A Probabilistic Approach for Failure Localization

Tania Panayiotou
Department of Electrical and Computer Engineering
University of Cyprus
Email: panayiotou.tania@ucy.ac.cy

Sotirios P. Chatzis
Department of Electrical Engineering, Computer Engineering, and Informatics
Cyprus University of Technology
Email: sotirios.chatzis@cut.ac.cy

Georgios Ellinas
Department of Electrical and Computer Engineering
University of Cyprus
Email: gellinas@ucy.ac.cy

***Abstract*—This work considers the problem of fault localization in transparent optical networks. The aim is to localize single-link failures by utilizing statistical machine learning techniques trained on data that describe the network state upon current and past failure incidents. In particular, a Gaussian Process (GP) classifier is trained on historical data extracted from the examined network, with the goal of modeling and predicting the failure probability of each link therein. To limit the set of suspect links for every failure incident, the proposed approach is complemented with the utilization of a Graph-Based Correlation heuristic. The proposed approach is tested on a dataset generated for an OFDM-based optical network, demonstrating that it achieves a high localization accuracy. The proposed scheme can be used by service providers for reducing the Mean-Time-To-Repair of the failure.**

I. Introduction

In practice, localization of network faults is usually performed by knowledgeable network operations teams, who work with associate on-call engineers to resolve problems in real time with the help of monitoring data. Such an approach can be time consuming; this may be further worsen by monitoring noise and the increasing size of the networks. As the scale of a network grows, automated fault localization becomes increasingly important, since it can reduce MTTR and service disruption.

In optical networks, the failure of a network element can result in the failure of several lightpaths and can, thus, cause huge data losses. This problem becomes more crucial when lightpaths are migrated to high bit rates, such as 40, 100 Gbps and beyond (with the latter being expected to be accommodated in future OFDM-based elastic optical networks). Commonly observed faults in optical networks are caused by fiber cuts, equipment failures, excessive bit errors, and human error. After the detection of a failure, protection/restoration techniques are invoked so that the traffic is recovered prior to the localization and repair of the faulty component [2]–[4]. Most of the existing works deal with the single-link failure scenario, since the occurrence of more than one simultaneous link failures in an optical network is not very common.

In opaque optical networks, it is relatively simple to perform failure detection and localization [11]. However, in transparent optical networks, failure localization becomes a challenging task. Existing fault localization approaches [6], [11]–[15], where the term *fault* may refer to both equipment and/or link failures, assume the use of monitoring equipment for accurately localizing failures. Monitoring equipment is able to send alarms and notifications when the optical signal deviates from its expected value. Thus, failure management relies on the information collected from the network through captured signal alarms. Even though monitoring does not influence optical signal transmission, it is costly, it may result in noisy alarms, it leads to extra bandwidth resource utilization, as well as to additional delay and extra control complexity at the electrical domain.

In this work, we propose localizing single-link failures in transparent optical networks, as accurately as possible, by applying advanced statistical machine learning techniques on historical data that can be found readily available in network management databases, and provide information on the network state upon failure incidents. We assume that a Path Computation Element (PCE) is present (i.e., the ABNO architecture [5]) that is resource aware and is able to maintain a centralized traffic engineering (TE) database with detailed spectrum availability information. Thus, it is capable of specifying the full details of each link and each lightpath. From this traffic engineering database, information can be extracted and stored in a knowledge database for training/validating/applying the fault localization model proposed. Further, we assume that a failure is detected (at the destination nodes of the established lightpaths) through a number of monitoring alarms capable of identifying, among the established lightpaths, the affected lightpaths. In the data plane, the optical network is considered to be equipped with monitors, e.g., installed in the digital signal processing of the coherent receivers (in a similar manner as the one assumed in [6]). The affected/unaffected lightpahts can be appropriately correlated for reducing the number of links suspected for causing the failure. As the lightpath correlation procedure may not unambiguously identify the suspect link [6], a link failure probability is calculated aiming at indicating the truth likelihood of each suspect link to be the faulty link.

