On Integrating Failure Localization with Survivable Design

by

Wei He

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2013

© Wei He 2013

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

In this thesis, I proposed a novel framework of all-optical failure restoration which jointly determines network monitoring plane and spare capacity allocation in the presence of either static or dynamic traffic. The proposed framework aims to enable a general shared protection scheme to achieve near optimal capacity efficiency as in Failure Dependent Protection(FDP) while subject to an ultra-fast, all-optical, and deterministic failure restoration process. Simply put, Local Unambiguous Failure Localization(L-UFL) and FDP are the two building blocks for the proposed restoration framework.

Under L-UFL, by properly allocating a set of Monitoring Trails (m-trails), a set of nodes can unambiguously identify every possible Shared Risk Link Group (SRLG) failure merely based on its locally collected Loss of Light(LOL) signals. Two heuristics are proposed to solve L-UFL, one of which exclusively deploys Supervisory Lightpaths (S-LPs) while the other jointly considers S-LPs and Working Lightpaths (W-LPs) for suppressing monitoring resource consumption. Thanks to the "Enhanced Min Wavelength Max Information principle", an entropy based utility function, m-trail global-sharing and other techniques, the proposed heuristics exhibit satisfactory performance in minimizing the number of m-trails, Wavelength Channel(WL) consumption and the running time of the algorithm.

Based on the heuristics for L-UFL, two algorithms, namely MPJD and DJH, are proposed for the novel signaling-free restoration framework to deal with static and dynamic traffic respectively. MPJD is developed to determine the Protection Lightpaths (P-LPs) and m-trails given the pre-computed W-LPs while DJH jointly implements a generic dynamic survivable routing scheme based on FDP with an m-trail deployment scheme. For both algorithms, m-trail deployment is guided by the Necessary Monitoring Requirement (NMR) defined at each node for achieving signaling-free restoration. Extensive simulation is conducted to verify the performance of the proposed heuristics in terms of WL consumption, number of m-trails, monitoring requirement, blocking probability and running time.

In conclusion, the proposed restoration framework can achieve all-optical and signaling-free restoration with the help of L-UFL, while maintaining high capacity efficiency as in FDP based survivable routing. The proposed heuristics achieve satisfactory performance as verified by the simulation results.

Acknowledgements

Firstly, I would like to thank Prof. Pin-Han Ho, my supervisor, who consistently suggests innovative ideas, provides useful comments and gives deep insights into my research topic, patiently while professionally clarifies my confusions, and helps me address various issues during my PhD program.

Secondly, I would like to thank Prof. Raouf Boutaba, Prof. Paul A.S. Ward, Prof. Otman Basir and Prof. Changcheng Huang who kindly serve as my PhD committee members. Thanks Prof. Patrick Lam for attending my defence and giving insightful comments. Special thanks goes to Prof. Raouf Boutaba and Prof. Paul A.S. Ward for providing constructive suggestions regarding my research and thesis. I should also thank Prof. Bernard Wong and Prof. David Taylor who attended my PhD research seminars and provided useful suggestions and comments; thank Ms. Margaret Towell and Ms. Paula Zister, who patiently and kindly helped me during my graduate studies. In addition, I feel grateful to all professors who have taught me at University of Waterloo.

I would also like to thank my co-authers: Dr. Bin Wu and Dr. Jànos Tapolcai. Without your cooperation, I would not be able to publish so many high-quality papers.

Special thanks should be given to Prof. Hongping Tan for your kind help and care when I first arrived in Canada. I would also like to thank Prof. Hui Li, Prof. Tianyi Zhang, Prof. Yuesheng Zhu (Peking University), Prof. Zhiying Zhao and Prof. Xin Wang (Fudan University), who recommended me to study in Canada.

Lastly, I wound like to express my gratefulness to all my my friends and everyone I know. Especially, Jiewen Wu, Lin He, Xi Han, Zhenhuan Wei, Shujiao Wang, Ming Gong, Jiajia Han, Tiantian Bian, Nan Hu, Qi Zhang, Qu Chen, Le Li, Xiang Gao and Cong Guo. Without your help, I would not be able to make this possible.

Dedication

This is dedicated to my parents for their love, support and understanding.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Network Survivability

With the advancement of telecommunication technologies, networks are playing an increasingly important role in our daily life. Apart from traditional applications such as emails and web browsing, numerous network-based applications have mushroomed in recent years, such as video chatting using Skype, online games, social networking using Facebook and Twitter, Document Processing with Google Doc, Online banking and shopping, interactive map searching and even online Anti-Virus services. To enjoy all these conveniences and meet the diversified Quality of Service(QoS) requirements of global users, optical networks such as Synchronous Optical Networking (SONET)[14] and Wavelength Division Multiplexing (WDM) networks, have been widely deployed (see Fig. 1.1) for their high speed, huge capacity and less signal interference.

At the same time, network failures[9](node failure, link failure and software failure) did occur for various reasons(imperfection of software systems, digging accident, earthquakes, flood, cables chewed by animals etc.) and cannot be completely eliminated[16, 19, 3]. According to [68], fiber cut rate is about 4.39/year/1000 sheath miles. In 2009, typhoon Morakot caused significant cable damage from Kaoping Canyon downstream all the way to the Manila Trench[28]. In 2012, U.S. level of network outrages (shown by system logs) doubled at about 0.4% after superstorm Sandy and took about 4 days to recover[26].

Generally speaking, node failure refers to the breakdown of network devices located at the network nodes, such as Optical Cross Connects(OXCs), Optical Line Systems(OLSs), or routers. Link failure is usually due to fiber cut, in which case, some network link(s)

Figure 1.1: Optic-fiber deployment[59](Countries without Optics darkly shaded)

are down. [17] showed that 84% of critical link failures are due to optical layer problems based on IP link failure logs collected from Sprint network between February 1, and June 30, 2003. [39] further demonstrated that 70% of the unplanned failures are individual link failures caused by a variety of problems, through monitoring the Sprint network from April to October, 2002. Software failures are resulted from the malfunctioning of software systems such as misconfiguration or buggy design of protocols.

In optical networks, even a short service disruption due to a single fiber cut can affect a large number of network users, lead to tremendous data loss and economic loss. In fact, the WDM technology further aggravates data loss, since it improves the transmission capacity of optical networks by several magnitudes through splitting a single fiber into multiple non-overlapping channels [50]. By the year 2010, 100G router ports together with 100G optical transport interface have been commercialized[71] to meet the soaring user demand(See Fig. 1.2). As regard to economic loss, data center failures cost an average of $5,600 per minute according to a Ponemon Institute study[20]. Table 1.1 lists other downtime costs given in [47].

On the other hand, average fiber repair time is 12 hours[68], submarine cable repair may even take 10 days according to [28] while most IP networks require sub-second recovery [48] and some services have very stringent recovery time requirement (e.g. 50ms for voice traffic)[16]. Consequently, **a cost-effective survivability scheme should be applied, before the failures can be fully repaired, as a first aid to maintain service continuity of end users, minimize data loss and economic loss, and qualify the**

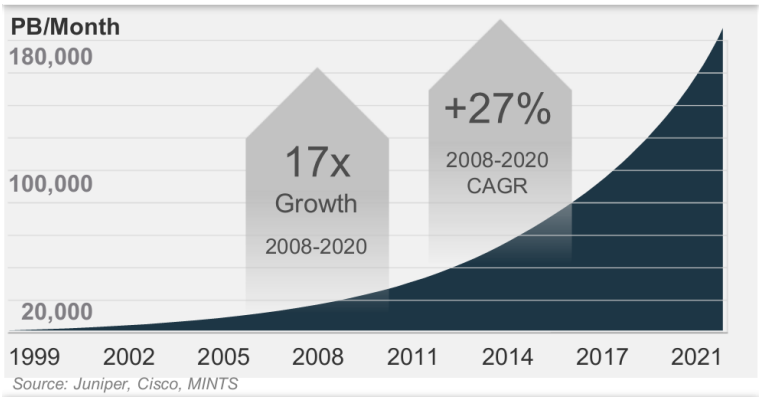


Figure 1.2: Explosive Bandwidth Growth[70](CAGR: compound annual growth rate)

network as *survivable* against network failures. In this thesis, *I focus on the optical survivability schemes for link failure.*

1.2 Optical Survivability Schemes

In WDM networks, each connection is an end-to-end lightpath usually established on a specific wavelength. The traffic demand for a network is usually expressed by a set of end-to-end connection requests. For each connection request, a lightpath called working lightpath (W-LP) is set up to transmit data. In order to deal with network failure, at

Table 1.1: Downtime Costs

	Downtime Costs (per Hour)
Brokerage operations	$6,450,000
Credit card authorization	$2,600,000
Ebay	$225,000
Amazon.com	$180,000
Package shipping services	$150,000
Home shopping channel	$113,000
Catalog sales center	$90,000
Airline reservation center	$89,000
Cellular service activation	$41,000
On-line network fees	$25,000
ATM service fees	$14,000

least one protection lightpath (P-LP) is used to protect a specific segment of the working path for that connection request. With reference to the Generalized Multi-Protocol Label Switching (GMPLS) protocol, a typical *network restoration* involves several steps: *failure localization, failure notification* and *device configuration*. Specifically, failure localization and failure notification are referred to as *network monitoring*. In a typical restoration process, after a failure is detected by an upstream node through network monitoring, a protection path will merge back to the working path at a preselected downstream node called merging node[40] to recover the data traffic.

According to [15], "Survivability is a system's capability to fulfill its mission, in a timely manner, in presence of attacks, failures, or accidents". Schemes which can achieve survivability are referred to as *survivability schemes*. Optical survivability schemes can be classified [54] by computation timing as pro-active/reactive (protection/restoration, pre-planned restoration/real-time searching [42]), based on the scope of backup path as being path-/link- based [40], by dependence on failure location as failure-dependent and failure-independant[53], and capacity sharing as dedicated/shared.

Among various design objectives to achieve survivability, *hardware cost*, *capacity efficiency* and *fault management complexity* are widely considered. *Hardware cost* refers to the costs brought by the installation and maintenance for optical devices, such as optical monitors, wavelength converters, add-drop multiplexers. *Capacity efficiency* concerns the spare capacity consumption, which is the capacity in terms of wavelength channels (WLs) reserved but not necessarily configured for recovery. *Fault management complexity* is reflected by the total restoration time, memory consumption at the network nodes and the signaling complexity.

1.3 Challenges

In reality, multiple network layers may have their individual survivability schemes with different restoration granularity[53]. For instance, wavelength-level restoration can be performed at the optical layer, Label-switched Path(LSP) level at layer 2.5 by MPLS Fast Reroute[55], IP-route level at layer 3 by IP Fast Reroute(IP-FRR)[36, 48] or other routing protocols as BGP, OSPF etc. Generally speaking, a higher layer survivability scheme operates on a finer-granularity but is time-consuming if the restoration is just performed at that layer. In contrast, a lower layer scheme works on a coarser granularity while usually faster[53, 34].

Without an efficient coordination and communication mechanism between different layers, a single failure event may trigger concurrent restoration at different layers, consume

a huge amount of network resources and computing powers, and may even lead to service disruption[34, 16, 32]; meanwhile, network failure would be reported inefficiently at each layer, causing "alarm flooding"[77]. Naturally, it is expected to restore the network in an efficient way such that unnecessary restoration actions can be prevented and the interference of restoration actions at different layers can be minimized[34].

One commonly adopted approach is to isolate the failure restoration within the optical domain by deploying optical monitors, whose alarms are collected by the control plane in a timely manner when failure occurs. In the best case, network failures can be handled instantly by the optical layer and completely hidden from upper layers. Various optical layer monitoring structures such as simple and non-simple m-cycle, m-path, and m-trail [77] have been proposed for unambiguous failure localization(UFL). All those schemes adopt an all-optically pre-cross-connected supervisory lightpath to transmit an optical supervisory signal. A dedicated monitor is equipped at the end node of the supervisory lightpath to check the on-off status of the supervisory signal. If there is any link failure on the supervisory lightpath, the monitor will detect a Loss of Light (LOL) and then generate an alarm, which is denoted by using a binary bit of 1 (0 for no alarm). Thus, by properly allocating a set of supervisory lightpaths, each failure event can be identified by its unique alarm code associated with the status of these supervisory lightpaths.

Among all the monitoring structures, m-trail is the most flexible one, because it can be either an open or a closed supervisory lightpath, and can be most freely routed in the network as long as it traverses any directed link at most once. It is shown that all other monitoring structures can be taken as special cases of m-trails, and using m-trails can always ensure unambiguous link failure localization with the minimum monitoring resources (i.e., supervisory wavelength-links and monitors).

The second issue to be addressed is "electronic alarm dissemination". Under the traditional UFL framework, the set of m-trails deployed for monitoring may terminate at different nodes, electronic alarm dissemination/flooding has to be performed at the end node of each m-trail, such that a remote routing entity can collect the flooded (electronic) alarm signals (i.e., binary bits) to form a complete alarm code for UFL. This makes the scheme unqualified as an all-optical monitoring scheme. As the network size grows, such a scheme can hardly meet the rigid time constraint for optical recovery. To eliminate alarm dissemination and speed up localization, the Local Unambiguous Failure Localization(L-UFL)[76] framework was proposed which enables a set of nodes to independently achieve UFL by tapping the m-trails passing through them. Unfortunately, the Integer Linear Program(ILP) proposed[76] for L-UFL only deals with single link failure, is time-consuming and could hardly reach optimality. Thus, in Chapter 4 , we proposed two novel heuristics to solve the m-trail allocation problems for achieving L-UFL.

Another issue is related to the selection of survivability schemes. There is a compromise between capacity efficiency and fault management complexity in the spectrum of shared protection, in which Failure Dependent Protection (FDP) and pre-configured protection (such as p-Cycle) are the two extremes. With FDP, each connection is assigned with multiple P-LPs, and one is activated for the restoration purpose according to the identified failure event. By allowing the reuse of the released working capacity for failure restoration (or called *stub release*), FDP has long been recognized with the optimal capacity efficiency and widely taken as the performance benchmark in the design of shared protection schemes. However, the FDP restoration process is subject to the highest control and signaling complexity that possibly yields the longest restoration time, mostly because the switching node has to precisely localize the failure event for real-time selection of a P-LP.

The other extreme is the pre-configured schemes that require minimum signaling, such as 1+1 and p-Cycle. They require the minimum fault management complexity and thus achieves very fast restoration since the only after-failure real-time action is the reconfiguration of the two nodes responsible for switching and merging the affected working traffic. Thus in contrast to any other counterpart, the pre-configured schemes can minimize the fault management complexity and possibly be implemented completely in the optical domain. Such simplicity and fast restoration speed are nonetheless at the expense of consuming the highest redundancy.

The above observation naturally brings up an interesting question: *Can we restore an optical layer fault without relying on any electronic signaling, while achieving optimal capacity efficiency as with FDP?* It is answered in Chapter 5 by introducing a novel framework of all-optical and signaling-free restoration that can work in parallel with the existing GMPLS based recovery process.

1.4 Our Approach and Contribution

To meet the afore-mentioned challenges, we proposed a signaling-free restoration framework incorporating a state-of-the-art optical layer monitoring system with the conventional spare capacity allocation task, in which a set of supervisory lightpaths(m-trails) is deployed and allowed to reuse the spare capacity reserved for P-LPs. As a consequence, each node can instantly obtain the network failure status and start the predefined restoration process by monitoring the on-off status of the traversing m-trails. To achieve the claimed signaling-free restoration, only a subset of SRLG pairs need to be differentiated at the network nodes, which are novelly defined as the Necessary Monitoring Requirements(NMR) to guide the

m-trail deployment. Simply put, L-UFL and FDP are the two most important building blocks for the proposed restoration framework.

In particular, a novel static survivable routing scheme, namely Monitoring and Protection Joint Design Heuristic (MPJD)[22], together with a dynamic survivable routing scheme, namely Dynamic Joint design Heuristic (DJH)[27], are proposed for the signaling-free restoration framework. MPJD is featured as a failure dependent protection (FDP) scheme under static traffic patterns (i.e. W-LPs are prerouted for meeting the connection requests). MPJD is capable of jointly allocating P-LPs for protecting the given W-LPs, as well as m-trails, such that all W-LPs can be restored in an all-optical fashion. By contrast, DJH is an FDP scheme under dynamic traffic patterns, which can jointly allocate a W-LP and its P-LPs that satisfies an newly arrived connection request, as well as the m-trails to be newly added to the network, such that the W-LP can be restored all-optically. Chapter 5 details these two heuristics.

To efficiently achieve L-UFL for the signaling-free restoration framework, we proposed a heuristic to solve L-UFL under SRLG failures with m-trails in [25]. Armed with some advanced techniques such as "Min Wavelength Max Information Principle" and m-trail global sharing, the proposed heuristic demonstrates satisfactory performance and verified through simulations. To further save monitoring resources for L-UFL, we proposed another heuristic [23] characterized by jointly considering the working lighpaths and m-trails to achieve L-UFL. In particular, this joint monitoring heuristic allocates W-LPs and a set of m-trails by a utility function in tandem so as to achieve unambiguous failure localization. The built-in utility function, which predicts the final solution quality from its current partial solution, further helps reduce the resource consumption. Chapter 4 details the above two algorithms and verifies their performance through extensive simulations.

Here is a list of papers published or submitted during my PhD program:

1. Wei He, Pin-Han Ho, Bin Wu, and J. Tapolcai. On integrating failure localization with network survivable design. In *IEEE ICC 2013*, accepted 2013.

2. Wei He, Pin-Han Ho, Bin Wu, and J. Tapolcai. On identifying SRLG failures in all-optical networks. *Optical Switching and Networking*, 10(1):77-88, 2013.

3. Wei He, Bin Wu, Pin-Han Ho, and J. Tapolcai. Monitoring trail allocation for fast link failure localization without electronic alarm dissemination. In *Optical Network Design and Modeling (ONDM)*, 2011 15th International Conference on, pages 1-6, feb. 2011.

4. Wei He, Bin Wu, Pin-Han Ho, and J. Tapolcai. Monitoring trail allocation for SRLG failure localization. In *Global Telecommunications Conference (GLOBECOM 2011)*, 2011 IEEE, pages 1-5, Dec.

5. Pin-Han Ho, Wei He, J. Tapolcai, and Bin Wu. A framework of dynamic survivable routing in all-optical networks. *Journal of Optical Communications and Networking*, submitted.

1.5 Organization of the thesis

The rest of the thesis is organized as follows. Chapter 2 and 3 explain the basic concepts and techniques for network monitoring and survivability schemes respectively. Concepts and heuristics related to L-UFL are introduced in Chapter 4 which will be used as building blocks for the novel signaling-free restoration framework elaborated in Chapter 5. Chapter 6 concludes.

Chapter 2

Background of Optical Network Monitoring

Network monitoring is one of the most important steps in the restoration process and provides critical information to trigger further actions to prevent data loss. Usually, a monitoring scheme considers a set of pre-defined shared risk link groups (SRLGs) on a given network topology and it infers the network state by reading the Loss of Light status of monitoring structures. First, the network model for Wavelength Division Multiplexing (WDM) network is explained in Section 2.1. As a building block for network monitoring, various kinds of monitoring structures are introduced in Section 2.2. Next, the Unambiguous Failure Localization problem is explained in Section 2.3. Lastly, literature review on network monitoring will be addressed in Section 2.4.

2.1 Wavelength-routed WDM networks

As Fig. 2.1a shows, a WDM network can be represented by a network graph $G(V, E)$ where the vertex set V corresponds to the set of network *nodes* and the edge set E represent the (logical) *links* interconnecting these nodes. A node usually employs Optical Cross-Connects (OXCs) to switch a wavelength from one link to another(see Fig. 2.1b). A link can be composed of multiple fibers, each of which contain multiple Wavelength Channels (WLs)(see Fig. 2.1c). A *lightpath* is defined as a node to node connection, established on a particular wavelength(see Fig. 2.1a). As wavelength-converter is quite expensive, lightpath establishment must obey the wavelength-continuity constraint when no wavelength-conversion is

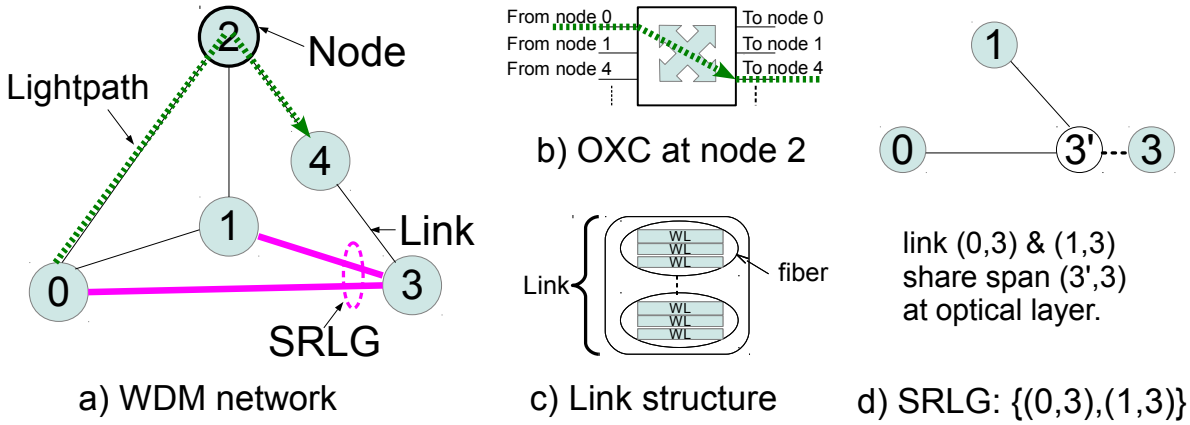


Figure 2.1: WDM network basics

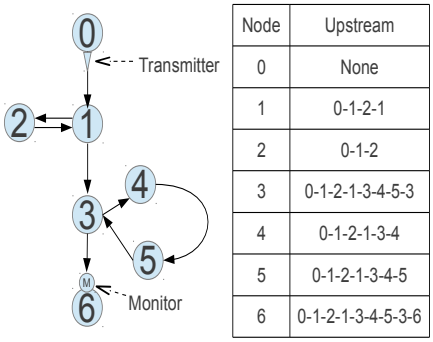
available: each link along the path must use the same wavelength. An Shared Risk Link Group (SRLG) refers to a group of links which may *fail simultaneously* due to a *common physical attribute*. For instance, link (0,3) and (1,3) form an SRLG as depicted by Fig. 2.1a, since these two links share a common fiber span $(3', 3)$ at the optical layer as Fig. 2.1d illustrates.

2.2 Monitoring Structures

Definition 1. *A Monitoring Structure (m-structure) is referred to as a monitoring subsystem in which light signal traverses through every link in a connected network subgraph.*

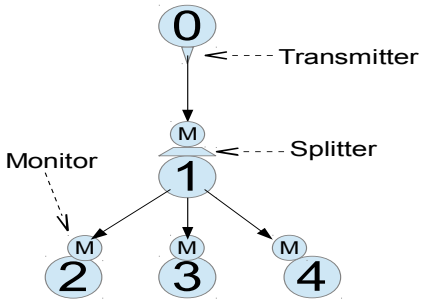
Usually, the structure is built upon transmitter(s), monitor(s) and other optical components. Once the internal light signal is interrupted due to an failure, the monitor(s) of the structure will sent out alarm(s) to help identify the failure source. Fig. 2.2 shows two types of m-structures: Monitoring Trail (m-trail)[65] and Monitoring Tree (m-tree)[12].

An m-trail is referred to as a lightpath that traverses any edge in each direction at most once while a node possibly for multiple times. In case the m-trail lightpath is a simple path or a cycle, it is also called Monitoring Path (m-path)[4] and Monitoring Cycle (m-cycle)[81] respectively. Clearly, m-trail generalizes all these m-structures. With regard to m-tree (see Fig. 2.2b for example), a single laser diode is installed at a node to transmit uni-directional supervisory signal over a single link and multiple monitors are installed. When hitting a node, the signal can be terminated, forwarded through a single outgoing link, or duplicated



Node	Upstream
0	None
1	0-1-2-1
2	0-1-2
3	0-1-2-1-3-4-5-3
4	0-1-2-1-3-4
5	0-1-2-1-3-4-5
6	0-1-2-1-3-4-5-3-6

(a) m-trail



(b) m-tree

Figure 2.2: Monitoring Structures

and sent to multiple outgoing links. As numerous works and our current research focus on using m-trail or its special case to achieve failure localization, we will elaborate on the m-trail concept in this section.

Usually, two types of lightpaths can be used by an m-trail: Supervisory Lightpath (S-LP) and Working Lightpath (W-LP). An S-LP is defined to be a lightpath used merely for network monitoring and carries supervisory light signal. On the other hand, a W-LP only carries data traffic by default. With the help of an end-to-end protocol or other techniques, W-LPs can also perform monitoring tasks.

An m-trail forms a subgraph in the network topology, in which a lightpath is launched at the source node and traverses its on-trail nodes in a fixed order. Such an order of links along the m-trail is based on a *pre-cross-connected pattern* of the lightpath, so that any on-trail node can sense a failure event affecting its *upstream* link(s) along the m-trail. Fig. 2.2a shows an m-trail routed from node 0 to node 6 and lists the upstream information for each on-trail node. The source node has to allocate a transmitter (laser diode) to support the lightpath for the m-trail and the destination node is equipped with a monitor to check the on-off status of the whole lightpath. Optionally, each intermediate node can also install a monitor to tap the optical signal that is locally available at the node.

Note that the *source* and *destination* of an m-trail can be located at a common node when the m-trail is in a shape of cycle. In that case, an on-trail node can obtain the failure status of the whole lightpath and every on-trail link is *defined to be in upstream* for every on-trail node. This is achieved by assuming a notification mechanism from the monitor to the transmitter at a common node, which turns off the transmitter once the monitor

(a) (b)

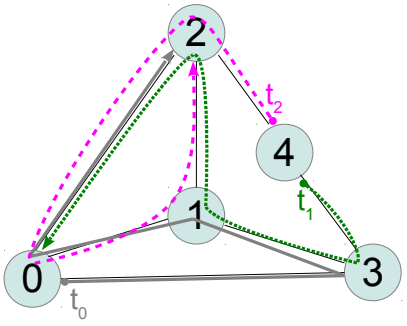
Figure 2.3: cycle notification mechanism (□ denotes a node; monitor & transmitter in one common node)

senses a *loss of light (LOL)*. For example, the light signal is transmitted normally along the cycle, as shown in Fig. 2.3a. At some moment, link (a, b) fails (see Fig. 2.3b), path segment $a - b - s$ will lose light. Then the monitor detects LOL and asks the transmitter to turn off the signal. After that, all on-trail nodes detect the failure due to LOL of the whole trail.

2.3 Unambiguous Failure Localization

Definition 2. *Given a network graph $G(V, E)$ where V and E represent the set of nodes and links in the graph respectively. Let $\Phi = \{\phi_0, \phi_1 \ldots\}$ denote the set of Shared Risk Link Groups (SRLGs) under consideration and assume that exactly one of the SRLGs can fail during the failure localization process. The Unambiguous Failure Localization (UFL) problem asks to uniquely identify any single SRLG failure by analyzing the light signal status of a set of monitoring structures.*

Fig. 2.4 illustrates how to achieve UFL by deploying m-trails. For the network topology shown in Fig. 2.4a, three m-trail t_0, t_1 and t_2 are deployed and 8 SRLGs are considered. As Fig. 2.4b shows, a UFL solution can be represented by an Alarm Code Table(ACT); ϕ_0 represents the normal case(i.e. no failure occurs) while ϕ_1 to ϕ_7 correspond to each link failure respectively. Each row in an ACT corresponds to a specific SRLG failure and shows the combination of m-trails which will send out alarms when that failure occurs. Each binary bit in an alarm code corresponds to an m-trail, using 1 to indicate that an alarm would be sent out by that m-trail. For example, the alarm code of ϕ_4 (i.e. link (1,2)) is "011"(see Fig. 2.4b) which means that when link (1,2) fails, the m-trails t_1 and t_2 will be disrupted and send out alarms while t_0 is intact. Thus, it can be verified that UFL is achieved with the deployed m-trails(see Fig. 2.4a), since each failure event under consideration would trigger a unique set of m-trails to send out alarms (i.e. unique alarm codes in Fig. 2.4b).



(a) M-trail Monitoring

SRLG	$t_0t_1t_2$
ϕ_0(normal)	000
$\phi_1(0,1)$	101
$\phi_2(0,2)$	111
$\phi_3(0,3)$	100
$\phi_4(1,2)$	011
$\phi_5(1,3)$	110
$\phi_6(2,4)$	001
$\phi_7(3,4)$	010

(b) Alarm Code Table (ACT)

Figure 2.4: UFL example with m-trails

The UFL problem can be viewed as a Combinatorial Group Testing (CGT)[13] problem with additional constraints. First, we briefly introduce the CGT model. Suppose there are n *items* each can be either *positive* or *negative* and the problem is to identify all positive items. All combinations of positive items under consideration compose the *sample space*. For example, if the number of positive items is upper bounded by an integer d, then all combinations of upto d items form the sample space. This model is referred to as the $(n, \bar{d})$ model while the (n, d) model is used when there are exactly d positive items. By conducting a set of *group tests*, the positive item combination can be uniquely identified. Each test is applicable to an arbitrary subset of items and has two possible *outcomes*. A negative outcome indicates all items in the subset are negative while a positive outcome shows otherwise. The goal is to minimize the number of tests.

Now consider the UFL problem. Each network link is regarded as an item and the SRLGs under consideration form the sample space in CGT. Each CGT test is conducted by launching a specific m-structure and test outcome is positive if Loss of Light is detected and negative otherwise. Unlike in CGT, where the items in a test can be chosen freely, the items(links) in a UFL test must form a connected subgraph. Just like CGT, by properly coding the SRLGs, any single SRLG failure can be uniquely identified.

Based on the above observation, we can use the terms in CGT to classify the UFL schemes based on how tests are conducted. A *sequential* scheme runs the test one by one and allows a later test to use the outcomes of all previous tests. A *nonadaptive* scheme conducts all tests simultaneously, thus forbidding using the outcome of one test to design another test. Between the two schemes, there are the $s-stage$ scheme for which all tests

in a stage must be conducted simulateously, but the stages are sequential.

