Chapter 1

FAILURE LOCATION IN WDM NETWORKS

Carmen Mas,[1] Hung X. Nguyen[2] and Patrick Thiran[2]
[1]*AIT, GR-19002 Athens, Greece,* [2]*LCA, EPFL, CH-1015 Lausanne, Switzerland*

Abstract Fault identification and location in optical networks must cope with a multitude of factors: (i) the redundancy and the lack of coordination (internetworking) of the managements at the different layers (WDM, SDH/SONET, ATM, IP); (ii) the large number of alarms a single failure can trigger; (iii) the difficulty in detecting some failures and the resulting need to cope with missing or false alarms.

This chapter first details the behavior of network components in transparent WDM networks when a failure occurs. Using this model, we then describe an efficient algorithm (Fault Location Algorithm, FLA) pointing out the element(s) which is (are) most likely to be the cause of the received alarms. Although the problem of multiple failure diagnosis is known to be NP-hard, the non-polynomial complexity of the algorithm is pushed ahead in a pre-computational phase, which can be done off-line, and not at the time of a failure. The diagnosis phase is therefore very rapid. We discuss the time and space complexity of the FLA.

Keywords: WDM network components, failure model, failure management, complexity.

1. Introduction

Because of the huge data rates that a single optical fiber can carry using WDM technology, a ribbon break yields the interruption of hundreds of thousands of flows, and the loss of thousands of megabits of data [1]. Survivability of optical networks, which includes fault identification and location, has thus become a crucial problem.

When a failure occurs at the physical layer, the lightpaths that are affected have to be restored as soon as possible so that higher layers do

not see the failure and do not start their own restoration mechanisms. In the meantime, the failure has to be located and repaired.

Failures are located from the alarms received by the management system. When there are two or more simultaneous failures, the number of alarms considerably increases, the alarms arrive intermingled at the management system and the problem of locating the failures becomes even more difficult. The problem of locating multiple failures has been shown to be NP-hard by Rao [2].

The location of the failure(s) must be fast and accurate, so that a small set of faulty candidates can be rapidly identified, before expensive repair actions are undertaken. Failures are less rare than one might expect; [3] has recently reported a failure rate of 1 per year per 300km of fiber. Submarine cables, which are vulnerable to damage from submarines, anchors and fishing gears, have to be repaired once every five weeks [4].

Network management is essential to ensure the good functioning of these networks. Every network management performs several functions, which have been classified into five different functional areas by OSI, and are briefly recalled here.

- **Configuration Management** deals with the initialization of the network components, the establishment, maintenance and updating of relationships among them. These relationships are based on the connections established and cleared down in the network. Configuration management must also reconfigure the network whenever necessary and include routines that are able to inform about any change in the configuration (for example, when a protection switch changes its position, the manager should be informed about the new paths of the established channels).

- **Performance Management** monitors and controls the components in the network. *Monitoring* is the function that tracks activities in the network, whereas the *control* function enables adjustments in the components to improve network performance. The main performance issues are: network capacity utilization, existence of excessive traffic and of bottlenecks, and increase of response time. Performance management collects information from the network and analyzes it so that the management system can recognize situations of performance degradation.

- **Security Management** deals with the generation, distribution and storage of encryption keys. It also monitors and controls access to computer networks and to management information.

- **Accounting Management.** Many network services are charged to the users. Network management performs not only the internal accounting but also other tasks, such as checking allowed access privileges.

- **Fault Management** deals with [5]:

  - fault detection, to know whether there is a failure or not in the network,
  - fault location, to know which is/are the component(s) that has/have failed and caused the received alarms,
  - fault isolation (so-called protection) in order for the network to continue to operate, which is the fast and automated way to restore interrupted connections. In general, it is implemented with protection switches that change positions when the optical powers drop below a certain threshold.
  - network (re-)configuration (so-called restoration) that minimizes the impact of a fault by restoring the interrupted connections using spare equipments. This involves some processing to discover the best paths to re-route the connections.
  - replacement of the failed component(s).

  Protection and restoration mechanisms in optical networks is an active field of research. This chapter focuses on the fault location problem. This problem is not specific to optical networks, but is encountered in many fields, such as electrical power plants monitoring, medical diagnosis, electric circuit analysis, nuclear power station maintenance and management of communication networks at large. Here we consider only its application to optical networks, although many methods are actually valid or have been developed in more general settings. Fault location is particularly important for optical networks, where the quantity of information carried is much larger than over other physical media, making the effects of a failure much more severe.

2. Fault Location Problem definition

Optical communication networks, and all networks in general, need a fault management system that performs fault diagnosis, that is, able to identify the faults that occur from the information given by the network components. A *fault* can be defined as an unpermitted deviation of at least one characteristic parameter or variable of a network element

from acceptable/usual/standard values, whereas a *failure* can be defined as the manifestation of the fault. For example, if the ventilator in a laser stops, and if the temperature increases and overpasses an accepted temperature limit, the fault is the stopped ventilator and the failure is the temperature of the laser, which is too high. Because both terms are closely related, we use them indistinctly.

Fault detection relies on the monitoring of the state of the network components. Simple fault detection mechanisms are often based on locally monitored variables. The faulty values reached by these variables are logged as errors. Critical errors are sent to the network manager as alarms. However, it is not always possible to detect complex faults on the sole basis of locally monitored variables: it is then necessary to have a global knowledge of the network and to do some processing to diagnose the presence (or absence), the nature and the location of the fault. Also, and because the fault can propagate to components that depend on the failed component, the influences of faulty components on other components have to be taken in account to perform an efficient fault management.

