A 3-layered self-reconfigurable generic model for self-diagnosis of telecommunication networks

Serge Romaric Tembo Mouafo, Jean-Luc Courant, Sandrine Vaton

SAI Intelligent Systems Conference 2015
November 10-11, 2015 | London, UK

A 3-Layered Self-Reconfigurable Generic Model For Self-Diagnosis of Telecommunication Networks

Serge Romaric Tembo
Orange Labs
2 Avenue Pierre Marzin
22300 Lannion, France
Email: serge.tembo@orange.com

Jean-Luc Courant
Orange Labs
2 Avenue Pierre Marzin
22300 Lannion, France
Email: jeanluc.courant@orange.com

Sandrine Vaton
Telecom Bretagne
655 Avenue du Technopole
29200 Brest, France
Email: sandrine.vaton@telecom-bretagne.eu

***Abstract*—The dynamic and distributed nature of telecommunication networks makes complex the design of model-based approaches for network fault diagnosis. Most model-based approaches assume the prior existence of the model which is reduced to a static image of the network. Such models become rapidly obsolete when the network changes. We propose in this paper a 3-layered self-reconfigurable generic model of fault diagnosis in telecommunication networks. The Layer 1 of the model is an undirected graph which models the network topology. Network behavior, also called fault propagation, is modeled in Layer 2 using a set of directed acyclic graphs interconnected via the Layer 1. We handle uncertainties of fault propagation by quantifying strengths of dependencies between Layer 2 nodes with conditional probability distributions estimated from network generated data. Layer 3 is the junction tree representation of the loopy obtained Layer 2 Bayesian networks. The junction tree is the diagnosis computational layer since exact inference algorithms fail on loopy bayesian networks. This generic model embeds intelligent self-reconfiguration capabilities in order to track some changes in network topology and network behavior. These self-reconfiguration capabilities are highlighted through some example scenarios that we describe. We apply this 3-layered generic model to carry out active self-diagnosis of the GPON-FTTH access network. We present and analyze some experimental diagnosis results carried out by running a Python implementation of the generic model.**

Keywords*—*Self-diagnosis; Self-reconfiguration; fault propagation; Bayesian network; Probabilistic Inference; GPON; FTTH.

I. Introduction

Reliability, robustness, availability, accessibility are the most important requirements that telecommunication networks should guarantee. As the design of network architectures, network management also became a central issue for telecommunication operators, which have triggered significant researches about autonomic networking. The main goal of autonomic networking is to automate as much as possible numerous tedious operations of network management like fault management.

Fault diagnosis is a central aspect of network fault management [27]. Traditionally, fault diagnosis has been performed manually by an expert or a group of experts experienced in managing communication networks [28]. However, the development of telecommunication networks has increased the size and complexity of their architectures. Fault diagnosis has become too complex for humans, who can keep track of only few hypotheses in their reasonings. Humans need a long training to fully master their network segment [6]. Automating fault diagnosis is critical in large scale telecommunication networks. A network fault or failure is a root cause of one or multiple network anomalies observed in the form of alarms or parameters outside of the standard. Alarms or symptoms are external manifestations of failures [8]. A telecommunication network is naturally a distributed system, a fault occurred spreads, triggering other faults and alarms which in turn trigger further faults and alarms. The consequence of both fault and alarm propagation is that a single root cause may result in a complex and distributed pattern of subsequent failures and their corresponding alarms [6]. This is especially true when multiple faults propagate simultaneously.

Fault diagnosis also called alarm correlation or root cause diagnosis [8], [13], [29], isolates the most probable set of faults based on their external manifestations. It is a process of searching intelligible explanations to observed alarms. Fault diagnosis is a complex process due to the non-deterministic nature of fault propagation phenomenon. A single fault may generate multiples alarms and a single alarm may be triggered by several faults.

The fault diagnosis problem was addressed in the past two decades and numerous techniques have been proposed. Expert systems [20], [25] encode specialized reasonings on specific diagnosis tasks in computer applications. Model-based approaches [4] develop reasonings based on an explicit representation of the network. The techniques based on machine learning algorithms like artificial neural networks [22], [1], probabilistic networks [28], [21] and case-based reasoning [14] infer diagnosis based on past experiences.