The novelty of this work stems from the fact that the proposed fault localization scheme calculates a link failure probability for each one of the links being suspect of causing the failure, aiming at reducing the Mean-Time-To-Repair (MTTR). In particular, the proposed fault localization scheme consists of two phases: (A) The first phase is activated upon the detection of a failure and consists of correlating the affected and unaffected lightpaths. Similar to [6], we assume that once a failure is detected, the affected lightpaths are identified according to a number of alarms triggered by the monitoring equipment at the data plane. For the path correlation phase, a Graph-Based Correlation (GBC) heuristic is developed. (B) The second phase is activated only if the GBC heuristic reports that more than one links are suspect for causing the failure. This phase consists of computing a failure probability for each suspect link, aiming at identifying as accurately as possible the failed link. The failure probability is computed by a Gaussian Process (GP) classifier trained on a set that describes past failure incidents. In particular, the GP classifier is trained according to a set that describes both the time dependencies between the successively occurred failures and the network state upon each failure incident.

As the proposed approach targets the accurate localization of the failure, it can be used by service providers for reducing the MTTR, and consequently the human effort required for the identification and repair of the faulty component [2]. The main advantage of the proposed approach is that it does not assume the utilization of any probing lightpaths (extra monitoring equipment), thus reducing the network cost and the control management complexity. For ensuring the network resiliency we assume that a fault protection scheme is invoked immediately after the detection of the failure, independently from the fault localization procedure.

II. Approach Motivation

In this work we define as *link failure* the incident caused by the abnormal operation of a fiber link or any component attached to a fiber link. The proposed fault localization scheme is based on the exploitation of the Mean-Time-Between-Failures (MTBF). The MTBF is usually modeled by well known distributions, such as the Exponential distribution or the Weibull distribution. Both distributions have a scale parameter describing the MTBF. The Weibull distribution, has also a shape parameter that describes how MTBF changes over time. Thus, the Weibull distribution is also capable of describing the ageing effects of the network components. In this work, and according to [7], we assume that the MTBF follows the Weibull distribution, with each one of the network links being described by their own scale and shape parameters. In particular, in [7] it was shown that the empirical CDF of the MTBF (of optical network related failures) is well approximated by a Weibull distribution in which the failure rate changes over time. This outcome, was the result of analyzing a dataset consisting of failure information for all links in the continental US for a period of seven months.

Specifically, we generate a dataset of failure information, assuming that the MTBF of each link in the network follows the Weibull distribution. To create a dataset that captures the time dependencies between the successively occurred failures, we count (1) the total number of failures in the network $C(i)$ and (2) the number of failures $c_j(i)$ associated with each link e_j in the network, up to the last known failure incident i. Then, by dividing $c_j(i)$ with $C(i)$, for each incident i, we manage to keep failure information regarding the failure rate of each link, which is relative to the failure rate of the other links in the network. In the dataset we also include information describing the network state upon each incident i (i.e., affected and non affected connections).

Such historical information can then be utilized by a probabilistic model for finding the truth likelihood of a link e_j to be the failed link upon incident i. In this work we use a GP classifier for calculating such probability. The class of GPs is one of the most widely used families of stochastic processes for modeling dependent data observed over time. Thus, GPs are useful for sequential data, such as time-series and tracking applications and can be particularly used for active data selection in such systems [8]. GPs constitute one of the most important Bayesian machine learning approaches and are based on a particularly effective method for placing a prior distribution over the space of regression functions. They have a small number of tunable parameters, can be trained on relatively small training sets, and exhibit significant robustness to outliers and the ability to handle sparse data without becoming prone to overtraining. Compared to another popular form of discriminative kernel machines, i.e., the Support Vector Machine (SVM) [9], GPs possess several advantages, with the most significant being that the GP model produces an output with a clear probabilistic interpretation, providing a measure of uncertainty for the obtained predictions, contrary to SVMs which merely provide point predictions [10].

III. Problem Formulation

The approach assumes a network topology represented by the graph $G = (V, E)$, where V corresponds to the set of network nodes, and E to the set of links in the network. Specifically, we denote $E = \{e_j | j = 1, ..., D\}$, where D is the total number of links in G. As pointed out, upon a link failure the affected destination nodes will report the abnormal network behavior. Upon failure detection, the Graph-Based Correlation (GBC) heuristic described in detail in Section IV-A is utilized for reducing the number of suspect links in the network. Briefly, the GBC heuristic, upon failure i, takes as input the set of affected paths $P(i)$ and the set of unaffected paths $P'(i)$, and through a path correlation procedure returns the set of suspect failed links $S(i)$. Assuming that for every incident i the GBC returns a set $S(i)$ with more than one suspect links, the set $S(i)$ is utilized for creating the observation vector $\boldsymbol{x}(i) = [x_1(i), ..., x_D(i)]$ (*independent* variable), where

$$x_j(i) = \begin{cases} -\frac{c_j(i)}{C(i)}, & \text{if } e_j \in S(i) \\ 0, & \text{otherwise.} \end{cases} \quad \forall e_j \in E \tag{1}$$