Three criteria are widely used to compare different monitoring schemes: *localization time*, *m-structure consumption* and *wavelength link/channel consumption*. Sequential schemes require fewer number of tests in average due to the extra information provided by previous tests. In other words, less lightpaths have to be launched at a time and possibly consumes less wavelength links compared to nonadaptive schemes. On the other hand, nonadaptive schemes are faster as all tests are conducted in parallel.

2.4 Literature Review on Network Monitoring

2.4.1 S-LP Monitoring

Numerous studies have focused on the topic of S-LP deployment so as to achieve UFL under various scenarios. A greedy algorithm for the purpose, which is characterized by Random Code Assignment (RCA) and Random Code Swapping (RCS), was proposed in [65] to solve the single link UFL problem; while the same problem was previously formulated into an ILP and solved in small topologies [78]. An important technique developed in [65] is that link code is firstly randomly assigned, followed by RCS which tries to improve the quality of the final solution by randomly swapping the alarm codes for a pair of links. The study of [87] improved the algorithm in [65] by sequentially generating valid m-trails in a deterministic manner to shorten the running time. In particular, the scheme in [87] generates an m-trail by interconnecting some fragments using shortest paths, which are in turn constructed by picking some weighted links. The longest fragment will be chosen as the new m-trail if the fragments are still disconnected.

The authors of [21] firstly applied combinatorial group testing (CGT) techniques to non-adaptive UFL for SRLGs with up to d links, targeting at minimizing the number of parallel probes. To achieve this goal, a fault-free subgraph is identified in the network through a group of parallel probes and served as a hub to route other necessary probes to pinpoint failure. The upper and/or lower bounds on the number of non-adaptive probes needed for different classes of topologies are also discussed. In [63], failure identification for SRLGs with up to d links is studied based on greedy code swapping and $\bar{d}$-separable CGT code assignment. In [7], the SRLGs with adjacent links are considered. The proposed heuristic, which partitions the network to remove the dependency of codes assigned to each link, performs good in terms of monitoring locations and computation efficiency.

Since the set of S-LPs in a UFL solution may terminate at different nodes, the scheme in [76] imposes a significant overhead in collecting the alarms for the UFL decision. To avoid

such overhead, the study in [76] allows each on-trail node to tap the optical supervisory signal and thus share the status information of the m-trail. Both ILPs (Integer Linear Program) [76] and heuristics [24, 25] were developed, which demonstrated effectiveness of the framework in suppressing the consumed monitoring resources.

The lower bounds of the required number of monitoring paths for single link UFL are analyzed in [44]. Based on the derived bounds, the proposed bisection method produces a set of monitoring paths terminated at a set of predefined nodes to achieve UFL. In [4], both monitoring paths and monitoring cycles are deployed to identify any SRLG failure by using a heuristic and an ILP. The feasibility conditions on the network connectivity as well as the placement of monitoring locations are addressed.

To save the hardware cost of monitors, [38] used a super monitor which is placed at the junction of a set of m-cycles. The supervisory signals from the laser inside each super monitor are split simultaneously onto multiple m-cycles using an optical splitter. Since a small number of super monitors is desired, the problem of determining the optimal number and locations of super monitors was formulated as an add-on feature of any existing cycle-based scheme. Candidate cycle sets are enumerated first and the placement of the super monitors are decided by an ILP.

As an endeavour to suppress the supervisory wavelength-links (WLs) consumption and the cost of transmitters, the monitoring-tree (m-tree) approach is introduced in [12]. In the m-tree approach, a single laser diode is placed at a node to transmit uni-directional supervisory signal over a single link. When hitting a node, the signal can be terminated, forwarded through a single outgoing link, or duplicated and sent to multiple outgoing links. The m-tree approach is advantageous in that it usually requires one laser diode and keeps the number of wavelength-links to minimum. However, the number of monitors consumed is at least half of the number of links even for single link UFL.

2.4.2 W-LP Monitoring

Other studies only utilizes the W-LPs to pinpoint the failure source in a best effort manner. In [83], an end-to-end fault detection and localization protocol is proposed. The source node of each W-LP keeps sending hello packets to the destination node along the lightpath. The alarm for an W-LP is triggered once a certain number of consecutive hello packets are missed within a given time. Then all alarms are reported to the centralized network management system (NMS) to locate the network fault. As such a protocol has no prior knowledge on the network topology and only relies on the alarms collected simultaneously from W-LPs to locate the failure source, this protocol cannot guarantee UFL.

In [30], a probabilistic failure localization method is proposed based on establishing normalized fault vectors and construction of components matrix according to current allocation of monitors. Each fault vector corresponds to a network part to identify and a situation vector is defined to report the current status of all monitors. In consideration of the frequencies of all alarms, a weighted component matrix is constructed based on fault vectors. Similar to [83], the alarms are reported to a centralized managerial centre of the network and this method can not guarantee UFL. Unlike [83], the probabilistic method outputs a prioritized list of possible failure candidates by computing the cross-product of the situation vector and the weighted component matrix. Moreover, this method can also handle false or lost alarms to some extent.

In [57], the authors investigated the failure localization problem by minimizing the active monitors along the W-LPs for centralized and hierarchically-distributed management. Just like [83, 30], the technique proposed in their paper can only provide failure localization in a best effort way as the W-LPs and the monitor allocation can not guarantee UFL of the network parts under consideration. They proved the NP-completeness of the optimal monitor activation problem and presented its Integer Linear Programming (ILP) formulation as well as a greedy heuristic. The alarms are collected either centrally or hierarchically depending on the adopted management scheme. The authors in [4] exercised the idea of monitoring with W-LPs at the cost of additional optical-splitters when there are no dedicated bandwidth for monitoring. The main observation in the paper is that the inclusion of an optical splitter is equivalent to adding a new lightpath to the monitoring system. They formulate the problem as an ILP which aims to find the minimum number of splitters and their placement such that the failure of an SRLG can be uniquely localized. Note that this is also an best-effort approach as UFL is feasible only when the W-LPs meet certain criteria.

In the context of IP over optical networks, researchers formulated an ILP in [43] to solve the single (physical) link failure localization problem through selecting a set of precomputed W-LPs, each of which maps to a logical link in the upper (IP) layer. The objective of their ILP is to minimize: the number of unidentifiable physical links (primary); the number of physical links with identical alarm code (secondary) and the number of physical links used by W-LPs (tertiary). Numeric results show their ILP can achieve best-effort failure localization.

In [31], the authors adopt risk modeling to translate high-level failure notifications (e.g. alarms from W-LPs) to low-level causes (e.g. failure of an SRLG in the optical layer). Two inputs are fed into the proposed localization engine: risk model and failure signature. In a risk model, a physical object (e.g. a fiber) usually represents a shared risk for a group of logical entities at an upper layer (e.g. IP layer). A failure signature is defined to be a

set of monitoring observations (e.g. alarms of W-LPs). Note that these observations may contain "false alarms" like [30] and thier best-effort greedy heuristic can effectively reduce the number of the suspicious fault causes and the number of "false causes" returned by the algorithm.

2.4.3 Joint Monitoring of S-LPs and W-LPs

Some recent studies started to allocate S-LPs in consideration of the W-LPs. In [37], the authors suggested to manipulate the alarm code table (ACT) to come up with a feasible m-trail solution for single link failure localization, where "do not care" terms are introduced and manipulated. In their approach, the W-LPs are filled into the ACT so that the S-LPs can be deployed later to achieve UFL. In [11], path labels, which are composed of link labels, are inserted into the W-LPs to passively monitor the network link failure status. When some labels are found missing by the monitors on the W-LPs, active probes are launched sequentially to detect more link states if the passive monitoring result cannot fully identify the failure source. Note that an active probe is a lightpath constructed adaptively according to the result of the previous probe. This approach is advantageous in requiring almost no dedicated bandwidth and less monitors; however, it only deals with single link failure and more time is consumed compared to the non-adaptive m-trail method. In [56], the authors assume that all Optical Cross Connects(OXC) can detect in-band Loss of Light(LOL) and the faults propagate and raise alarms downstream from the point of failure along all affected lightpaths. Both ILP and a heuristic are given to solve the problem. However, the heuristic just pick S-LPs from the pre-computed k-shortest paths which can consume a lot of monitoring resources.

It is considered of ultimate importance to come up with an effective approach to jointly consider W-LPs and S-LPs such that the consumed resources are the minimal. As far as we know, no systematic approach has been available to exactly quantify how each newly added S-LP contributes to the identification of a set of SRLGs before we proposed the heuristic in [23].

Chapter 3

Background of Network Survivability Schemes

This chapter introduces the basic concepts, together with a brief comparison between different survivability schemes.

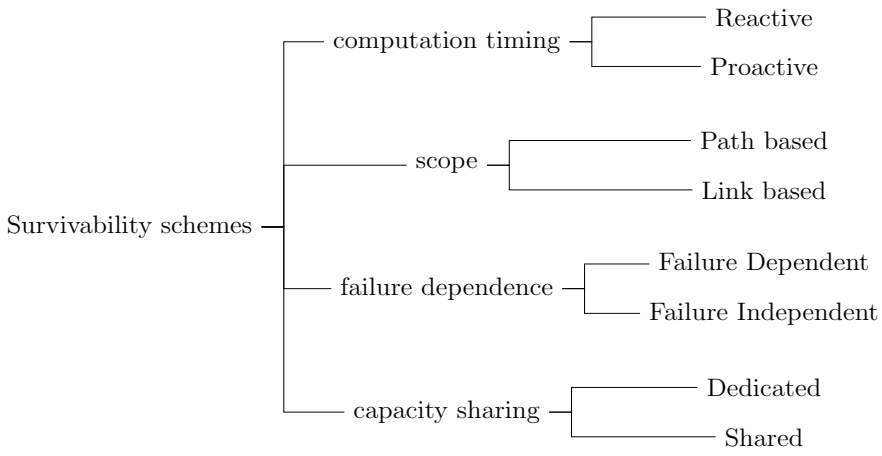


Figure 3.1: Classification of Survivability Schemes

3.1 Basic Concepts in Network Survivability

In a wavelength-routed network as WDM network, each connection is an end-to-end lightpath usually established on a specific wavelength. The traffic demand for a network is usually expressed by a set of end-to-end connection requests. For each connection request, a lightpath called working path (W-LP) is set up to transmit data. In order to deal with network failure, at least one Protection Lightpath (P-LP) is used to protect a specific segment of the working path for that connection request. In a typical restoration process, after a failure is detected by an upstream node through network monitoring, a protection path will merge back to the working path at a preselected downstream node called merging node[40] to recover the data traffic.

Without confusion, *Working Capacity* is defined to be the set of WLs taken by the W-LPs, *Spare Capacity* is referred to as the set of WLs have been reserved for protection purposes while *Free Capacity* refers to the WLs which can be freely allocated for W-LPs, P-LPs or monitoring purposes.

Among various design objectives to achieve survivability, *hardware cost*, *capacity efficiency* and *fault management complexity* are widely considered. Hardware cost refers to the costs brought by the installation and maintenance for optical devices, such as optical monitors, wavelength converters, add-drop multiplexers. *Capacity efficiency* is defined as the ratio of Working Capacity to Spare Capacity. Fault management complexity is reflected by the total restoration time, memory consumption at the network nodes and the signaling complexity.

As shown in Fig. 3.1, survivability schemes can be classified [54] by computation timing as pro-active/reactive (protection/restoration, pre-planned restoration/real-time searching [42]), based on the scope of backup path as being path-/link- based [40], by dependence on failure location as failure-dependent and failure-independant[53], and capacity sharing as dedicated/shared.

A *proactive* scheme precomputes the backup paths for all failure scenarios and reserves resources as soon as the working lightpaths are established. On the other hand, a *reactive* scheme dynamically searches for backup path(s) to detour the failure component(s) without reserving resources.

Capacity sharing among the working and protection paths is either *dedicated* or *shared*. In dedicated protection, spare resources are specifically allocated for a particular working lightpath[53](see Fig. 3.2a[29]). In 1+1 dedicated protection, both paths carry the same copy of data while the destination selects the better of the two paths with a decision circuitry. In contrast, in 1:1 dedicated protection, the protection path is not activated

until an failure occurs on the working lightpath[53]. In shared protection, however, multiple working paths may reserve the same network resources. As Fig. 3.2b shows, the W-LPs A-A' and B-B' are protected by protection lightpaths A-C-C'-A' and B-C-C'-B' respectively, which share the spare capacity on link CC'.

For shared protection, protection resources can be either *pre-configured* or *pre-planned*. In the former case, protection resources are untouched at normal operation and only need to be activated upon the failure occurs while in the later case, protection resources can be used by best effort traffic and must be configured first before the data traffic can be switched over.

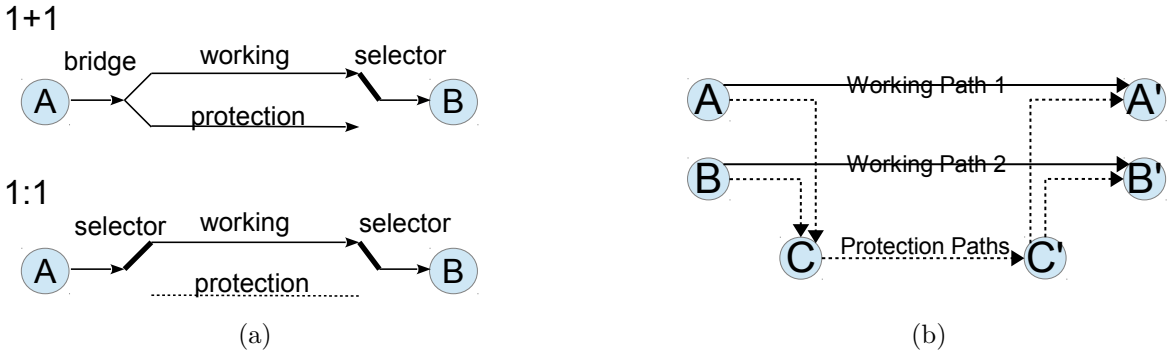
As this thesis focuses on shared path protection, the differences and key features of Path-/Link- based protection will be explained in Section 3.2.

3.2 Path Based Protection Vs. Link Based Protection

In this Section, the constraint imposed by the Shared Risk Link Group and the shared protection schemes will be introuced first. Next, the Path-/Link- Based protection will be illustrated with examples.

3.2.1 Shared Risk Link Group Constraint

To guarantee 100% restorability after a failure occurs, the Shared Risk Link Group(SRLG) constraint[42] stipulates that a set of working paths in the same SRLG cannot have their protection paths taking the same spare capacity.



(a) (b)

Figure 3.2: Dedicated Protection Vs. Shared Protection

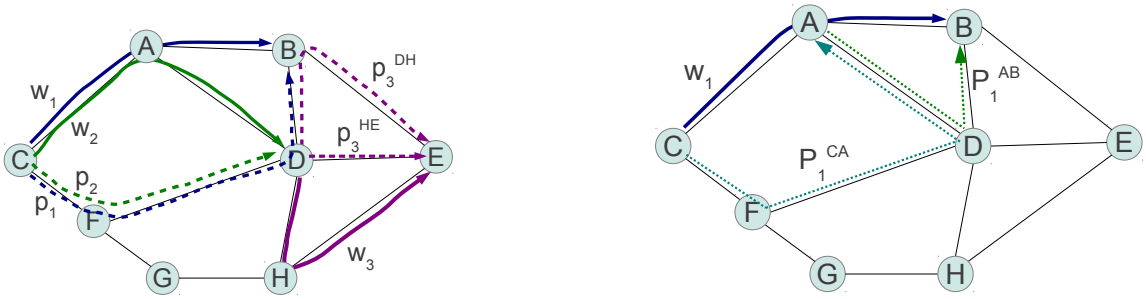
See Fig. 3.3a for example, each W-LP w_i is protected by some P-LPs labelled by p_i^j and all single links are the SRLGs under consideration. According to the SRLG constraint, p_1 and p_2 cannot share resources on path segment C-F-D since w_1 and w_2 need simultaneous protection when link AC fails. On the other hand, p_1 and p_3^{DH} can share capacity on link DB as no single link failure would disconnect both w_1 and w_3.

3.2.2 Path-Based Protection

In path protection, one or more P-LPs between the source and destination nodes are used to protect each W-LP. In case a W-LP is disrupted due to an SRLG failure, the data transmission task will be taken over by a corresponding P-LP.

Depending on whether different P-LPs will be used upon various SRLG failures, path protection schemes fall into two categories: failure-independent path protection (FIPP) and failure-dependent path protection (FDPP). In FIPP, each W-LP is protected by exactly one P-LP, which simultaneously bypasses all SRLGs affecting that W-LP. The well-known Shared Backup Path Protection(SBPP) belongs to this category. Alternatively, in an FDPP strategy, a W-LP can be assigned several P-LPs, each of which recovers the W-LP under some specific SRLG failure(s).

In Fig. 3.3a, W-LPs w_1 and w_2 adopts FIPP while w_3 adopts FDPP. When link DH fails, P-LP p_3^{DH} will be used to recover W-LP w_3. Similarly, p_3^{HE} will be used when link HE fails.



(a) (b)

Figure 3.3: Path Protection Vs. Link Protection

3.2.3 Link-Based Protection

Link-Based Protection protects each network link independently. Upon any link fails, a protection path would be established between the two end nodes of that failed link to restore the services. Compared to path protection, a lower restoration time is expected for link protection since the protection signaling is handled by the end nodes of the failed link rather than the end nodes of the W-LP.

Take the W-LP w_1 in Fig. 3.3b for example, when link CA fails, the protection path p_1^{CA} (C-F-D-A) would replace CA in w_1 and the resulting new path C-F-D-A-B would be used to transmit data. Similarly, p_1^{AB} (A-D-B) would replace link AB and generate a new path C-A-D-B to recover the traffic of w_1 when link AB fails.

Preconfigured Protection Cycle (p-Cycle) [85, 84] can be viewed as another type of link-based protection. Several p-Cycles may be deployed in the network to deal with any link failure. Each p-Cycle can protect every on-cycle link and any straddling link. See Fig. 3.4 for example, the p-Cycle A-B-G-D-E-F-A can protect all the on-cycle links: AB, BG, GD,DE and EF, as well as all the straddling links: FG and EG. When an on-cyle link fails, the remaining part of the cycle would compose the protection path. When a straddling link fails, two on-cycle paths are available to recover the traffic. For example, when the on-cycle link AB fails, the path A-F-E-D-G-B can recover the failure while when the straddling link FG fails, both F-A-B-G and F-E-D-G can be used. Note that the allocated p-Cycles cannot share WLs.

3.3 Comparisons of different Survivability Schemes

Compared to reactive strategies, proactive strategies can achieve shorter restoration time for static traffic conditions since the P-LPs are precomputed and network resources for

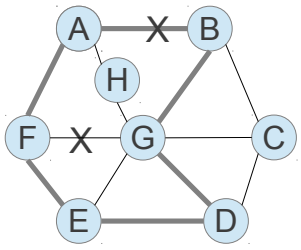


Figure 3.4: p-Cycle Example

protection can be pre-planned or even pre-configured. Nonetheless, the reactive strategies are more suitable to handle dynamic traffic as the protection paths are computed in an online fashion.

Compared to dedicated protection, shared protection achieves better capacity efficiency while consuming more restoration time due to configuring the optical network elements and processing the control messages along the protection path. When a fault occurs on a W-LP, the destination node of the W-LP will notify the source node to activate a switch-over once it detects the fault. Then, the source node will properly configure all nodes along the protection path by a wakeup packet and then switch over the traffic to the protection path.

Compared to link protection, path protection usually takes longer restoration time since the recovery process is handled by the two end nodes of the W-LP rather than the end nodes of the failed link. However, path protection is more capacity efficient[53, 49]. As pointed out in [42], the failed link and its downstream neighbour node require separate protection segments in link protection, which may impair capacity efficiency.

Compared to failure independent schemes, failure dependent schemes are more flexible in survivable routing. For example, failure independent schemes as Shared Backup Path Protection (SBPP) may subject to "trap condition" in survivable routing since each P-LP needs to be link-disjoint from its W-LP when all single-link failures are under consideration; in contrast, failure dependent schemes are more flexible in routing, since a P-LP may choose to only bypass some links traversed by the W-LP. On the other hand, memory consumption and signaling complexity of failure dependent schemes are relatively higher since a failure dependent scheme may needs to activate different P-LP under various failure events, which means an on-line decision for choosing P-LPs must be made and more information of P-LPs should be kept at a node.

Chapter 4

On Identifying SRLG failures in All-Optical Networks

4.1 Introduction

Generally, a failure localization scheme relies on a monitoring mechanism which considers a set of predefined SRLGs on the given network topology. Fault management is defined under the control framework of Generalized Multiprotocol Label Switching (GMPLS) [45, 8] as a set of real-time tasks performed sequentially right after the occurrence of any failure, including failure localization, failure notification and device configuration. The former two tasks (i.e., localization and notification) in the optical network domain are defined via a series of electronic signaling mechanisms. By applying Link Management Protocol (LMP) coupled with a signaling protocol such as Resource Reservation Protocol - Traffic Engineering (RSVP-TE), each downstream node of a failed working lightpath is subject to Loss of Light (LOL) and will send an alarm to its upstream node. After receiving the alarm, the upstream node checks the corresponding input port and forwards the alarm to further upstream if the node is also subject to LOL. Otherwise, the faulty link is determined in the downstream, and the upstream node initiates the predefined failure recovery procedure via RSVP-TE again to "wake up" each intermediate node along the protection lightpath.

The aforementioned GMPLS based approaches are subject to many weaknesses. Firstly, alarms could be simultaneously issued by multiple downstream nodes, and the number of alarms is determined by the number of lightpaths traversing through the faulty link(s) and the length of the lightpaths. Theoretically, such a large number of alarms could easily lead to an *alarm storm* in the control plane and bring risks of crashing the whole network.

Secondly, the aforementioned approaches cannot handle multi-link SRLG failure events, due to the fact that a node can only be aware of the faulty link which is in the downstream of any lightpath traversing the node, but has no way to know the status of a link that none of the traversing lightpaths of the node goes through the link. Therefore, when a multi-link SRLG fails, a node may only be able to identify the failure of a subset of the links in the SRLG, and thus may select a protection path for restoration that is nonetheless subject to the failure, too. Thirdly, due to extensive electronic signaling mechanism and nodal processing, the aforementioned approaches may take hundreds milliseconds of delay which is added up to the overall recovery time, which may not only cause data loss but also impose vicious impact on the upper network/transport layer protocols such as OSPF and TCP.

To improve the GMPLS based approaches, link monitoring [65] has been considered such that every link is exclusively monitored via a single-hop supervisory lightpath (S-LP). Once a failure occurs, the monitor(s) subject to LOL will issue an alarm, which is further shipped to the network controller or the corresponding decision nodes (e.g., edge routers) for subsequent restoration processes.

Although being an effective solution that can mitigate the awkward situation in the conventional GMPLS based approach, the link-based monitoring approach requires $|E|$ WLs along with $|E|$ transmitters/monitors that are considered precious resources in optical networks. Besides, similar to the GMPLS based approaches, link monitoring strongly relies on electronic signaling for failure notification and resource reservation, which leads to considerable control complexity and long recovery time. Thus, numerous schemes based on sophisticated designs and various assumptions were extensively reported in the past decades. One of the reported advances is by using multi-hop monitoring structures for the purpose, such as m-cycle[81], m-path[4] etc., which are further generalized by monitoring trail (m-trail)[65]. Essentially, an m-trail is referred to as a monitoring structure whose S-LP can traverse through each undirected edge of a network subgraph in any direction by at most once. Many schemes including integer linear programming (ILP) [76] and heuristics [65][25] have been introduced to accomplish the m-trail allocation task under various constraints and SRLG scenarios, including single link failure, sparse/dense SRLG failure, adjacent link failure, limitations on the length of m-trails and the number of transmitters at each node, and the consideration of the physical impairment, etc.

Unfortunately, the total monitoring WL consumption (or termed total coverlength) of these schemes is much higher than that by link monitoring and grows significantly with respect to the scale of the problem in terms of the size of the network topology and the number of considered SRLGs. Furthermore, similar to LMP based schemes, most of these studies rely on control plane signaling for alarm collection before failure localization

decision can be made except for [76, 24, 25] that aimed at a signaling-free and all-optical framework for failure monitoring and localization.

To avoid the alarm collection overhead, the study in [76] allows each on-trail node to tap the optical supervisory signal and thus share the status information of the m-trail. Due to the one-way nature of the directed S-LPs, a monitor equipped at an upstream on-trail node may not be able to detect LOL caused by a downstream on-trail link failure. To resolve this situation, the work in [76] only considered closed m-trails (i.e., simple or non-simple m-cycles), where LOL of any S-LP can practically propagate along the cyclic monitoring structure. In practice, it is possible that only a subset of all nodes, denoted as monitoring nodes (MNs), need to carry out failure localization. The proposed *Local Unambiguous Failure Localization (L-UFL)* problem sets its goal as follows: by properly allocating a set of m-trails, each MN can unambiguously identify every possible link failure merely based on its locally collected LOL signals.

Since the ILP approach for L-UFL [76] is time-consuming and can hardly reach optimality, we proposed a heuristic to tackle L-UFL with S-LPs [25], verified through extensive simulations. Unfortunately, the total coverlength of the heuristic in [25] is even worse compared to the UFL framework due to lack of an efficient mechanism for suppressing monitoring resources. For example, coverlength consumption for 20-node 60-link random networks is about $15|E|$ to enable each node to identify any dual link SRLG failure[25].

To further save monitoring resources for L-UFL, we proposed another novel algorithm in [23] to localize SRLG failure via S-LPs (m-trails) together with the working lightpaths. In particular, this joint monitoring heuristic allocates a set of S-LPs by a utility function in tandem so as to achieve unambiguous SRLG failure localization. The built-in utility function, which predicts the final solution quality from its current partial solution, further helps reduce the resource consumption. Smulation results are examined to verify the proposed approach in terms of its efficiency and solution quality.

In this chapter, I generalize our published papers [25, 23] under the Local Unambiguous Failure Localization (L-UFL) framework, in which a set of network nodes, called monitoring nodes (MNs), can locally perform UFL for the considered SRLGs by inspecting the failure status of the traversing lightpaths. In Section 4.2, I briefly introduce the definitions for L-UFL together with some basic concepts and techniques for the proposed heuristics. After discussing the feasibility conditions for L-UFL in Section 4.3, an m-trail generation algorithm is analyzed in Section 4.4, which would be used as building-blocks for achieving L-UFL. Finally, Section 4.5 and 4.6 detail our L-UFL heuristic using S-LPs [25] and the joint monitoring L-UFL heuristic [23] respectively.

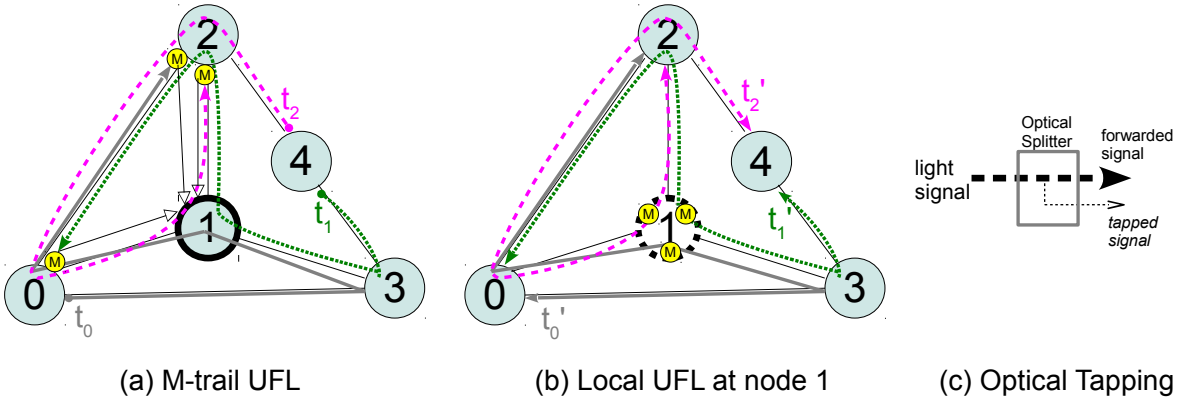


Figure 4.1: UFL vs. L-UFL

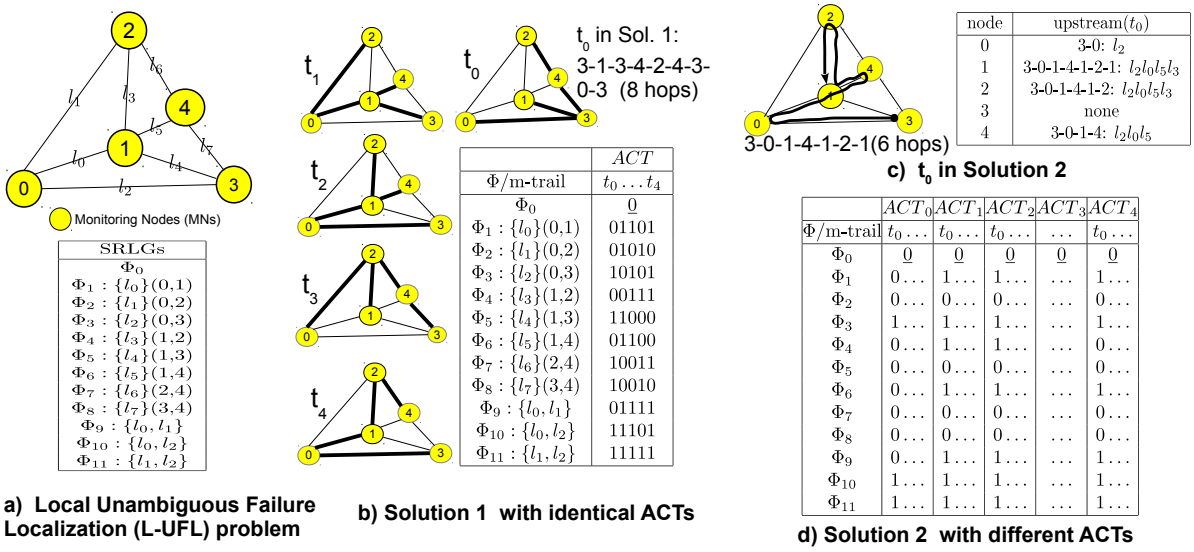
4.2 Algorithm Background

In this section, backgrounds on L-UFL will be presented. First, the definition of the L-UFL problem will be given in Subsection 4.2.1. Since the solution to the L-UFL problem is represented by Alarm Code Tables (ACTs), they are introduced in Subsection 4.2.2. Subsection 4.2.3 explains two important concepts: *Entropy and Mutual Entropy of an ACT*, which serve to evaluate the *"progress"* of an ACT towards L-UFL and the information contribution of an m-trail respectively. Finally, to improve the performance of the proposed algorithm by evaluating partial solutions, a utility function is derived in Subsection 4.2.4.

