for SmallNet. As a comparison, see the solution for UFL with alarm code dissemination in [7].

routed to achieve NL-UFL of all single link *and* node failures?

III. Problem Definition

The problem input is an undirected graph $G = (V, E)$ with node set V and link set E, where the number of nodes is denoted by $n = |V|$ and the number of links by $m = |E|$. The NL-UFL m-trail allocation problem for single-link and node failure is to establish a small set of m-trails, denoted by $\mathcal{T} = \{T_1, \ldots, T_b\}$ where $b = |\mathcal{T}|$ is the number of m-trails, such that each m-trail T_i is a connected subgraph of G, and each node $v_j \in V$ can achieve L-UFL according to the on-off status of m-trails in T^j - the subset of $\mathcal{T}$ containing the m-trails passing through v_j.

The set of m-trails T^j for v_j must satisfy the following two requirements:

(R1): Every link e should be passed by a unique set of m-trails in T^j, such that every link and node has a unique alarm code seen by v_j.

(R2): T_i for $1 \leq i \leq b$ must be a connected subgraph of G.

IV. Bound Analysis

In this section we derive lower bounds on the number of m-trails required to satisfy the NL-UFL m-trail allocation constraint. Note that an analytical study on single link failure localization under NL-UFL was reported in [1], which takes advantage of a suite of spanning trees. Thus a novel method should be developed such that each node has a unique ACT where every other node and link is traversed by a different set of m-trails seen at the node.

TABLE I
Notation list

Notation	Description
$G = (V, E)$	undirected graph representation of the topology
$n = \|V\|$	the number of nodes in G
$m = \|E\|$	the number of links in G
b	the number of m-trails
$\mathcal{T} = \{T_1, \ldots, T_b\}$	a solution with b (b)m-trails
T_i	the i^{th} (b)m-trail, which is a set of links in G
$\|T_i\|$	number of nodes the i^{th} m-trail traverses
$\mathcal{C}^*(n, k)$	minimum number of tests to localize a faulty item among n using tests of average size k
$H(p)$	the binary entropy function
b_v	number of tests containing node v
k_v^*	average size of tests at node v
δ	average nodal degree
a_e	the alarm code of link $e \in E$
$a^e_{<i>}$	the bitwise pair of a_e at the i^{th} position
$a_{e,[j]}$	the j^{th} bit of the alarm code of link $e \in E$
$\|\|\mathcal{T}\|\|_E$	normalized cover length, see (17)

Thus a novel method should be developed such that each node has a unique ACT where every other node and link is traversed by a different set of m-trails seen at the node.

The optimal length of each m-trail is of interest and should be discussed first. The binary search or half-interval search algorithm intuitively suggests that an ideal test should contain half of the nodes. This is in contrast to the case of localizing only link failures as considered in [1] where having each m-trail to traverse all the nodes (via a spanning tree) is the most efficient, as its on-off status is visible at every node.

A. Lower Bounds for Combinatorial Group Testing (CGT)

To obtain lower bounds on the number of necessary m-trails for any single link and node failure, a simplified problem is considered first. Let a network $G = (V, E)$ contain n nodes and m links; our goal is to localize a single node failure using dedicated bi-directional m-trails. For better understanding, in the first step we ignore link failures and just focus on single node failures; it is clear that the derived lower bound will still be a lower bound for the original problem.

We treat this simplified problem as a CGT problem where there are items (i.e. nodes) and we need to define group tests on the items to identify at most one faulty item. In this model only nodes are considered. In particular, tests are subsets of nodes.

The first problem we take is where the average size of the group tests is restricted. A lower bound was proved by Katona [16]. Let $\mathcal{C}(n,k)$ denote the smallest number of tests needed to localize a faulty item among n items using tests of size *exactly* k and $\mathcal{C}^*(n,k)$ using tests of *average size* k, respectively. In [16], Theorem 5 gives a lower bound

$$\frac{\log_2 n}{H(\frac{k}{n})} \leq \mathcal{C}(n,k)\,, \tag{1}$$

where $H(p)$ denotes the binary entropy function,

$$H(p) = -p\log_2 p - (1-p)\log_2(1-p),$$

for $p \in [0,1]$ and $k \leq 2n$.

Ahlswede [17] proved[4] that

$$\frac{\log_2 n}{H(\frac{k}{n})} \leq \mathcal{C}^*(n,k)\,. \tag{2}$$

Next, suppose that we have a set of *observatories* in the input, and each observatory knows the outcome of a given subset of the group tests. We need to ensure that every observatory can identify the faulty item according to the group testing result of the given subset provided there is at most one faulty item. This version of the problem is somewhat similar to the case when a group test may give a false outcome. Basically we need to ensure that a subset of the tests provides sufficient information to identify the failed item.

An interesting special case is when there are b tests and $\binom{b}{k}$ observatories, each seeing a different subset of $b-k$ tests. In this case the code of any two items should have a Hamming distance of at least $k+1$, because if there are two items with a Hamming distance of at most k, the observatory that can see exactly the complementary set of $b-k$ tests cannot distinguish the failure of these two items. In other words, the items should be assigned with alarm codes that are error-correcting in nature. In NL-UFL problem this special case is only possible on complete graphs.