$c_j(i)$ is the number of times link e_j has failed up to incident $i-1$, and $C(i)$ is the total number of failures in the system up to incident i, such that $C(i) = \sum_{j=1}^{D} c_j(i) + 1$. Each vector $\boldsymbol{x(i)}$ is associated with a corresponding *dependent* variable vector $\boldsymbol{y(i)} = [y_1(i), ..., y_D(i)]$, where

$$y_j(i) = \begin{cases} 1, & \text{if } e_j \text{ has failed at } i \\ -1, & \text{otherwise} \end{cases} \quad \forall e_j \in E \tag{2}$$

and $\sum_{j=1}^{D} y_j(i) = -D+1$, as we are only considering a single-link failure scenario. Thus, a training set $\mathcal{D} = \{(\boldsymbol{x(i)}, \boldsymbol{y(i)})\}|i = 1, ..., n\}$ is created, with n being the total number of known failure incidents. The dataset $\mathcal{D}$ is then utilized for training the used GP classifier, as described in detail in Section IV-B. The generation procedure of the dataset $\mathcal{D}$ follows in Section V.

IV. Link Failure Localization

As pointed out, a GP classifier is trained with the dataset $\mathcal{D}$ for learning to predict the link failures in the network. According to Eq. 1, the creation of dataset $\mathcal{D}$ depends on a set of candidates $S = \{S(i)|i = 1, ...n\}$, calculated by means of the GBC heuristic. As such, we first describe the GBC heuristic, and then proceed with a brief presentation of the GP classification technique.

A. GBC Heuristic

For the description of the GBC heuristic we define as $P'(i) = \{p'_m(i)|m = 1, ..., t'(i)\}$ the set of unaffected paths upon incident i, and as $P(i) = \{p_m(i)|m = 1, ...t(i)\}$ the set of affected paths upon incident i. Note that a path is considered *affected* if it passes through the failed link. Otherwise, it is considered *unaffected*. Additionally, a path is defined as the set of links it traverses. On this basis, the GBC heuristic is described in Algorithm 1.

Algorithm 1 GBC heuristic

Input: The sets P and P', where $P = \{P(i)|i = 1, ...n\}$ and $P' = \{P'(i)|i = 1, ...n\}$.
Output: The set S, where $S = \{S(i)|i = 1, ...n\}$.

```
1: for $i = 1$ to $n$ do
2:     $A(i) = \bigcap_{m=1}^{t(i)} p_m(i)$
3:     $A'(i) = \bigcup_{m=1}^{t'(i)} p'_m(i)$
4:     if $A(i) = \emptyset$ then
5:         $A(i) = p_1(i)$
6:     end if
7:     $S(i) = A(i) - (A(i) \bigcap A'(i))$
8: end for
9: return $S$
```

The basic idea of the GBC heuristic is to intersect the affected paths in order to get the set of links $A(i)$ that are common to every affected path. If, however, only a single path is affected, then the set $A(i)$ is just the set of links traversing this path. Then, the GBC removes from set $A(i)$ the links that cannot be considered suspect as they also belong to the set of unaffected links $A'(i)$. The heuristic terminates by returning the set $S(i)$. Note that GBC assumes that there always exists at least one affected path that triggers the fault localization procedure.

B. GP Classification

If we consider an observation space $\mathcal{X}$, then a GP $f(\boldsymbol{x})$, where $\boldsymbol{x} \in \mathcal{X}$, is defined as a collection of random variables, any finite number of which have a joint Gaussian distribution [16]. A GP is completely defined by its mean function and covariance function. If the mean function for a real process $f(\boldsymbol{x})$ is defined as $m(\boldsymbol{x})$ and the covariance function evaluated at $\boldsymbol{x}$ and $\boldsymbol{x'}$ is defined as $k(\boldsymbol{x}, \boldsymbol{x'})$, then we have

$$m(\boldsymbol{x}) = \mathbf{E}[\mathbf{f}(\boldsymbol{x})] \tag{3}$$

and

$$k(\boldsymbol{x}, \boldsymbol{x'}) = \mathbf{E}[(f(\boldsymbol{x}) - m(\boldsymbol{x}))(f(\boldsymbol{x'}) - m(\boldsymbol{x'}))]. \tag{4}$$

Hence, the GP can be written as

$$f(\boldsymbol{x}) \sim \mathcal{N}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x'})) \tag{5}$$

where $\mathcal{N}(\cdot|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is a Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$.