Communication networks are built on several layers, each performing fault management functions independently. When a failure occurs, several symptoms or event indications are issued to the network manager from different management layers, and fault management functions start in parallel. Research is carried out to allow interoperability between different layers, to avoid task duplication and increase efficiency.

The fault location problem is solved by Fault Management Systems that take the events generated by the network elements as input (these events can be alarms, warnings or parameters of the network elements), and produce an output, which is the set of network elements whose failures explain the input events.

These fault management systems differ in:

1 the way they solve the problem: using neural networks [6, 7], Finite State Machines [8–10], the history of previous cases [11], a (detailed) modeling of the network [12]. A classification of these methods into model-based methods and Black box learning-based approaches is made in [13]. The first one relies on an abstract model of the network, capturing the dependency relations between the elements and capable of pointing out the element(s) which is (are) most likely to be the cause of the received alarms. The second one does not attempt to model the network in detail, but leaves it as a black box. An abnormal situation is then diagnosed from a set of rules obtained by learning or thanks to expertise of the human manager.

2 the information they need: failure propagation probabilities [14], timestamps, set of established channels [12], etc.

3 the assumptions on which they rely: existence of only single failures [15], existence of multiple failures [12, 16, 17], etc.

4 the quantity of memory they use: large memory requirements [11, 12] to store failure history, modest memory requirements or even absence of memory requirements [15, 14, 9, 8, 10], etc.

5 intolerance [11, 18] or tolerance [6, 12] to false and missing alarms.

3. A failure model of optical networks

We distinguish two classes of network components: (i) Optical components , which take care of the optical signal transmission and are not able to send alarms, except, in some cases, for their own failures (but never when the received optical signals are not as expected); and (ii) Monitoring equipments that are able to send alarms and notifications when the monitored optical signals are not the expected one. The alarms sent by monitoring equipments depend on the equipment type and characteristics. Failure of a monitoring equipment does not interrupt/modify the data transmission, it may only result in the loss of an alarm. It is not as relevant as the failure of an optical component, and therefore is not considered in the present study.

Optical equipment

The optical equipment in a transparent optical network can be listed as follows.

- Transmitters (Txs), which are located at the beginning of an optical channel, are lasers or laser arrays converting electrical signals into optical ones at a certain wavelength. New lasers used in advanced WDM networks are tunable and can change the emission wavelength within a prescribed range. Some lasers do include a wavelength locker so that when the emitted wavelength deviates from the expected value due, for example, to temperature changes, it resets the transmitter to the original wavelength.
- Receivers (Rxs), which are located at the end of an optical channel, convert the received optical signal of a certain wavelength, into an electrical one.
- Optical switches: There are different switches architectures, each of them having different crosstalk characteristics: crossbar, Clos,

Spanke, Benes, and Spanke-Benes. Different technologies can be used for their implementation (except MEMS, all these technologies are used in crossbar architectures: Micro-Electro-Mechanical System (MEMS), Bulk mechanical, Bubble-based waveguide, Liquid crystal, Thermo-optical, SOA).

- Amplifiers, which output a signal at a higher power level than the input signal, usually add distortion to the signal. A fault may occur when the pump laser (in the case of EDFA and Raman amplifiers) fails or when the fiber or a passive component within the amplifiers fails. Other faults may involve the failure of the gain monitoring system causing gain variations. They send alarms for example when the pump laser does not work properly, or when the incoming power falls below a threshold value.

- Optical regenerators and wavelength converters: These two types of elements are included in the same category since they are based upon similar physical principles and technologies. There are three techniques to perform optical wavelength conversion and regeneration: optical gating, interferometric, and wave mixing based.

- Couplers (Splitters/combiners): These elements are included in some demultiplexers/multiplexer architectures. Their key performance parameter is their insertion loss that should be kept low so that when included in serial architectures the overall loss is still acceptable.

- Optical filters: These components have two important applications: to be used to multiplex and demultiplex wavelengths in a WDM system, and to provide equalization of the gain and filtering of noise in optical amplifiers. The most important characteristics of optical filters are: insertion loss, temperature coefficient, flat passband, and sharp passband skirts.

- Protection switches, which receive multiple optical signals, and select the one with an acceptable power level. Note that these elements could also be considered as monitoring equipments, since they send alarms when they change the switch positions due to unacceptable incoming optical powers.

Monitoring equipment

Different types of monitoring equipment exist in the market and are used in transparent optical networks [19, 20]. They monitor the optical

signals using tapping couplers. We assume that for monitoring purposes, the optical signal can be converted to the electrical domain. We distinguish six different types of monitoring equipment:

- Optical Power Meter: detects any change in the power of the optical signal over a wide band. It may be able to send an alarm when the measured power is different from the expected one. When the power decrease is very slow, it takes a longer time to be detected and an alarm takes longer to be generated. In some cases, the BER can be severely degraded without a corresponding decrease in the optical power. For example, amplifiers with automatic gain control deliver signals with the expected power even if they contain too much noise.
- Optical Spectrum Analyzer [21] performs analog optical signal monitoring by measuring the spectrum of the optical signal. The parameters that can be measured are channel power, channel center wavelength and optical signal to noise ratio (OSNR), which is useful and provides important information on the health and quality of the optical signal. For example, it is able to detect OSNR changes (even if they do not cause optical power variations) and the unexpected out-of-band signals. However, this equipment suffers from slow responses and possible sampling errors.
- Eye Monitoring derives from the eye diagram information on the time distortion and interferences. The resulting histogram is used to study the statistical characteristics of the optical signal [22, 23].
- BER Monitoring: After converting the signal to the electrical domain, this equipment is able to calculate the Bit Error Rate which is sensitive to noise and time distortion. BER Testers compare the expected with the received bit pattern and the differences that are found, give the estimated BER. This equipment is sensitive to impairments such as crosstalk, chromatic and polarization mode dispersion, and optical non-linearities. Most of the BER techniques are based on the synchronous [24, 25] or asynchronous [23, 26] sampling of the optical signal.
- Wavemeter is an accurate monitoring equipment able to detect any variation in the used wavelength.
- Pilot tones are signals that travel with the data but can be retrieved easily. For example, pilot tones may use different carrier frequencies from the transmitted signals (in WDM systems), different time slots (in TDM systems), or different codes (in CDMA