4.2.1 Local Unambiguous Failure Localization

Definition 3. *Given a network graph $G(V, E)$ where V denotes the set of nodes and E refers to the set of undirected edges, Local Unambiguous Failure Localization (L-UFL) is defined for each node in a set $M \subseteq V$. For simplicity, the nodes in M are referred to as Monitoring Nodes (MNs). Let $\Phi = \{\Phi_0, \Phi_1, \ldots\}$ correspond to the set of SRLGs under consideration with each entry Φ_i represented by a set of links contained in the SRLG. Specifically, Φ_0 is an empty link set, which means no failure.*

The L-UFL m-trail allocation problem is to determine a set of S-LPs and W-LPs, such that each MN can perform L-UFL under the given set of SRLGs solely based on its locally collected alarm signals.



	ACT
Φ/m-trail	$t_0 \dots t_4$
Φ_0	$\underline{0}$
$\Phi_1 : \{l_0\}(0,1)$	01101
$\Phi_2 : \{l_1\}(0,2)$	01010
$\Phi_3 : \{l_2\}(0,3)$	10101
$\Phi_4 : \{l_3\}(1,2)$	00111
$\Phi_5 : \{l_4\}(1,3)$	11000
$\Phi_6 : \{l_5\}(1,4)$	01100
$\Phi_7 : \{l_6\}(2,4)$	10011
$\Phi_8 : \{l_7\}(3,4)$	10010
$\Phi_9 : \{l_0, l_1\}$	01111
$\Phi_{10} : \{l_0, l_2\}$	11101
$\Phi_{11} : \{l_1, l_2\}$	11111

node	upstream(t_0)
0	3-0: l_2
1	3-0-1-4-1-2-1: $l_2 l_0 l_5 l_3$
2	3-0-1-4-1-2: $l_2 l_0 l_5 l_3$
3	none
4	3-0-1-4: $l_2 l_0 l_5$

	ACT_0	ACT_1	ACT_2	ACT_3	ACT_4
Φ/m-trail	$t_0 \dots$	$t_0 \dots$	$t_0 \dots$	$\dots$	$t_0 \dots$
Φ_0	$\underline{0}$	$\underline{0}$	$\underline{0}$	$\underline{0}$	$\underline{0}$
Φ_1	$0\dots$	$1\dots$	$1\dots$	$\dots$	$1\dots$
Φ_2	$0\dots$	$0\dots$	$0\dots$	$\dots$	$0\dots$
Φ_3	$1\dots$	$1\dots$	$1\dots$	$\dots$	$1\dots$
Φ_4	$0\dots$	$1\dots$	$1\dots$	$\dots$	$0\dots$
Φ_5	$0\dots$	$0\dots$	$0\dots$	$\dots$	$0\dots$
Φ_6	$0\dots$	$1\dots$	$1\dots$	$\dots$	$1\dots$
Φ_7	$0\dots$	$0\dots$	$0\dots$	$\dots$	$0\dots$
Φ_8	$0\dots$	$0\dots$	$0\dots$	$\dots$	$0\dots$
Φ_9	$0\dots$	$1\dots$	$1\dots$	$\dots$	$1\dots$
Φ_{10}	$1\dots$	$1\dots$	$1\dots$	$\dots$	$1\dots$
Φ_{11}	$1\dots$	$1\dots$	$1\dots$	$\dots$	$1\dots$

Figure 4.2: L-UFL definition

When $M = \varnothing$, the problem is trivially the UFL problem while it is called Network-wide Local Unambiguous Failure Localization (NWL-UFL) problem when $M = V$. Thus, the techniques and algorithms for L-UFL presented in this chapter are also applicable to UFL with minor modification. Particularly, under UFL:

1. no MNs are defined, thereby an m-trail is not required to traverse any specific node
2. one ACT suffices to represent the solution.

Fig. 4.1 compares the traditional UFL approach with the signaling free L-UFL framework. Suppose that only node 1 needs to monitor the whole network. As Fig. 4.1a shows, the UFL approach deploys three m-trails whose monitors are located at two different nodes (i.e. node 0 and 2). Upon a failure event, these monitors need to report their status to node 1 through electronic signaling. In contrast, with the help of optical-tapping (see Fig. 4.1c), node 1 can independently and locally get the network status through tapping the set of m-trails passing through it under the L-UFL framework, as Fig. 4.1b illustrates..

Fig. 4.2a exemplifies a NWL-UFL problem for a 8-link network and 12 SRLGs under consideration while Fig. 4.2b-d provide two feasible solutions to the problem.

4.2.2 Alarm Code Table

.

A solution to the L-UFL problem can be represented by Alarm Code Tables (ACTs). An ACT of an MN keeps the mapping between the received failure status "seen" at the MN and the corresponding SRLG. An m-trail of a feasible m-trail solution can provide meaningful information to all or just a subset of MNs that it traverses through. For simplicity, we use ACT_i $(i = 0 \dots |M| - 1)$ to denote the ACT of the i_{th} MN.

By looking up ACT_i, the i_{th} MN can independently identify the failure of any predefined SRLG. In specific, each row in an ACT stores an *alarm code* corresponding to an SRLG that can be identified at the MN, and each column is called a *trail code* which is calculated based on the *upstream links* [1] traversed by the m-trail. As usual, each binary bit in an alarm code corresponds to the on-off status of an m-trail seen at a MN. Note that ACT_i only contains those m-trails which provide meaningful information to the i_{th} MN.

In case each m-trail has identical trail codes in all ACTs, the m-trail solution can be simply expressed by one ACT. Fig. 4.2b provides such an example. Note that each m-trail in that solution (e.g. t_0 in Fig. 4.2b) is in shape of a cycle (some links are traversed twice, although not drawn explicitly) and spans all MNs. Consequently,all these m-trails have identical trail codes. For instance, the alarm code of link $(0, 1)$ in any ACT (see Fig. 4.2b) is "01101" and the trail code of m-trail t_0 in that solution is "000101011011".

Fig. 4.2d gives a general m-trail solution represented by distinct ACTs in which every node can perform L-UFL for a network with 8 links. As shown in Fig. 4.2d, the m-trail t_0 drawn in Fig. 4.2c corresponds to different trail codes in various ACTs.

Fig. 4.3 further explains how Solution 1 shown in Fig. 4.2b can achieve L-UFL. As shown in Fig. 4.3a, all m-trails span all MNs and are transmitting light signals normally when no failure occurs(i.e. normal case). Thus, the status of the m-trails can be represented by an all-zero alarm code under normal case. Suppose at some moment link (0,1) and (0,2) fail, all m-trails except for t_0 are affected by the failure and send out alarms, which leads to a different alarm code: "01111"(see Fig. 4.3b). Similarly, it can be verified that under different predefined SRLG failures, different combinations of m-trails will send out alarms and detected by the MNs through tapping. In other words, L-UFL is achieved by the deployed m-trails since each SRLG is assigned a unique alarm code in the ACT as shown in Fig. 4.2b.

As all links in an SRLG fail simultaneously, in any ACT, an alarm code of an SRLG is always the bitwise-OR of the alarm codes of all the links in the SRLG. Specifically, Φ_0's

[1] see Section 2.2 for detail

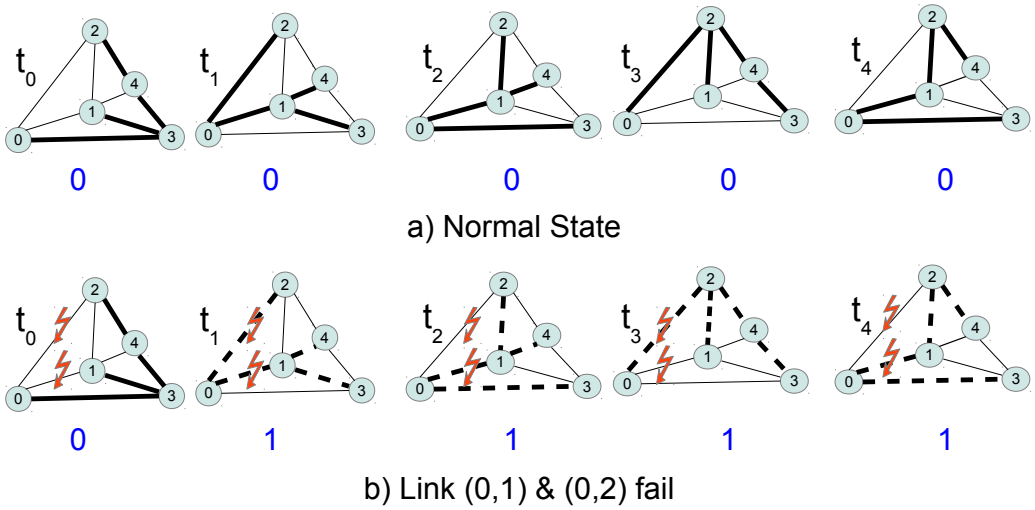


Figure 4.3: L-UFL example

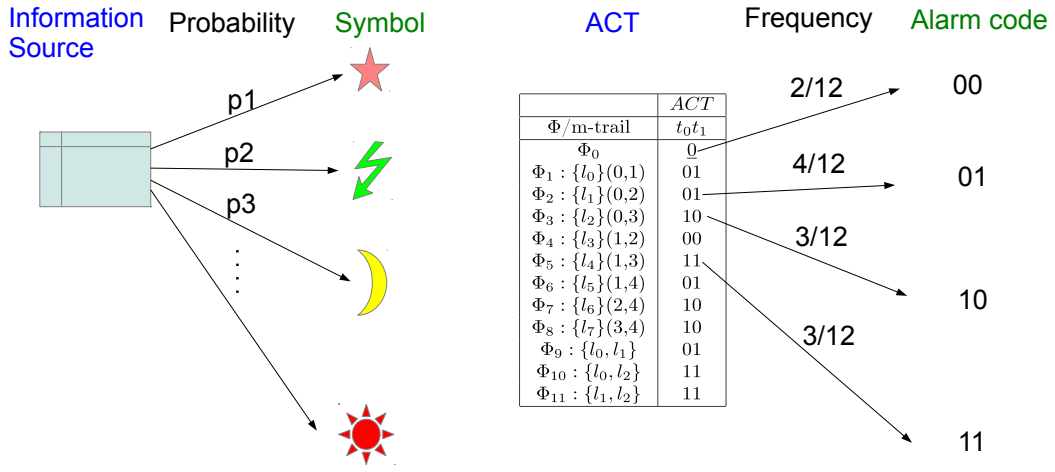
alarm code is a zero vector (represented by $\underline{0}$) as it indicates no failure. As Fig. 4.2b shows, the alarm codes for $\Phi_1 = \{l_0\}$ and $\Phi_2 = \{l_1\}$ are 01101 and 01010 respectively in Solution 1. Then, the alarm code for $\Phi_9 = \{l_0, l_1\}$ is $01101 \vee 01010 = 01111$.

For each link l_k in the network, we define a binary vector I_k^l of length $|\Phi|$ as its *link-SRLG* Incidence Vector (IV) The j_{th} bit in I_k^l corresponds to $\Phi_j \in \Phi$ $(j = 0 \ldots |\Phi| - 1)$, and the bit is set to 1 if $l_k \in \Phi_j$. For the problem in Fig. 4.2a, the IV of l_0 is [010000000110] as Φ_1, Φ_9 and Φ_{10} contains l_0. Given an m-trail and an on-trail MN, by the Bitwise-OR operation on the IVs of the upstream links, the trail code in the ACT of that MN can be obtained.

4.2.3 Entropy and Mutual Entropy of an ACT

In an ACT, all SRLGs with the same alarm code compose an *alarm code group*. Let $\Delta_i = \{\Delta_i^0, \Delta_i^1 \ldots\}$ represent the set of alarm code groups in ACT_i $(i = 0 \ldots |M| - 1)$.

As Fig. 4.4 shows, by viewing an ACT as an *information source*, an alarm code as a *symbol* to be generated by the source with certain probability, and the frequency with which an alarm code appears in the ACT as the *probability* for a symbol, the *entropy* definition in Information Theory can be applied to *evaluate the ambiguity of an ACT*, i.e. how much progress has been made towards a monitoring task(e.g. UFL).



Φ/m-trail	ACT t_0t_1
Φ_0	0
$\Phi_1 : \{l_0\}(0,1)$	01
$\Phi_2 : \{l_1\}(0,2)$	01
$\Phi_3 : \{l_2\}(0,3)$	10
$\Phi_4 : \{l_3\}(1,2)$	00
$\Phi_5 : \{l_4\}(1,3)$	11
$\Phi_6 : \{l_5\}(1,4)$	01
$\Phi_7 : \{l_6\}(2,4)$	10
$\Phi_8 : \{l_7\}(3,4)$	10
$\Phi_9 : \{l_0, l_1\}$	01
$\Phi_{10} : \{l_0, l_2\}$	11
$\Phi_{11} : \{l_1, l_2\}$	11

Entropy: $H(\mathcal{A}) = -\sum_{\alpha} p(\alpha)\log(p(\alpha)) = -\frac{2}{12}\log\frac{2}{12} - \frac{4}{12}\log\frac{4}{12} - 2\cdot\frac{3}{12}\log\frac{3}{12}$

Figure 4.4: Entropy Definition and calculation when Sol. 1 only contains t_0 and t_1

Thus, the *Entropy* of ACT_i is defined as:

$$H(ACT_i) = -\sum_{j=0}^{|\Delta_i|-1} p_j \log p_j \tag{4.1}$$

where $p_j = |\Delta_i^j|/|\Phi|$.

Fig. 4.4 exemplifies the entropy computation of the ACT for Solution 1(see Fig. 4.2b) when two m-trails are available. As another example, Table. 4.1 shows the alarm code group assignment and entropy calculation when Solution 2 in Fig. 4.2d only contains its first m-trail t_0.

After a new m-trail is generated, it helps "zoom in" the failure event(SRLG) to be identified (i.e. reduce the ambiguity). For simplicity, $p_j \log p_j$ is defined as the *"group entropy"* for an alarm code group. After adding a new trail, if some alarm code groups formed by previous trails split, they are called *info-gaining* groups as their entropy are increased. The *mutual entropy* (i.e. information gain) $\delta = H(\mathcal{C}') - H(\mathcal{C})$ where $\mathcal{C}'(\mathcal{C})$ denotes the ACT at a node after(before) adding the trail. Fig. 4.5 gives an example for mutual entropy computation given two trails t_0 and t_1.

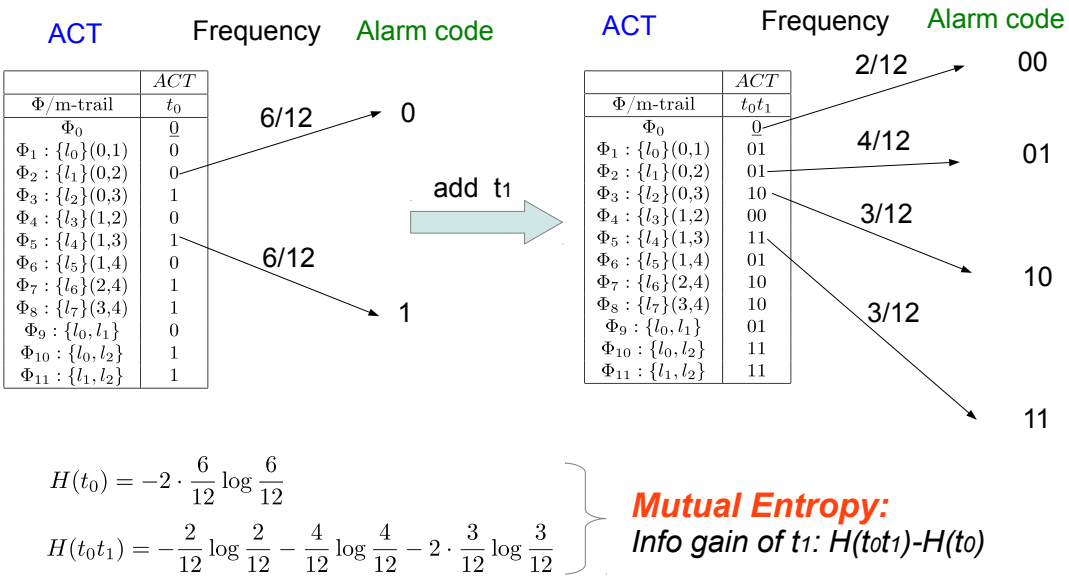


Figure 4.5: Mutual Entropy

The *enhanced "Min Wavelength Max Information" principle* means that an m-trail with high information gain and low wavelength channel consumption is preferred. Hence, *trail efficiency*: $\eta = \delta/\omega$ is computed and it serves as the metric to select a good m-trail, where

Table 4.1: Entropy calculation when Solution 2 only contains t_0

	alarm code group(size)	$H(ACT)$
ACT_0	$\Delta_0^0 = \{\Phi_{0-2,4-9}\}$ (9) $\Delta_0^1 = \{\Phi_{3,10-11}\}$ (3)	$-\frac{3}{12} \log \frac{3}{12} - \frac{9}{12} \log \frac{9}{12}$ $= 0.81$
ACT_1	$\Delta_1^0 = \{\Phi_{0,2,5,7-8}\}$ (5) $\Delta_1^1 = \{\Phi_{1,3-4,6,9-11}\}$ (7)	$-\frac{5}{12} \log \frac{5}{12} - \frac{7}{12} \log \frac{7}{12}$ $= 0.98$
ACT_2	$\Delta_1^0 = \{\Phi_{0,2,5,7-8}\}$ (5) $\Delta_1^1 = \{\Phi_{1,3-4,6,9-11}\}$ (7)	$-\frac{5}{12} \log \frac{5}{12} - \frac{7}{12} \log \frac{7}{12}$ $= 0.98$
ACT_3	$\Delta_3^0 = \Phi$ (12)	0
ACT_4	$\Delta_4^0 = \{\Phi_{0,2,4-5,7-8}\}$ (6) $\Delta_4^1 = \{\Phi_{1,3,6,9-11}\}$ (6)	$-2\frac{6}{12} \log \frac{6}{12} = 1$
$\hat{H}$	$(0.81 + 0.98 + 0.98 + 0 + 1)/(5 \log 12) = 0.21$	

ω stands for the wavelength cost of the current trail (i.e., the number of WLs consumed by the trail).

4.2.4 Evaluating a solution

A complete or partial m-trail solution can be given by a set of m-trails $T = \{t_0, t_1, \ldots\}$ with each entry t_i stands for an m-trail.

When T corresponds to a complete solution, as an important policy in the study, the target is to minimize a generic cost function on the complete solution:

$$Cost(T) = \gamma \cdot \omega(T) + |T| \tag{4.2}$$

where ω is a function which returns the *coverlength* (i.e. total number of WL consumed) for a solution, $|T|$ refers to the number of m-trails in the solution and γ is a weighting that can be manipulated to compromise the effects of the two measures.

On the other hand, when T corresponds to a partial solution, more m-trails are required to get a complete solution. To meet the desired feature of the proposed algorithm that evolves for better solution quality in each iteration of algorithm progressing, a utility function is devised to evaluate T:

$$Utility(T) = -\gamma \cdot [\, \omega(T) + \omega(T^r) \,] - [\, |T| + |T^r| \,] \tag{4.3}$$

where T^r refers to the unknown set of m-trails needed based on T to achieve L-UFL. Note that $T \cup T^r$ is a complete m-trail solution and the utility of T is computed by negating the cost of $T \cup T^r$. $|T^r|$ is assigned the lower bound given in Lemma 1 and $\omega(T^r)$ is obtained by multiplying $|T^r|$ with the average number of wavelength-channels consumed by an m-trail in T. For instance, when Solution 2 in Fig. 4.2d only contains one m-trail t_0, the corresponding alarm code groups are listed in Table. 4.1 and we compute the utility for $T = \{t_0\}$ as follows. Note that $\omega(t_0) = 6$ (t_0 consumes 6 WLs (i.e. 6 hops) as shown in Fig. 4.2c) and the size of biggest alarm code group Δ_3^0 is 12(see Table. 4.1). Hence, $|T^r| = \lceil \log 12 \rceil$ and

$$Utility(T) = -\gamma \cdot [\, \omega(T) + \frac{\omega(T)}{|T|}|T^r| \,] - [\, |T| + |T^r| \,]$$

$$= -\gamma \cdot [\, 6 + \frac{6}{1}\lceil \log 12 \rceil \,] - [\, 1 + \lceil \log 12 \rceil \,]$$

.

Lemma 1. *For a partial solution given by the m-trail set T, the lower bound for $|T^r|$ is $\lceil \log_2 Z \rceil$ where Z is the size of the biggest "alarm code group" in the ACTs.*

Proof. Each m-trail can provide up to 1 bit information; thus an alarm code group with z SRLGs needs at least $\lceil \log_2 z \rceil$ m-trails to differentiate them. As all alarm code groups must be differentiated, the best we can achieve is that the m-trails in T^r simultaneously differentiate all groups and $|T^r| = \lceil \log_2 Z \rceil$, where Z is the size of the biggest group. □

Furthermore, we define the *"entropy ratio"* of an m-trail solution by evaluating its ACTs:

$$\hat{H} = \frac{1}{|M|} \sum_{i=0}^{|M|-1} \frac{H(ACT_i)}{\log |\Phi|} \tag{4.4}$$

Note that $\hat{H}$ is a ratio number between 0 and 1. For a complete solution, the $\hat{H}$ value is 1. To see this, just consider the ACT for one MN. Since each SRLG is uniquely coded for a complete solution, there are $|\Phi|$ alarm code groups in that ACT. Thus, $H(ACT) = \log |\Phi|$. According to (4.4), $\hat{H} = 1$. Generally, a less ambiguous m-trail solution has a higher $\hat{H}$ value(see Table. 4.1 for calculation example).

4.3 Feasibility of an L-UFL problem

Lemma 2. *Two SRLGs with link sets Φ_i and Φ_j are distinguishable at an MN if and only if:*

1. *there exists an m-trail t_k which traverses through some link l in Φ_i* ***while disjoint from*** *Φ_j* ***(or*** *Φ_j* ***while disjoint from*** *Φ_i)**;*
2. *the MN is in downstream of l.*

Proof. Condition 1) guarantees that at least one alarm bit in the respective alarm code is different for the two SRLGs. Without loss of generality, let t_k traverse through some link in Φ_i while disjoint from Φ_j and provides the f_{th} alarm bit in the ACT of the MN.

As the alarm code of an SRLG is the bitwise-OR of all alarm codes of those links in the SRLG, traversing any link l in Φ_i will set the f_{th} alarm bit for Φ_i to 1. This is because the failure of Φ_i will interrupt t_k and turn the f_{th} bit from 0 to 1. On the other hand, as t_k is disjoint from Φ_j, the f_{th} alarm bit for Φ_j is 0 as the failure of Φ_j will not interrupt t_k. Consequently, the alarm codes for two SRLGs are always different if condition 1) is met.

Condition 2) ensures the failure due to an SRLG be detected by the MN as an MN is only aware of LOL due to upstream links unless the m-trail is in a shape of cycle.

□

According to Lemma 2 and the definition of L-UFL, we have the following conclusion:

Definition 4. *Consider an L-UFL problem and an m-trail solution given by a set of m-trails T. Let $T_i \subseteq T$ $(i = 0 \dots |M| - 1)$ represent the m-trails in ACT_i. T is a feasible solution to the L-UFL problem if and only if :*

1. *Each m-trail in T_i passes through the i_{th} MN;*
2. *any two SRLGs under consideration are distinguishable to the i_{th} MN by an m-trail in T_i.*

Corollary 1. *If graph G is $(d + 1)$ edge-connected, there always exists a valid m-trail solution to identify up to d-link failure events for any MN.*

Proof. As graph G is $(d + 1)$ edge-connected, removing links belonging to any SRLG with no more than d links will not disconnect the graph. According to Lemma 2 and Definition 4, a valid m-trail solution can always be found. □

Definition 5. *As a valid L-UFL m-trail solution, any ACT at a MN must satisfy:*

1. *Any SRLG under consideration is assigned a unique alarm code*
2. *the alarm code of an SRLG is the bitwise-OR of all links' alarm codes in that SRLG*
3. *the trail code of an m-trail is computed based on the upstream links of the MN*

Corollary 2. *As a valid L-UFL m-trail solution for identifying all upto d link failures, any ACT at a MN must satisfy:*

1. *the alarm codes in the ACT form a $\bar{d}$-separable CGT code*
2. *the trail code of an m-trail is computed based on the upstream links of the MN*

Proof. When all upto d link failures is under consideration, the first two conditions in Definition 5 can be replaced with the first condition in Corollary 2. By viewing each link as an "item" and each m-trail as a "group test", the first condition in Corollary 2 directly follows from the definition of a CGT code. Section 2.3 provides the detailed analogy between the CGT model and the UFL model. □

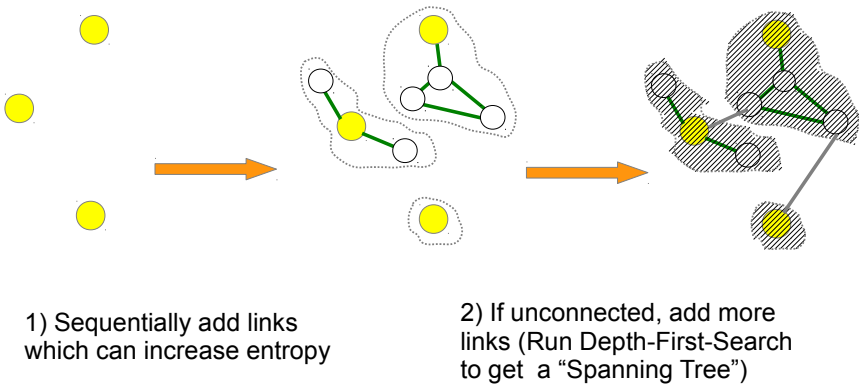


Figure 4.6: Simplified version for m-trail generation

Note that Definition 5 and Corollary 2 will be used by the proposed algorithms in Section 4.5 and 4.6 for validating the generated L-UFL solution.

4.4 Generating m-trails

Fig. 4.8 shows the main framework of generating a valid and info-gaining m-trail. Depending on whether the m-trail needs to pass through all unidentified MN or just one unidentified MN, the m-trail generation schemes can be classified as m-trail *global sharing* scheme and m-trail *local sharing* scheme respectively (An MN is *unidentified* if it has not achieved L-UFL). Since the only difference between the two schemes is the definition of a valid m-trail, we use one general algorithm to unify them.

Fig. 4.6 illustrates the high-level idea of the m-trail generation algorithm, where the colorful nodes represent MNs. The first step is sequentially adding links which can differentiate as many as possible SRLG pairs. In case the selected links in the first step form a valid m-trail, the algorithm stops; Otherwise, additional links are chosen carefully to obtain a valid m-trail.

The detailed procedure is shown in Fig. 4.8. First, a tentative trail code is generated based on the ACT of a randomly unidentified MN (**line 1-4**). Links which can increase the entropy of that ACT are added sequentially. Fig. 4.7 exemplifies the link-adding process for m-trail t_0 in Solution 1 (see Fig. 4.2). Next, a valid m-trail is obtained (checked in **line 5-6**) if the tentative code meets the connectivity constraints:

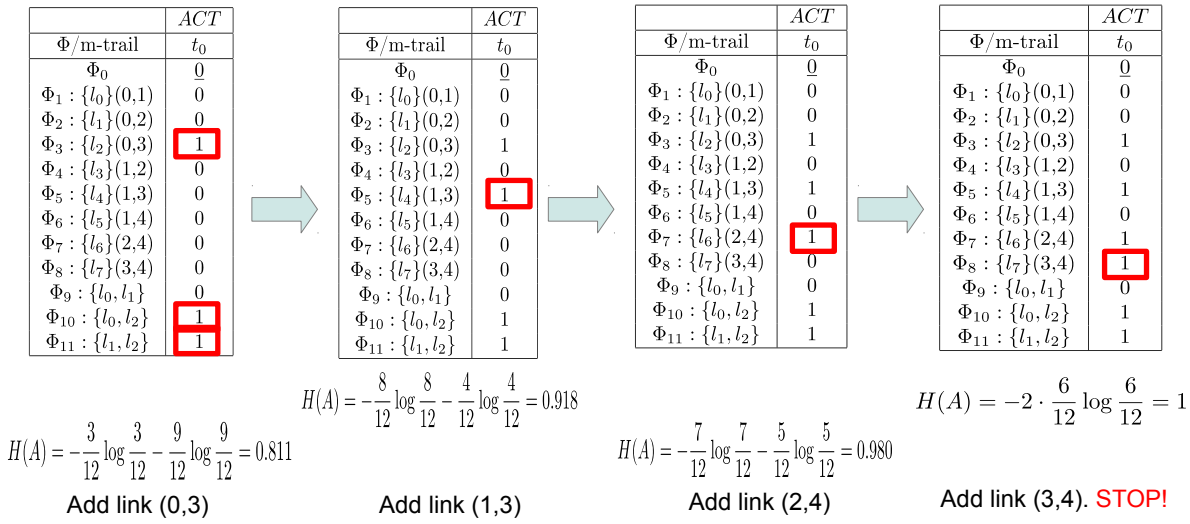


Figure 4.7: Sequentially Adding Entropy-increasing links

1. the subgraph induced by the code is connected
2. spans all/one unidentified MN(s) if Global/Local Sharing

Otherwise, an SRLG ϕ_s coded '0' is selected, which belongs to an *info-gaining* alarm code group in τ (**line 7**). For an info-gaining group, there exists at least two SRLGs which are coded as 0 and 1 in τ respectively.