Given the dataset $\mathcal{D} = \{(\boldsymbol{x(i)}, \boldsymbol{y(i)})\}|i = 1, ..., n\}$, with the D-dimensional variables $\boldsymbol{x(i)}$ and the targets $\boldsymbol{y(i)}$ defined in Section III, we wish to make predictions for new inputs $\boldsymbol{x(*)}$ that have not been observed in the training set. In particular, we wish to predict the corresponding outputs $\boldsymbol{y(*)}$ based on the information contained in the training set $\mathcal{D}$. This, however, corresponds to a multiclass classification problem with D classes. Specifically, each link $e_j \in E$ is represented by the class j, where $j = 1, ..., D$. GP classification can then be formulated as either a single GP multiclass classification problem, or as multiple GP binary classification problems; in the latter case, we train D models, where the jth model is trained to recognize the jth class versus all other classes.

In this work, we have formulated the problem according to the GP binary classification in order to avoid the scalability issues that may arise as the network grows. Thus, the dataset $\mathcal{D}$ is decomposed into D datasets such that $\mathcal{D} = \{\mathcal{D}_j|j = 1, ..., D\}$, where $\mathcal{D}_j = \{(\boldsymbol{x(i)}, y_j(i)\}|i = 1, ..., n\}$. Then, each dataset $\mathcal{D}_j$ is utilized by a separate binary GP classifier, that produces one probabilistic model for each link in the network. Therefore, in the discussion that follows we focus on the GP-based binary classification problem. Note that for brevity, notation j is omitted in the remainder of this section.

1) Model Formulation and Training: In binary GP classification the postulated model attempts to predict the probability of the dependent variable being "on," i.e., $y = 1$. Specifically, binary GP classification postulates $\pi(\boldsymbol{x}) \triangleq p(y = 1|\boldsymbol{x}) = \sigma(f(\boldsymbol{x}))$, where $\sigma(.)$ is the logistic function, and $f(\boldsymbol{x})$ is a real-valued latent function that constitutes the main modeling mechanism of the GP classifier. Binary GP classification, being a Bayesian inference approach, proceeds by imposing a Gaussian prior over the inferred latent function $f(\boldsymbol{x})$, jointly over all the training instances $\boldsymbol{X} = \{\boldsymbol{x(i)}\}_{i=1}^{n}$ and the test instance $x(*)$, yielding

$$\begin{bmatrix} \boldsymbol{f(X)} \\ \boldsymbol{f(x(*))} \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} \boldsymbol{K(X,X)} & \boldsymbol{k(x(*))} \\ \boldsymbol{k(x(*))} & k(\boldsymbol{x(*)}, \boldsymbol{x(*)}) \end{bmatrix}\right) \tag{6}$$

where

$$\boldsymbol{k(x(*))} \triangleq [k(\boldsymbol{x(1)}, \boldsymbol{x(*)}), ..., k(\boldsymbol{x(n)}, \boldsymbol{x(*)})]^T \tag{7}$$

and $\boldsymbol{K}$ is the matrix of the covariances between the n training data points usually referred to as the *gram* matrix (shown in Eq. 8 below).

$$\boldsymbol{K}(\boldsymbol{X},\boldsymbol{X}) \triangleq \begin{bmatrix} k(\boldsymbol{x(1)},\boldsymbol{x(1)}) & k(\boldsymbol{x(1)},\boldsymbol{x(2)}) & ... & k(\boldsymbol{x(1)},\boldsymbol{x(n)}) \\ k(\boldsymbol{x(2)},\boldsymbol{x(1)}) & k(\boldsymbol{x(2)},\boldsymbol{x(2)}) & ... & k(\boldsymbol{x(2)},\boldsymbol{x(n)}) \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ k(\boldsymbol{x(n)},\boldsymbol{x(1)}) & k(\boldsymbol{x(n)},\boldsymbol{x(2)}) & ... & k(\boldsymbol{x(n)},\boldsymbol{x(n)}) \end{bmatrix} \quad (8)$$