[4]Ahlswede proved the bound in Theorem 1 of [17] only for $k \leq \frac{n}{2}$ and for tests of average size at most k. The bound for $C^*(n,k)$ and for arbitrary $0 \leq k \leq n$ follows by a slight modification of the proof in [17].

B. Localizing Node Failures

The NL-UFL problem for node failures has a recursive nature, as node failures should be localized at nodes that tap the m-trails traversing them. This is captured by the model where each observatory corresponds to an item, and the set of tests the observatory can see is the set of tests that contain the corresponding item. Also, we require that every item hosts an observatory.

We divide the cost of each test equally among the observatories (or equivalently, the number of nodes that an m-trail traverses), and represent the cost in a matrix Ω which has n columns and b rows, where

$$\omega_{v,i} = \begin{cases} \frac{1}{|T_i|} & \text{the } i\text{th m-trail traverses node } v, \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

where $|T_i|$ denotes the number of nodes the test T_i passes through. The total number of tests b can be expressed as

$$\sum_{i=1}^{b}\sum_{v=1}^{n}\omega_{v,i} = \sum_{i=1}^{b} 1 = b\,, \tag{4}$$

which can be reordered as

$$b = \sum_{i=1}^{b}\sum_{v=1}^{n}\omega_{v,i} = \sum_{v=1}^{n}\sum_{i=1}^{b}\omega_{v,i} = \sum_{v=1}^{n}\left(\sum_{i|v\in T_i}\frac{1}{|T_i|}\right)\,. \tag{5}$$

Let k_v^* denote the average size of the tests at node v, formally

$$k_v^* = \frac{\sum_{i|v\in T_i}|T_i|}{b_v}\,, \tag{6}$$

where b_v is the number of tests containing node v. Note that b_v is at least $\mathcal{C}^*(n,k_v^*)$ because v is an observatory. The inequality of harmonic and arithmetic means states that

$$\frac{b_v}{\sum_{i|v\in T_i}\frac{1}{|T_i|}} \leq \frac{\sum_{i|v\in T_i}|T_i|}{b_v} = k_v^*\,. \tag{7}$$

Let σ_v denote the inner sum in the right side in (5) for node v, for which we have the following lower bound

$$\sigma_v = \sum_{i|v\in T_i}\frac{1}{|T_i|} \geq \frac{b_v}{k_v^*} \geq \frac{\mathcal{C}^*(n,k_v^*)}{k_v^*}\,. \tag{8}$$

Using the bound in (2) we have

$$\sigma_v \geq \frac{1}{k_v^*}\frac{\log n}{H(\frac{k_v^*}{n})}\,. \tag{9}$$

To simplify further computations we define $\alpha := \frac{k_v^*}{n}$, and substitute it into (9) to get

$$\sigma_v \geq \frac{1}{\alpha\, n}\frac{\log_2 n}{H(\alpha)} = \frac{\log_2 n}{n}\cdot\frac{1}{\alpha\, H(\alpha)} = \frac{\log_2(n)}{n}g(\alpha)\,, \tag{10}$$

where $g(\alpha) := \frac{1}{\alpha\, H(\alpha)}$.

Fig. 2 shows $g(\alpha)$ for $0 \leq \alpha \leq 1$. Let $G(\alpha) = \frac{1}{g(\alpha)} = \alpha\, H(\alpha)$. Finding the minimum of $g(\alpha)$ is equivalent to finding the maximum of $G(\alpha)$ when $0 \leq \alpha \leq 1$. In order to find the

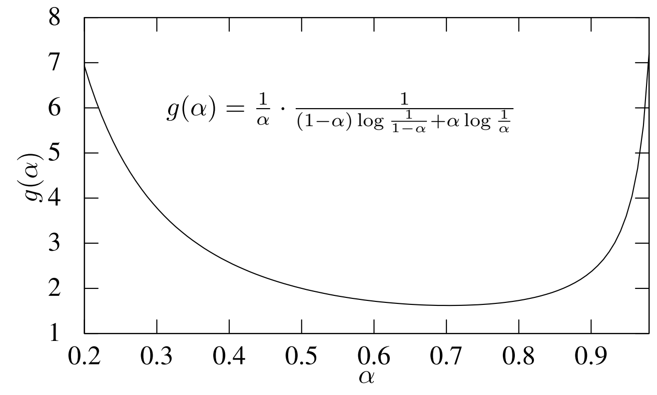


Fig. 2. A plot on $g(\alpha)$ for $0 < \alpha < 1$.

maximum of $G(\alpha)$ we take the derivative w.r.t. α and search for the root.

$$\frac{\mathrm{d}}{\mathrm{d}\alpha}G(\alpha) = \frac{\mathrm{d}}{\mathrm{d}\alpha}\alpha\, H(\alpha) = H(\alpha) + \alpha \cdot \log_2 \frac{1-\alpha}{\alpha}. \tag{11}$$

Simplification yields the following equation:

$$2\alpha \cdot \log_2 \frac{\alpha}{1-\alpha} - \log_2 \frac{1}{1-\alpha} = 0 \tag{12}$$

Let α^* denote the solution of (12) for α. Solving (12) numerically we get $\alpha^* \approx 0.7035$ with $g(\alpha) \geq 1.62088$.