Only new links not in ϕ_s will be added such that the connectivity constraint is met (**line 8-12**). By not adding links in ϕ_s, the refined trail code guarantees to increase the entropy of that ACT [25]. The detailed process is as follows. First, each link not in ϕ_s is saved to a hyperedge $h(j,k)$ if it connects the jth and kth Connected Component(CC) (**line 8-9**). Then, a new graph $G'(H,C)$ is constructed by randomly adding each CC as a node and each hyperedge containing at least one link as edge (**line 10-11**). Finally, a Depth First Search (DFS) is carried out to get a spanning tree on G'. During the process, whenever a hyperedge is added, the link with highest utility within that hyperedge is picked and the trail code τ is modified accordingly(**line 12**).

After an m-trail is successfully generated, we need to fill its trail code into the related ACTs and also record the current progress by changing the *Active Group List* maintained along with each ACT.

Generally speaking, an active group list is a double linked list whose element is a list of identically coded SRLGs. Fig. 4.9 shows three different active group lists which contains 1,

```
Input: G(V, E),Φ, M, msol
Result: trail code τ
begin
    /* Generate a tentative code first                                   */
1   τ ← 0
2   randomly shuffle the link candidates                    /* O(|E|)    */
    /* add link if entropy increases                                     */
3   foreach link candidate l do
4       if entropy increases then
            add the link to τ
                                                            /* O(|E|·|Φ|) */
    /* Refine trail code if it is invalid                                */
5   get Connected Component(CC) set C based on τ and M   /* DFS+Building Disjoint
    Sets :O(|E| + |V| + |V| log |V|)  */
6   if τ is invalid then
7       find a '0' coded SRLG φ_s in an info-gaining group of τ   /* O(|Φ|) */
8       foreach link ∉ φ_s do
9           save to hyperedge h(j, k) if connects jth and kth CC
                                                            /* O(|E| log |V|) */
10      randomly shuffle hyperedge set H                    /* O(|H|)    */
11      Construct a new Graph G'(C, H) using adjacency list /* O(|H| + |C|) */
12      Run modified DFS on G'                  /* O(|H| + |C| + |E|·|Φ|) */
        (pick max utility link for each chosen hyperedge and modify τ )
13  return τ
```

Figure 4.8: Algorithm: m-trail Generation(MG)

2 and 3 groups respectively and the process of splitting the groups. To access each group in $O(1)$ time, an array of group pointers are maintained which points to each group and a pointer is set to NULL if that group is deleted.

Fig. 4.10 describes how to split the active alarm code groups and save the results to an m-trail partial solution $msol$, after an m-trail τ is generated. The main idea is as follows. Each group in the current active group list $\mathcal{AG}$ is checked sequentially (**line 1**). Whenever a group is visited, the number of 1 coded SRLGs are counted (**line 2**). If not all SRLGs are identically coded (**line 3**), the current group is divided into two subgroups: the minority subgroup (sz) and the majority subgroup (bz) (**line 4**). Each subgroup contains the identically coded SRLGs and the majority subgroup contains more SRLGs than its counterpart. If the minority group contains at least two SRLGs (**line 5**), then a new group g' is appended to $\mathcal{AG}$ and the minority SRLGs in g are moved to g'. Accordingly, the group ids for each moved SRLG and the group sizes for g and g' are updated (**line 6-8**). Otherwise, the minority SRLG are labelled as identified the group size of g decreases by one (**line 9-10**). Finally, if the group size of the adjusted majority group is equal to one, it is removed from $\mathcal{AG}$ (**line 11**).

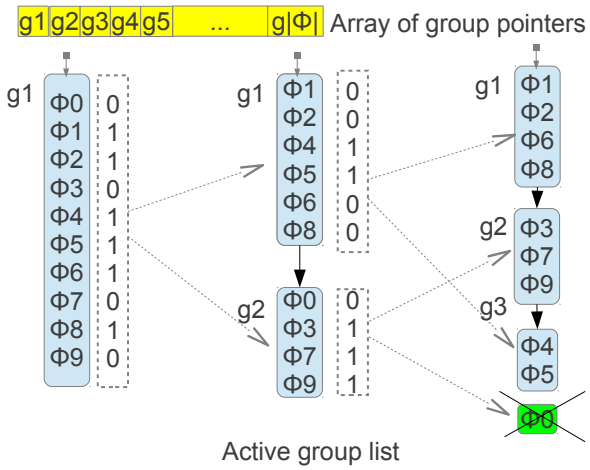


Figure 4.9: Splitting Group Example

Input: τ, $\mathcal{AG}$
Result: msol
begin

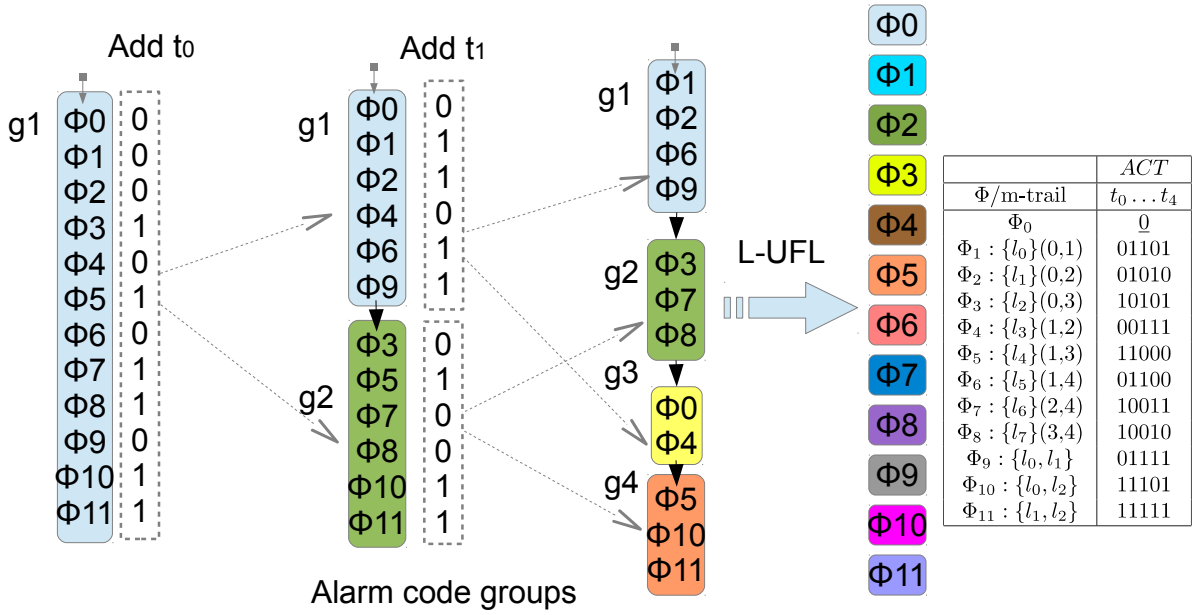
/* $O(|\Phi|)$ */

1 **foreach** *group* $g \in \mathcal{AG}$ **do**
2 count # of 1s in g based on τ
3 **if** $0 < \#of1 < size(g)$ **then**
4 $sz \leftarrow \min(\#1, \#0)$; $bz \leftarrow size(g) - sz$
5 **if** $sz > 1$ **then**
6 Append a new group g' to $\mathcal{AG}$, record position
7 move minority SRLGs in g to g'
8 update group size(g,g') and group ids for moved SRLGs
9 **else**
10 Remove the minority SRLG from g
 update group size(g), label the SRLG as identified
11 **if** $bz = 1$ **then**
 remove g from $\mathcal{AG}$,label the SRLG as identified

Figure 4.10: Algorithm: Active Group Splitting (AGS)

See Fig. 4.9 for a detailed example to split groups based on two trail codes: 0110111010 and 0001110101, on an active group list with ten SRLGs.

Fig. 4.11 demonstrates how Solution 1 shown in Fig. 4.3 is obtained through group-splitting.



Φ/m-trail	ACT $t_0 \dots t_4$
Φ_0	0
$\Phi_1 : \{l_0\}(0,1)$	01101
$\Phi_2 : \{l_1\}(0,2)$	01010
$\Phi_3 : \{l_2\}(0,3)$	10101
$\Phi_4 : \{l_3\}(1,2)$	00111
$\Phi_5 : \{l_4\}(1,3)$	11000
$\Phi_6 : \{l_5\}(1,4)$	01100
$\Phi_7 : \{l_6\}(2,4)$	10011
$\Phi_8 : \{l_7\}(3,4)$	10010
$\Phi_9 : \{l_0, l_1\}$	01111
$\Phi_{10} : \{l_0, l_2\}$	11101
$\Phi_{11} : \{l_1, l_2\}$	11111

Figure 4.11: Group Splitting Process of Solution 1

Input: $G(V, E)$,Φ, M
Result: $msol$
begin
1 **while** *L-UFL unachieved at some MNs* **do**
2 randomly select an unidentified MN
3 generate an info-gaining m-trail (MG)
`/* global sharing: m-trail should span all unidentified MNs */`
`/* local sharing: m-trail just pass through the selected MN */`
4 update the alarm code groups (AGS), remove identified MNs
5 save the m-trail into $msol$
6 return $msol$

Figure 4.12: Algorithm: achieving L-UFL with m-trails

4.5 Monitoring with S-LPs

In this Section, a heuristic [25] is built upon the m-trail generation procedure (MG) and the active group splitting procedure (AGS) introduced in Section 4.4, to solve the L-UFL problem with S-LPs(m-trails).

4.5.1 L-UFL Heuristic with m-trails

As shown in Fig. 4.12, the proposed heuristic keeps on generating m-trails **(line 3)** which can differentiate more SRLG pairs at an MN or multiple MNs, thereby to achieve L-UFL eventually. Both the m-trail global-sharing and local-sharing algorithm can be applied for m-trail generation as Fig. 4.12 illustrates ([25] adopts m-trail global-sharing). After an m-trail is generated, the alarm code groups and unidentified MNs are updated accordingly **(line 4)** and the m-trail is saved to current solution *msol* **(line 5)**.

Theorem 1. *The algorithm shown in Fig. 4.12 converges to achieve L-UFL.*

Proof. First, each m-trail generated forms a connected subgraph spanning all unidentified MNs or one selected MN. Moreover, each m-trail reduces the ambiguity of at least one alarm code group at an unidentified MN. Consequently, each m-trail improves entropy gradually and eventually rules out ambiguity. □

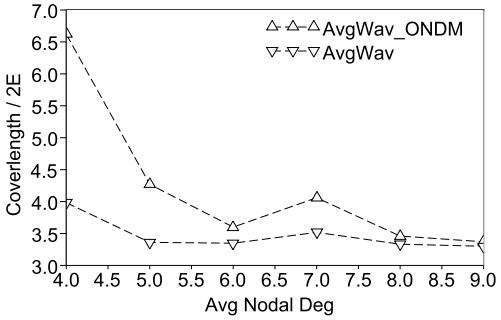
4.5.2 Simulation

Coverlength ratio $Wav = \omega/(2E)$ and trail ratio $T_r = J/(\lceil \log_2 |\Phi| \rceil)$ are used as two metrics to evaluate the performance of the proposed m-trail allocation algorithm, where ω, E and J denote coverlength, the number of links in the network, and the number of required m-trails, respectively.

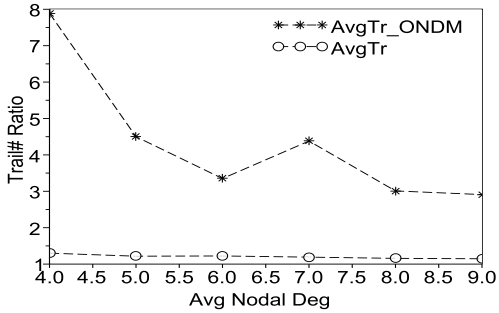
To check the performance of the algorithm shown in Fig. 4.12 in general networks, we use the method in [86] to generate 20-nodes and 30-nodes random networks (one 20-node/30-node random topology for each average nodal degree as shown in Fig. 4.14 and Fig. 4.15 ; each plotted point represent 40 simulation results for a given topology), where all nodes are defined as MNs. The average nodal degree ranges from 4 to 9. Note that *AvgWav* and *AvgTr* in Fig. 4.13 are the average coverlength ratio and trail ratio, whereas *BestWav* and *BestTr* denote the best results obtained.

As indicated by Fig. 4.13a-4.13b , the new algorithm outperforms previous algorithm in [24] (labelled "ONDM" in figures) in reducing the coverlength and trail ratio to identify single-link failure for 20 node networks. Compared to [25], the heuristic proposed in [24] only deals with single link failures, which greedily tackles the L-UFL problem MN by MN through random code assignment, trail splitting and other techniques. [2]

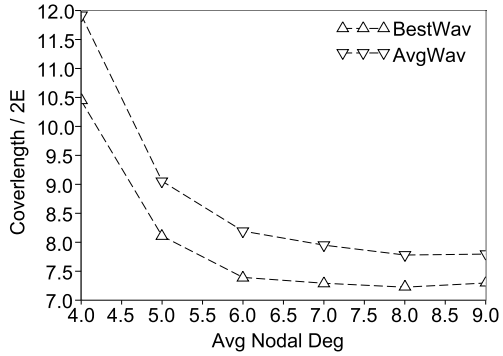
[2]the proposed algorithm is further validated with the published m-cycle scheme[4, 37] as shown in Fig. 4.21a to Fig. 4.21d.



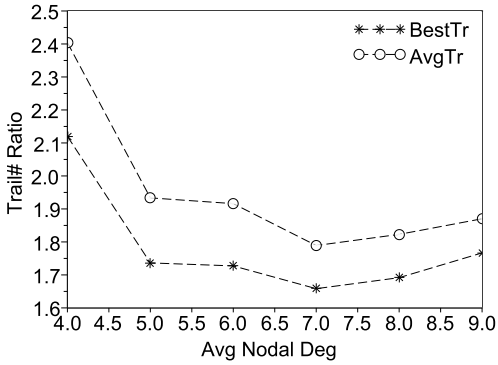
(a) Wav: 20-Node(single)



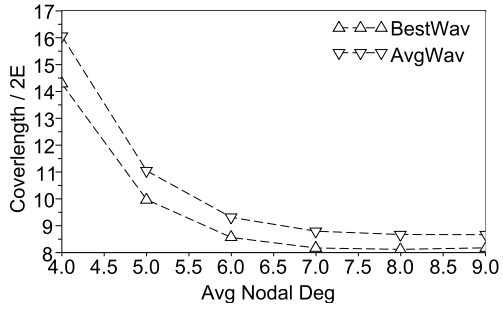
(b) Tr: 20 Node(single)



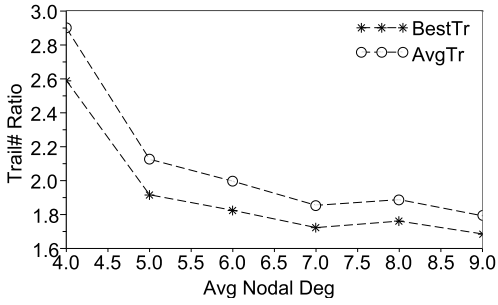
(c) Wav: 20-Node(dual)



(d) Tr: 20 Node(dual)



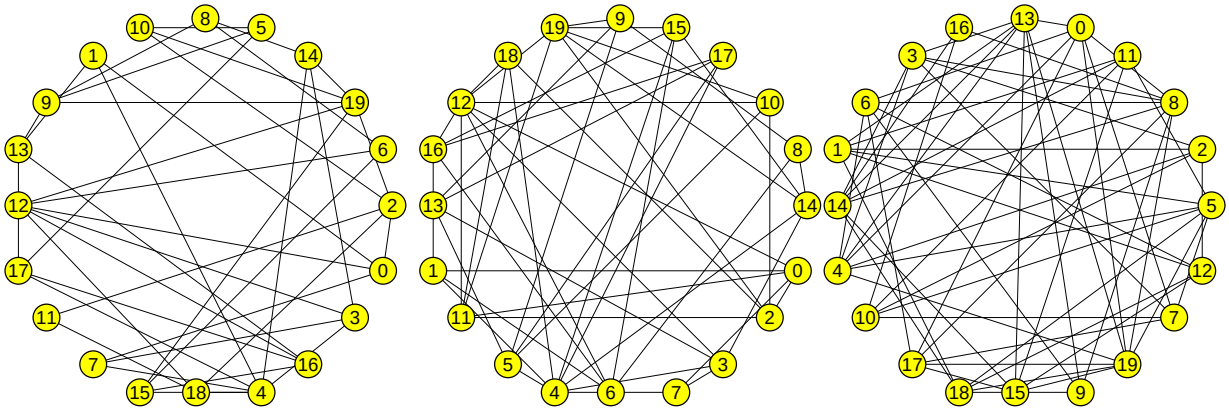
(e) Wav: 30-Node(dual)



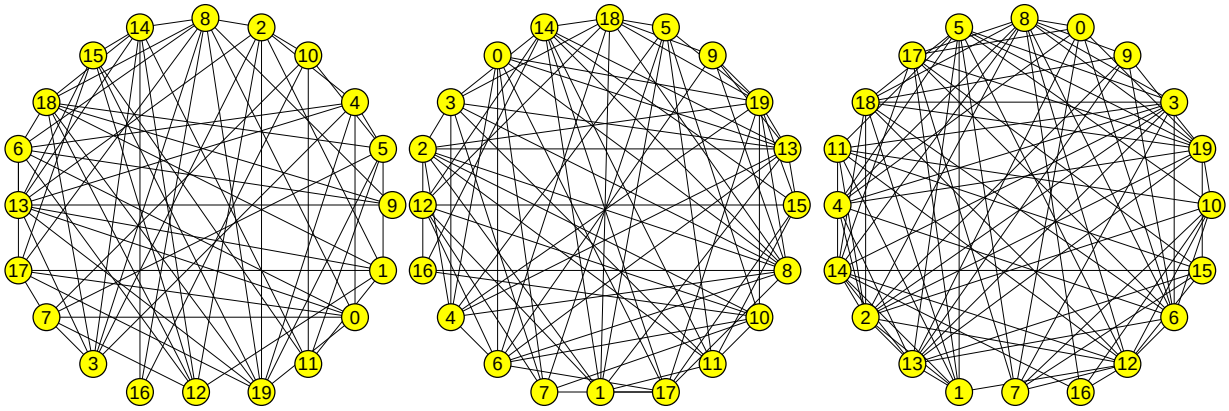
(f) Tr: 30-Node(dual)

Figure 4.13: Simulation Results for random networks

Figs. 4.13c-4.13d and Figs. 4.13e-4.13f deal with failure events (up to 2 links) for 20-node and 30-node networks respectively. Thanks to m-trail global sharing and enhanced "Min Wavelength Max Information" principle proposed in [25], trail ratio is almost optimal for single-failure L-UFL tasks and usually less then $2(\lceil \log_2 |\Phi| \rceil)$ for dual failure case which



(a) 20-node: deg 4 (b) 20-node: deg 5 (c) 20-node: deg 6

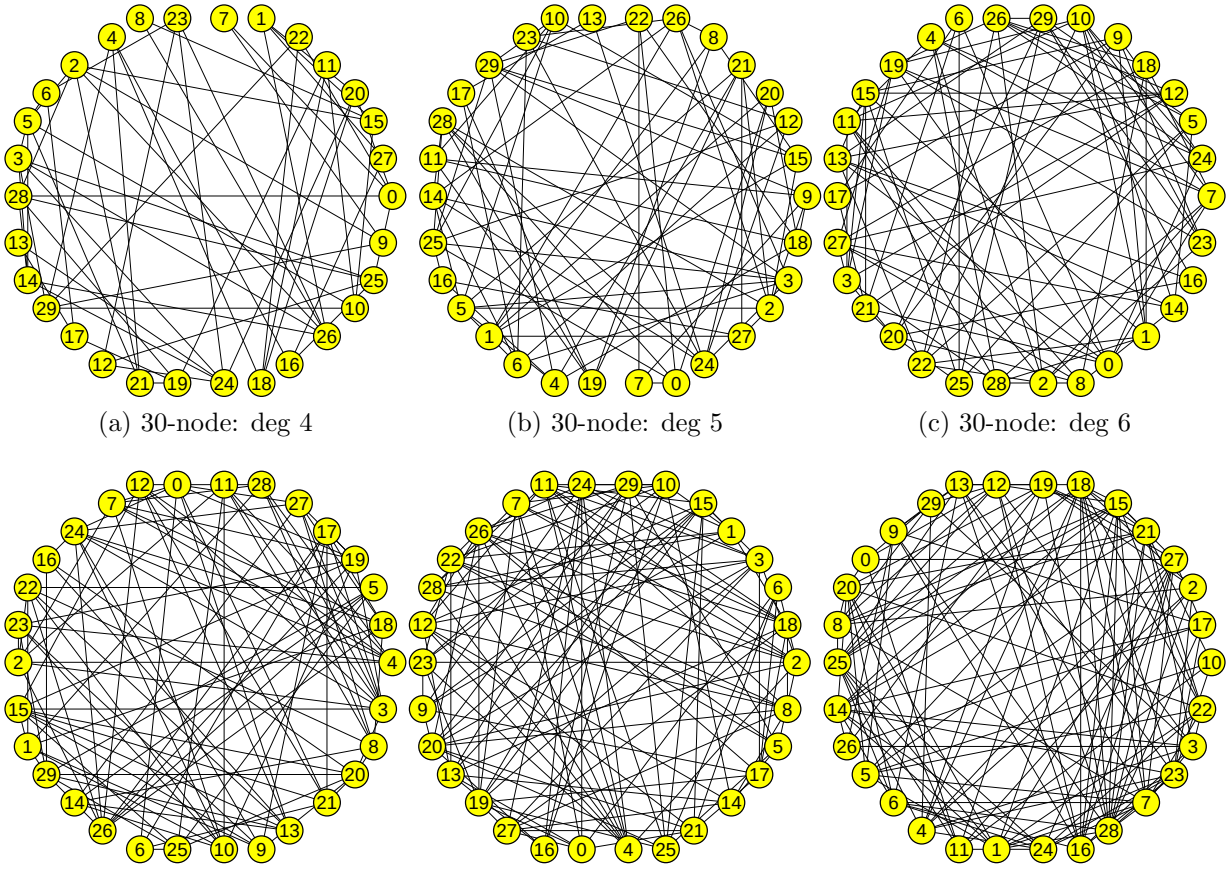


(d) 20-node: deg 7 (e) 20-node: deg 8 (f) 20-node: deg 9

Figure 4.14: 20-node Random Topologies

is also acceptable. In addition, as the average nodal degree increases, network topology is more connected and coverlength is suppressed because the difficulty of finding a valid trail is reduced.

We now check the running time of the algorithm. On a work station with 2.8G CPU and 1G memory, the m-trail allocation algorithm takes minutes to find 100 feasible dual-link solutions for a 30-node, 270-link network.



(a) 30-node: deg 4 (b) 30-node: deg 5 (c) 30-node: deg 6

(d) 30-node: deg 7 (e) 30-node: deg 8 (f) 30-node: deg 9

Figure 4.15: 30-node Random Topologies

4.6 Joint Monitoring with S-LPs and W-LPs

A novel algorithm for L-UFL, which jointly considers W-LPs and S-LPs, is explained in this Section. To begin with, two important procedures, i.e., the W-LP evaluation procedure (EVW) and the Partial Solution Production procedure (PPS), are discussed in Subsections 4.6.1-4.6.2. As will be explained in Subsection 4.6.3, the proposed algorithm first analyzes the monitoring contribution of W-LPs by calling EVW and saves useful W-LPs as an initial partial solution. Afterwards, PPS are called in a iterative fashion such that a complete solution to the L-UFL problem can be obtained.

Input: M,T^w
Result: T^1
begin
1 $T^1 \leftarrow \varnothing$; Randomly shuffle W-LPs in T^w
2 **foreach** *W-LP* $t \in T^w$ **do**
3 **for** $i := 0$ *to* $|M| - 1$ **do**
4 get the upstream links for the i_{th} MN in t
5 get the trail code for t in ACT_i by bitwise-ORing the IVs of the upstream links
6 **if** *trail code increases* $H(ACT_i)$ **then**
save the trail code to ACT_i
7 **if** *trail code of* t *is saved in some ACT* **then**
add t to T^1
8 return T^1

Figure 4.16: Algorithm: Evaluating W-LPs for L-UFL (EVW)

4.6.1 Evaluating W-LPs (EVW)

In this Subsection, we first prove that the problem of choosing an optimal subset of W-LPs for network monitoring is NP-complete. Next, a heuristic procedure EVW used by our L-UFL algorithm is analyzed in detail.

Minimum W-LP Set (MWS) Problem:

Given a network graph G, a set of Monitoring Nodes, a set of SRLGs $\Phi = \{\Phi_0, \Phi_1 \ldots\}$, a set of W-LPs T^w and an integer q, decide whether there exists a subset $T_s^w \subseteq T^w$ such that $|T_s^w| = q$ and the entropy ratio of the solution given by T_s^w is the same as the ratio of the solution given by T^w.

Redundant Monitor Deactivation Problem (RMDP):[57] (rephrased)

Given an integer m, a set of k monitors $M = \{M_1 \ldots M_k\}$ and a set of n faults $\Phi = \{\Phi_0, \ldots \Phi_{n-1}\}$ such that the ACT based on M achieves UFL (i.e. each fault has a unique alarm code; specifically, the alarm code of Φ_0 is a zero vector which corresponds to no failure). Decide whether there exists a subset $D \subseteq M$, where D is the set of redundant monitors to be deactivated such that the ACT based on $M \setminus D$ still achieves UFL when $|D| = m$.

Theorem 2. *The MWS problem is NP-complete.*

Proof. It is easy to see that the MWS problem is in NP. Given a solution to the MWS Problem (i.e. a set of W-LPs, denoted as T_s^w), we can easily verify if $|T_s^w| = q$ and the entropy ratio is met.

To show the MWS problem is NP-hard, we will reduce the RMDP, which is NP-complete, to the MWS problem.

Let $M = \{M_1 \dots M_k\}$, a set of n faults $\Phi = \{\Phi_0, \dots \Phi_{n-1}\}$ and an integer m be an arbitrary input to the RMDP problem. We construct an instance for MWS as follows.

First, create a complete graph K_n with n nodes. Let $G = K_n$ and $q = k - m$. Then, arbitrarily pick one node ,say N_u as the only MN. Hence we only have one ACT. Map each fault Φ_i $(i = 1 \dots n-1)$ in RMDP to a distinct edge incident to N_u in K_n. Let these $n-1$ edges and the no failure case be the n SRLGs under consideration. For each monitor M_i, we create a W-LP which passes through the same subset of the edges as in the ACT of the original RMDP instance. Note that as each W-LP is composed of edges incident to N_u, these edges must be connected and a valid W-LP can be formed. Clearly, the entries in the ACT created for our MWS instance is the same as the ACT for the RMDP instance. Moreover, the created W-LP set T^w achieves UFL (i.e. entropy ratio is 1) since the monitors in M achieves UFL by definition.

Claim: the MWS instance can achieve UFL with $q = k - m$ W-LPs if and only if m monitors can be deactivated for the RMDP instance. (As the entries in the ACTs for both instances are exactly the same, proof directly follows.)

Based on the above claim, the MWS problem is NP-hard. Since the problem is also in NP, it is NP-complete.

□

Let the set of W-LPs be denoted as T^w, the algorithm shown in Fig. 4.16 saves the W-LPs as a partial solution to an L-UFL problem and returns a set of m-trails T^1 corresponding to that solution. At the beginning, T^1 is empty and the W-LPs are randomly shuffled (**line 1**). Then, each W-LP, denoted by t is considered sequentially (**line 2**); and the algorithm iterates through all the MNs to quantify the differentiation contribution of each t (**line 3**). It takes three steps to analyze the differentiation contribution of t regarding each MN. First, the upstream links of each traversing W-LP of each MN are obtained(**line 4**). This can be done by traversing along t from the destination node until hitting the MN; and the unvisited links are the upstream links. Second, the trail code for t can be obtained by the bitwise-OR operation on all the link-SRLG incidence vectors of the upstream links (**line 5**). Third, the trail code is saved to the ACT of the MN if it increases the entropy of that ACT(**line 6**). As long as the trail code of t has been saved to an ACT, t is added to T^1(**line 7**).

```
Input: Ĥ_cur, Ĥ_exp
Result: T_p
begin
1   | T_p ← ∅; copy current ACTs to BCTs
2   | while Ĥ_cur < Ĥ_exp do
3   |     | add an entropy-increasing m-trail to T_p
4   |     | save the new m-trail to the BCTs
5   |     | Ĥ_cur ← entropy ratio of BCTs
6   | return T_p
```

Figure 4.17: Algorithm: Producing Partial Solution (PPS)

4.6.2 Producing Partial Solution (PPS)

In the process of generating a complete solution, the set of m-trails in the solution of the current iteration and its entropy ratio is denoted as T_c and $\hat{H}_{cur}$ respectively. By comparing $\hat{H}_{cur}$ and the expected entropy ratio, denoted as $\hat{H}_{exp}$, the algorithm in **Fig. 4.17 aims to produce a partial solution with m-trail set T_p, such that the solution yielded by the m-trail set $T_p \cup T_c$ has an entropy ratio no less than $\hat{H}_{exp}$.**

Initially, T_p is an empty set and a copy of current ACTs, namely BCTs, is stored(**line 1**). As long as the current entropy ratio $\hat{H}_{cur}$ of BCTs is less than the expected ratio $\hat{H}_{exp}$ (**line 2**), a new m-trail which guarantees to increase the entropy ratio is generated by any possible method (e.g. m-trail method in [25] or m-cycle method in [4]) and added to T_p (**line 3**). After that, the trail code of the new trail is filled into the BCTs accordingly (**line 4**) and the entropy ratio for BCTs is recalculated (**line 5**). Finally, a new iteration starts over again if $\hat{H}_{cur} < \hat{H}_{exp}$ (**line 2**); otherwise, the set of m-trails T_p is returned (**line 6**).