Note that, in the above equations, $k(\boldsymbol{x(z)},\boldsymbol{x(m)})$ is the *kernel* function of the postulated GP model, which expresses the similarity between two data points $\boldsymbol{x}(k)$ and $\boldsymbol{x}(l)$ and takes nonnegative values in $\mathbb{R}^+$. In our application, we employ the automatic relevance determination (ARD) kernel [17]. Our selection is due to the capability of the ARD kernel to determine how relevant each input component is, thereby omitting input components that are deemed irrelevant [16]. The ARD kernel reads

$$k(\boldsymbol{x(z)},\boldsymbol{x(m)}) = \theta_0 \exp\{-\frac{1}{2}\sum_{j=1}^{n} \eta_j (x_j(z) - x_j(m))^2\} \quad (9)$$

Here, θ_0 and $\{\eta_j\}_{j=1}^{D}$ are hyperparameters of the kernel function, that are optimized as part of the training procedure of the GP classifier. For this purpose, we resort to maximization of the marginal log-likelihood of the model w.r.t. θ_0 and $\{\eta_j\}_{j=1}^{D}$, as discussed in [17].

2) Prediction Generation: Inference is divided into two steps. First, we use Eq. 6 to derive the posterior distribution of the latent function value corresponding to the given test case

$$p(f(*)|\boldsymbol{X},\boldsymbol{y},\boldsymbol{x(*)}) = \int p(f(*)|\boldsymbol{X},\boldsymbol{x(*)},\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{X},\boldsymbol{y})d\boldsymbol{f} \quad (10)$$

where we denote $\boldsymbol{y} = \{y(i)\}_{i=1}^{n}$, $\boldsymbol{f} = \boldsymbol{f(X)}$, and $f(*) = f(x(*))$, while

$$p(\boldsymbol{f}|\boldsymbol{X},\boldsymbol{y}) = p(\boldsymbol{y}|\boldsymbol{f})p(\boldsymbol{f}|\boldsymbol{X})/p(\boldsymbol{y}|\boldsymbol{X}) \quad (11)$$

is the posterior over the latent function values on the training data points. Subsequently, we can use this posterior over the latent $\boldsymbol{f(*)}$ to produce a probabilistic prediction

$$\begin{aligned}\pi &\triangleq p(y(*) = +1|\boldsymbol{X},\boldsymbol{y},\boldsymbol{x(*)}) \\ &= \int \sigma(f(*))p(f(*)|\boldsymbol{X},\boldsymbol{y},\boldsymbol{x(*)})df(*) \end{aligned} \quad (12)$$

Note that, for brevity, we denote $\boldsymbol{y} = \{y(i)\}_{i=1}^{n}$, $\boldsymbol{f} = \boldsymbol{f(X)}$, and $f(*) = f(x(*))$. As the integral in Eq. 10 is analytically intractable, we here employ a first-order Taylor expansion of the integral around its mean. This approach, commonly referred to as the Laplace approximation, yields the predictive distribution expression [16]

$$\bar{\pi} \simeq \boldsymbol{E_q}[\pi(*)|\boldsymbol{X},\boldsymbol{y},\boldsymbol{x(*)})] = \int \sigma(f(*))q(f(*)|\boldsymbol{X},\boldsymbol{y},\boldsymbol{x(*)})df(*) \quad (13)$$

where $q(f(*)|\boldsymbol{X},\boldsymbol{y},\boldsymbol{x(*)})$ is the *Gaussian* approximation of $p(f(*)|\boldsymbol{X},\boldsymbol{y},\boldsymbol{x(*)})$ obtained by the Laplace technique; its corresponding mean and variance expressions, $\boldsymbol{E_q}[f(*)|\boldsymbol{X},\boldsymbol{y},\boldsymbol{x(*)})]$ and $\boldsymbol{V_q}[f(*)|\boldsymbol{X},\boldsymbol{y},\boldsymbol{x(*)})]$, respectively, are omitted from here due to space restrictions, but can be found in Eqs. (3.21) and (3.24) of [16].

V. Dataset Generation

In this work, the dataset $\mathcal{D}$ was generated for an OFDM-based elastic optical network. In general, the dataset generation process of our approach is divided into two phases. In the first phase, a list of sequential, in time, failures is created. Then, these failures are injected into a dynamic Routing and Spectrum Allocation (RSA) system for creating the network incidents from which the network features are extracted.