Substituting it back into (10) eventually gives for b:

$$b \geq \sum_{v=1}^{n} g(\alpha)\frac{\log_2(n)}{n} = g(\alpha)\log_2(n) \geq 1.62088\log_2(n). \tag{13}$$

Theorem 1: The number of tests necessary to localize any single node failure at every node is at least

$$b \geq \lceil 1.62088 \log_2(n) \rceil, \tag{14}$$

where n is the number of items.

Tightness of the bound in Theorem 1: Next, we examine the tightness of (14) by providing graphs with NL-UFL solutions close to the bound in Theorem 1. We focus on the problem where the task is to localize every single node failure at every node and still ignore link failures. We construct a graph $G^* = (V, E)$ and $2\lceil\log_2|V|\rceil + 1$ m-trails that can localize every node failure locally at every node, while the graph has only nodes with degrees at most 4. This means that the gap for the lower bound in Theorem 1 can be as low as 23% even on realistic topologies.

The graph is a path with some extra links. It has node set $V = \{v_0, v_1, \dots, v_{n-1}\}$. First we assign codes to the nodes, then the alarm code of each link e is computed as the bitwise AND of the codes assigned to the two nodes incident to e.

The first $b' = \lceil\log_2|V|\rceil$ bits of the codes assigned to nodes $v_0, v_1, \dots, v_{n-1}$ are a series of unique binary codes, where two successive values differ in only one bit (see also Fig. 3). We consider these as column vectors. Using Gray codes, such a node coding process is feasible since $2^{b'} \geq |V|$. The next b'

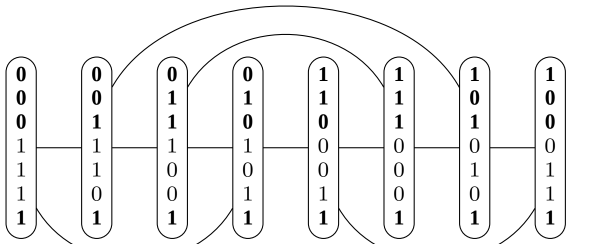


Fig. 3. An example of the Gray-codes mapped to a graph with $b' = 3$.

bits of the node codes will be exactly the complements of the first b' bits allocated to the node. For example, if node v_i has the first bit as 0, then its $(b'+1)$-th bit should be 1, and so on. This results in the fact that the row corresponding to the j-th bit position (denoted as R_j) is the complement of row $R_{j+b'}$, $(1 \leq j \leq b')$. Finally, the last bit of the node codes is 1 for every node. With the $2b'+1$ rows the construction is complete.

G^* has links (v_i, v_{i+1}) for $i = 0, \dots, n-2$, which form a path. Now we add some extra links. For each node v_i we add at most one extra link. Let j be the position $1 \leq j \leq b'$, where the i-th and $(i+1)$-th Gray codes differ. We add at most one extra link (v_i, v_k) with $k > i$ as follows. If $v_i[j] = 1$ and $v_{i+1}[j] = 0$, then let k be the first index which is greater than i and for which $v_k[j] = 1$, provided that there is such a k. Also, if $v_i[j] = 0$ and $v_{i+1}[j] = 1$, then let k be the first index which is greater than i and for which $v_k[j] = 0$, provided that such k exists. These extra links are used to connect the disjoint segments of the j-th and $(b'+j)$-th m-trail, respectively.

The m-trails are edge sets. Trail T_j contains a path from v_l to v_k iff $v_k[j] = v_l[j] = 1$ holds. By construction every T_j is connected. We assume in this model that if a node is down then every link incident to it is down. In particular, a failure of node v will be detected at every node along T_j, provided that T_j is incident to v.

The constructions of G^*, the node-codes and the trails imply that $v \in V$ is incident to trail T_j if and only if the j-th bit of the code of v is 1. To prove NL-UFL, we must verify that every node w can correctly identify a single node failure. Let H be the set of bit positions j where the code of w has value 1. It suffices to verify that the nodes have pairwise different codes when restricted only to bit positions (rows) in H. We call positions $1 \leq i, j \leq 2b$ *complementary* if $|i - j| = b'$ holds. Now observe that H is big in the sense that any position i or its complementary pair is in H. This implies that if the code vectors for nodes u and v agree on positions belonging to H, then they agree everywhere by complementation; this is possible only when $u = v$. The last bit position ensures that the no-failure state is recognized properly.