systems). Pilot tones can detect transmission disruptions, but not in-band jamming problems unless they affect the pilot tone frequency (which in this case may not affect the transmitted signals).

- Optical Time Domain Reflectometry (OTDR) techniques are based on pilot tones. However, the difference is that OTDR analyzes the pilot tone's echo. This method is used to detect fiber cut and bending that causes data signal loss. OTDR may be able to detect some faults that pilot tones cannot. For example, a jamming signal may not be detected by a pilot tone if it does not cover its frequency, but the pilot tone's echo may contain remainders of the jamming signal that have reflected.

Table 1.1 summarizes the monitoring properties of the components listed above, for a transparent WDM network.

Table 1.1. Failure detection capabilities of monitoring equipment.

Monitoring Component	*Power*	*In-band Jamming*	*Out-band Jamming*	*Wavelength misalignment*	*Time distortion*
Opt. Power Meter	yes	no	no	no	no
Opt. Sp. Anal.	yes	no	yes	no	no
Eye Monitoring	yes	no	yes	no	yes
BER Monitoring	yes	yes	yes	no	yes
Wavemeter	yes	no	no	yes	no
Pilot Tones	yes	no	no	no	yes
OTDR	yes	sometimes	sometimes	no	yes

A WDM network is formed by a number of optical channels, which are composed of the different components listed above. The classification of this section enables us to abstract the network components in two categories: optical (passive) components, and monitoring components. To keep the exposition of the Fault Location Algorithm sufficiently simple in the next section, we assume that only power is monitored, which is the only variable that will be detected by all monitoring components. In reality, the algorithm is much more complex, as each monitoring component belongs to a different sub-category, as defined by the alarms that can be generated according to Table 1.1 (see also [27, 28]). The classification becomes even richer, when we include opaque networks.

Figure 1.1 shows an example of two channels in a transparent network, abstracted using the models elaborated in this section.

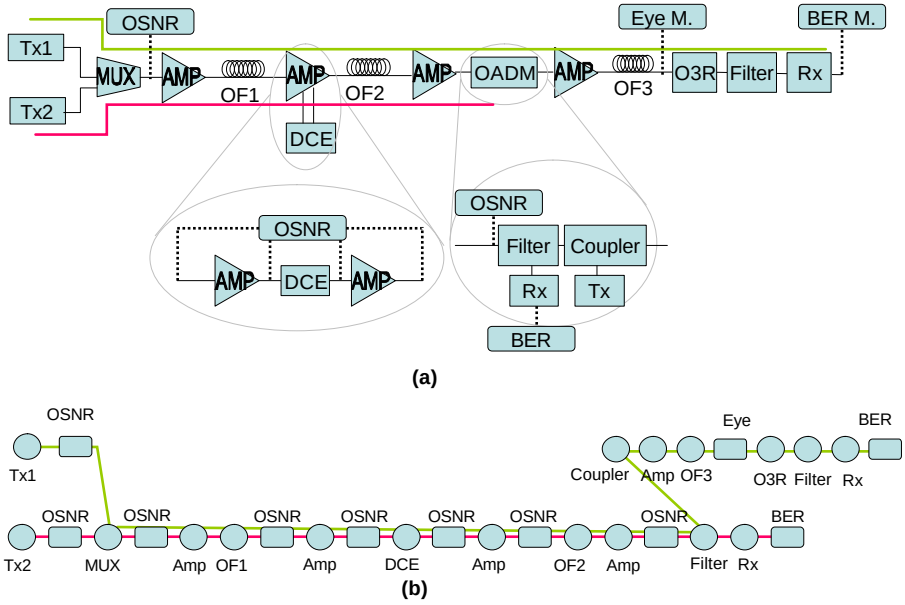

Figure 1.1. Example of a transparent ultra long haul WDM network including some monitoring equipments at some relevant components such as an amplifier with a Dynamic Spectrum Equalizer (DSE) or an Optical Add-Drop Multiplexer with Wavelength Selective architecture. The figure also shows two established channels that have been modeled: the optical components are reprensented by circles, and the monitoring equipments by rectangles.

4. Fault location algorithm (FLA)

Time to locate failure(s) is critical, any good fault location algorithm must be able to locate fault as quick as possible. Unfortunately, the fault localization problem has been shown to be NP-Hard by Rao in [2] even in the ideal scenarios where there are no erroneous alarms. Nevertheless, the computation that has to be carried out when alarms reach the manager can be kept short despite the potentially large size of the network, if we follow the fault location algorithm (FLA) proposed in [12] to pre-compute, as much as possible, the functions that can be executed independently of the received alarms. This phase is called the *pre-computation phase* (PCP). The pre-computation phase is executed only when relationships between fault sources and alarm elements change, for example when a channel is set up or torn down, not when the alarms are received. Once the manager starts receiving alarms from

the network, the algorithm does not have to perform complex computation but simply traverses a binary tree. This minimizes the time the algorithm needs to deliver results to the manager when failures occur.