A. Link Failure Generation

For generating the specific points in time that each link has failed, we assumed that the times between the successive failures of each specific link e_j follow the Weibull distribution as explained in Section II. Each link e_j, is thus characterized by two parameters, the scale parameter λ_j indicating the statistical dispersion of the probability distribution, and the shape parameter β_j indicating how quickly the failure rate of the link increases with time. Then, a number of failure times for each link are drawn by $L_j \ \sim Wei(\lambda_j, \beta_j)$, where L_j is the list in which the failure times of link e_j are reported. Once the lists L_j are generated, the list L is created by sequentially adding into L the link with the minimum time of occurrence among all the times reported in lists L_j. Thus, a list L is created, in which links appear sequentially according to their times of occurrence. Specifically, $L = \{e_k(i)|i = 1,....,n\}$ and $e_k(i) \in E$ is the failed link upon incident i.

B. Feature Extraction

Connection requests arrive dynamically into the network according to a Poison process with exponentially distributed holding times. The source-destination pair and the spectrum demand for each connection request are randomly generated. Then, for each request, the RSA algorithm is used for finding a route and a free spectrum allocation for establishing the connection. For the routing sub-problem, Dijkstra's [18] algorithm is used while for the spectrum allocation procedure the first-fit algorithm is utilized. According to the first-fit algorithm, the connection is established on the first free spectrum slots that meet the spectrum continuity and sub-carrier consecutiveness constraints [1]. A connection request is accepted into the network if a route is found that meets the the spectrum continuity and sub-carrier consecutiveness constraints; otherwise it is blocked.

As the network evolves, link failures are injected sequentially in order to capture the network state upon their occurrence. In particular, with every arriving request, a single-link failure is injected, following the sequence of links in list L. Upon the injection of each failure i, the network state is inspected to identify the set of affected paths $P(i)$ and the set of unaffected paths $P'(i)$. A counter $c_j(i)$ is also kept for each link $e_j \in E$ and for each incident i, according to Eq. 14:

$$c_j(i) = \begin{cases} c_j(i-1), & \text{if } e_j \neq e_k(i-1) \in L \\ c_j(i-1)+1, & \text{if } e_j = e_k(i-1) \in L. \end{cases} \quad \forall e_j \in E \quad (14)$$

Finally, the counter $C(i)$ is computed for each incident i as $C(i) = C(i-1) + 1$. Then, according to the above

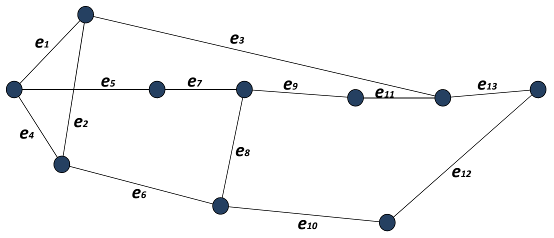

Fig. 1. Network topology with links e_j denoted.

procedure, if the number of total incidents is n, the sets $P = \{P(i)|i = 1, ...n\}$ and $P' = \{P'(i)|i = 1, ...n\}$ are generated as inputs to the GBC heuristic, as described in Section IV-A. In addition, the sets $C = \{C(i)|i = 1, ...n\}$ and $c_j = \{c_j(i)|i, j = 1, ...n\}$ are generated and used in conjunction with the GBP output set $S = \{S(i)|i = 1, ...n\}$ for the creation of the final dataset $\mathcal{D} = \{(\boldsymbol{x(i)}, \boldsymbol{y(i)})\}|i = 1, ..., n\}$, as described in Section III.

VI. Performance Evaluation

The network topology of Fig. 1 was utilized for evaluating the proposed fault localization scheme. The exact distances for each link are shown in Table I. For generating the list L, the λ_j and β_j parameters of the Weibull distribution were generated for each link. In particular, each λ_j parameter was generated by multiplying the inverse of the distance of each link with a random number generated by the uniform distribution. This was done in order to assign to each link a scale parameter relevant to its distance (i.e., longer links are expected to fail more frequently than shorter links due to the longer geographical distance that they are spanning and due to the larger number of components expected to be attached to that link). Each β_j parameter was generated by multiplying the distance of each e_j link to a random number generated by the uniform distribution. To avoid the generation of large shape parameters, that very quickly increase the failure rate, the β_j parameters were divided by a large constant number. The parameters used in the simulations are shown in Table I.