4.6.3 Proposed Algorithm for L-UFL

The algorithm shown in **Fig. 4.19 generates a complete m-trail solution to an L-UFL problem in an iterative approach**, where n specifies the number of partial solutions generated when adding S-LPs, s indicates how much entropy ratio should be increased in an iteration ($0 < s \leq 1$), M is the set of MNs, T^w is the set of W-LPs, variable i counts the number of iterations that have gone through, and T_c represents the set of m-trails in solution of the current iteration.

In the first iteration, the W-LPs are analyzed by calling the function namely EVW and only the W-LPs which can increase the entropy ratio are stored in T_c as the initial solution (**line 1**). Next, the entropy ratio of the solution in the current iteration, denoted by $\hat{H}_{cur}$, is computed (**line 2**). If $\hat{H}_{cur}$ is less than 1, then the algorithm begins generating S-LPs(**line 3**). In iteration i ($i \geq 2$), the expected entropy ratio, denoted as $\hat{H}_{exp}$, is

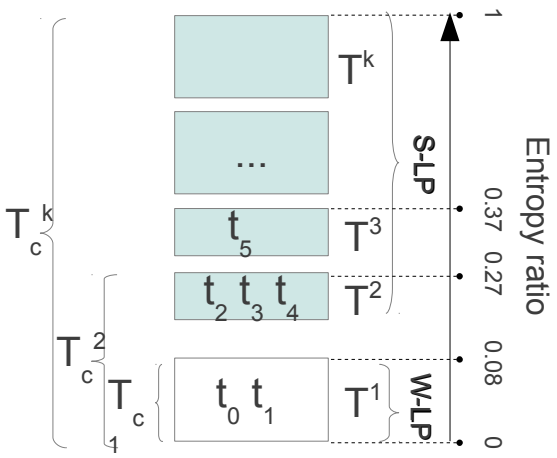


Figure 4.18: L-UFL algorithm with k iterations ($s = 0.1$,T_c^i is an abbreviation for "T_c in iteration i")

Input: n,s,M,T^w
Result: T_c
begin
1 $i \leftarrow 1$; $T_c \leftarrow EVW(M, T^w)$
2 $\hat{H}_{cur} \leftarrow$ entropy ratio of current ACTs
3 **while** $\hat{H}_{cur} < 1$ **do**
4 $i \leftarrow i + 1$; $\hat{H}_{exp} \leftarrow \min\{\hat{H}_{cur} + s, 1\}$
5 call $PPS(\hat{H}_{cur}, \hat{H}_{exp})$ n times to get n partial solutions which achieve $\hat{H}_{exp}$
6 pick a partial solution named T_p^* such that $T_c \cup T_p^*$ has highest utility
7 $T^i \leftarrow T_p^*$; adjust ACTs; $T_c \leftarrow T_c \cup T^i$
8 $\hat{H}_{cur} \leftarrow$ entropy ratio of ACTs
9 **return** T_c

Figure 4.19: Algorithm: Achieving L-UFL with S-LPs and W-LPs

computed based on $\hat{H}_{cur}$ and the parameter s (**line 4**). Then, a new partial solution T^i is stored on top of the existing solution T_c such that the solution yielded by $T^i \cup T_c$ has an entropy ratio no less than $\hat{H}_{exp}$. Note that $T^i \cup T_c$ has the highest utility among n partial solutions generated by PPS (**line 5-6**) and T_c is set to $T^i \cup T_c$ (**line 7**). The next iteration starts if the recalculated entropy ratio (**line 8**) is not 1; otherwise, a complete solution is obtained (**line 9**). Fig. 4.18 gives an example when the algorithm finishes in k iterations and s is set to 0.1. In iteration i ($i \geq 2$), the solution for iteration i, i.e. T_c^i, is obtained by stacking T^i over the existing solution T_c^{i-1} and the entropy ratio increases by at least s ($s = 0.1$ in this example).

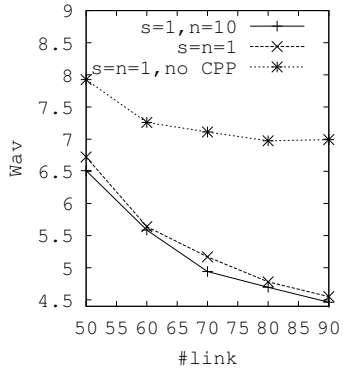
4.6.4 Simulation

Average coverlength ratio $Wav = \omega(T)/(2|E|)$ and trail ratio $Tr = |T|/(\lceil \log_2 |\Phi| \rceil)$ are used as two metrics to evaluate the results, where $\omega(T)$, $|E|$ and $|T|$ denote coverlength, the number of links in the network, and the number of required m-trails, respectively. Without confusion, these ratios are computed according to the best results obtained for different topologies. We implement the m-trail generation in three different ways including the m-cycle local-sharing scheme, m-trail global-sharing scheme and the m-trail local-sharing scheme. Besides, all simulations aim to achieve NWL-UFL of upto 2-link failures. Except for Fig. 4.20e-4.20f , the ratio γ in the utility function is set to 0.2.

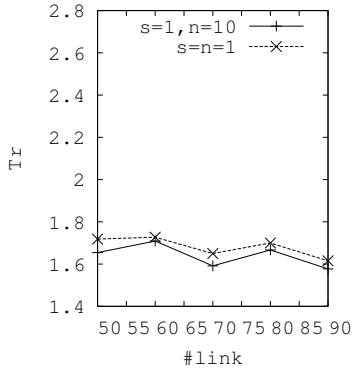
The m-cycle local-sharing scheme [4][37] is easy to implement, which generates cycles passing through a MN by two shortest paths. However, the scheme only works on 4-edge connected graphs for upto 2-link UFL[4]. Thus, we use the method in [65] to generate 20-nodes random networks and only pick $4-$edge connected topologies. The link number for these topologies ranges from 50 to 90.

Contrary to the m-cycle scheme which intends to differentiate one pair of SRLGs by an m-cycle, both the global and local m-trail sharing schemes try to differentiate as many SRLG pairs as possible. To minimize the wavelength consumption of a particular m-trail, the algorithm for Chinese Postman Problem is used for both m-trail schemes to calculate the optimal S-LP.

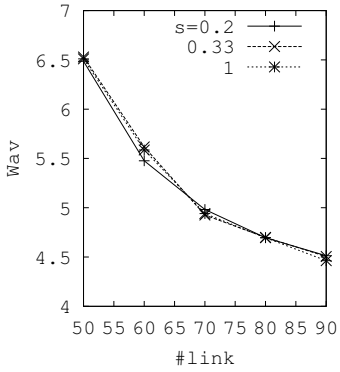
As regard to W-LP generation, a $|V|$ by $|V|$ binary matrix is used to represent the traffic demand of W-LPs. The entry at row i column j ($i \neq j$; diagonal entries are set to 0) is set to 1 if there should be a W-LP starting from node i and ends at node j; Otherwise, it is set to 0 . Suppose η denotes the traffic load, then $\eta = 1$ corresponds to the full traffic matrix with $|V|^2 - |V|$ non-zero entries. Given a particular traffic load , $\lceil \eta \cdot (|V|^2 - |V|) \rceil$ W-LPs (simple paths) are generated by Random Depth First Search(RDFS) . RDFS differs



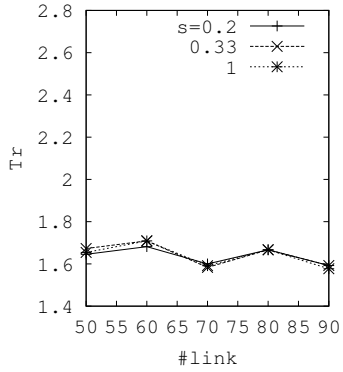
(a) *Wav*: n varies, $s = 1$, $\gamma = 0.2$



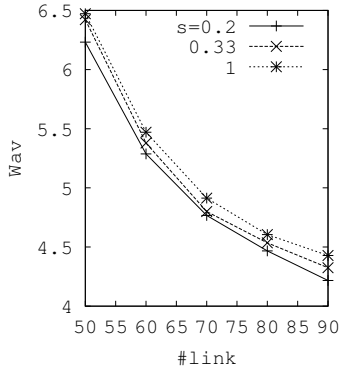
(b) *Tr*: n varies, $s = 1$, $\gamma = 0.2$



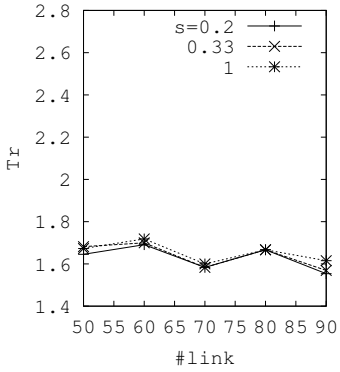
(c) *Wav*: s varies, $n = 10$, $\gamma = 0.2$



(d) *Tr*: s varies, $n = 10$, $\gamma = 0.2$



(e) *Wav*: s varies, $n = 10$, $\gamma = 5$



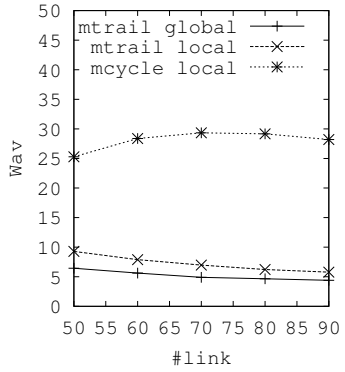
(f) *Tr*: s varies, $n = 10$, $\gamma = 5$

Figure 4.20: global m-trail sharing with different n, s and γ values

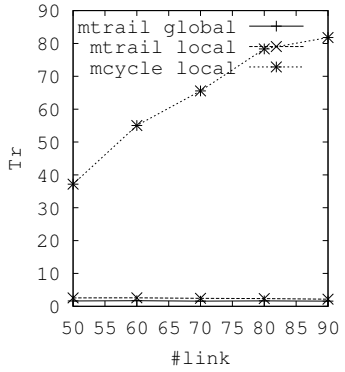
from depth first search in that it randomly selects the next hop from the adjacency link list instead of following a fixed order.

For each network topology, 10 random traffic demand matrices for each given load are constructed ($\eta \in \{0.3, 0.6, 0.9\}$) . Afterwards, a set of W-LPs are generated accordingly for each matrix while the W-LP set with median entropy ratio is picked as the input to the algorithm in Fig. 4.19.

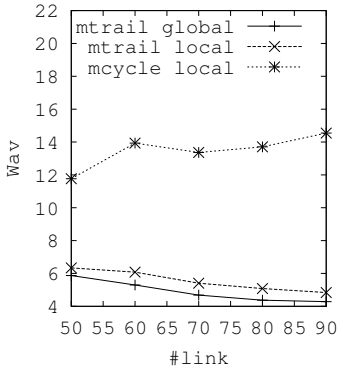
To check the correctness and performance of L-UFL algorithm, we adopt the m-trail global-sharing scheme in [25] to generate m-trails. Fig. 4.20a-4.20b demonstrates the usefulness of the utility function by changing the parameter n; the notation "No CPP" in Fig. 4.20a means the wavelength link consumption of an m-trail is calculated by doubling



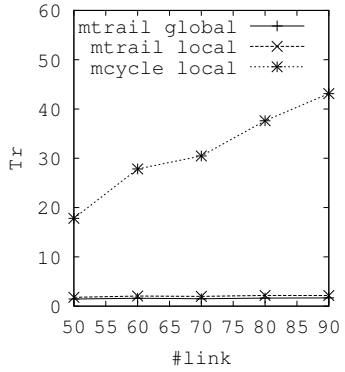
(a) *Wav*: method varies ($\eta = 0$)



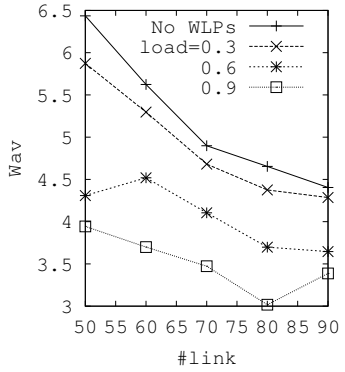
(b) *Tr*: method varies ($\eta = 0$)



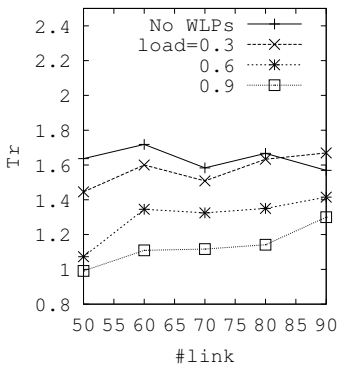
(c) *Wav*: method varies ($\eta = 0.3$)



(d) *Tr*: method varies ($\eta = 0.3$)



(e) *Wav*: load varies (m-trail global-sharing)



(f) *Tr*: load varies (m-trail global-sharing)

Figure 4.21: Comparison between different m-trail generation methods and the effects of WLPs ($n = 10$, $s = \gamma = 0.2$)

the number of undirected edges it traverses. Note that the setting of $s = n = 1$ and "No CPP" corresponds to our previous work in [25]. By setting a higher n value, the resource consumption can be slightly reduced. Fig. 4.20a also shows that when the wavelength-link consumption of an m-trail is computed by the Chinese Postman Algorithm on undirected graphs (see Chap. 15.7 in [58]), the coverlength can be significantly suppressed. Fig. 4.20c-4.20f examines the effect of parameter s. According to the simulation results, a small s value (so that the utility function is applied more frequently) leads to better performance. This effect is more obvious when γ is set to a higher value.

Fig. 4.21a-4.21d compare the performance of different sharing schemes. Fig. 4.21a-4.21b examine the difference when no W-LPs are considered while Fig. 4.21c-4.21d focus on the case when W-LPs with load 0.3 are present. Based on the simulation results, the m-trail global sharing scheme outperforms the other two in both coverlength and the number of m-trails while the performance gap is narrowed when W-LPs are considered

To study the affect brought by W-LPs, we further test the performance of m-trail global sharing under different W-LP loads. As shown in Fig. 4.21e-4.21f, with the increase of random W-LPs, the monitoring resource consumption of S-LPs are significantly suppressed.

As regard to the running time, on a work station with 2.8G CPU and 1G memory, the m-trail global-sharing scheme takes minutes to find 100 feasible dual-link solutions for a 20-node, 90-link network when no W-LPs are present. But for local-sharing schemes, it takes much longer as more m-trails have to be generated and the ACTs are not identical. When W-LPs are present, the global m-trail sharing computation time is similar to m-trail local-sharing as ACTs are usually not the same in that case.

To conclude, the monitoring resource consumption for signaling-free failure localization is greatly reduced with the help of the proposed utility function, S-LP routing based on Chinese Postman algorithm, as well as the W-LPs.

4.7 Conclusion

In this chapter, I introduced a signaling-free failure localization framework: L-UFL, under which a set of monitoring nodes can individually perform local unambiguous failure localization (L-UFL) for SRLGs by inspecting the locally available m-trail status. Two heuristics [25, 23] are explained to solve the L-UFL problem.

Section 4.5 presents the heuristic in [25] for L-UFL using S-LPs. The Enhanced *Min Wavelength Max Information* principle and other techniques help to reduce the number of m-trails and the coverlength of the solution. Simulations based on various network topologies showed that the proposed heuristic algorithm exhibits a satisfactory performance in minimizing the number of m-trails, coverlength and the running time of the algorithm.

Section 4.6 presents the novel approach in [23] for further reducing the monitoring resource consumption of the signaling-free monitoring trail allocation problem. The proposed approach is characterized by jointly considering the working lightpaths and newly added supervisory lightpaths as m-trails, where each intermediate node of an m-trail detects loss of light due to failure on any upstream link along the m-trail. We claim that this is the

first study that investigates L-UFL using S-LPs and WLPs. The problem of selecting an optimal set of working lightpaths for L-UFL was shown to be NP-complete and a heuristic was provided. Techniques were developed to solve the L-UFL m-trail allocation problem which iteratively constructs the desired m-trail solution using a novel entropy based utility function. Simulation results on various network topologies demonstrated the effectiveness of the proposed algorithm and its performance advantage against other methods.

Finally, Fig. 4.22 outlines the main ideas and techniques discussed in this chapter with a mind map.

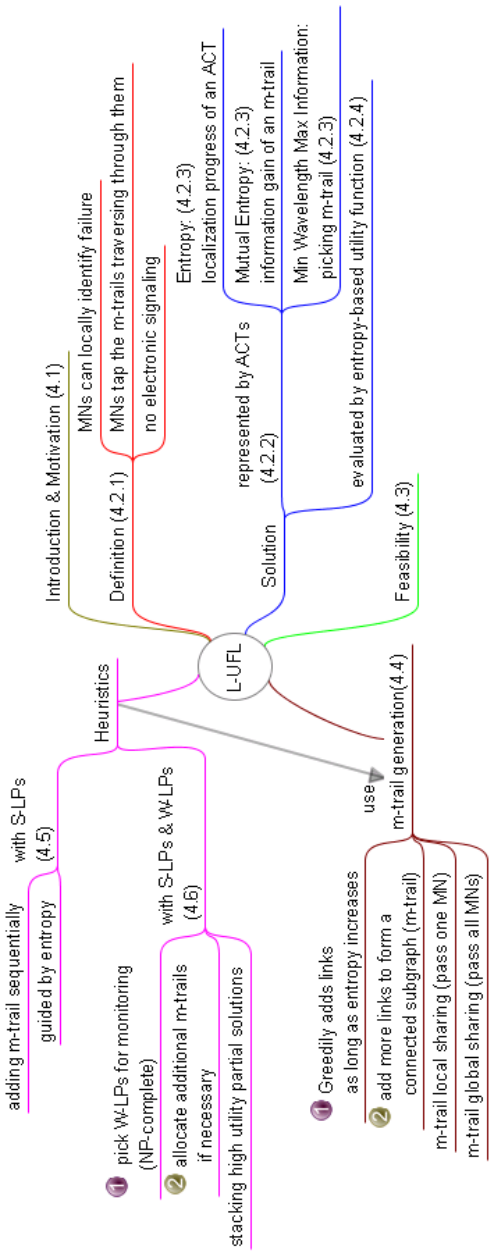


Figure 4.22: mind map for L-UFL

Chapter 5

Integrating Failure Localization with Survivable Design

5.1 Introduction

A general approach to increase availability of each connection is to pre-plan one or multiple *protection lightpaths* (P-LPs) for each *working lightpath* (W-LP). Fortunately, network failures are rare events thus it is a widely accepted strategy to share the allocated spare capacity among multiple P-LP(s) that are assumed not to be activated at the same time. This is also referred to as *shared protection*. In contrast to *dedicated protection*, a shared protection scheme relies on a suite of after-failure real-time mechanisms to restore the failed W-LPs, including failure localization, failure notification and device configuration; and the former two are also known as *fault management* defined under Generalized Multi-Protocol Label Switching (GMPLS) based recovery[45].

Numerous research efforts were addressed in the design of shared protection schemes, including shared backup path protection (SBPP), segment shared protection (SSP), link protection, failure dependent protection (FDP, or referred to as *path restoration*), and pre-configured protection [35, 18, 41, 51, 74, 52]. Two objectives are widely considered in their design when claimed in the context of all-optical mesh networks, namely *capacity efficiency* and *fault management complexity*. The former concerns the amount of consumed *spare capacity*, which is the capacity in terms of wavelength channels (WLs) reserved (but not necessarily configured) for the P-LPs; while the latter can by measured by *restoration time*, defined as the time period from the instant that the traffic is unexpected interrupted to that the affected traffic is completely restored.

In the spectrum of shared protection scheme design, there exists a compromise between capacity efficiency and fault management complexity, in which FDP and pre-configured protection (such as p-Cycle) are the two extremes. With FDP, each W-LP is assigned with multiple P-LPs, and one is activated for the restoration purpose according to the identified failure event. By allowing the reuse of the released working capacity for failure restoration (or called *stub release*), FDP has long been recognized with the optimal capacity efficiency and widely taken as the performance benchmark in the design of shared protection schemes. However, the FDP restoration process is subject to the highest control and signaling complexity that possibly yields the longest restoration time, mostly because the switching node has to precisely localize the failure event for real-time selection of a P-LP.

The other extreme is the pre-configured protection schemes that are free from any electronic signaling mechanism, such as 1+1 and p-Cycle. They require the minimum fault management complexity and thus achieves very fast restoration since the only after-failure real-time action is the reconfiguration of the two nodes responsible for switching and merging the affected working traffic. Thus in contrast to any other counterpart, the pre-configured schemes can minimize the fault management complexity and possibly be implemented completely in the optical domain. Such simplicity and fast restoration speed is gained nonetheless at the expense of consuming the highest redundancy.

The above observation naturally brings up an interesting question: *Can we restore an optical layer fault without relying on any electronic signaling, while achieving optimal capacity efficiency as with FDP?* The question is answered in this Chapter. In specific, the proposed framework of survivable routing incorporates a state-of-the-art optical layer monitoring system with the conventional spare capacity allocation task, in which a set of supervisory lightpaths, called *monitoring trails* (m-trails), is deployed and is allowed to reuse the spare capacity reserved for P-LPs. As a consequence, each node can instantly obtain the network failure status and start the predefined restoration process by monitoring the on-off status of the traversing m-trails.

The chapter firstly details how the proposed restoration process can be realized purely in the optical domain. Then, a novel static survivable routing scheme, namely Monitoring and Protection Joint Design Heuristic (MPJD)[22], together with a dynamic survivable routing scheme, namely Dynamic Joint design Heuristic (DJH)[27], are introduced. MPJD is featured as a failure dependent protection (FDP) scheme under static traffic patterns (i.e. W-LPs are prerouted for meeting the connection requests). MPJD is capable of jointly allocating P-LPs for protecting the given W-LPs, as well as m-trails, such that all W-LPs can be restored in an all-optical fashion. By contrast, DJH is an FDP scheme under dynamic traffic patterns, which can jointly allocating a W-LP and its P-LPs that satisfies an newly arrived connection request, as well as the m-trails to be newly added to

the network, such that the W-LP can be restored all-optically.

Since both the m-trails and W-LPs are lightpaths with constant optical signal, a W-LP that bears user traffic can also be used as an m-trail for some other nodes in the meantime, and vice versa. On the other hand, by reusing spare capacity, the amount of WLs dedicated for the m-trails can be further minimized.

Extensive simulation is conducted to evaluate the performance of the proposed survivable routing schemes under various network topologies and traffic loads. MPJD outshines two reported schemes: NWL-UFL restoration[62] and GNFL[61], regarding WL consumption, number of transmitters, monitoring requirement, and running time. Specifically, NWL-UFL restoration[62] requires each network node to localize all the SRLG failures no matter it is needed or not, unnecessary monitoring resource consumption was found and investigated in [61]. By using very short m-trails and in-band monitoring (i.e., taking the on-off status of the W-LPs into the monitoring plane) in GNFL[61], the roles of m-trails and W-LPs become interchangeable. Such flexibility and simplicity in the transport plane management is gained at the expense of taking an excessive number of transponders.

The blocking probability of DJH is compared with those of a couple of counterparts, namely PWCE [86] and DJHnt. The former is the only reported shared protection scheme that copes with dynamic traffic without relying on electronic signaling, and the later is a representative of conventional FDP based shared protection scheme without m-trails, which is taken as the benchmark indicating how far the proposed DJH scheme is away from the optimal due to the launching of m-trails. We will show that the DJH scheme only causes very limited additional redundancy in the presence of dynamic traffic, thereby leading to significant better performance than PWCE and similar to the case of DJHnt, especially under the scenarios with denser topologies and larger link capacity. In other words, the all-optical restoration of the proposed scheme can be gained with almost zero additional redundancy.

This chapter is organized as follows. Section 5.2 discusses the GMPLS based recovery and the basis of all-optical monitoring based on m-trails. Section 5.3 presents the proposed all-optical restoration framework and describes how it can achieve the said signaling-free restoration process. Section 5.4 formally defines the static and dynamic joint design problem and the Necessary Monitoring Requirements for achieving all-optical restoration. Section 5.5 and Section 5.6 present the proposed static survivable routing scheme MPJD and the dynamic survivable routing scheme DJH respectively. Section 5.7 concludes.

5.2 Background

5.2.1 GMPLS based Recovery

GMPLS provides a complete support of W-LP and P-LP setup in the optical domain. At a connection request arrival, the W-LP and its P-LP(s) are calculated based on a specific survivable routing scheme. One or multiple P-LPs are pre-planned for the restoration of the W-LP once unexpectedly interrupted, where the mapping between the failure events and the P-LPs should be defined; i.e., which P-LP is activated at a specific failure event to restore the W-LP. Then the ingress node launches a path setup message to configure the optical cross-connect (OXC) at each intermediate node along the W-LP, as well as the notification of the intermediate nodes along the P-LP(s) about the reservation. Note that under shared protection, the WLs along the P-LP(s) are reserved but not configured except for pre-configured protection schemes such as p-Cycle.

Fault management is defined under GMPLS as a set of sequentially performed after-failure real-time tasks, including failure localization, failure notification, and device configuration. The former two tasks (i.e., localization and notification) in the optical network domain are defined as a series of electronic signaling mechanisms. By applying Link Management Protocol coupled with a resource reservation protocol (such as RSVP-TE), each downstream node of a failed W-LP subject to loss of light (LOL) should send an alarm to its upstream node. After receiving the alarm, the upstream node checks the corresponding input port and forwards the alarm to further upstream if the node is also subject to LOL. Otherwise, the faulty link is determined in the downstream, and the upstream node initiates protection/restoration procedures.

Such a GMPLS based fault management process is subject to some weaknesses. Firstly, the alarms could be simultaneously issued by multiple downstream nodes, and the number of alarms is determined by the number of W-LPs traversing through the faulty link(s) and the length of the W-LPs. The numerous alarms could easily lead to an alarm storm in the control plane and bring risks of crashing the whole network. Secondly, the GMPLS fault management could be vulnerable to multi-link Shared Risk Link Group (SRLG) failure events, due to the fact that a node can only be aware of the faulty link which is in the upstream of any W-LP traversing the node, but has no way to get the status of a link that is not traversed by any W-LP. Therefore, when a multi-link SRLG fails, a node may only be able to identify the failure of a subset of the links in the SRLG, and thus may select a P-LP that is nonetheless subject to the failure, too. Thirdly, due to extensive electronic signaling mechanism and nodal processing, the GMPLS fault management may take hundreds of milliseconds for failure localization and notification, and the delay is

added up to the overall restoration time. Note that a slow restoration not only causes data loss but also imposes vicious impact on the upper network/transport layer protocols such as OSPF and TCP.

To make the GMPLS fault management process better fit in the optical domain, link-based monitoring [80, 73, 4, 12] has been considered such that every link is exclusively monitored via a single-hop supervisory lightpath (S-LP). Once a failure occurs, the monitors subject to LOL issue alarms to the network controller or the corresponding decision nodes (e.g., ingresses) for the subsequent restoration process.

Although being an effective solution for mitigating the awkward situation in the conventional GMPLS based approach, the link-based monitoring approach strongly relies on electronic signaling for alarm collection and failure notification, which leads to considerable control complexity and long restoration time. Thus, numerous solutions based on various assumptions and design premises were reported in the past decade.

5.2.2 Signaling-Free Failure Localization

Failure localization using multi-hop S-LPs, or referred to as monitoring trails (m-trails) in the following context without loss of generality, has been extensively studied in the past decade [79, 33, 69, 6, 72, 66, 21, 64]. Local Unambiguous Failure Localization (L-UFL) [5, 75, 23, 64] is an interesting scenario of m-trails, aiming at signaling-free failure restoration that can be purely operated in the optical domain. With a 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.

[75] further explored the idea in [5] by considering the situation where not only the terminating node but also an intermediated node of an m-trail can obtain its on-off status via lambda monitoring. The study attempted to enable L-UFL for a given set of nodes via an integer linear program, and discovered the fact that the total coverlength scales very well with the number of L-UFL capable nodes, mostly due to the sharing of on-off status among the nodes traversed by a common m-trail. Motivated by the result, [23] provided a novel heuristic approach to determine the set of m-trails in presence of a set of L-UFL capable nodes. [64] further investigated the case where all the nodes are made L-UFL capable under any single link failure, referred to as Network-wide L-UFL (NWL-UFL), in which an efficient heuristic was developed for allocating m-trails in a shape of spanning tree via link code swapping.

In [60], NWL-UFL was taken to facilitate all-optical and signaling-free failure localization. The basic idea is to leave the two post-failure tasks, namely fault management and

P-LP device configuration, to be autonomously performed in the optical domain without any aid of a multi-hop signaling protocol. To achieve this, the switching, intermediate, and merging nodes of a P-LP can start configuring their OXCs to form the required cross-connect right after the identification of the failure, thanks to NWL-UFL which allows every node to unambiguously localize any link failure. Thus, the affected W-LP can be switched over to the P-LP without waiting for the cross-layer signaling mechanisms as in GMPLS.

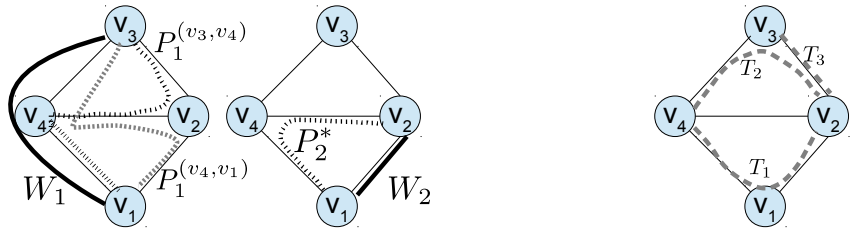
Although serving as viable solutions to the respectively considered scenarios, the previous schemes [23, 64, 60] are subject to poor scalability. Given that every node has to be L-UFL capable for all the links, the lengths of m-trails could possibly be as lengthy as $\frac{|E|}{2}$ hops. Note that a lengthy m-trail not only introduces a long monitoring latency in the resultant failure localization system, but also stands for a significant consumption on regenerators/amplifiers to support the function of all-optical monitoring. Accordingly, [67] resolved the problem by using short m-trails and having each node to be aware of the SRLG failure status only if the node needs to react to the failure.