FLA to localize single failure in the ideal case

We first start by solving the problem of localizing single failure. In this case, only a single network component can fail at a time and when a network component fails, the monitoring components that follow it on a channel will issue alarms.

Let us illustrate the steps of the algorithm on the network in Figure 1.1. For the sake of simplicity in our description, we will only consider a fraction of the network as in Figure 1.2, this is equivalent to assume that all optical components preceding $OF2$ in both channel 1 and channel 2 never fail. There are two established channels in the network ($\mathcal{CH} = \{CH_1, CH_2\}$). We label the network elements by p for optical (passive) components and e for the monitoring components. In this example, there are 10 passive elements $p_1, ..., p_{10}$ and 4 monitoring elements $e_1, ..., e_4$.

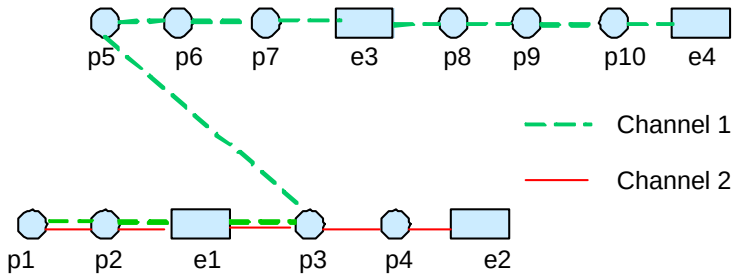

Figure 1.2. Reduced network example

The PCP consists in the following steps.

1 Compute the *domain* for each optical component. The *domain* of an optical component is defined as the set of monitoring equipments that will trigger alarms when that optical component fails. In the example, we obtain

$$\begin{aligned}
Domain(p_1) &= \{e_1, e_2, e_3, e_4\} \text{ , } Domain(p_2) = \{e_1, e_2, e_3, e_4\},\\
Domain(p_3) &= \{e_2, e_3, e_4\} \text{ , } Domain(p_4) = \{e_2\},\\
Domain(p_5) &= \{e_3, e_4\} \text{ , } Domain(p_6) = \{e_3, e_4\},\\
Domain(p_7) &= \{e_3, e_4\} \text{ , } Domain(p_8) = \{e_4\},\\
Domain(p_9) &= \{e_4\} \text{ , } Domain(p_{10}) = \{e4\}.
\end{aligned}$$

2 Group all identical domains into *fault classes*

$$\begin{aligned} C_1 &= Domain(p_1) = Domain(p_2) = \{e_1, e_2, e_3, e_4\}, \\ C_2 &= Domain(p_3) = \{e_2, e_3, e_4\}, \\ C_3 &= Domain(p_5) = Domain(p_6) = Domain(p_7) = \{e_3, e_4\}, \\ C_4 &= Domain(p_4) = \{e_2\}, \\ C_5 &= Domain(p_8) = Domain(p_9) = Domain(p_{10}) = \{e_4\}. \end{aligned}$$

3 Compute the set $P(C_i)$ of elements that belong to the same fault class C_i, in other words, the optical components whose failures cannot be distinguished by the network manager from the monitoring information. In this example, these sets are:
$P(C_1) = \{p_1, p_2\}$, $P(C_2) = \{p_3\}$, $P(C_3) = \{p_5, p_6, p_7\}$, $P(C_4) = \{p_4\}$ and $P(C_5) = \{p_8, p_9, p_{10}\}$.

4 Construct the dependency matrix where each column of the dependency matrix is a binary vector $Bin(C_i)$ with as many elements as the number of monitoring components in the established channels (4 in this example). The jth component of $Bin(C_i)$ is equal to 1 if the jth monitoring equipment fires alarm when elements in the fault class C_i fail, and 0 otherwise. In the example, let us denote by G the dependency matrix, then

$$G = \begin{bmatrix} 1\,0\,0\,0\,0 \\ 1\,1\,0\,1\,0 \\ 1\,1\,1\,0\,0 \\ 1\,1\,1\,0\,1 \end{bmatrix}.$$

5 Build a binary tree for single failures: The binary tree has a depth equal to the number of alarm elements and leaves correspond to different binary combinations. Occupied leaves point to the fault class C_i whose corresponding column in the dependency matrix, $Bin(C_i)$, is the path from the root of the tree to the leaves. The binary tree constructed for this example is shown in Figure 1.3.

FLA to localize multiple failures in the ideal case

Let us now consider the case where several failures may happen in a short interval of time so that the alarms reaching the manager are intermingled. The algorithm has to be able to distinguish the failures based on the received alarms. To solve this problem, the binary tree is extended by adding leaves which correspond to multiple failures. This amounts to computing the domains of simultaneous failures which are the union

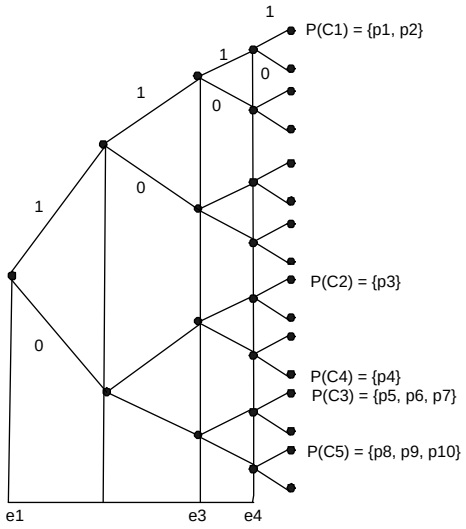

Figure 1.3. Binary tree for single failures

of the domains of single failures. We begin with double failures. The construction are as follows. Let $C_i = Domain(p_i)$, $C_j = Domain(p_j)$ and C_k be the domain for double failures of p_i and p_j , then

$$Bin(C_k) = Bin(C_i \cup C_j) = Bin(C_i) \vee Bin(C_j),$$

where $\vee$ stands for the point-wise OR operation between $Bin(C_i)$ and $Bin(C_j)$.