TABLE I
Network and Dataset Information

Link	Distance (km)	λ_j	β_j	Link	Distance (km)	λ_j	β_j
e_1	1100	471	2.77	e_8	800	229	1.77
e_2	1600	202	1.55	e_9	800	629	2.22
e_3	2800	131	1.45	e_{10}	1200	674	2.65
e_4	600	971	2.53	e_{11}	700	263	2.43
e_5	1100	321	1.62	e_{12}	900	636	2.5
e_6	2000	67	2.71	e_{13}	700	1094	2.1
e_7	600	1454	2.1	–	–	–	–

The dataset $\mathcal{D}$ was created by simulating dynamic point-to-point connection requests for three different network scenarios: For a network load of (a) 7, (b) 10, and (c) 20 Erlangs. For all three cases, each spectral slot in the network was set at 12.5 GHz, with each fiber link utilizing 400 slots. Fifteen-thousand $(15,000)$ connection requests were generated with the connection bandwidth of each request being randomly distributed in the set of bandwidth slots $B = \{2, 4, 7, 8, 13, 16, 32, 64, 80, 100\}$. Note that such lightly-loaded networks (7 to 20 Erlangs) were chosen in order to create a large number of network incidents i for which the GBC heuristic returns a set $S(i)$ with more than one suspect links in it (for heavier loads it is expected that the GBC heuristic will succeed to isolate the failed link). The network statistics related with the aforementioned simulation parameters are shown in Table II. In summary, Table II reports that for all network cases, the blocking probability is 0, and that as the network load increases more lightpaths are expected to be simultaneously established on the network upon a failure incidence; that is, as the network load increases, more information is passed to the GBC heuristic for isolating the failed link.

For the creation of the dataset $\mathcal{D}$, $10,000$ failures were injected into the network. Thus, a dataset $\mathcal{D}$ was created for $n = 10,000$ incidents. The dataset $\mathcal{D}$ was then divided into the training dataset $\mathcal{D}^{train}$ and into the test dataset $\mathcal{D}^{test}$ that is utilized for evaluating the efficiency of the proposed approach. Specifically, the first $7,000$ incidents were utilized for training the GP classifier while the last $3,000$ incidents were kept for evaluation purposes.

As described in Section IV-B, the binary GP classifier was applied to our data and thus the training was performed in a "one versus the others" fashion. Thus, a probabilistic model was computed for each link in the network. In our training example, the training set $\mathcal{D}^{train}$ was modified for each link $e_j \in E$, in such way that from each $\mathcal{D}_j^{train}$ the incidents for which only one link was reported as suspect and $x_j(.) = 0$ were removed. The final number of training incidents for each training dataset $\mathcal{D}_j^{train}$ and for each network scenario examined, are shown in Table III.

After the training procedure, the evaluation phase was performed on the $\mathcal{D}^{test}$. Amongst the $3,000$ incidents for which we want to identify the failed link, a number of incidents for which the GBC heuristic accurately identified the failed link, were removed. Information regarding the $\mathcal{D}_r^{test}$, which is the set created after the removal of the incidents identified by the GBC heuristic is shown in Table IV. Further, Table IV denotes the average, minimum and maximum number of suspect link observed among all incidents in the $\mathcal{D}_r^{test}$.

For the incidents in $\mathcal{D}_r^{test}$, a probabilistic value was gener-

TABLE II
Simulation Statistics

Traffic load (Erlangs)	7	10	20
# of Blocked Requests	0	0	0
Max. # of Established lightpaths	20	23	38
Av. # of Established lightpaths	4.2	5.41	10.58
Min. # of Established lightpaths	1	1	1

TABLE III
# OF INCIDENTS IN $\mathcal{D}_j^{train}$

Link	7 Erlangs	10 Erlangs	20 Erlangs
e_1	771	550	299
e_2	1937	1568	1388
e_3	1819	1697	1542
e_4	1102	762	424
e_5	743	570	410
e_6	3807	3685	3570
e_7	444	242	61
e_8	927	774	608
e_9	546	313	135
e_{10}	686	467	153
e_{11}	812	472	202
e_{12}	632	415	58
e_{13}	694	481	238

TABLE IV
INFORMATION FOR $\mathcal{D}_r^{test}$ PASSED TO THE GP CLASSIFIER

Traffic load (Erlangs)	7	10	20
# of Incidents	1541	1184	686
Min. # of Suspect Links	2	2	2
Max. # of Suspect Links	12	9	5
Av. # of Suspect Links	3.24	2.77	2.18

ated according to each trained link model and the link with the maximum value was chosen as the failed link. Table V summarizes the results. In particular, Table V denotes the GP accuracy for each network load and the total accuracy of the proposed approach; that is, the percentage of all the correctly classified incidents resulting from both the GP classifier and the GBC heuristic over the total number of incidents tested. The results clearly show that the approach achieves an overall high accuracy (93% to 99%), for networks that are lightly loaded (7 to 20 Erlangs). The results, also show that as the load increases the approach accuracy increases significantly. This is the case since the GBC heuristic performs better, as more paths are simultaneously established for correlation. Further, the GP classifier performs better as the average number of suspect links in $\mathcal{D}_r^{test}$ is reduced (i.e., the uncertainty is reduced). Note that the GP classifier required approximately 1 hour for the training procedure and approximately 1.3 sec to classify a single incident on our MATLAB machine with a CPU @2.60 GHz and 8 GB RAM.