C. Lower Bound on the Number of M-trails

We now extend our results to single link and node failures. We modify the above model as follows: the items are the nodes and links, the observatories correspond to nodes, and the set of tests the observatory can see is the set of tests that contain the corresponding node. This is a simplified model, where a test may contain any set of nodes and links. We ignore the graph connectivity, and also the fact that an m-trail traversing a link must traverse the adjacent nodes.

Each node must be traversed by at least $\lceil \log_2(m+n+1) \rceil$ m-trails to have a unique alarm code for each failure state, where n is the number of nodes and m is the number of links in the network. This means (8) now becomes

$$\sigma_v \geq \frac{1}{k_v^*} \max\left\{\mathcal{C}^*(n,k_v^*), \lceil \log_2(m+n+1) \rceil\right\} . \qquad (15)$$

It is easy to see that the bound is tighter than Theorem 1 only if the intersection of $\mathcal{C}^*(n,k_v^*)$ and $\lceil \log_2(m+n+1) \rceil$ is at $k_v^* < 0.7035n$. This is the case if

$$1.62088 \log_2(n) < \frac{1}{0.7035} \log_2(m+n+1),$$

which holds if

$$n^{1.1411} < n+m+1$$

after raising both sides to the base 2. This holds when

$$n^{1.1411} \leq n + \frac{\delta}{2} n ,$$

where δ is the average nodal degree. It can be written as

$$n^{0.1411} = \sqrt[7.087]{n} \leq 1 + \frac{\delta}{2} .$$

This leads to the following bound.

Theorem 2: If $n \leq (1+\frac{\delta}{2})^{7.087}$, the number of m-trails to localize a single node failure at every node is at least

$$b \geq \left\lceil \frac{1}{1-\alpha'} \lceil \log_2(m+n+1) \rceil \right\rceil ,$$

where $\alpha' \geq 2$ is the solution for

$$H(\alpha') = \frac{\log(n)}{\lceil \log_2(m+n+1) \rceil} \leq \frac{\log_2(n)}{\log_2(m+n)} = \log_{(m+n)}(n) . \qquad (16)$$

As for practical values, the theorem is valid for $n \leq 661$ if $\delta = 3$, and $n \leq 2406$ if $\delta = 4$. This intuitively shows that for large networks node failure localization is the key difficulty and localizing link failures does not necessarily require additional m-trails.

V. The Heuristic Approach

This section presents a novel heuristic algorithm to solve the NL-UFL m-trail allocation problem for single node and link failures. A *failure scenario* is defined as the failure of a single link, a single node, or both.

Algorithm 1 gives the pseudo code of the proposed heuristic algorithm. In Step (1) the initial number of m-trails b is computed according to Theorem 2. Next, in Step (2), b random trees with at most $\alpha|V|$ nodes are generated, where α is an input parameter from the range $[0.5, 0.95]$. In our implementation the method of Aldous/Broder [18], [19] is adopted for this purpose (See also Algorithm 2).

Algorithm 1: M-Trail Design Problem for L-UFL

```
  Input: $G(V,E)$, $\alpha$
  begin
1     Set $b_{ini}$ as Theorem 2
      for $b := b_{ini}$ to $n-1$ do
2         Generate $b$ random trees of size $\alpha|V|$ with Alg. 2
3         Count $\chi_v$ the unique alarm codes seen at $\forall v \in V$
4         Count $\hat{\eta}_e$ the number of code conflicts for $\forall e \in E$
5         Sort the alarm codes in descending order of $\hat{\eta}_e$
          for $j := 1$ to $j_{max}$ do
              for iterate through the sorted links $e$ do
                  for $i := 1$ to $b$ do
6                     if $a^e_{<i>}$ gives no code conflict for $e$ then
7                         change link code of $e$ to $a^e_{<i>}$
              if every link has unique alarm code then
8                 return succeed
```

Algorithm 2: Aldous/Broder random tree generator

```
    Input: $G(V,E)$, $\alpha$
    begin
2.1     Start at a random node $v$.
        while the tree has less than $\alpha|V|$ nodes do
2.3         Choose a random neighbor $v^*$ of $v$.
2.4         if $v^*$ is not part of the tree then
                add edge $(v^*,v)$ to the tree.
2.5         $v := v^*$
```

These trees, denoted as $T' = [t_1, t_2, \ldots, t_b]$, are used to determine the initial assignment of alarm codes for every link e (denoted as a_e), where the alarm code a_e has the j-th bit as 1 if t_j traverses through e, and 0 otherwise.

We define a *collision* of two codes at node v if the codes are identical and used by at least two failure scenarios at a given node v. Let χ_v denote the number of failure scenarios minus the number of possible different codes seen at node $v \in V$ as a result of a single failure. If $\chi_v = 0$ we have an L-UFL solution at node n. Let $\chi = \sum_{\forall v \in V} \chi_v$. If $\chi = 0$ we have a valid NL-UFL solution. For each failure scenario z we define η_z which is the total number of nodes where z does not have a unique code. Similarly, for each link e we define $\hat{\eta}_e$ which is the sum of η_z for all failure scenarios z having link e. We call a failure scenario *detectable* at node v if it has a unique alarm code at node v. Similarly, we say a failure scenario is detectable if it has a unique alarm code at every node, i.e., $\eta_z = 0$. We say an alarm code is suitable for link e if $\hat{\eta}_e = 0$.