If $Bin(C_k)$ is equal to $Bin(C_l)$ for an existing failure class C_l, the leaf is already occupied by the domain of a single failure. We can reasonably assume that single failure is more likely to occur than multiple failures, hence the occupied leaf points to the more likely single failure. Conversely, if $Bin(C_k)$ is different from any of the existing leaves, a new leaf is then occupied and pointed to the double failures. Once all the new leaves corresponding to double failures are filled, we proceed likewise for triple failures, etc. Note that if at some point of this procedure, there is a C_k corresponding to a single failure which is such that for all the already computed C_i's,

$$Bin(C_i) \vee Bin(C_k) = Bin(C_i) \text{ or } Bin(C_i) \vee Bin(C_k) = Bin(C_k),$$

then C_k will not contribute to any new leaf anymore. Therefore, it needs not be considered for further steps with more failures. This property allows us to decrease the number of binary vectors corresponding to single failures needed for computing the domain of multiple ones. The process finishes when the set of single failures becomes empty.

Let us consider the example shown in Figure 1.2. The output of this part of the algorithm is the following:

$Bin(C_4) \vee Bin(C_5) = Bin(C_6)$.

One new failure class C_i has been found. It is defined by the vector $Bin(C_6) = (0, 1, 0, 1)$. The resulting binary tree is shown in Figure 1.4.

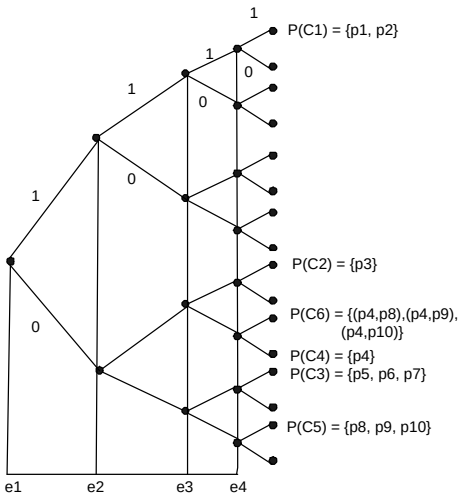

Figure 1.4. Binary tree for multiple failures

FLA to localize multiple failures in the non-ideal case

Alarm errors are unavoidable in network diagnosis. There are two kinds of erroneous alarms: missing alarms and false alarms . Missing alarms occur when some alarms are lost, or arrive with such a delay that they cannot be considered during the ongoing computation of the algorithm or if a monitoring component is itself defective. False alarms can occur when, due to abnormal situations, a monitoring component sends an alarm although there is no real failure. The empty leaves in the binary tree for multiple failures in Figure 1.4 are those corresponding to missing or false alarms.

The tree can be viewed as a particular block error-correcting code, whose codewords have the property that the logical OR of any two codewords is another codeword. One empty leaf of the tree corresponds to an erroneous word, and the error correction task would be to replace the erroneous word by the correct codeword whose Hamming distance with the received word is minimal. The problem of finding the near-

est codeword to an arbitrary word can be shown to be NP-complete, but however, there are special instances where this problem is easy to solve. We will discuss these cases in the next section when we discuss the computational complexity of the FLA. In the rest of this section, we will show how the binary tree can be extended to deal with erroneous alarms.

Contrary to the use of error-correcting codes for data transmission, the manager of a network does not require unique decoding. Indeed, he/she will prefer to get the set of all faulty candidates whose domains are close to the received alarms. In fact, we can use the bounded distance decoding approach by giving all the codewords that realize a given alarm mismatching threshold as possible answers. For example, if we tolerate a maximum of m_1 missing alarms and m_2 false alarms, for a set of alarms $\mathcal{R}$, the codewords that fall within this margin from the binary vector $Bin(\mathcal{R})$ are the codewords that have a '1' when a $Bin(\mathcal{R})$ has a '0' in at most m_1 positions, and have a '0' when $Bin(\mathcal{R})$ has a '1' in at most m_2 positions.

The error correction part of the algorithm can be computed off-line, in the PCP module, or on-line, when the alarms are received. In the first case, one computes for each occupied leaf (i.e., for each codeword), the binary vectors whose number of 0's and 1's differ to this codeword respectively by at most m_1 and m_2 positions. The corresponding fault classes C_i are added to the list pointed by the leaf (see for example in Figure 1.5 the new binary tree when $m_1 = 1$ and $m_2 = 0$ and in Figure 1.6 the new binary tree when $m_1 = 0$ and $m_2 = 1$). The on-line approach is discussed in the next section when trade-offs between storage requirement and time requirement for the FLA are considered.

Let us illustrate the algorithm with different scenarios in the example of Figure 1.2 when the set of received alarms is $\mathcal{R} = (a_2, a_3)$ (a_2 is issued by e_2, and a_3 is issued by e_3). Hence, $Bin(\mathcal{R}) = (0110)$, which corresponds to an empty leaf of the tree in Figure 1.4. Let us check the following scenarios:

- $m_1 = 1$, $m_2 = 0$: One missing alarm and no false alarm are tolerated. In this case (Figure 1.5), the output of the algorithm is the leaf $Bin(C_2) = (0111)$ with one mismatch which corresponds to the following solution:

  Failure of p_3 with one mismatch.