Obviously, all of them are on static traffic, and there has not been any research for dynamic survivable routing which can deal with connection requests arriving one after the other without any prior knowledge of future arrival. Based on the above observations, in Section 5.5, I will introduce a novel heuristic for signaling-free restoration under static traffic which aims to reduce the monitoring resource consumption by defining the Necessary Monitoring Requirements(NMR) and other advanced techniques. In Section 5.6, a heuristic under dynamic traffic will be elaborated.

5.3 An Overview of All-Optical Failure Restoration

5.3.1 Introduction

Fig. 5.1 shows an example of the desired all-optical and signaling-free restoration process by using a 4-node 5-link topology. Two W-LPs denoted as W_1 and W_2 are present, each being provisioned with two physical lightpaths on the same route in opposite directions. W_1 is protected by two P-LPs, namely $P_1^{(v_3,v_4)}$ for link failure (v_3, v_4), and $P_1^{(v_4,v_1)}$ for (v_4, v_1) as in Fig. 5.1a, while W_2 is protected by a single P-LP denoted as P_2^* as in Fig. 5.1b. To ensure signaling-free restoration for W_1 and W_2, v_1 should be able to unambiguously identify the failure of (v_3, v_4), (v_4, v_1), and (v_2, v_1), such that W_1 can be switched over to $P_1^{(v_3,v_4)}$ (or $P_1^{(v_4,v_1)}$) when (v_3, v_4) (or (v_4, v_1)) fails and W_2 can be switched over to P_2^* when (v_2, v_1) fails. Thus, v_1 must be able to all-optically detect the failure status of the three links (v_3, v_4), (v_4, v_1), and (v_2, v_1).



SRLG	T_1	T_2	W_1	switch to
(v_3, v_4)	0	1	1	$P_1^{(v_3,v_4)}$
(v_4, v_1)	1	0	1	$P_1^{(v_4,v_1)}$
(v_2, v_1)	1	0	0	P_2^*
(v_3, v_2)	0	1	0	
(v_2, v_4)	0	0	0	

(d) RT at v_4

SRLG	T_2	T_3	W_1	switch to
(v_3, v_4)	1	0	1	$P_1^{(v_3,v_4)}$
(v_4, v_1)	0	0	1	$P_1^{(v_4,v_1)}$
(v_3, v_2)	1	1	0	
(v_2, v_1)	0	0	0	
(v_2, v_4)	0	0	0	

(e) RT at v_3

SRLG	T_1	W_1	W_2	switch to
(v_3, v_4)	0	1	0	$P_1^{(v_3,v_4)}$
(v_4, v_1)	1	1	0	$P_1^{(v_4,v_1)}$
(v_2, v_1)	1	0	1	P_2^*
(v_3, v_2)	0	0	0	
(v_2, v_4)	0	0	0	

(f) RT at v_1

SRLG	T_1	T_2	T_3	W_2	switch to
(v_3, v_4)	0	1	0	0	$P_1^{(v_3,v_4)}$
(v_4, v_1)	1	0	0	0	$P_1^{(v_4,v_1)}$
(v_2, v_1)	1	0	0	1	P_2^*
(v_3, v_2)	0	1	1	0	
(v_2, v_4)	0	0	0	0	

(g) RT at v_2

Figure 5.1: An illustrative example for the restoration process under the proposed framework (dashed square shows the embedded ACT)

Similarly, v_2 and v_4 should also be respond to the failure status of (v_4, v_1), (v_3, v_4) and (v_2, v_1), such that an all-optical failure restoration for W_1 and W_2 can be achieved. On the other hand, since v_3 is traversed by two P-LPs (i.e., $P_1^{(v_3,v_4)}$ and $P_1^{(v_4,v_1)}$), it needs to react to any failure upon W_1. Therefore, v_3 should be able to all-optically obtain the failure status of (v_3, v_4) and (v_4, v_1).

To enable each node to all-optically detect the failure status of the SRLGs of interest, W-LPs and m-trails should be properly allocated. As shown in Fig. 5.1c, three m-trails T_1, T_2 and T_3 are deployed, by which the Restoration Table (RT) for each node is formed as shown in Fig. 5.1d-g. Each RT is extended from the ACT at the node, which further maintains the mapping between each possible alarm code and the corresponding restoration configuration to be performed upon that failure event. In the embedded ACT at a node, each column corresponds to a W-LP or an m-trail passing through the node and a bit is set to 1 if the corresponding W-LP or m-trail traverses through some link(s) in that

SRLG. Each row of an RT on top of the separator corresponds to an SRLG failure state of interest, and the rows below the separator are the alarm codes that the node may see but does not care because they are caused by failures that the node does not need to respond. As introduced in [22], a node may just keeps the rows of the RT above the separator for saving memory.

Any alarm code in a RT above the separator should be different from any code below it, such that the interested failure states can be separated from the others. In addition, *any two alarm codes above the separator should be different if they have distinct restoration configurations* (i.e. different "switch to" values). In this way, each node can precisely decide which restoration action to be taken upon a failure event.

For example, v_1 keeps the RT as in Fig. 5.1f by observing the on-off status of T_1, W_1 and W_2, so as to uniquely identify the failure on (v_3, v_4), (v_4, v_1) or (v_2, v_1) of interest. If v_1 finds that T_1 and W_2 become unexpectedly off while W_1 is still on, an alarm code "101" is obtained; so the node will consider link (v_2, v_1) as failed by matching the third row of its RT and be ready to switch W_2 over to P_2^*. Meanwhile and in parallel, v_2 and v_4 will be able to identify the failure of (v_2, v_1) by matching the third row in their RTs as in 5.1g and d, respectively, and instantly configure their OXCs to support P_2^*. Thus W_2 can be restored in an all-optical and deterministic fashion upon the failure of (v_2, v_1).

In the above case all the nodes on P_2^* can all-optically localize the failure on (v_2, v_1) and immediately respond to the failure up W_2. Similarly, all the nodes on $P_1^{(v_3,v_4)}$ and $P_1^{(v_4,v_1)}$ can all-optically localize the failure on (v_3, v_4) and (v_4, v_1) respectively, and immediately respond to any link failure upon W_1. In other words, both W_1 and W_2 can be restored from any possible link failure in an all-optical manner. We define in such a situation W_1 and W_2, respectively, meets the Necessary Monitoring Requirement (NMR)[1]. It is clear that a W-LP meeting the NMR can be restored from a failure event in an all-optical and signaling-free manner; and by exercising the NMR, each network node can initiate a proper OXC configuration upon any predefined failure event merely based on the on-off status of the m-trails and W-LPs (if in-band monitoring is enabled) traversing through it.

5.3.2 Restoration Time Analysis

The after-failure real-time tasks under the conventional GMPLS based recovery includes *failure localization/isolation, failure notification* and *device configuration*. The former two

[1] formally defined in Subsection 5.4.2

rely on electronic multi-hop signaling, while the last one is mainly based on nodal processing. According to [46, 45, 16]: [2]

- Failure localization provides information about the location of the failure to the deciding entity — an entity that makes the recovery decision or selects the recovery resources;
- Failure notification phase is used:
  1. to inform intermediate nodes that W-LP(s)/link(s) failure has occurred and has been detected and
  2. to inform the recovery deciding entities — which participate in the recovery of the W-LPs/link(s) — that the corresponding W-LP/link is not available.

The three after-failure real-time tasks have to be performed in sequence, and their latencies are added up to the total failure restoration time. In the following we argue that the proposed framework can significantly simplify the real-time recovery process and be performed completely in the network optical domain.

Firstly, since the switching node of an interrupted W-LP is aware of the failure event, the time for failure localization is simply the propagation delay of the m-trails interrupted by the failure, while the time for failure notification is completely removed. Since the propagation of optical signals is deterministic in terms of its speed, the time for failure localization is deterministic. Further, since all the intermediate nodes of the P-LPs corresponding to the failure event can localize the failure event thanks to NWL-UFL, they can start configuring their OXCs based on the collected alarms without waiting for P-LP setup request from any other network entity. Thus the W-LP setup latency can be minimized as well. This leads to a completely all-optical and deterministic restoration process.

In specific, the total latency t_r can be modeled as:

$$t_r = t_l + t_d \ ,$$

where t_l and t_d is the time for failure localization, and the time for device configuration for the P-LP setup, respectively. Here, t_l is determined solely by the light propagation delay along the m-trails. On the other hand, t_d should be close to the time for OXC configuration at a single node, since all the intermediate nodes of the P-LP configure their OXCs almost in parallel.

[2] I rephrase it slightly using the terminologies in this thesis.

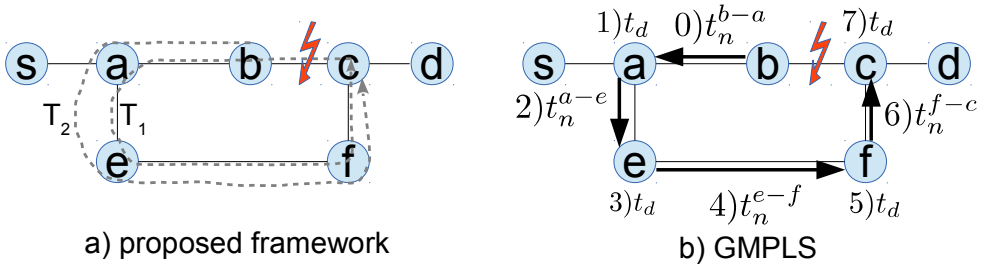


Figure 5.2: Restoration time analysis

Fig. 5.2a is an illustration for the restoration process under the proposed framework. Given a W-LP $a-b-c$ and a P-LP $a-e-f-c$ protecting it, let there be two m-trails $T_1 : c-b-a-e-f$ and $T_2 : b-a-e-f-c$ which help nodes along the P-LP, namely a, e, f, and c, to localize the failure at link $b-c$. With this, node a can identify the failure at $b-c$ by taking time t_l^{b-a} right after the failure of $b-c$, and node a looks up its RT and immediately switches over the traffic by taking another latency of t_d. Meanwhile, nodes e, f and c can also identify the failure by taking a propagation time t_l^{b-a-e}, $t_l^{b-a-e-f}$, and $t_l^{b-a-e-f-c}$, respectively, and they look up their RTs and immediately configure their OXCs to realize the P-LP which takes t_d. Thus, the total consumed time is the longest of the propagation time plus a single device configuration time.

In contrast, Fig. 5.2b shows the restoration process under the GMPLS protocol with link monitoring, under which the failure localization time can be ignored (failure is just one hop away from the detecting entities). When link $b-c$ fails, the GMPLS-based restoration needs to go through a series of signaling mechanism to complete the restoration. Firstly, node b which is adjacent to the failed link $b-c$, will notify the decision node a about the failure. Secondly, node a will initiate a recovery process by a device configuration process, in which the OXCs of a, e, f, c along the P-LP will be notified by a wake-up message, mostly done in a sequential manner. Thus, the GMPLS-based restoration process may take hundred of milliseconds of restoration time due to such a multi-hop signaling mechanism and cross-layer operation. In this example, t_r is $t_n^{b-a-e-f-c} + 4t_d$, where t_n is the time for failure notification based on electronic signaling.

5.4 Algorithm Preliminaries

5.4.1 Joint Design Problem

The detailed design goals of the optical restoration strategy via m-trails can be formulated as the joint design problem.

Definition 6. *Given a host of connection requests and a set of SRLGs, the* ***Joint Design Problem*** *asks to route Working Lightpaths (W-LPs), Protection Lightpaths (P-LPs) and M-trails for a network topology $G(V, E)$ such that:*

1. *W-LPs meet the connection requests;*
2. *all W-LPs are survivable through any single SRLG failure with the help of P-LPs;*
3. *Each node can react to any failure in parallel with sufficient information reported by m-trails;*
4. *resources consumed by W-LPs, P-LPs and m-trails are minimized.*

Note that Definition 6 will be used to validate the joint design solution generated by the algorithms introduced in Section 5.5 and 5.6. Specifically, Necessary Monitoring Requirements for signaling-free restoration are checked using the methods in Section 5.4.2.

In addition, we use the following cost function to evaluate a joint design solution:

$$Cost = \gamma \cdot (\#\text{ of m-trails}) + \sum_{e \in E} c_e \max\{m_e, p_e\} \tag{5.1}$$

where γ is a cost ratio for balancing the importance of an m-trail and a Wavelength Channel(WL), c_e is the predefined cost for using a WL on link e; m_e and p_e represents the number of WLs consumed by m-trails or P-LPs on the link e respectively. Thus, $\max\{m_e, p_e\}$ reflects that we allow m-trails and P-LPs to share WLs.

Definition 7. ***Static Joint Design Problem:*** *a joint design problem, for which all connection requests arrive simultaneously and W-LPs for meeting these requests are pre-computed.*

Definition 8. ***Dynamic Joint Design Problem:*** *a joint design problem, for which connection requests arrive one after the other without any prior knowledge of future arrivals.*

Table 5.1: P-LP assignment

P-LP assignment	$s_1^{(v_3,v_4)}$	$s_1^{(v_4,v_1)}$	s_2^*
SRLG/P-LP	3-2-4-1	3-4-2-1	1-4-2
ϕ_0	0	0	0
(v_3, v_4)	1	0	0
(v_4, v_1)	0	1	0
(v_2, v_1)	0	0	1
(v_3, v_2)	0	0	0
(v_2, v_4)	0	0	0

Table 5.2: Monitoring Requirements Matrices

node	1	2	3	4
SRLG	$s_1^{(v_3,v_4)}s_1^{(v_4,v_1)}s_2^*$	$s_1^{(v_3,v_4)}s_1^{(v_4,v_1)}s_2^*$	$s_1^{(v_3,v_4)}s_1^{(v_4,v_1)}$	$s_1^{(v_3,v_4)}s_1^{(v_4,v_1)}s_2^*$
ϕ_0	000	000	00	000
(v_3, v_4)	100	100	10	100
(v_4, v_1)	010	010	01	010
(v_2, v_1)	001	001	00	001
(v_3, v_2)	000	000	00	000
(v_2, v_4)	000	000	00	000

5.4.2 Necessary Monitoring Requirements

To get the necessary monitoring requirements, a binary matrix, called **"monitoring requirement matrix"(MRM)** is derived for each node. Consider the W-LPs and P-LPs shown in Fig. 5.1a-b for example. First, a bit vector s_i^j is assigned to each P-LP P_i^j. A bit is set to 1 if that P-LP is used to protect the corresponding SRLG; otherwise, it is set to 0. For instance, the vector $s_1^{(v_3,v_4)}$ for P-LP $P_1^{(v_3,v_4)}$ is 010000 as shown in Table 5.1, since it protects SRLG (v_3, v_4). After allocating all the bit vectors for P-LPs, all P-LPs' bit vectors passing through a particular node are stacked together. For instance, since node 1 is traversed by all three P-LPs , its monitoring requirements can be represented by stacking the bit vectors of these P-LPs as demonstrated by Table 5.2.

For simplicity, each row in a node's MRM is called a **"configuration code"** for an SRLG. For example, at node 1, SRLG (v_3, v_4) 's configuration code is 100. In this way, ***all SRLGs with identical configuration code fall into a configuration group***.

Thus, the *Necessary Monitoring Requirements* for achieving signaling-free restoration can be rephrased as follows:

Definition 9. ***Necessary Monitoring Requirement(NMR):*** *For each network node, two SRLGs in different configuration groups should be assigned different alarm codes.*

We can verify that the monitoring requirements in Table 5.2 (concisely represented using separators in Fig. 5.1d-g) are fully met by the m-trail allocation in Fig. 5.1c.

To quantify the monitoring requirement of a restoration strategy, we can compute the total number of SRLG pairs required to be distinguished by the nodes using (5.2):

$$\#(\text{SRLG pairs}) = \sum_{k \in V} \sum_{i<j} |g_i^k| \cdot |g_j^k| \tag{5.2}$$

where $|g_i^k|$ represents the size of the i_{th} *Configuration Group (CG)* at node k.

Note that ***NMR is a much looser constraint*** compared to other strategies as NWL-UFL [62] and GNFL [61]. The former regards each SRLG under consideration as a CG, thus requiring to distinguish $\binom{|\Phi|}{2}$ SRLG pairs at each node, where $|\Phi|$ denotes the number of SRLGs. The later looses the requirements under single link failures by defining a *neighbourhood* for each node. GNFL asks each link in the neighbourhood to form a CG and the remaining links to form another CG. However, *since the links in a neighbourhood may share restoration configurations, they may not need to be fully distinguished.*

Lemma 3. *NMR requires to distinguish no more SRLG pairs than GNFL and NWL-UFL.*

Proof. Note that the CGs defined by NMR are subdivided into small CGs under GNFL or NWL-UFL. Thus, for each CG g_i^k defined by NMR, we have: $|g_i^k| = \sum_l |r_{i,l}^k|$, where $r_{i,l}^k$ is a small CG defined by another strategy under comparison (i.e. GNFL or NWL-UFL). From Cauthy Inequality and (5.2), the SRLG pairs required by NMR would not be larger. □

We can use NWL-UFL as a benchmark to evaluate the performance of the others. The ***"pair ratio"*** is defined by:

$$\text{PairRatio} = \frac{\sum_{k \in V} \sum_{i<j} |g_i^k| \cdot |g_j^k|}{|V| \cdot \binom{|\Phi|}{2}} \tag{5.3}$$

Clearly, NWL-UFL's pair ratio is 1, which is a upper bound of all monitoring requirements.

5.5 Heuristic for Static Joint Design

This section introduces the proposed heuristic for the static joint design problem. Assume all the W-LPs are given and shortest-path routed. The basic idea of the proposed heuristic is to iteratively look for a solution with the highest utility (i.e. lowest cost according to (5.1)).

5.5.1 M-trail Allocation Procedure

The m-trail global-sharing scheme[25] introduced in Section 4.4, can be used by the proposed heuristic to generate m-trail solutions with two modifications:

1. the algorithm stops when NMR is met
2. when generating an m-trail, a priority queue is used to keep track of the current WL consumption on each link and the link with lower WL consumption is preferred.

Roughly speaking, the m-trail ***global-sharing*** scheme generates m-trails spanning all unidentified MNs (all nodes are MNs under NMR; an MN is unidentified if it has not met NMR). First, a tentative trail code is generated by greedily adding entropy-increasing links, based on the ACT of an MN. Next, a valid m-trail is obtained if the subgraph induced by the code is connected and spans all unidentified MNs. Otherwise, by not adding links in a '0' coded SRLG, a refined trail code is produced to increase the entropy of that ACT and meet the connectivity constraint (see [25] for more details).

5.5.2 Proposed Heuristic

The proposed heuristic is given in Fig. 5.3. At the beginning the current best solution S_B is initialized with the lowest utility $-\infty$ **(line 1)**. In each iteration starting at line 2 (totally *it* iterations), a joint design solution **(line 3-5)** is produced. The process to generate a joint solution takes two steps: the P-LP selection and the m-trail generation, which is performed one after the other. Firstly, a set of P-LPs (P_c) is selected (as in Fig. 5.4), which picks up useful P-LPs from a pool of precomputed k-shortest paths **(line 3)**. Next, the monitoring requirements for each node is derived from the P-LP solution using the method in Subsection 5.4.2 **(line 4)**. Then, nt m-trail solutions will be generated based on the WL consumption and the monitoring requirements of P_c. Combining P_c with the m-trail solutions, we get nt joint solutions **(line 5)**. Afterwards, the joint solution with highest utility S_H is selected by a utility function **(line 6)**. If S_H achieves higher utility than current best solution S_B, it overwrites S_B **(line 7)**.

With regard to P-LP selection, the algorithm in Fig. 5.4 selects at most *plim* P-LPs for each W-LP ω in the given W-LP set W sequentially. First, the P-LP candidates for ω are randomly shuffled **(line 4)**. Then, the algorithm continues testing different combinations of *plim* P-LPs until a feasible combination is found **(line 5-14)**. When testing a P-LP combination, the SRLGs which can interrupt ω are saved to a list Φ^{ω} **(line 7)**. Next, each

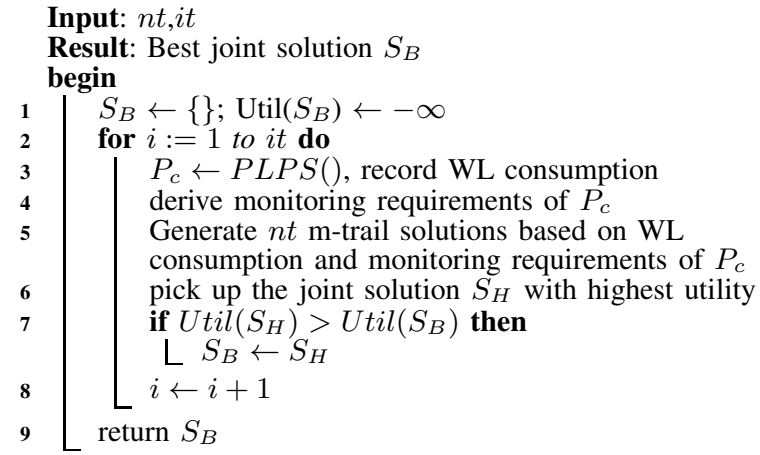


Figure 5.3: Algorithm: Monitoring and Protection Joint Design (MPJD)

P-LP in that combination is tested sequentially **(line 8-9)**; all SRLGs in Φ^ω which can be protected by current P-LP are removed in the process **(line 10-11)**. In case current P-LP ρ bypasses all SRLGs handled by previously selected P-LPs for ω, the algorithm will overwrite the P-LP set R^ω with ρ **(line 12)**. Otherwise, current P-LP will be appended to R^ω **(line 13)**. After processing all P-LPs in current combination, a feasible combination is found if the SRLG list Φ^ω is empty. Otherwise, the next combination is tested **(line 6)**.

5.5.3 Simulation Results

Wavelength Channel Consumption ratio $Wav = \omega/(2|E|)$ and trail ratio $T_r = J/(\lceil \log_2 |\Phi| \rceil)$ are used to evaluate the performance of the proposed heuristic, where ω, $|E|$ and J denotes the total WLs consumed for joint design, the number of links in the network, and the number of m-trails required, respectively. For cost function (5.1), γ is set to 5. As usual, the WLs consumed for monitoring is called *Coverlength*.

To check the performance of the heuristic in Fig. 5.3 (implemented using LEMON[2]) under various traffic loads (load= $\{0.2, 0.4, 1\}$), we generate 50 20-node random topologies with 50 links to 90 links (using the method in [65], implemented using [1]). Without confusion, all ratios plotted are the best results averaged over 10 different topologies. Full traffic load implies a bi-directional W-LP connecting any two nodes (e.g. 20 node network implies $\binom{20}{2}$ W-LPs). For each load ($load = \{0.2, 0.4\}$) and a given topology, a set of W-LPs are randomly picked from 10 W-LP sets generated by the shortest path algorithm. 10 P-LP candidates for each node pair are precomputed using k shortest path algorithm, as

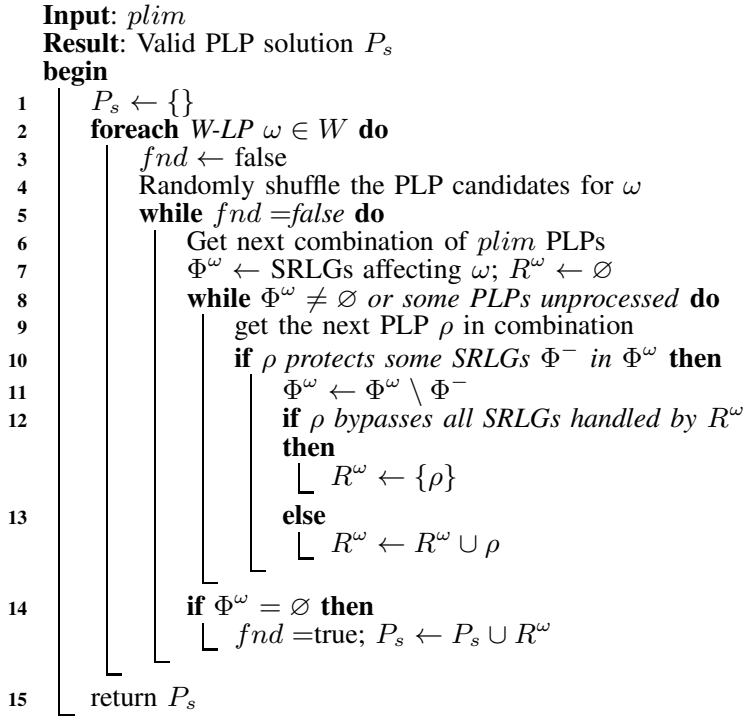
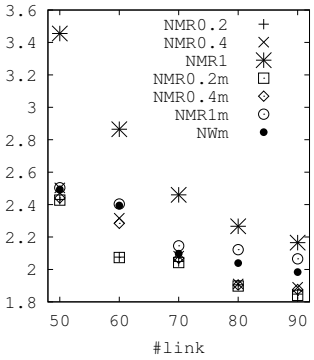


Figure 5.4: Algorithm: PLP Selection (PLPS)

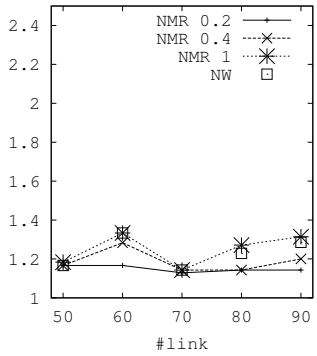
another input to the heuristic. Specifically, $plim = 3$ (see Fig. 5.4) in simulation, allowing up to 3 P-LPs to protect each W-LP.

Fig. 5.5 demonstrates the performance of the proposed heuristic, NWL-UFL [62] and GNFL[61], which are labelled by "NMR", "NW" and "GNFL" respectively (For NWL-UFL, only monitoring resource consumption is shown). For NMR, instead of localizing all the SRLG failures, *each node is only required to distinguish a subset of SRLG pairs* ***necessary*** *for achieving the all-optical restoration.*This is a further improvement against [62] and [61], where the former (i.e. NWL-UFL) forces all nodes to distinguish every pair of SRLGs, while the latter (i.e. GNFL), although defined neighborhood for each node under single link failures, still requires to distinguish more SRLG pairs than necessary, and to consume an excessive number of transmitters (or transponders) for using shortest paths as m-trails.

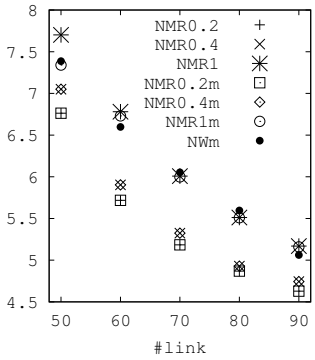
The number part in a data label represents the traffic load. A data label ends with m if it merely indicates monitoring resource consumption (i.e. consumed by m-trails); otherwise, it indicates the total network resource consumed (i.e. by P-LPs and m-trails/paths).



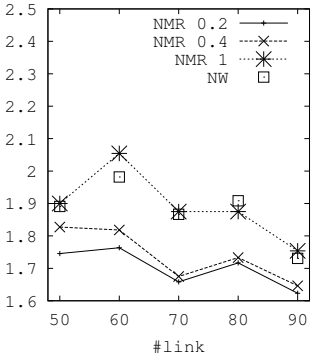
(a) Wav: 20-Node(single)



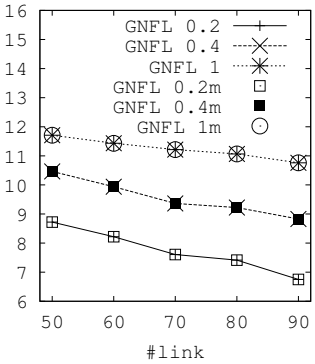
(b) Tr: 20 Node(single)



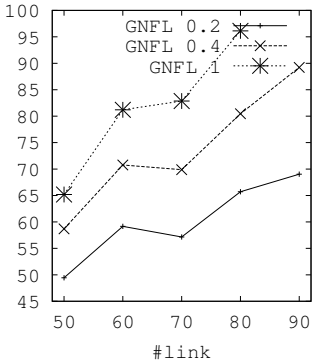
(c) Wav: 20-Node(dual)



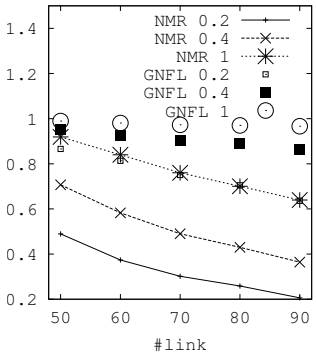
(d) Tr: 20 Node(dual)



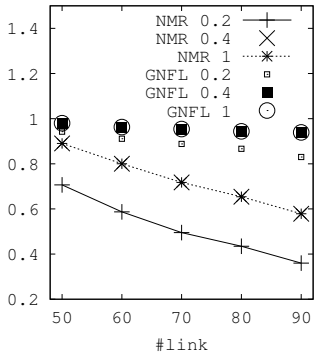
(e) Wav: 20-Node(single) GNFL



(f) Tr: 20 Node(single) GNFL



(g) SRLG pair ratio (single)



(h) SRLG pair ratio (dual)

Figure 5.5: Simulation Results for random networks

The proposed heuristic outperforms the other two in consuming less WLs and transponders (no. of m-trails/paths). As shown by Fig. 5.5a, 5.5c and 5.5e, total WL consumption grows with the increase of traffic load for NMR and GNFL. When network size is larger, a lower WL consumption ratio is witnessed since the topology is more densely connected for both NMR and GNFL. Moreover, Coverlength dominates the total WL consumption in most cases except for some high traffic load for the proposed heuristic. Fig. 5.5b, 5.5d and 5.5f show the trail number grows with traffic load and NMR outperforms the others. Fig. 5.5a-5.5d also show the coverlength and trail ratio achieved by NMR under high load is close to the one under NWL-UFL while NMR outshines NWL-UFL in other cases.