During the greedy random search our goal is to find suitable alarm codes for each link in the network. In each greedy step we try to remove all possible collisions by modifying the collided codes, where the code modification operations include adding and removing a link to and from an m-trail (also referred to as *bit-flipping*). This can greatly simplify the tracking of the consequences of modifications and eventually help minimizing the computation in each step toward the final result.

Let the bitwise pair at the i-th position of alarm code a_e be denoted as $a^e_{<i>}$, which is the code with all identical bits as a_e except for the i-th bit. For example, 011100 is the bitwise pair for the third position of 010100. Bit-flipping of link e at position i means its alarm code is changed from a_e to $a^e_{<i>}$. The following rules of thumb are adopted in the bit flipping process:

- Only incident links to the m-trail can be added.
- Only leaf links are allowed to be removed from an m-trail.
- If a leaf link with leaf node v is removed from an m-trail, then the node v should be on at least $\lceil \log_2(m+n+1) \rceil$ m-trails.

We say that a link e at position i is *flippable* if changing its alarm code from a_e to $a^e_{<i>}$ does not affect the code uniqueness of the other nodes' ACTs. Specifically, by taking the links in descending order of $\hat{\eta}_e$, until $\hat{\eta}_e > 0$, Step (7) attempts to remove the bit collision of e by checking each bit-flipping possibility iteratively upon each bit position $i = 1, \ldots, b$, in order to search for any flippable bit along the code of e. If such a code exists, the link code for e is changed to $a^e_{<i>}$ to resolve the code collision.

If there are no more flippable bits, the algorithm increases b until it finds a valid solution in Step (9). We maintain a tabu list to ensure that a code at a given position is not flipped twice. To avoid infinite loops, the algorithm stops if Step (6-8) is executed over $j_{max} = 500$ times; however, the heuristic always terminated with a valid solution at Step (8) in our evaluation.

To reduce computation time, an incremental update is performed on an internal data structure that stores whether a failure scenario has a conflicted code at a given node or not. The set of failure scenarios stored in the internal data structure contains every single link and every single node. The alarm codes for each failure scenario at each node are stored in a balanced binary search tree (e.g., `std::map` in C++), which provides fast lookup and modification procedures. When a bit i is swapped for link e, we need to update these trees at every node involved in the m-trail T_i. For the end nodes of e we may need to rebuild these trees, but for the rest of the nodes involved in T_i we just need to modify the alarm codes of the three failure scenarios with link e (e and the terminal nodes of e).

VI. Simulation Results

Simulations on some well-known network topologies taken from [20] were conducted. The performance metrics of interest are the number of m-trails, the normalized cover length of the solution (a measure of the cumulative bandwidth of the m-trails, formally defined in (17)) and the running time. Our primary goal is to analyze the performance of the proposed heuristic with the derived lower bounds on realistic network topologies.

We consider three failure scenarios: (a) single link failures (b) single node failures, and (c) single link and node failures. To localize single link failures (a) we implemented *RSTA+GLS* [1]. For (b) and (c) we launched the proposed heuristic with different sets of failure scenarios.

Table II summarizes our results. The number of nodes, links, and the diameter in hops of every topology graph is also shown in the first three columns of the table. The next two columns show the lower bound of the theorems in Section IV. It is followed by the columns on the smallest number of m-trails, denoted by b, obtained among 10 runs for each failure scenario. The normalized cover length over the number of links, denoted as $||\mathcal{T}||_E$ is also shown in the table for each failure scenario. $||\mathcal{T}||_E$ is a measure of the average number of monitoring wavelength channels (WLs) traversing each link, formally

$$||\mathcal{T}||_E = \frac{\sum_{i=1}^{b} |T_i|}{m} \ . \tag{17}$$

Note that the average values of $||\mathcal{T}||_E$ and b are shown on Fig. 4. Finally the average running time of the heuristics is shown.

We have observed that localizing a single node failure requires significantly more network resources in terms of cover length and the number of m-trails than the localizing a link failure. Nonetheless, it requires little additional network resources to localize a link failure besides a node failure, even if the number of links is much larger than the number of nodes in the network. This demonstrates that localizing a node failure requires significantly more resources than localizing a link failure.

We have also investigated the impact on the heuristic performance due to the assigned initial length of m-trails. We observed a trend similar to our theoretical analysis in Fig. 2, where the ideal size of m-trails was $0.7\,|V|$ for both CGT and realistic network topologies. This shows that the underlying CGT bound introduced in Section IV dominates the solution quality of the m-trail allocation problem.

Fig. 5(a) and (b) show the performance of the proposed heuristic algorithm by using randomly generated network topologies, aiming to gain some possible insight on performance impact due to topology density. We used the random graph generator [21] to generate planar 2-connected backbone networks; it first generates nodes randomly with a uniform distribution over the unit square then adds links with small physical lengths to keep the graph planar with each facet of an equal size.