- $m_1 = 0$, $m_2 = 1$: One false alarm and no missing alarm are tolerated. In this case (Figure 1.6) the output of the algorithm is the leaf $Bin(C_4) = (0100)$ with one mismatch which corresponds to the following solution:

Failure of p_4 with one mismatch.

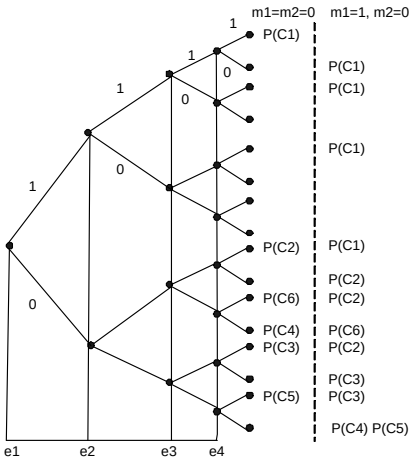

Figure 1.5. Binary tree when $m_1 = 1$ and $m_2 = 0$

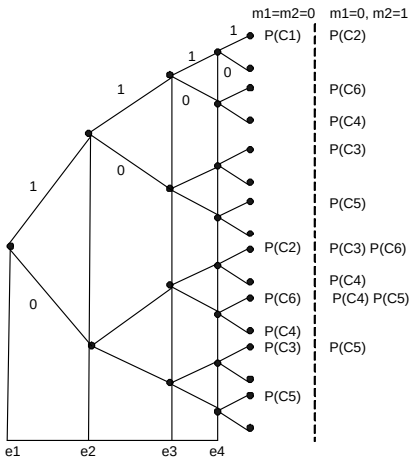

Figure 1.6. Binary tree when $m_1 = 0$ and $m_2 = 1$

In the considered example, there are only four monitoring elements, the tolerance $m_1 = m_2 = 1$ is too loose because it amounts to accepting 50% erroneous alarms. For this reason, we will not consider this tolerance or other tolerances with higher values of m_1 and m_2.

5. Algorithmic complexity

In this section, we present the complexity of the FLA and show how the algorithm can be modified to meet certain complexity requirements.

Let us denote by n the number of different components in the established channels, by n_a the number of alarming elements and by $n_{na} = n - n_a$ the passive (non-alarming) elements. n_a is also the depth of the binary tree and therefore the binary vector size. Let us also denote by t the number of single fault classes and by c the number of established channels.

Pre-Computing Phase (PCP)

Time requirements.

1 Computation of *Domains* : The computational complexity for this step is at most of $O(\frac{n(n-1)c}{2}) = O(n^2)$ because the number of channels c is much less than the number of network components.

2 Grouping of *Domains* into *fault classes* and production of the dependency matrix: each domain is associated to one n_a-dimensional binary vector. The computation time needed to group identical domains, and to compute the codeword $Bin(C_i)$ is at most $O(n^2 n_a)$.

3 Computation of the domains of multiple failures and of the corresponding codewords. If d is the total number of different classes C_i (or equivalently of different codewords), the computational time of this step is $O(\frac{d(d-1)}{2} n_a)$. Since $d \leq 2^t$, the (very) worst case bound is $O(4^t n_a)$.

4 Computation of codewords with mismatching thresholds m_1 and m_2. This step can be implemented as follows. For each codeword $\overrightarrow{x} = Bin(C_i)$, it finds all the vectors $\overrightarrow{y}$ such that $\sum_{k=1}^{n_a} [x_{ik} - y_k]^+ \leq m_1$ and $\sum_{k=1}^{n_a} [y_k - x_{ik}]^+ \leq m_2$ where $[z]^+ = z$ if $z > 0$ and 0 otherwise. The worst case computation time is $O(2^{n_a} \sum_{k=1}^{m_1+m_2} C_k^{n_a})$, where $C_k^{n_a} = \frac{n_a!}{(n_a-k)!k!}$ is the combinatorial coefficient (n_a, k).

Storage requirements. The storage memory for a binary tree which has 2^{n_a} leaves is $O(2^{n_a})$.

FLA Main Part

1 Compute $Bin(\mathcal{R})$: This module only requires a computational time of $O(n_a)$.

2 Find $P(C_i)$'s of the $Bin(\mathcal{R})$ leaf: This module also requires a computational time of $O(n_a)$.

The total computational time of the on-line part of the algorithm is $O(n_a)$. This low complexity comes at a cost of a high computation time in the PCP which is $O(4^t n_a)$ and an exponential storage memory of $O(2^{n_a})$.

The Error Correction Problem

Let us consider a received alarm vector $\overrightarrow{r} = Bin(\mathcal{R})$, which is a binary vector of dimension n_a. The problem of correcting the erroneous alarms amounts to finding a codeword $Bin(C_i)$, for some fault class C_i, with the minimum Hamming distance to the received word; that is, such that the number of 1's by which $\overrightarrow{r}$ and $Bin(C_i)$ differ is minimal for all possible fault classes C_i.

Note at this point that the code is not linear, and that it will have very poor performances in terms of minimal distance. Nevertheless,

the following results are important for managing monitoring equipments in a transparent optical network. When there is no missing alarm, i.e., $m_1 = 0$, finding the nearest codeword to a received word can be achieved in polynomial time. The polynomial time algorithm first finds all the codewords that do not have 1s at the positions where the received word has 0s, and then sequentially remove all the common '1' entries between the received word and each of the codewords identified above. The '1' entries left in the received word after this operation correspond to the false alarms. On the other hand, when there are missing alarms, i.e. $m_1 > 0$, finding the nearest codeword to a received word is NP-complete. This is due to the fact that when $m_1 > 0$ and $m_2 = 0$, the problem of finding the nearest codeword becomes the red-blue set cover problem as defined in [29]. The red-blue set cover problem is an extension of the set cover problem which is an NP-complete problem, hence it is also NP-complete.