Fig. 5.5g-5.5h compare the monitoring requirements specified by NMR, GNFL and NWL-UFL under various traffic loads using (5.3). As shown by the figures, NMR dramatically reduces the number of required SRLG pairs under mild traffic loads and outperforms the others in both single link and dual link failure scenarios.

With regard to the running time, the heuristic takes about 3 minutes to find 100 feasible solutions (i.e. $nt = it = 10$ in Fig. 5.3) on a work-station with 2.8G CPU and 1G memory. The tested joint design problem asks to find solutions for a 20-node, 90-link random network under full traffic load, and all upto 2-link SRLGs are under consideration. NWL-UFL and GNFL takes longer since their monitoring requirement is more strict than NMR.

5.6 Heuristic for Dynamic Joint Design

The proposed dynamic survivable routing scheme, namely Dynamic Joint design Heuristic (DJH), is presented in this section. It aims to solve the W-LP, P-LPs and a set of m-trails corresponding to a connection request, such that each node along the P-LPs meets the NMR of the connection; i.e., each node along the P-LPs can be aware of the SRLG failure status that may possibly affect the W-LP via the inspected on-off status of the traversing m-trails and other existing W-LPs, so as to ensure an all-optical failure restoration that is free from any after-failure real-time signaling. With the failure status knowledge, the nodes on the P-LPs can react to the identified failure immediately for the W-LP restoration without waiting for any control plane notification.

For simplicity, we assume each connection request $C_i = (ns_i, nt_i)$ is bidirectional, where ns_i and nt_i denotes the two end nodes, respectively. A wavelength channel (WL) reserved for an m-trail can be reused by a P-LP. When a new connection request C_i arrives, it is accepted only if the dynamic joint design algorithm can successfully: 1) establish a W-LP

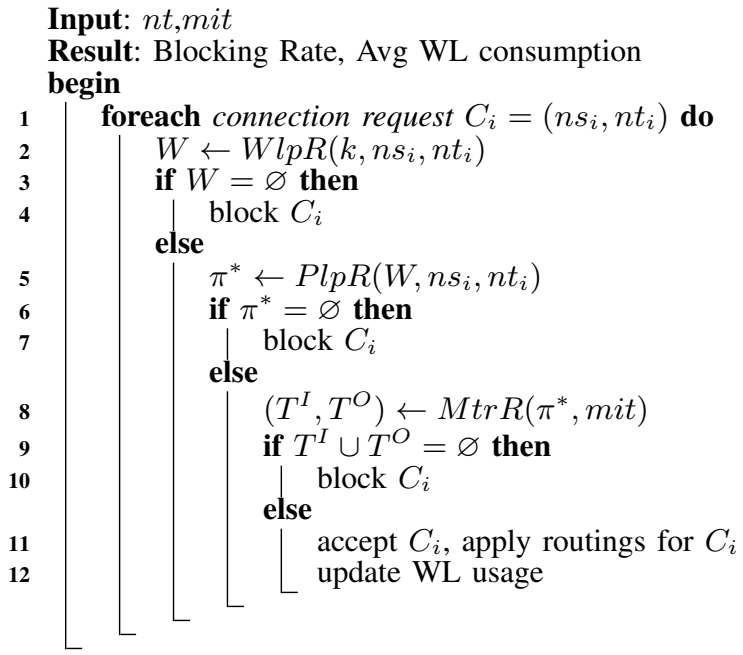


Figure 5.6: Algorithm: Dynamic Joint Design Heuristic (DJH)

for the request, 2) allocate the corresponding P-LP(s) for protecting the W-LP from any predefined failure event, and 3) modify the m-trail set to meet the NMR corresponding to C_i and all the other connections currently supported in the system. Otherwise, the request will be blocked.

Fig. 5.6 details the proposed dynamic joint design heuristic (DJH), which handles each arriving request C_i in three steps. Firstly, the heuristic calls procedure $WlpR$ to generate some feasible W-LP candidates (**line 2**). These candidates could be pre-computed and stored in a database for each node pair. Secondly, with the W-LP candidates, $PlpR$ tries to assign P-LP(s) for each W-LP candidate and selects the combination with highest utility (i.e. less SRLG pairs to differentiate and less WL consumption), denoted as π^* in (**line 5**). Finally, procedure $MtrR$ (**line 8**) tries to deploy m-trails for meeting the NMR of all connections currently in system as well as π^*.

During the whole process, the request C_i will be simply blocked if any of the above steps fails (**line 3-4, 6-7, 9-10**). Otherwise, C_i is accepted and launched in the network (**line 11-12**).

Note that $Wlpr$ generates a number of k W-LP candidates corresponding to the connection request C_i by using Yen's Algorithm[10] on the residual graph. With the candidate

W-LPs, the SRLGs that may fail each W-LP will be given. Thus, Dijkstra's algorithm is used by $PlpR$ to derive the P-LPs for each candidate W-LP, where each P-LP of a candidate W-LP is diversely routed from one or multiple SRLGs that intersects with the W-LP, such that every link along the W-LP is protected by at least one P-LP from any considered SRLG failure event.

The main novelty of the proposed heuristic is in the task of m-trail allocation corresponding to a connection request defined in the function $MtrR$, which is given in Fig .5.7. To make the presentation clearer, we refer an *"out-of-band"* m-trail to as a lightpath exclusively used for the monitoring purpose, while an *"in-band"* m-trail to as an existing W-LP in the network whose failure status is also monitored and considered in the monitoring plane.

5.6.1 Overview of m-trail allocation

The m-trail allocation takes three steps. First, the W-LPs in system are randomly shuffled (**line 3**) and a W-LP is configured as an in-band m-trail if it helps meet NMR(**line 4**). Second, in case the in-band m-trails cannot fully meet NMR, some out-of-band m-trails in system are selected(**line 5-7**). Finally, additional out-of-band m-trails will be launched online if the first two steps fail to meet NMR (**line 8-11**).

The detailed process for launching additional m-trails is as follows. As long as the current set of m-trails cannot meet NMR(**line 8**), a node which needs further monitoring information, n_i, and a pair of undifferentiated SRLG pair (ϕ_j, ϕ_k) (**line 9**) are chosen. Then, procedure $GenMtr$ is called to generate an m-trail which starts from node n_i and can differentiate ϕ_j and ϕ_k (**line 10**). If no m-trail can be generated, $MtrR$ stops; otherwise, the generated m-trail is saved to the out-of-band m-trail set T^O, WL usage are updated accordingly (**line 11**), and $MtrR$ tries to generate another m-trail if the NMR is not fully met(**line 8**).

Fig. 5.9 gives a detailed example for the m-trail allocation process. Suppose that when Connection 3 arrives, the first two connections are not released; thereby their W-LPs, P-LPs and m-trails are still in system (see Fig. 5.9a,b and d; T_4, W_3 and P_3^* have not been deployed at the moment). Suppose that connection 3 asks to establish a connection between v_2 and v_3. Thus, a new W-LP W_3 is created along with a P-LP P_3^* to protect it as Fig. 5.9c shows. Note that W_3 is established by configuring the out-of-band m-trail T_3 as an in-band one. To achieve all-optical restoration for W_3, all nodes on P_3^* should be aware of the link status of (v_2, v_3). In other words, the monitoring requirements at all nodes

```
Input: WLP-PLP canidate π, tit
Result: inband m-trail set: T^I; out-of-band m-trail set: T^O
begin
1   {T^I, T^O} ← ∅
2   compute NMR imposed by π and other connections in
    system
    /* deploy W-LPs as inband m-trails      */
3   randomly shuffle the WLPs in system W^c
4   foreach WLP ω ∈ W^c do
        T^I ← T^I ∪ ω if ω helps achieve NMR
    /* pick out-of-band m-trails in system
       */
5   while T^I ∪ T^O cannot achieve NMR do
6       pick up an out-of-band m-trail ρ in system
7       T^O ← T^O ∪ ρ if ρ helps achieve NMR
    /* launch additional out-of-band
       m-trails                              */
8   while T^I ∪ T^O cannot achieve NMR do
9       select an undifferentiated node n_i and an
        undifferentiated SRLG pair {φ_j, φ_k} at n_i
10      ρ ← GenMtr(n_i, φ_j, φ_k, tit)
11      if ρ = ∅ then
            break
        else
            T^O ← T^O ∪ ρ
            update WL usage
12  if T^I ∪ T^O cannot achieves NMR then
        {T^I, T^O} ← ∅
    return (T^I, T^O)
```

Figure 5.7: Algorithm: M-trail Routing (MtrR)

now further stipulate link (v_3, v_2) to be uniquely identified for switching to P_3^* (Compare Fig. 5.9e-h with Fig. 5.1d-g).

The m-trail allocation procedure $MtrR$ then begins allocating m-trails at each node. At node v_4, $MtrR$ first add W_1 as an in-band m-trail while in-system m-trails T_1 and T_2 are added later. Since T_1, T_2 and W_1 have met the NMR at v_4, no additional m-trail is generated(see Fig. 5.9e). Similarly, at v_3 and v_2, $MtrR$ meets the NMR by keeping the m-trails and W-LPs allocated for the first two connections(compare Fig. 5.9f,h with Fig. 5.1e,g). However, at v_1, just deploying W_1, W_2 and T_1 cannot differentiate (v_3, v_2) from link (v_2, v_4)(see Fig. 5.1f). To meet the NMR at v_1, a new m-trail T_4 will be generated by calling the m-trail generation procedure $GenMtr$.

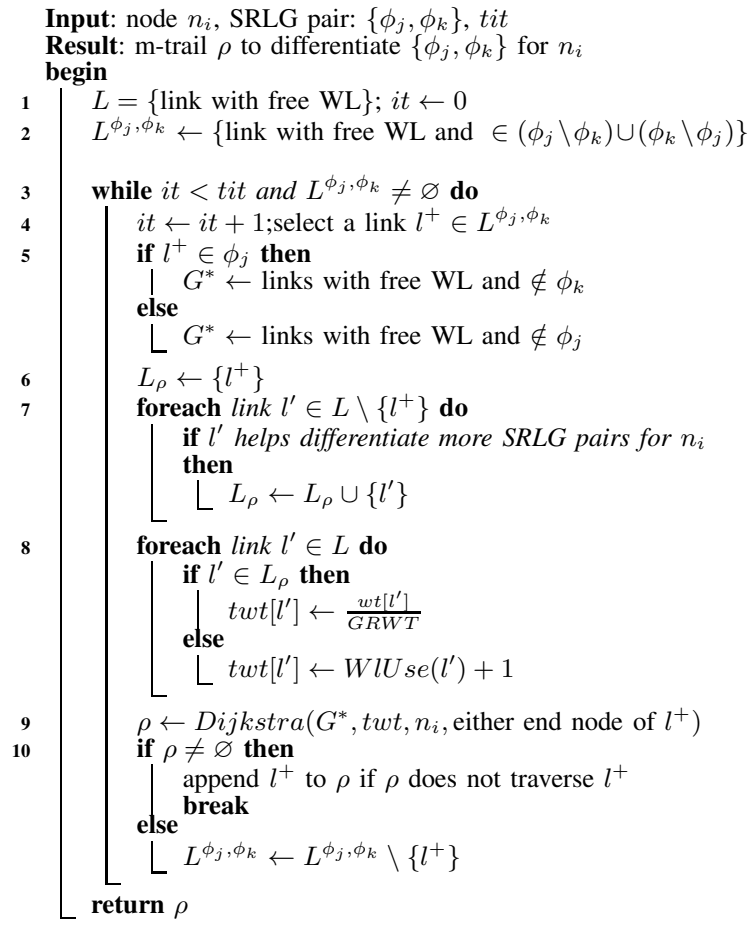
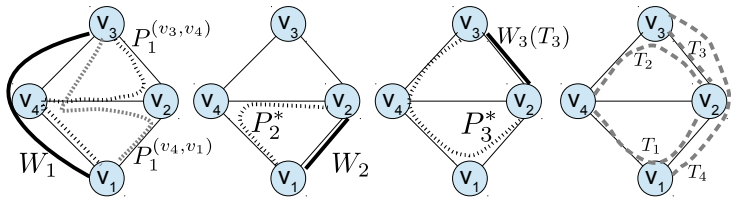


Figure 5.8: Algorithm: Generate M-trail (GenMtr)

5.6.2 Generating m-trails

The m-trail generation procedure *GenMtr* is shown in Fig. 5.8. Firstly, all links with free WL are recorded in the link set L (**line 1**) while the links with free WL and not contained by both SRLGs are saved to L^{ϕ_j,ϕ_k} (**line 2**). Next, *GenMtr* will run for at most *tit* iterations(**line 3-10**), which tries to use the link candidate in L^{ϕ_j,ϕ_k} to differentiate the SRLG pair.

At the beginning, a link l^+ will be selected randomly (**line 4**) and a pruned graph G^* will be constructed (**line 5**). Then, the link set L_ρ, which represents the m-trail to be generated, is initialized as $\{l^+\}$ (**line 6**). Additional links are added to the link set L_ρ



(a) Connection 1 (b) Connection 2 (c) Connection 3 (d) M-trails

SRLG	T_1	T_2	W_1	switch to
(v_3, v_4)	0	1	1	$P_1^{(v_3,v_4)}$
(v_4, v_1)	1	0	1	$P_1^{(v_4,v_1)}$
(v_2, v_1)	1	0	0	P_2^*
★(v_3, v_2)	0	1	0	P_3^*
(v_2, v_4)	0	0	0	

(e) RT at v_4

SRLG	T_2	T_3	W_1	switch to
(v_3, v_4)	1	0	1	$P_1^{(v_3,v_4)}$
(v_4, v_1)	0	0	1	$P_1^{(v_4,v_1)}$
★(v_3, v_2)	1	1	0	P_3^*
(v_2, v_1)	0	0	0	
(v_2, v_4)	0	0	0	

(f) RT at v_3

SRLG	T_1	T_4	W_1	W_2	switch to
(v_3, v_4)	0	0	1	0	$P_1^{(v_3,v_4)}$
(v_4, v_1)	1	0	1	0	$P_1^{(v_4,v_1)}$
(v_2, v_1)	1	1	0	1	P_2^*
★(v_3, v_2)	0	1	0	0	P_3^*
(v_2, v_4)	0	0	0	0	

(g) RT at v_1

SRLG	T_1	T_2	T_3	W_2	switch to
(v_3, v_4)	0	1	0	0	$P_1^{(v_3,v_4)}$
(v_4, v_1)	1	0	0	0	$P_1^{(v_4,v_1)}$
(v_2, v_1)	1	0	0	1	P_2^*
★(v_3, v_2)	0	1	1	0	P_3^*
(v_2, v_4)	0	0	0	0	

(h) RT at v_2

Figure 5.9: Demo DJH

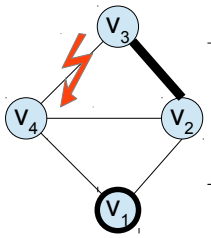
sequentially, such that as many SRLG pairs can be differentiated at n_i (**line 7**).

Since L_ρ may not compose a valid m-trail, the Dijkstra algorithm will try to traverse through as many links in L_ρ as possible with proper link weight setting(**line 8-10**). If Dijkstra's algorithm succeeds, a valid m-trail will be obtained; otherwise, current link candidate l^+ will be deleted from L^{ϕ_j,ϕ_k} and the algorithm tries to generate another m-trail using the next link candidate(**line 10**).

In **line 8**, $twt[l']$ stores the computed weight of link l' while $wt[l']$ denotes the given link cost on l'; $GRWT$ is a constant representing the total link cost of the network graph while $WlUse(l')$ represents the WL usage ratio on link l'. Thus, each link in L_ρ is assigned a small weight corresponding to its given link cost while all the other links with free WL are assigned bigger weight based on their WL usage.

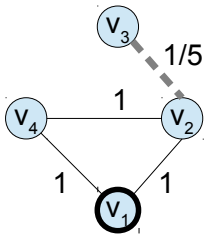
Fig. 5.10 illustrates how the m-trail T_4 is generated for meeting the NMR at v_1. To differentiate (v_3, v_2) from (v_2, v_4) at v_1, the generated m-trail should not pass through both (v_3, v_2) and (v_2, v_4). Suppose that $GenMtr$(Fig. 5.8) chooses (v_3, v_2) and (v_2, v_4) as the links which must be passed through (l^+) and bypassed respectively(see Fig. 5.10a).

Firstly, $GenMtr$ adds (v_3, v_2) to the m-trail's link set L_ρ. Since NMR is met (check



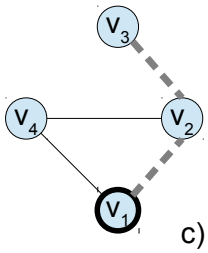
SRLG	T_1	T_4	W_1	W_2	switch to
(v_3, v_4)	0		1	0	$P_1^{(v_3,v_4)}$
(v_4, v_1)	1		1	0	$P_1^{(v_4,v_1)}$
(v_2, v_1)	1		0	1	P_2^*
(v_3, v_2)	0		0	0	P_3^*
(v_2, v_4)	0		0	0	

a) Pass(v2,v3), bypass(v2,v4); Pruned Graph G*



SRLG	T_1	T_4	W_1	W_2	switch to
(v_3, v_4)	0	0	1	0	$P_1^{(v_3,v_4)}$
(v_4, v_1)	1	0	1	0	$P_1^{(v_4,v_1)}$
(v_2, v_1)	1	0	0	1	P_2^*
(v_3, v_2)	0	1	0	0	P_3^*
(v_2, v_4)	0	0	0	0	

b) add link (v2,v3), set link weight



SRLG	T_1	T_4	W_1	W_2	switch to
(v_3, v_4)	0	0	1	0	$P_1^{(v_3,v_4)}$
(v_4, v_1)	1	0	1	0	$P_1^{(v_4,v_1)}$
(v_2, v_1)	1	1	0	1	P_2^*
(v_3, v_2)	0	1	0	0	P_3^*
(v_2, v_4)	0	0	0	0	

c) T_4 = shortest path v1->v2 + (v2,v3)

Figure 5.10: M-trail demo

v_1's RT shown in Fig. 5.10b), no further links will be added. Suppose that each link has a unit cost. Thus, the total weight of the given graph is 5. According to the weight setting rules given in *GenMtr*(Fig. 5.8: **line 8**), the links in L_ρ are set a smaller link weight: $\frac{1}{5}$ while other links' weight are set to 1.

Next, Dijkstra's algorithm is launched(see Fig. 5.10c), which connects v_1 to one end node of (v_3, v_2). Let $v_1 - v_2$ be the obtained path. By concatenating $v_1 - v_2$ with link (v_2, v_3), the m-trail $T_4 : v_1 - v_2 - v_3$ is generated successfully. After filling T_4 into v_1's RT, the NMR is met at v_1(see Fig. 5.9g).

5.6.3 Simulation Results

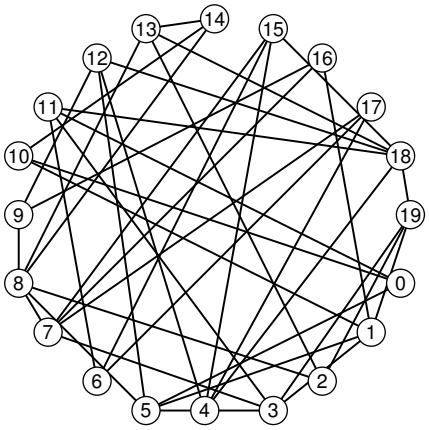
To evaluate the performance of the proposed DJH scheme, a discrete event-driven simulation was conducted to examine its blocking probability performance. For comparison, we implemented Protected Working Capacity Envelope (PWCE) [86], which is the only reported dynamic survivable routing scheme that guarantees an all-optical and signaling-free failure restoration for every W-LP. A simplified version of DJH, namely DJHnt, was also implemented to serve as a plain FDP scheme without considering monitoring. It is expected that $DJHnt$ serves as a benchmark and will yield no worse blocking performance than DJH. Such performance advantage is at the expense of taking electronic signaling for the restoration of failed W-LPs and longer restoration time.

In the simulation, each link contains Ψ WLs in both directions, where Ψ is 16, 32, and 64, respectively. A plotted data is as a result of $20,000$ connection requests, each for a bi-directional W-LP protected by a set of link-disjoint P-LPs. Each connection request arrives randomly upon the network node pairs one after the other without any prior knowledge of future arrivals, with an arrival rate of λ and an average holding time of $\frac{1}{\mu}$.

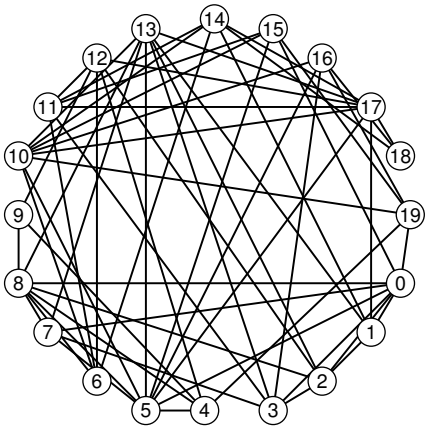
The three schemes (i.e., DJH, $PWCE$, and $DJHnt$) are examined using four topologies, in which two are randomly generated networks with 20 nodes and average nodal degree as 4 and 6 respectively, and another two are well known commercial network topologies namely ARPA2 and BellCore (see Fig. 5.11) . For DJH, up to three W-LP candidates are generated for each connection request, while the procedure $GenMtr$ will try at most twice to obtain a valid m-trail. For $PWCE$, two p-Cycles possibly with multiple copies are pre-computed. Note that although the proposed DJH can easily handle multi-link failures, we only consider single-link failures in this simulation.

Fig. 5.12 demonstrates the blocking probability of the three schemes under different traffic loads and network capacities Ψ for 20 node random networks. As expected, $DJHnt$ achieves the lowest blocking probability while DJH significantly outperforms $PWCE$. It is interesting to observe that the performance gap between DJH and $DJHnt$ is slightly increased when the traffic load is increased at the beginning, but the difference keeps still and even becomes smaller when the traffic load is further increased (see Fig. 5.12c and e). This is due to the effect of resource sharing between the m-trails and the P-LPs, which becomes more effective in slowing down the increase of the monitoring capacity assumption when the number of connections co-existing in the network is getting larger.

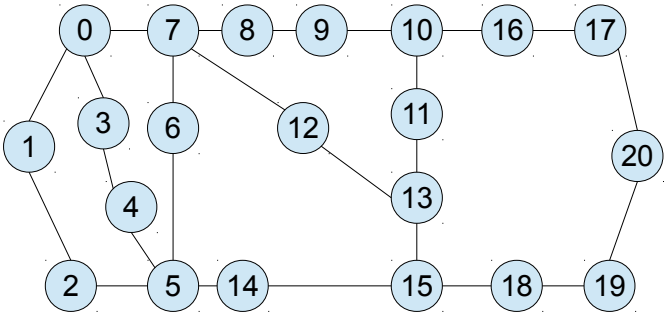
We further observed that the larger the network capacity (i.e., Ψ), the smaller the performance gap between DJH and $DJHnt$ is. For example, when the total traffic load is 175 Erlang, the gap of the blocking probability is around 0.2 with $\Psi = 16$ as shown in



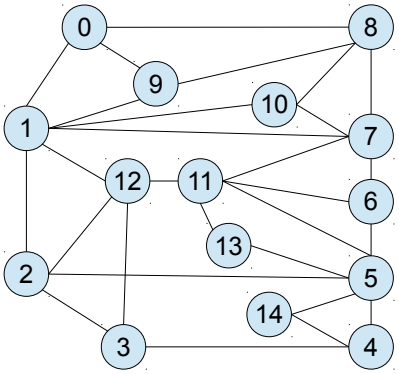
(a) 20-node random network: deg 4



(b) 20-node random network: deg 6



(c) ARPA2

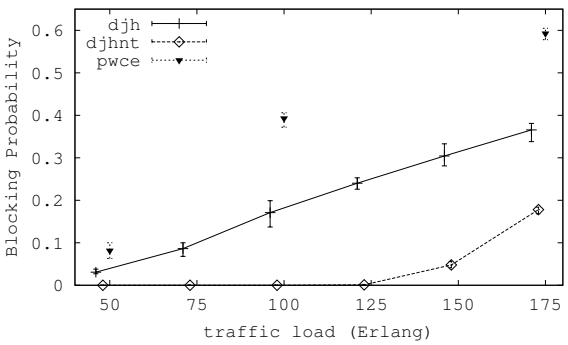


(d) BellCore

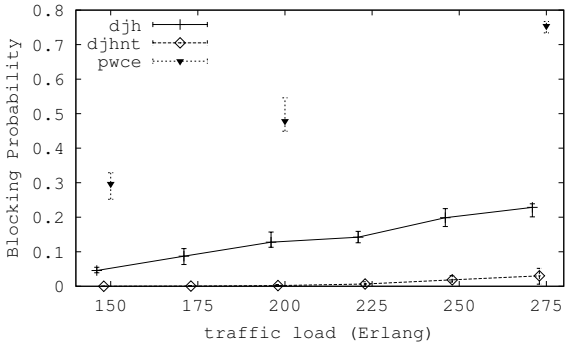
Figure 5.11: network topologies for simulation

Fig. 5.12a, while becoming almost zero with $\Psi = 32$ as shown in Fig. 5.12c. This is due to the fact that the monitoring resource consumption is upper bounded by that of NWL-UFL (i.e., the situation where every node can localize the failure of every SRLG). Thus, when the network capacity is larger to simultaneously accommodate more connections, a newly arrived connection has a better chance to reuse the existing m-trails for meeting the NMR.

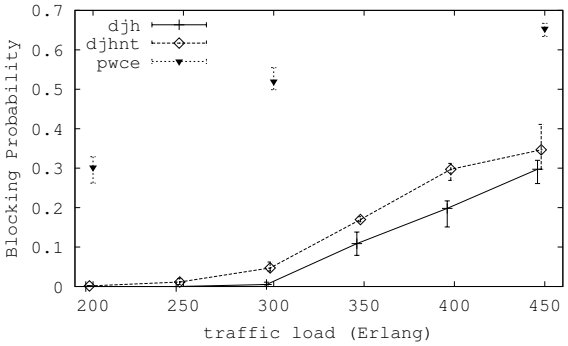
We also observed that the performance gap between DJH and $DJHnt$ is smaller in denser network topologies as demonstrated by Fig. 5.12, because in this case the m-trails only occupy a smaller portion of the total network capacity.