By experimenting on 250 such random 50-node networks with different nodal degrees, we found that the consumed network resources by the heuristic are very high when the network nodal degrees are low (e.g., 2.5 - 3) and decrease rapidly as the networks are more densely connected. Nevertheless the decrease almost stops and the curves become flat when the nodal degrees became larger than 4, since the number of links is significantly increased as well.

Note that the lower bounds on the number of m-trails predicts 42%-64% of the obtained m-trail solutions, which may be because the bounds are based purely on the CGT problem and ignore the underlying graph structure.

TABLE II
RESULTS BY THE PROPOSED *RSTA+GLS* [1] FOR SINGLE LINK FAILURES ONLY, AND BY THE PROPOSED METHOD FOR SINGLE NODE AND SINGLE LINK OR NODE FAILURES ON SOME WELL-KNOWN NETWORKS.

Network [20]	Graph			Theorem		#m-trails			$\|\|\mathcal{T}\|\|_E$			Time [s]		
key on Fig 4	n	m	diam.	1	2	*Link*	*Node*	*Node&Link*	*Link*	*Node*	*Node&Link*	*Link*	*Node*	*Node&Link*
Pan-European +	16	22	6	7	8	7	12	13	2.43	5.4	6.6	0.39	0.7	1.9
German	17	26	6	7	8	8	12	13	2.46	4.8	6.6	0.51	1.6	3.4
ARPA	21	25	7	8	8	6	14	16	2.80	7.9	11.3	0.36	1.3	4.4
European □	22	45	5	8	9	11	14	14	2.56	4.8	6.0	2.10	6.9	10.3
USA ■	26	42	8	8	9	9	15	16	2.72	7.0	8.0	2.16	6.0	10.4
Nobel EU ○	28	41	8	8	10	7	16	16	3.02	7.9	8.8	0.87	5.3	11.7
Italian ●	33	56	9	9	9	10	19	19	2.93	8.3	10.1	4.83	15.9	44.3
Cost 266	37	57	8	9	9	8	17	17	3.00	8.0	8.9	2.04	18.1	32.3
North Amer.	39	61	10	9	9	8	16	18	3.09	7.8	9.1	2.31	27.9	45.1
NSFNET ◇	79	108	16	11	11	9	23	26	3.51	13.1	15.7	6.05	129	289.61

The average number of WLs required for failure localization is $\sim$10, which may sound expensive. However, the latest technology available on the market for optical FlexGrid transmission technology allows switching at 6.25 GHz channel granularity at reconfigurable optical add-drop multiplexers (ROADMs). This allows cheap launching of any lightpaths with small bandwidth in the network, and makes real-time monitoring systems cost-efficient. For example, allocating 60–80Ghz in each optical fiber in the 1530–1560 nm range[5] occupies just $1.5-2\%$ of the total bandwidth, while allowing to launch up to 10-15 supervisory lightpaths for network monitoring. Further, the WLs taken by the m-trails could be reused as spare capacity for shared protection; this approach is referred to as the *monitoring resource hidden property* [14], where the consumed monitoring resources can be significantly reduced.

Finally, the computation efficiency of the proposed heuristics is examined. The heuristic should maintain a different ACT for each node, which can be seen in the increase of the computation time compared to *RSTA+GLS* where only a single ACT is maintained. Nevertheless, the largest network was solved in 5 minutes, which is a reasonable performance for a network planning tool.

To summarize the simulation results above, the proposed heuristic achieves the desired computation efficiency and performance in handling realistic networks, and its feasibility in the operation of future all-optical backbone is proved for achieving NL-UFL under single node and link failures using bi-directional m-trails.

VII. CONCLUSIONS

The paper studied the monitoring trail (m-trail) allocation problem for network-wide unambiguous failure localization (NL-UFL) for single link and node failures. In particular, we developed theoretical results based on combinatorial group testing (CGT) that can give some analytical bounds for the formulated problem. To solve the problem in realistic networks, a novel heuristic was developed. Simulation was conducted to examine the performance of the proposed heuristic and to analyze the formulated problem. Our conclusions are the following: (1) The considered NL-UFL problem can be modeled by using the Ahlswede-Katona theory which leads to a general CGT scenario. (2) The lower bound was obtained via a novel construction using Gray code, which was shown to be tight, i.e., within a small factor of about 1.23. (3) We found that the number of m-trails required to localize both node and link failures is only slightly larger than that necessary to localize node failures alone, and could significantly larger than the number required to localize single link failures only. (4) Simulation results verified the proposed heuristic and demonstrated the performance behavior of the considered problem in terms of the required monitoring resources, the number of m-trails, and the computation time.

[5]It is at least 4000Ghz.

REFERENCES

[1] J. Tapolcai, P.-H. Ho, L. Rónyai, and B. Wu, "Network-wide local unambiguous failure localization (NWL-UFL) via monitoring trails," *IEEE/ACM Transactions on Networking*, 2012.