From the above result, we see that for the error correction problem, it is more desirable to have false alarms than missing alarms. This has a significant practical influence as the most common source of alarm errors comes from setting wrong threshold values for monitoring equipments. Our analysis above indicates that one should set the threshold values high rather than low in order to restrict all missing alarms (with the possibility of producing more false alarms).

Trading time for storage

In this section, we look at the problem of reducing the exponential storage requirement for the binary tree. Since the storage memory is proportional to the number of leaves, the way to reduce the storage memory is to reduce the number of leaves in the tree. The leaf reduction can be achieved in two steps.

In the first step, we remove all the leaves that correspond to erroneous alarms (that is, those obtained for $m_1 > 0$ and/or $m_2 > 0$) and carry out the error correction module of the algorithm on-line. There are two reasons for carrying out this module in the on-line part of the algorithm : first, it helps speeding up the PCP phase of the algorithm; second, the overall complexity of the algorithm is lower, as the received alarm vector $\overrightarrow{r} = Bin(\mathcal{R})$ is compared only with the codewords corresponding to single or multiple failures in the absence of erroneous alarms (i.e., with $m_1 = m_2 = 0$), not with all the leaves. This is particularly true in cases where the problem of finding the nearest codeword can be solved in polynomial time (i.e., when $m_1 = 0$).

After this first step, we are left with a tree whose leaves point to at most one set $P(C_i)$ each. Each of these sets $P(C_i)$ is made of subsets of network components, the union of the domains of all elements within a subset is C_i. Recall that in the tree construction process for multiple faults, among all the subsets of faulty elements which satisfy the condition that the union of the domains of all elements within a subset is C_i, we only keep in $P(C_i)$ subsets with the smallest cardinality. Let us define a failure scenario associated to a set $P(C_i)$ as a set of fault classes for single failure which satisfies the condition that the unions of the fault classes in that set is C_i. In the second step of the leaf reduction process, we further reduce the leaves in the binary tree by exploiting the fact that the online decoding time for a leaf depends on the number of failure scenarios associated to the corresponding set $P(C_i)$. We proceed as follows. Pick a number $m \geq 1$. If the number of failure scenarios associated to $P(C_i)$ is less than m, then $P(C_i)$ is removed from the tree. Otherwise, it is maintained in the tree.

With such a trimmed tree, whenever the received codeword $\overrightarrow{r}$ corresponds to a set $P(C_i)$ that is associated to at least m failure scenarios, it is retrieved as before, since it is pointed by the corresponding leaves. Otherwise, if the received codeword $\overrightarrow{r}$ corresponds to a set $P(C_i)$ that is associated to less than m failure scenarios, we find these on-line, using an exhaustive search. This exhaustive search only needs to examine at most m different sets of single fault classes that correspond to $P(C_i)$ to find the optimal subsets. Let us denote by t the number of single failure codewords. Since there is a maximum of 2^t different possible subsets of single fault classes, there are less than $2^t/m$ codewords whose corresponding set $P(C_i)$ is associated to more than m different failure scenarios. Furthermore, the computational time to identify one subset of fault classes that corresponds to a codeword is $O(n_a t)$. Hence, by storing in the binary tree only the codewords whose corresponding set $P(C_i)$ that has more than m failure scenarios, one can store less than $2^t/m$ leaves in the binary tree and still guarantee a worst case decoding time of $O(mn_a t)$. By varying m, one can actually choose different levels of time and storage complexities for the FLA.

The on-line decoding strategy when a word $\overrightarrow{r} = Bin(\mathcal{R})$ is received is as follows. The algorithm iteratively identifies all codewords within a mismatching threshold m_1 and m_2 from $\overrightarrow{r}$. This step has a time complexity of $O(\sum_{k=1}^{m_1+m_2} C_k^{n_a})$. For each codeword, the algorithm first traverses the binary tree in $O(n_a)$ time, to find the optimal failure scenario. If the codeword is not in the binary tree, the tree search will fail, the algorithm will then carry out an exhaustive search in time $O(mn_a t)$ to find the optimal failure scenario, where m is the number of failure

scenarios that are associated with $\overrightarrow{r}$. The worst case bound for the on-line decoding module is $O(mn_at\sum_{k=1}^{m_1+m_2} C_k^{n_a})$.

6. Conclusion

Ideally, methods to locate faults in a WDM network should be sensitive to hard and soft failures, should rely on the analog signals or cope with lost and false alarms, should avoid the use of probabilities since they are time-varying parameters that are difficult to estimate, should be aware of the possible failure scenarios and how they propagated under correlation rules [30], or an appropriate taxonomy of the network elements as developed in this chapter and [12], and should rapidly identify an accurate set of faulty candidates. Despite the NP-hard complexity of the multiple-failure problem, an efficient algorithm that meets these requirements has been described in this chapter. We have also discussed the complexity of the various modules of the algorithm, and shown how the NP-hard part of the algorithm can be done off-line.

Without monitoring equipments along the optical physical layer, optical networks provide few information about failures. These monitoring components detect different signals and thus failures, and the Fault Location Algorithm should be extended in order to take this into account.