VII. CONCLUSION

A fault localization scheme is proposed that can be used by service providers for reducing the MTTR and the human effort required for fault localization purposes. The proposed fault localization scheme consists of two phases. Once a failure is detected, the GBC heuristic is performed for limiting the number of suspect failed links. Then, if the GBC heuristic returns more than one suspect links, the failure probabilities for each suspect link are generated according to the probabilistic link models computed by a state-of-the-art GP classifier. The proposed scheme achieved a high overall accuracy (reaches 99% for a network load of 20 Erlangs) without utilizing any lightpath probing (extra monitoring equipment). Practical feasibility issues, related to the proposed approach (i.e., how much data is enough for training an accurate data-driven model, how much time it would take for collecting such number of data, scalability of the approach as the network grows), are planned for future work.

TABLE V
APPROACH ACCURACY VS TRAFFIC LOAD

Traffic load (Erlangs)	7	10	20
# Incidents in $\mathcal{D}^{test}$	3000	3000	3000
# Correctly Classified Incidents by GBC	1459	1816	2314
# Incidents in $\mathcal{D}_r^{test}$ (Passed to GP)	1541	1184	686
# Correctly Classified Incidents by GP	1327	1068	655
GP Accuracy	0.86	0.9	0.95
Total Accuracy (GBC and GP)	0.93	0.96	0.99

REFERENCES

[1] K. Christodoulopoulos, et. al., "Elastic Bandwidth Allocation in Flexible OFDM-Based Optical Networks", *IEEE/OSA J. Lightwave Technol.*, 29(9):1354–1366, 2011.

[2] E. Bouillet, et. al.. *Path Routing in Mesh Optical Networks*. Wiley-Interscience, 2007.

[3] G. Ellinas, et. al., "Practical Issues for the Implementation of Survivability and Recovery Techniques in Optical Networks", *Optical Switching and Net.*, 14(2):179–193, 2014.

[4] J. Rak. *Resilient Routing in Communication Networks*. Springer, Berlin, 2015.

[5] D. King and A. Farel, "A PCE-based Architecture for Application-based Network Operations," IETF RFC 7491, March 2015.

[6] K. Christodoulopoulos, et. al., "Exploiting Network Kriging for Fault Localization," *Proc. IEEE/OSA Optical Fiber Communications Conference (OFC)*, Anaheim, CA, March 2016.

[7] A. Markopoulou, et. al., "Characterization of Failures in an IP Backbone," *Proc. IEEE Infocom*, Hong Kong, March 2004.

[8] R.A. Davis, "Gaussian Process: Theory", *Wiley StatsRef: Statistics Reference Online*, 2014.

[9] V. N. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

[10] S.P. Chatzis and Y. Demiris, "Echo State Gaussian Process," *IEEE Trans. Neur. Netw.*, 22(9):1435–1445, Sept. 2011.

[11] C. Mas, et. al., "Failure Location Algorithm for Transparent Optical Networks", *IEEE J. on Selected Areas in Com.*, 23(8):1508–1519, 2005.

[12] S. S. Ahuja, et. al., "Single-link Failure Detection in All-optical Networks Using Monitoring Cycles and Paths", *IEEE/ACM Trans. Netw.*, 17(4):1080–1093, 2009.

[13] B. Wu, et. al., "A Novel Framework of Fast and Unambiguous Link Failure Localization via Monitoring Trails", *Proc. IEEE Infocom*, San Diego, CA, March 2010.

[14] M. L. Ali, et. al., "Multi-link Failure Localization via Monitoring Bursts", *IEEE/OSA J. Opt. Commun. Netw.*, 6(11):952–964, 2014.

[15] H. Herodotou, et. al., "Scalable Near Real-time Failure Localization of Data Center Networks", *Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, NY, Aug. 2014.

[16] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006.

[17] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag, 2006.

[18] E. W. Dijkstra, "A Note on Two Problems in Connexion with Graphs", *Numer. Math.*, 1(1):269–271, 1959.