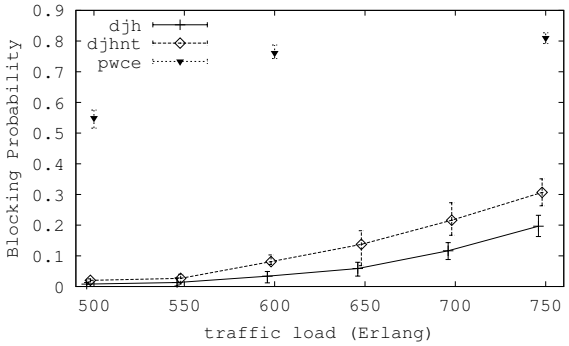
(a) $\Psi = 16$, Deg 4



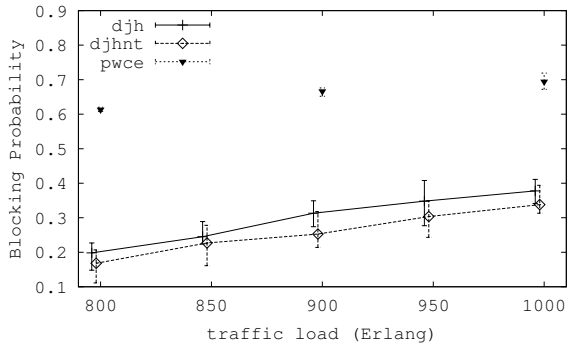
(b) $\Psi = 16$, Deg 6



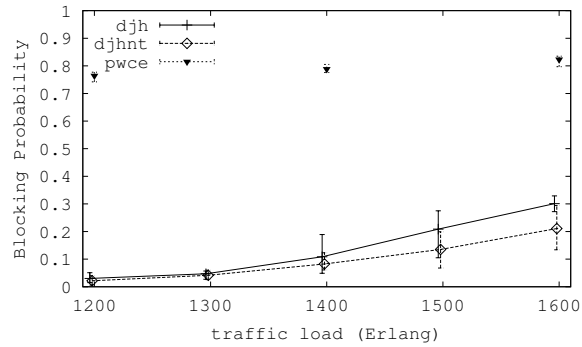
(c) $\Psi = 32$, Deg 4



(d) $\Psi = 32$, Deg 6



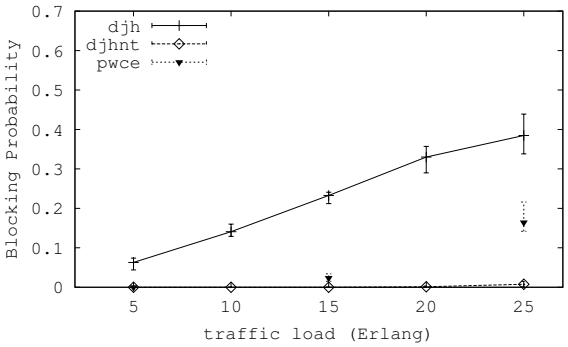
(e) $\Psi = 64$, Deg 4



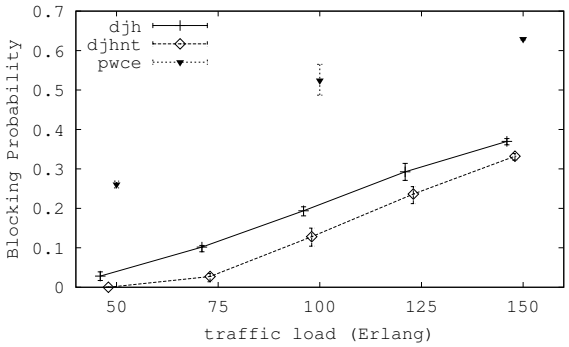
(f) $\Psi = 64$, Deg 6

Figure 5.12: Results for 20-node random networks

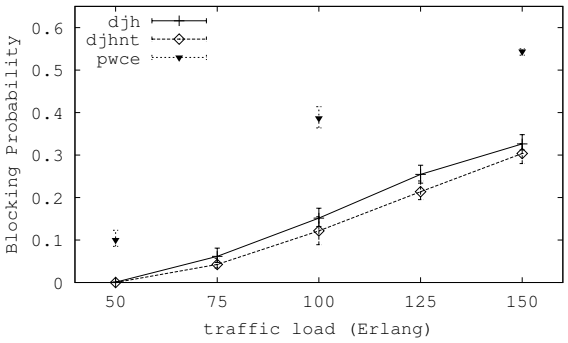
In Fig. 5.13, we examined the three schemes in two typical network topologies, namely



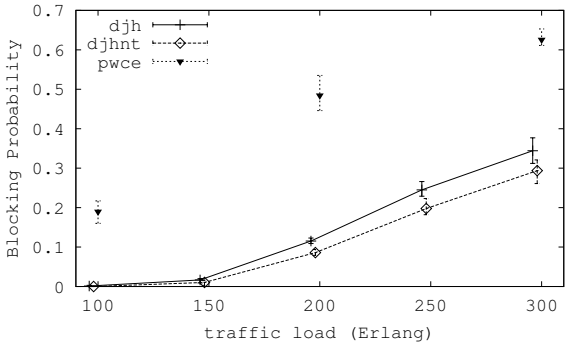
(a) $\Psi = 16$, ARPA2



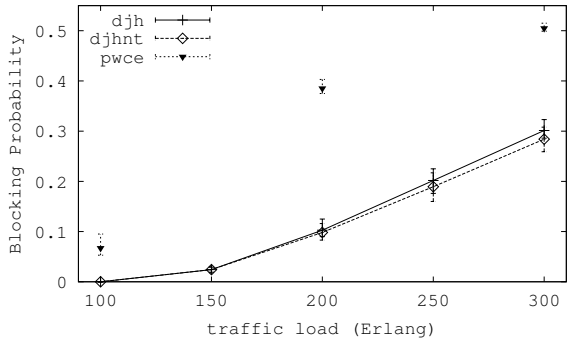
(b) $\Psi = 16$, BellCore



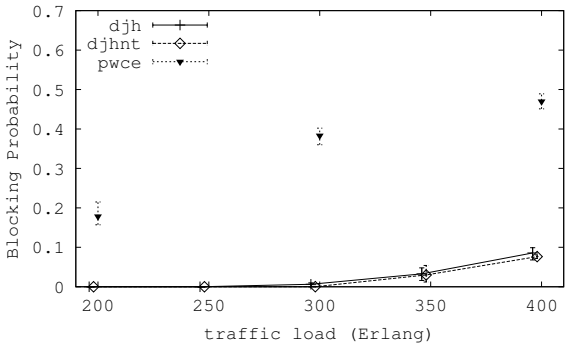
(c) $\Psi = 32$, ARPA2



(d) $\Psi = 32$, BellCore



(e) $\Psi = 64$, ARPA2



(f) $\Psi = 64$, BellCore

Figure 5.13: Results for typical networks

ARPA2 (21 nodes, 25 links) and BellCore (15 nodes, 28 links)[82]. We have seen quite

similar performance behavior of the three schemes as observed before, where the proposed DJH scheme leads to better blocking performance in network topologies with larger nodal degrees and link capacity. As shown in Fig. 5.13a, when the network is the sparsest and Ψ is the smallest, DJH is even outperformed by PWCE; but when Ψ is increased to 64, DJH becomes significantly better than PWCE and yields almost a zero performance gap with $DJHnt$.

As for running time, all the three schemes were run on a workstation with a 2.2G CPU and 1G memory. We note that DJH takes about a few hundreds milliseconds for settling each connection, while $DJHnt$ and $PWCE$ only needs less than 10 milliseconds, since $DJHnt$ waives the m-trail allocation process as in DJH, and $PWCE$ even has the p-Cycles pre-computed. Thus, DJH could be subject to a scalability issue as the network size is getting larger. One way to mitigate this problem is that a connection request is settled simply by using $DJHnt$ and the service is provisioned before the m-trails for NMR are in place. This can significantly reduce the connection setup time but is obviously at the risk that a failure occurs to the connection and has to be restored via conventional electronic signaling protocols.

5.7 Conclusions

This chapter introduced a novel framework of survivable routing in the network optical layer that jointly determines network monitoring plane and spare capacity allocation in the presence of either static or dynamic traffic. This chapter first discussed the feasibility of the targeted all-optical failure restoration process along with its implementation details. Based on the proposed framework, we developed two novel heuristics: $MPJD$ and DJH.

$MPJD$ was developed to determine the P-LPs and m-trails given the W-LPs, which was further examined by extensive simulation. We have witnessed in the simulation that the proposed approach can outperform the reported schemes in [62, 61] in terms of totally consumed WLs, required transponders (i.e.,the number of m-trails) and the pair of SRLGs required to be distinguished.

On the other hand, DJH, was developed for dynamic survivable routing as well as m-trail deployment in a single stage. Extensive simulation was conducted to validate the proposed framework and gain insights into the blocking probability performance. In particular, we compared DJH with a couple of its counterparts, namely $PWCE$ and $DJHnt$. The former is to the best of our knowledge, the only reported dynamic survivable routing scheme that can achieve all-optical restoration, while the later is simply a conventional

FDP scheme taken as a benchmark in the performance comparison.. We found that the proposed DJH scheme is is significantly advantageous against $PWCE$, and its performance is getting closer to that of $DJHnt$ as the network topologies are denser and link capacity is larger.

We conclude that the proposed framework can achieve all-optical and signaling-free restoration like in ring networks, while maintaining high capacity efficiency as in FDP based survivable routing. Fig. 5.14 summaries the main ideas and concepts in this chapter.

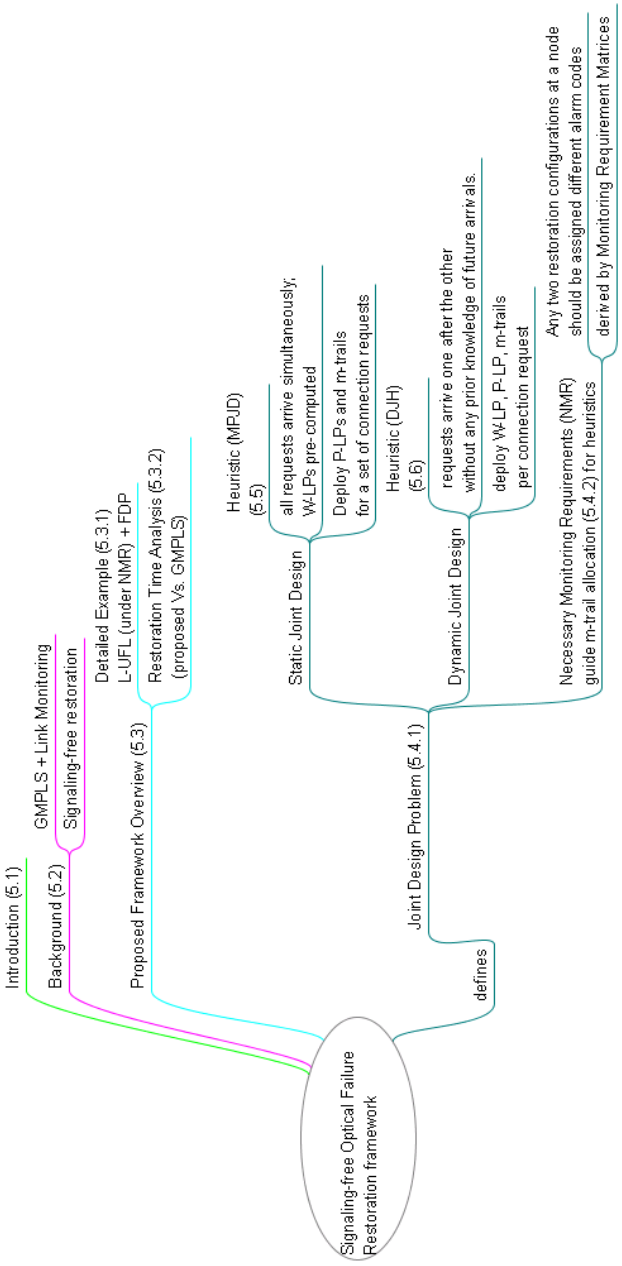


Figure 5.14: mind map for Signaling-free All-optical Restoration Framework

Chapter 6

Conclusion

In this thesis, I proposed a novel framework of all-optical failure restoration in the optical layer for jointly determining network monitoring plane and spare capacity allocation in the presence of either static or dynamic traffic. The proposed framework aims to enable a general shared protection scheme to achieve near optimal capacity efficiency as in FDP while subject to an ultra-fast, all-optical, and deterministic failure restoration process. Simply put, Local Unambiguous Failure Localization(L-UFL) and FDP are the two building blocks for the proposed restoration framework.

Under L-UFL, by properly allocating a set of m-trails, a set of nodes can unambiguously identify every possible SRLG failure merely based on its locally collected LOL signals. Two heuristics were first proposed by us to solve the L-UFL problem and introduced in Chapter 4. The heuristic in [25] achieves L-UFL using S-LPs. Thanks to the Enhanced *Min Wavelength Max Information* principle and other techniques, the proposed heuristic algorithm exhibits a satisfactory performance in minimizing the number of m-trails, coverlength and the running time of the algorithm. The other L-UFL heuristic in [23] further reduces the monitoring resource consumption by jointly considering the working lightpaths and newly added supervisory lightpaths as m-trails. This is the first study that investigates L-UFL using S-LPs and W-LPs. The problem of selecting an optimal set of working lightpaths for L-UFL was shown to be NP-complete. Techniques were developed to solve the L-UFL m-trail allocation problem which iteratively constructs the desired m-trail solution using a novel entropy based utility function. Simulation results on various network topologies demonstrated the effectiveness of the proposed algorithm and its performance advantage against other methods.

Based on the L-UFL heurisitcs, two novel heuristics: $MPJD$[22] and DJH[27] were

proposed under the novel restoration framework. $MPJD$ was developed to determine the P-LPs and m-trails given the W-LPs, which was further verified by extensive simulation. We have witnessed that the proposed approach can outperform the reported schemes in [62, 61] in terms of consumed WLs, required transponders (i.e.,the number of m-trails) and the pair of SRLGs required to be distinguished.

On the other hand, DJH, in which a generic dynamic survivable routing scheme based on FDP is jointly implemented with an m-trail deployment scheme. Extensive simulation was conducted to gain insights into the blocking probability performance. In particular, we compared DJH with a couple of its counterparts, namely $PWCE$ and $DJHnt$. We found that the proposed DJH scheme is advantageous against $PWCE$, and its performance is getting closer to that of $DJHnt$ as the network topologies are denser and link capacity is larger.

All in all, the proposed restoration framework can achieve all-optical and signaling-free restoration with the help of L-UFL, while maintaining high capacity efficiency as in FDP based survivable routing. The proposed heuristics achieve satisfactory performance as verified by the simulation results.

As regard to future work, Integer Linear Programs could be proposed for the novel restoration framework as a benchmark to validate the performance of the heuristics. Approximation algorithms are desired as well to guarantee the performance to some extent. The restoration framework can be further enhanced to support different QoS levels(e.g. some W-LPs does not need protection), different recovery granularities(e.g. WL-level rather than link level), as well as different traffic patterns(multicast, P2P traffic etc.). The deployment of wavelength-converters and amplifiers should be taken into consideration as well. Some multi-stage monitoring scheme could be developed as a compromise between recovery time and network resource consumption in a time interval. M-trail light signals can be transmitted periodically to save energy while other metrics such as bit-error-rate and signal degrade could be examined to handle more complex failure scenarios. Most importantly, a multi-layer survivability scheme is desired which can coordinate the restoration actions at each layer and efficiently while rapidly perform the recovery.

Acronyms

ACT Alarm Code Table. 27

CGT Combinatorial Group Testing. 12

GMPLS Generalized Multiprotocol Label Switching. 23

ILP Integer Linear Programming. 15

IV Incidence Vector. 28

L-UFL Local Unambiguous Failure Localization. 26

LOL Loss of Light. 23

m-cycle Monitoring Cycle. 9

m-path Monitoring Path. 9

m-structure Monitoring Structure. 9

m-trail Monitoring Trail. 9

m-tree Monitoring Tree. 9

MN Monitoring Node. 26

NMR Necessary Monitoring Requirement. 57

NWL-UFL Network-wide Local Unambiguous Failure Localization. 27

OXC Optical Cross-Connect. 8

p-Cycle Preconfigured Protection Cycle. 20

P-LP Protection Lightpath. 18

S-LP Supervisory Lightpath. 10

SONET Synchronous Optical Networking. 1

SRLG Shared Risk Link Group. 9

UFL Unambiguous Failure Localization. 11

W-LP Working Lightpath. 10

WDM Wavelength Division Multiplexing. 1, 2, 8

WL Wavelength Channel. 8

References

[1] igraph: library for complex network research. http://igraph.sourceforge.net/.

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

[3] P. K. Agarwal, A. Efrat, S. K. Ganjugunte, D. Hay, S. Sankararaman, and G. Zussman. The resilience of WDM networks to probabilistic geographical failures, 2013.

[4] S.S. Ahuja, S. Ramasubramanian, and M. Krunz. SRLG failure localization in all-optical networks using monitoring cycles and paths. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 700 –708, april 2008.

[5] S.S. Ahuja, S. Ramasubramanian, and M. Krunz. Single link failure detection in all-optical networks using monitoring cycles and paths. *IEEE/ACM Trans. Networking*, 17(4):1080–1093, 2009.

[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*, pages 2676–2680, 2002.

[7] P. Babarczi, J. Tapolcai, and Pin-Han Ho. Adjacent link failure localization with monitoring trails in all-optical mesh networks. *Networking, IEEE/ACM Transactions on*, 19(3):907 –920, june 2011.

[8] Ayan Banerjee, John Drake, Jonathan Lang, Brad Turner, Daniel Awduche, Lou Berger, Kireeti Kompella, and Yakov Rekhter. Generalized multiprotocol label switching: An overview of signaling enhancements and recovery techniques. *IEEE Communications Magazine*, 39:144–151, 2001.

[9] R. Bhandari. *Survivable networks: algorithms for diverse routing*. Kluwer international series in engineering and computer science. Kluwer Academic Publishers, 1999.

[10] Ernesto de Queiros Vieira Martins, Ernesto Queir, Vieira Martins, Marta Margarida, and Marta Margarida Braz Pascoal. A new implementation of yen's ranking loopless paths algorithm, 2000.

[11] Shen Dong, Tse Kam-Hon, and Chan Chun-Kit. Adaptive fault monitoring in all-optical networks utilizing real-time data traffic. *Journal of Network and Systems Management*, pages 1–21, 2011.

[12] Elias A. Doumith, Sawsan Al Zahr, and Maurice Gagnaire. Monitoring-tree: An innovative technique for failure localization in wdm translucent networks. In *GLOBECOM*, pages 1–6, 2010.

[13] Ding-Zhu Du and Frank K. Hwang. *Combinatorial Group Testing and Its Applications (Applied Mathematics)*. World Scientific Publishing Company, January 2000.

[14] Rudra Dutta, AhmedE. Kamal, and GeorgeN. Rouskas. Grooming mechanisms in SONET/SDH and next-generation SONET/SDH. In *Traffic Grooming for Optical Networks*, Optical Networks, pages 39–55. Springer US, 2008.

[15] R.J. Ellison, D.A. Fisher, R.C. Linger, H.F. Lipson, T.A. Longstaff, and N.R. Mead. Survivability: protecting your critical systems. *Internet Computing, IEEE*, 3(6):55–63, 1999.

[16] Adrian Farrel and Igor Bryskin. *GMPLS: Architecture and Applications*. Morgan Kaufmann, first edition, 2006.

[17] Yashar Ganjali, Supratik Bhattacharrya, and Christophe Diot. Limiting the impact of failures on network performance. Sprint ATL Research Report RR04-ATL-020666, Sprint ATL, February 2004.

[18] Wayne D. Grover. The protected working capacity envelope concept: An alternate paradigm for automated service provisioning. *IEEE Commun. Mag.*, 42(1):62–69, january 2004.

[19] M. Farhan Habib, Massimo Tornatore, Ferhat Dikbiyik, and Biswanath Mukherjee. Disaster survivability in optical communication networks. *Computer Communications*, 36(6):630 – 644, 2013. Reliable Network-based Services.

[20] Chandler Harris. Data center outages generate big losses. In *www.informationweek.com*, 2011.

[21] N.J.A. Harvey, M. Patrascu, Yonggang Wen, S. Yekhanin, and V.W.S. Chan. Non-adaptive fault diagnosis for all-optical networks via combinatorial group testing on graphs. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 697 –705, may 2007.

[22] Wei He, Pin-Han Ho, Bin Wu, and J. Tapolcai. On integrating failure localization with network survivable design. In *IEEE ICC 2013*, accepted 2013.

[23] Wei He, Pin-Han Ho, Bin Wu, and János Tapolcai. On identifying SRLG failures in all-optical networks. *Optical Switching and Networking*, 10(1):77–88, 2013.

[24] Wei He, Bin Wu, Pin-Han Ho, and J. Tapolcai. Monitoring trail allocation for fast link failure localization without electronic alarm dissemination. In *Optical Network Design and Modeling (ONDM), 2011 15th International Conference on*, pages 1 –6, feb. 2011.

[25] Wei He, Bin Wu, Pin-Han Ho, and J. Tapolcai. Monitoring trail allocation for srlg failure localization. In *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, pages 1–5, Dec.

[26] John Heidemann. Internet impacts of hurricane sandy. In *NANOG57*, 2013.

[27] Pin-Han Ho, Wei He, J. Tapolcai, and Bin Wu. A framework of dynamic survivable routing in all-optical networks. *Journal of Optical Communications and Networking*, submitted.

[28] Richard Kahn. APAC submarine cables and their impact to ip backbone design. In *NANOG52*, 2011.

[29] K. Kazi. *Optical Networking Standards: A Comprehensive Guide*. Springer Science+Business Media, LLC, 2006.

[30] Mehdi khani, Mohammad Ghazemzadeh, Fazlollah Adibnai, and Mehdi Sarram. A probabilistic failure localization in optical wdm networks. *Middle-East Journal of Scientfic Research 6(4)*, pages 324–328, 2010.

[31] Ramana Rao Kompella, Jennifer Yates, Albert Greenberg, and Alex C. Snoeren. Fault localization via risk modeling. *IEEE Trans. Dependable Secur. Comput.*, 7(4):396–409, October 2010.

[32] Hyang-Won Lee, Kayi Lee, and Eytan Modiano. Cross-layer survivability. In Suresh Subramaniam, Mat Brandt-Pearce, Piet Demeester, and Chava Vijaya Saradhi, editors, *Cross-Layer Design in Optical Networks*, volume 15 of *Optical Networks*, pages 243–262. Springer US, 2013.

[33] C.S. Li, R. Ramaswami, I.B.M.T.J.W.R. Center, and Y. Heights. Automatic fault detection, isolation, and recovery in transparentall-optical networks. *IEEE/OSA J. Lightwave Technol.*, 15(10):1784–1793, 1997.

[34] Siarhei Liakh. Multilayer network survivability(project report). 2009.

[35] Steven S. Lumetta, Muriel Médard, and Yung-Ching Tseng. Capacity versus robustness: A tradeoff for link restoration in mesh networks. *IEEE/OSA J. Lightwave Technol.*, 18(12):1765–1775, December 2000.

[36] S. Bryant M. Shand. IP Fast Reroute Framework. RFC 5714, January 2010.

[37] K.M. Maamoun and H.T. Mouftah. Using monitoring trails (m-trails) with established lightpaths to perform fault localization in all-optical networks. In *Computer Engineering Conference (ICENCO), 2010 International*, pages 68 –71, dec. 2010.

[38] Minjing Mao and K.L. Yeung. Super monitor design for fast link failure localization in all-optical networks. In *Communications (ICC), 2011 IEEE International Conference on*, pages 1 –5, june 2011.

[39] Athina Markopoulou, Gianluca Iannaccone, Supratik Bhattacharyya, Chen-Nee Chuah, Yashar Ganjali, and Christophe Diot. Characterization of failures in an operational ip backbone network. *IEEE/ACM Trans. Netw.*, 16(4):749–762, August 2008.

[40] J.L. Marzo, T. Stidsen, S. Ruepp, E. Calle, J. Tapolcai, and J. Segovia. *Graphs and Algorithms in Communication Networks*, chapter Network Survivability: End-to-End Recovery Using Local Failure Information, pages 137–161. Springer, 2009.

[41] M. Médard, R.A. Barry, S.G. Finn, W. He, and S.S. Lumetta. Generalized loop-back recovery in optical mesh networks. *IEEE/ACM Trans. Networking*, 10(1):164, 2002.

[42] H.T. Mouftah and P.H. Ho. *Optical networks: architecture and survivability*. Kluwer Academic, 2003.

[43] Wenda Ni, Yabin Ye, M. Schlosser, E. Patzak, and Hanyi Zhang. Survivable mapping with maximal physical-layer failure-localization potential in ip over transparent optical

networks. In *Optical Fiber Communication (OFC), collocated National Fiber Optic Engineers Conference, 2010 Conference on (OFC/NFOEC)*, pages 1 –3, march 2010.

[44] N. Ogino and H. Nakamura. All-optical monitoring path computation based on lower bounds of required number of paths. In *Communications (ICC), 2011 IEEE International Conference on*, pages 1 –6, june 2011.

[45] D. Papadimitriou and E. Mannie. Analysis of Generalized Multi-Protocol Label Switching (GMPLS)-based Recovery Mechanisms (including Protection and Restoration). RFC 4428 (Informational), March 2006.

[46] D. Papadimitriou and E. Mannie. Recovery (Protection and Restoration) Terminology for Generalized Multi-Protocol Label Switching (GMPLS). Technical report, RFC 4427, March, 2006.

[47] David A. Patterson. A simple way to estimate the cost of downtime. In *Proceedings of the 16th USENIX conference on System administration*, LISA '02, pages 185–188, Berkeley, CA, USA, 2002. USENIX Association.

[48] Stefano Previdi. IP fast reroute technologies. In *Asia Pacific Regional Internet Conference on Operational Technologies*, 2006.

[49] S. Ramamurthy and B. Mukherjee. Survivable WDM mesh networks. part I-protection. In *INFOCOM'99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 744–751 vol.2, 1999.

[50] S. Ramamurthy, L. Sahasrabuddhe, and B. Mukherjee. Survivable wdm mesh networks. *Lightwave Technology, Journal of*, 21(4):870 – 883, april 2003.

[51] D. A. Schupke, W. D. Grover, and M. Clouqueur. Strategies for enhanced dual failure restorability with static or reconfigurable p-cycle networks. In *Proc. IEEE ICC*, pages 1628–1633., Paris, France, June 2004.

[52] S. Sebbah and B. Jaumard. P-cycle based dual failure recovery in WDM mesh networks. In *Proc. IFIP Working Conference on Optical Network Design & Modelling (ONDM)*, 2009.

[53] J.M. Simmons. *Optical Network Design and Planning*. Optical networks. Springer London, Limited, 2008.

[54] Arun Somani. *Survivability and Traffic Grooming in WDM Optical Networks*. Cambridge University Press, New York, NY, USA, 2005.

[55] Joseph M. Soricelli. Tutorial: MPLS fast reroute. In *NANOG30*, 2004.

[56] S. Stanic and S. Subramaniam. Fault localization in all-optical networks with user and supervisory lightpaths. In *Communications (ICC), 2011 IEEE International Conference on*, pages 1 –6, june 2011.

[57] S. Stanic, S. Subramaniam, G. Sahin, H. Choi, and H.-A. Choi. Active monitoring and alarm management for fault localization in transparent all-optical networks. *Network and Service Management, IEEE Transactions on*, 7(2):118 –131, june 2010.

[58] Skiena Steven. *The Algorithm Design Manual*. Springer, 2 edition, 2008.

[59] Tim Stronge. International submarine cable development: New infrastructure, new prices? In *NANOG57*, 2013.

[60] J. Tapolcai, Pin-Han Ho, P. Babarczi, and L. Rónyai. On signaling-free failure dependent restoration in all-optical mesh networks. *Technical Report*, 2012. `http://opti.tmit.bme.hu/~tapolcai/papers/tapolcai2013fdp-mtrail.pdf`.

[61] J. Tapolcai, Pin-Han Ho, Peter Babarczi, and L. Ronyai. On achieving all-optical restoration via monitoring trails. In *IEEE INFOCOM 2013*, accepted.

[62] J. Tapolcai, Pin-Han Ho, Peter Babarczi, and L. Ronyai. On signaling-free failure dependent restoration in all-optical mesh networks. *IEEE Trans. on Networking*, submitted.

[63] J. Tapolcai, Pin-Han Ho, L. Ronyai, P. Babarczi, and Bin Wu. Failure localization for shared risk link groups in all-optical mesh networks using monitoring trails. *Lightwave Technology, Journal of*, 29(10):1597 –1606, may15, 2011.

[64] J. Tapolcai, Pin-Han Ho, L. Rónyai, and Bin Wu. Network-wide local unambiguous failure localization (NWL-UFL) via monitoring trails. *IEEE/ACM Transactions on Networking*, 2012.

[65] J. Tapolcai, Bin Wu, Pin-Han Ho, and L. Ro andnyai. A novel approach for failure localization in all-optical mesh networks. *Networking, IEEE/ACM Transactions on*, 19(1):275 –285, feb. 2011.

[66] J. Tapolcai, Bin Wu, Pin-Han Ho, and L. Rónyai. A novel approach for failure localization in all-optical mesh networks. *IEEE/ACM Trans. Networking*, 19(1):275 –285, feb 2011.

[67] János Tapolcai, Pin-Han Ho, Péter Babarczi, and Lajos Rónyai. On achieving all-optical failure restoration via monitoring trails. In *IEEE Infocom 2013*, 2013.

[68] M. To and P. Neusy. Unavailability analysis of long-haul networks. *Selected Areas in Communications, IEEE Journal on*, 12(1):100 –109, jan 1994.

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

[70] Juniper white paper. Converged supercore– disruptive innovation at the network core. 2012.

[71] P. Winzer. Beyond 100g ethernet. *Communications Magazine, IEEE*, 48(7):26 –30, july 2010.

[72] B. Wu, P.-H. Ho, and K.L. Yeung. Monitoring trail: On fast link failure localization in all-optical WDM mesh networks. *IEEE/OSA J. Lightwave Technol.*, 27(18):4175–4185, 2009.

[73] B. Wu, K.L. Yeung, and P.-H. Ho. Monitoring cycle design for fast link failure localization in all-optical networks. *IEEE/OSA J. Lightwave Technol.*, 27(10):1392–1401, 2009.

[74] B. Wu, K.L. Yeung, and P.-H. Ho. Ilp formulations for p-cycle design without candidate cycle enumeration. *IEEE/ACM Trans. Networking*, 18(1):284–295, 2010.

[75] Bin Wu, Pin-Han 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.

[76] Bin Wu, Pin-Han Ho, J. Tapolcai, and Xiaohong Jiang. A novel framework of fast and unambiguous link failure localization via monitoring trails. In *INFOCOM IEEE Conference on Computer Communications Workshops , 2010*, pages 1 –5, march 2010.

[77] Bin Wu, Pin-Han Ho, and K.L. Yeung. Monitoring trail: A new paradigm for fast link failure localization in wdm mesh networks. In *Global Telecommunications Conference, 2008. IEEE GLOBECOM 2008. IEEE*, pages 1 –5, 30 2008-dec. 4 2008.

[78] Bin Wu, Pin-Han Ho, and K.L. Yeung. Monitoring trail: On fast link failure localization in all-optical wdm mesh networks. *Lightwave Technology, Journal of*, 27(18):4175 –4185, sept.15, 2009.

[79] 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*, 11(3):277–286, 2006.

[80] H. Zeng and A. Vukovic. The variant cycle-cover problem in fault detection and localization for mesh all-optical networks. *Photonic Network Communications*, 14(2):111–122, 2007.

[81] Hongqing Zeng, Changcheng Huang, and A. Vukovic. Monitoring cycles for fault detection in meshed all-optical networks. In *Parallel Processing Workshops, 2004. ICPP 2004 Workshops. Proceedings. 2004 International Conference on*, pages 434 – 439, aug. 2004.

[82] Hongqing Zeng, Changcheng Huang, Alex Vukovic, and J. Michel Savoie. Fault detection and path performance monitoring in meshed all-optical networks, 2004.

[83] Hongqing Zeng, Alex Vukovica, and Changcheng Huang. A novel end-to-end fault detection and localization protocol for wavelength-routed WDM network. In *Proceedings of SPIE Photonics North 2005*, Sep. 2005.

[84] Feng Zhang and Wen-De Zhong. Extending p-cycles to source failure recovery for optical multicast media traffic. *J. Opt. Commun. Netw.*, 2(10):831–840, Oct 2010.

[85] Weiyi Zhang, Farah Kandah, Xiaojiang Du, and Chonggang Wang. Self-protecting networking using dynamic p-cycle construction within link capacity constraint. *Sec. and Commun. Netw.*, 5(6):689–708, June 2012.

[86] Zhenrong Zhang, Zhengbin Li, Yongqi He, and Anshi Xu. p-cycle-based strategy for adaptive pwce design. pages 71372K–71372K–8, 2008.

[87] Yangming Zhao, Shizhong Xu, Xiong Wang, and Sheng Wang. A new heuristic for monitoring trail allocation in all-optical wdm networks. In *GLOBECOM 2010, 2010 IEEE Global Telecommunications Conference*, pages 1 –5, dec. 2010.