Instead of deploying additional monitoring equipment, or in complement to this deployment, a second solution is to exploit the interoperability between the layers so that a single fault management system using information from the different layers could become more accurate and locate a larger number of failure scenarios.

Acknowledgments

This work was financially supported by grant DICS 1830 of the Hasler Foundation, Bern, Switzerland.

References

[1] C. Metz. IP Protection and Restoration. *IEEE Internet Computing*, pages 97–102, March-April 2000.

[2] N. S. V. Rao. Computational Complexity Issues in Operative Diagnosis of graph-based Systems. *IEEE Transactions on Computers*, 42(4):447–457, April 1993.

[3] P. Arijs, P. van Caenegem, P. Demeester, P. Lagasse, W. Van Parys, and P. Achten. Design of ring and mesh based wdm transport networks. *Optical Networks Magazine*, pages 20–45, 2000.

[4] Submarine Networks. *FibreSytems*, 3(5):26, June 1999.

[5] S. Abek H. Hegerin and B. Neumair. *Integrated Management of Networked Systems*. Morgan Kaufmann Publishers, 1998.

[6] R. Gardner and D. Harle. Alarm Correlation and Network Fault Resolution using Kohonen Self-Organising Map. *Globecom 97 proceedings*, pages 1398–1402, 1997.

[7] C. Rodriguez, S. Rementeria, J. I. Martin, A. Lafuente, J. Muguerza, and J.Perez. A Modular Neural Network approach to Fault Diagnosis. *IEEE Transactions on Neural Networks*, 7(2):326–340, March 1996.

[8] C. S. Li and R. Ramaswami. Fault Detection and Isolation in transparent All-Optical Networks. *IBM Research Report*, RC-20028, April 1995.

[9] C. Wang and M. Schwartz. Fault Detection with Multiple Observers. *IEEE INFOCOM Proc.*, pages 2187–2196, 1992.

[10] A.T. Bouloutas, G. W. Hart, and M. Schwartz. Fault Identification Using a FSM model with Unreliable Partially Observed Data Sequences. *IEEE Transactions on Communications*, 41(7):1074–1083, July 1993.

[11] H.-G. Hegering and Y. Yemini (Editors). *Integrated Network Management*. Elsevier Science Publishers B.V., 1993.

[12] Carmen Mas and P. Thiran. An efficient algorithm for locating soft and hard failures in WDM network. *JSAC special issue on Protocols and Architectures for next generation optical WDM networks*, end 2000.

[13] Carmen Mas and P. Thiran. A review on fault location methods and their application to optical networks. *Optical Networks Magazine*, 2(4), July/August 2001.

[14] R. H. Deng, A. A. Lazar, and W. Wang. A probabilistic Approach to Fault Diagnosis in Linear Lightwave Networks. *IEEE JSAC*, 11(9):1438–1448, December 1993.

[15] N. S. V. Rao. On Parallel Algorithms for Single-Fault Diagnosis in Fault Propagation Graph Systems. *IEEE Transactions on Parallel and Distributed Systems*, 7(12):1217–1223, December 1996.

[16] J. Kleer and B.C. Williams. *Diagnosing multiple faults-Artificial Intelligence*, volume 32. Elsevier Science Publishers, 1987.

[17] Y. Y. Yang and R. Sankar. Automatic failure isolation and reconfiguration. *IEEE Network*, pages 44–53, September 1993.

[18] I. Katzela and M. Schwartz. Schemes for Fault Identification in Communication Networks. *IEEE/ACM Transactions on Networking*, 3(6), December 1995.

[19] M. Medard, S. R. Chinn, and P. Saengudomlert. Node wrappers for QoS monitoring in transparent optical nodes. *Journal of High Speed Networks*, 10:247–268, 2001.

[20] R. Habel, K. Roberts, A. Solheim, and J. Harley. Optical domain performance monitoring? optical fiber communication conference. Baltimore, 2000.

[21] K. J. Park S. K. Shin and Y. C. Chung. A novel optical signal-to-noise ratio monitoring technique for wdm networks. Baltimore, 2000.

[22] K. Mueller et al. Application of amplitude histograms for quality of service measurements of optical channels and fault identification. Madrid, 1998.

[23] I. Shake, H. Takara, S. Kawanishi, and Y. Yamabayashi. Optical signal quality monitoring method based on optical sampling. *Electronic Letters*, 34(22):2152, October 1998.

[24] S. Ohteru and N. Takachio. Optical signal quality monitor using direct Q-factor measurement. *IEEE Photonics Technology Letters*, 11:1307, 1999.

[25] R. Wiesmann, O. Bleck, and H. Heppne. Cost-effective performance monitoring in wdm systems. Baltimore, 2000.

[26] N. Hanik, A. Gladisch, C. Caspar, and B. Strebel. Application of amplitude histograms to monitor performance of optical channels. *Electronic Letters*, 35:403, 1999.

[27] Carmen Mas and Ioannis Tomkos. Failure detection for secure optical networks. *International Conference on Transparent Optical Networks*, Vol.1, Mo.C2.1, 2003.

[28] Carmen Mas, Ioannis Tomkos, and Ozan Tonguz. Optical Networks Security: A Failure Management Framework. *Information technologies and Communications ITCom 2003*, 2003.

[29] R. D. Carr, S. Doddi, G. Konjevod, and M. Marathe. On the red-blue set cover problem. *Symposium on Discrete Algorithms*, 2000.

[30] A.T. Bouloutas, S. Calo, and A. Finkel. Alarm Correlation and Fault Identification in Communication Networks. *IEEE Transactions on Communications*, 42(2/3/4):523–533, Feb/March/April 1994.