[2] E. Mannie and D. Papadimitriou, "Rfc 4427 - recovery (protection and restoration) terminology for generalized multi-protocol label switching (GMPLS)," 2006.

[3] H. Zeng, C. Huang, and A. Vukovic, "A Novel Fault Detection and Localization Scheme for Mesh All-optical Networks Based on Monitoring-cycles," *Photonic Network Communications*, vol. 11, no. 3, pp. 277–286, 2006.

[4] C. Li, R. Ramaswami, I. Center, and Y. Heights, "Automatic fault detection, isolation, and recovery in transparentall-optical networks," *IEEE/OSA J. Lightwave Technol.*, vol. 15, no. 10, pp. 1784–1793, 1997.

[5] Y. Wen, V. Chan, and L. Zheng, "Efficient fault-diagnosis algorithms for all-optical WDM networks with probabilistic link failures," *IEEE/OSA J. Lightwave Technol.*, vol. 23, pp. 3358–3371, 2005.

[6] C. Assi, Y. Ye, A. Shami, S. Dixit, and M. Ali, "A hybrid distributed fault-management protocol for combating single-fiber failures in mesh based DWDM optical networks," in *Proc. IEEE GLOBECOM*, 2002, pp. 2676–2680.

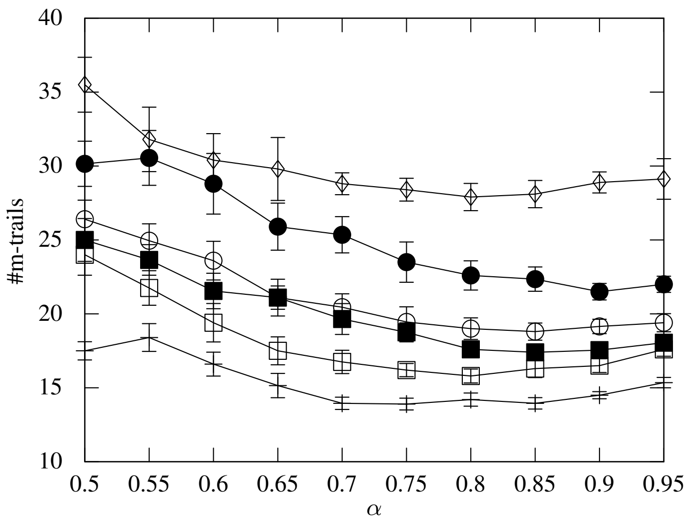
[7] B. Wu, P.-H. Ho, and K. Yeung, "Monitoring trail: On fast link failure localization in all-optical WDM mesh networks," *IEEE/OSA J. Lightwave Technol.*, vol. 27, no. 18, pp. 4175–4185, 2009.

[8] J. Tapolcai, B. Wu, P.-H. Ho, and L. Rónyai, "A novel approach for failure localization in all-optical mesh networks," *IEEE/ACM Trans. Networking*, vol. 19, no. 1, pp. 275 –285, feb 2011.

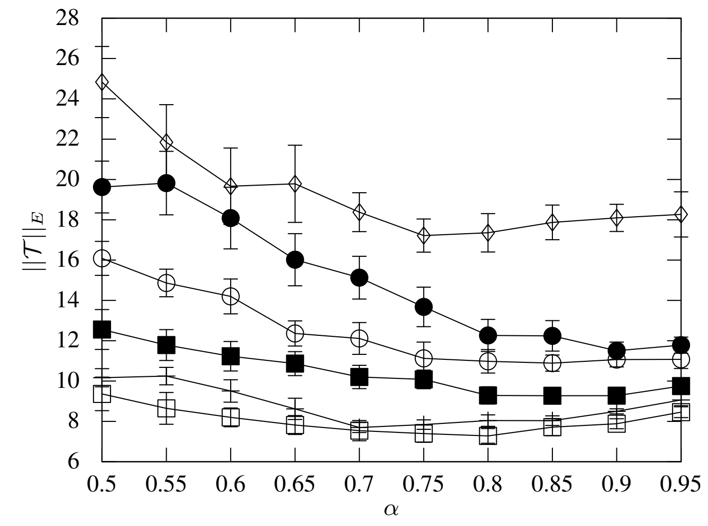
[9] N. Harvey, M. Patrascu, Y. Wen, S. Yekhanin, and V. Chan, "Non-Adaptive Fault Diagnosis for All-Optical Networks via Combinatorial Group Testing on Graphs," in *Proc. IEEE INFOCOM*, 2007, pp. 697–705.

[10] S. Ahuja, S. Ramasubramanian, and M. Krunz, "Single link failure detection in all-optical networks using monitoring cycles and paths," *IEEE/ACM Trans. Networking*, vol. 17, no. 4, pp. 1080–1093, 2009.

[11] B. Wu, P.-H. Ho, J. Tapolcai, and X. Jiang, "A novel framework of fast and unambiguous link failure localization via monitoring trails," in *IEEE INFOCOM WIP*, San Diego, 2010, pp. 1–5.

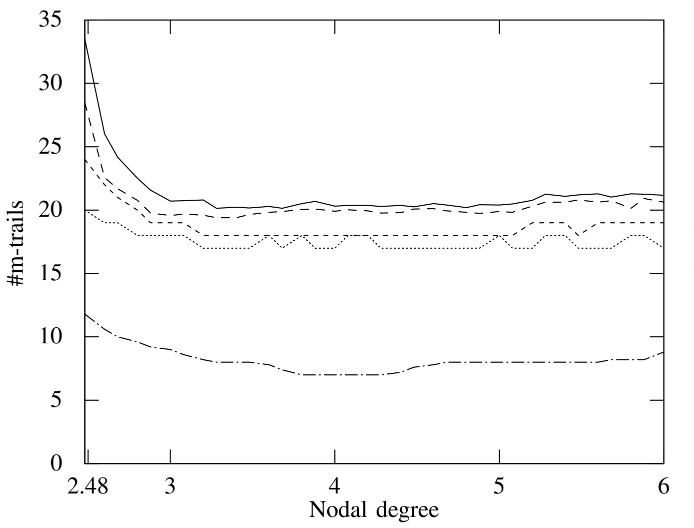


(a) The average number of m-trails for different α parameter values.

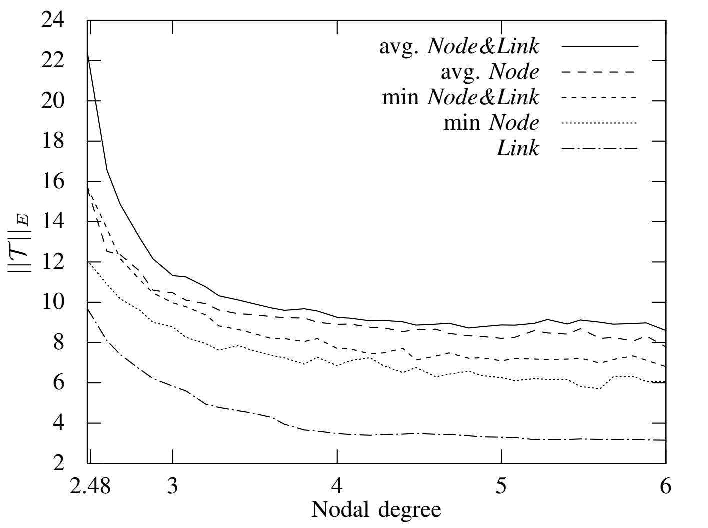


(b) The average normalized cover length for different α parameter values.

Fig. 4. The effect of parameter α on the performance of the algorithm. α defines the initial size of randomly generated m-trails. For each networks of Table II and α setting the algorithm was launched 20 times, and the 95% confidence intervals are the bars of each point on the figure.



(a) The average number of m-trails for random networks.



(b) The average normalized cover length for random networks.

Fig. 5. The effect of topology diversity on the performance of the algorithm. 250 random 50-node networks were generated with different nodal degree. For each networks the algorithm was launched 5 times, and the 95% confidence intervals are the bars of each point on the figure.

[12] W. He, P.-H. Ho, B. Wu, and J. Tapolcai, "On identifying SRLG failures in all-optical networks," *Elservier Journal on Optical Switching and Networking (OSN)*, vol. 10, no. 1, pp. 77 – 88, jan 2013.

[13] M. Ali, P.-H. Ho, J. Tapolcai, and B. Shihada, "M-burst: A framework of SRLG failure localization in all-optical networks," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 4, no. 8, pp. 628–638, 2012.

[14] J. Tapolcai, P.-H. Ho, P. Babarczi, and L. Rónyai, "On signaling-free failure dependent restoration in all-optical mesh networks," *IEEE/ACM Transactions on Networking*, 2013.

[15] ——, "On achieving All-Optical failure restoration via monitoring trails," in *Proc. IEEE INFOCOM Mini-Symposium*, Turin, Italy, Apr. 2013.

[16] G. Katona, "On separating systems of a finite set," *Journal of Combinatorial Theory*, vol. 1, no. 2, pp. 174–194, 1966.

[17] R. Ahlswede, "Ratewise-optimal non-sequential search strategies under constraints on the tests," *Discrete Applied Mathematics*, vol. 156, no. 9, pp. 1431–1443, 2008.

[18] D. Aldous, "The random walk construction of uniform spanning trees and uniform labelled trees," *SIAM Journal on Discrete Mathematics*, vol. 3, no. 4, pp. 450–465, 1990.

[19] A. Broder, "Generating random spanning trees," in *Annual Symposium on Foundations of Computer Science*. IEEE Computer Society, 1989, pp. 442–447.

[20] S. Orlowski, M. Pióro, A. Tomaszewski, and R. Wessäly, "SNDlib 1.0–Survivable Network Design Library," in *Proc. Int. Network Optimization Conference (INOC)*, April 2007.

[21] "LEMON: A C++ Library for Efficient Modeling and Optimization in Networks," http://lemon.cs.elte.hu.