A brief description and discussion of related work on fault diagnosis as well as our objectives is presented in section 2. We describe and formalize in section 3, our contribution: a generic three layered self-reconfigurable probabilistic model for telecommunication networks fault diagnosis. In section 4, we focus on self-reconfiguration capabilities of the generic model by showing how this model can be turn into an intelligent autonomous system for self-diagnosis purposes. In section 5, we present an application of the generic model to fault diagnosis of the FTTH (Fiber To The Home) access networks based on GPON (Gigabit capable Passive Optical Network). In section 6, some experimental diagnosis results are presented and analysed with respect to the GPON-FTTH network. We conclude and present future works in section 7.

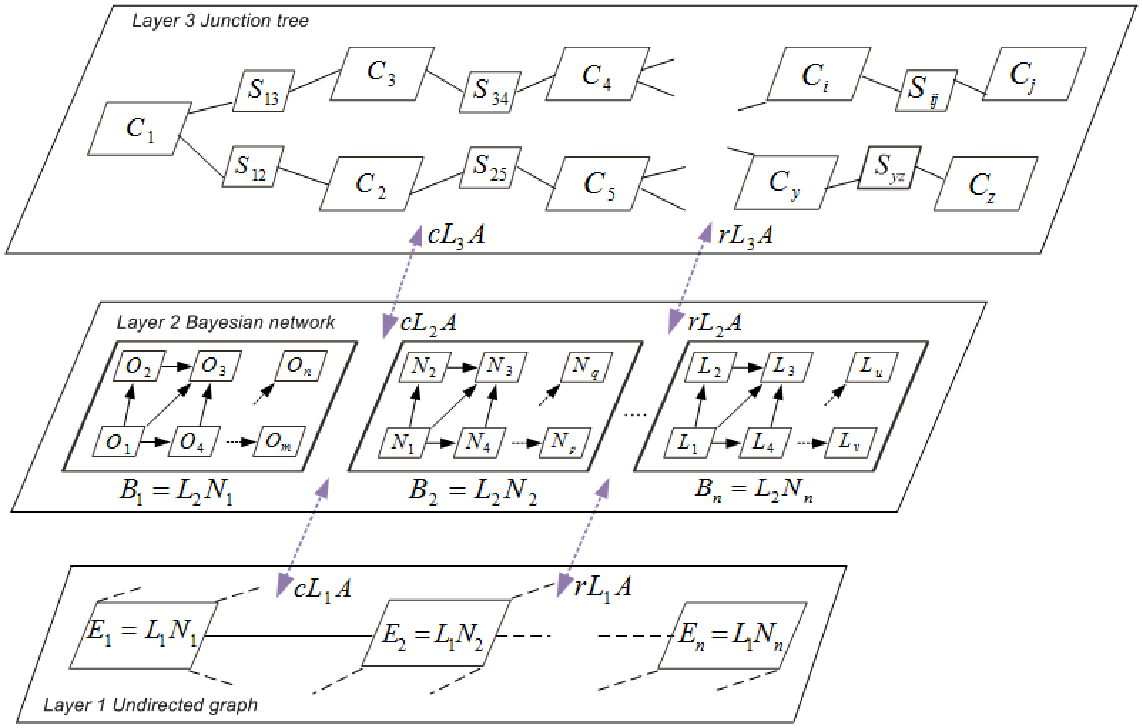

Fig. 1: The 3-Layered Generic Model.

II. Related works

Early self-diagnosis approaches were called expert systems [20], [25]. An expert diagnosis system attempts to infer the cause of a problem from symptoms recognized in sensor data [6]. It is a problem-solving software that embodies specialized reasonings on narrow diagnosis tasks usually performed by a trained skilled human called expert. The specialized reasonings can be formalized with rules, list of facts, logic predicates, etc. The inference engines are commonly based on forward-backward [15] chaining algorithms. However, expert systems mainly suffer of brittleness. The system fails when it faces a novel problem that comes out of its expertise.

Model-based diagnosis approaches [5], [10], [9], [11], [12], [23] develop reasonings based on formal and explicit representation of network structure and network behavior. Network structure describes the network architecture. Network behavior describes the process of alarm propagation and alarm correlation [7]. Network structure and network behavior are then modeled [4]. The obtained model is the support of reasoning algorithms which must be designed. The model-based approach is easy to deploy and is appropriate for a large-scale network if information on network resources is available [16]. It has the ability to deal with novel problems although its performances degrade in this case.

Model-based approach seems natural when relationships between objects are graph-like and easy to obtain [27]. The model can be designed in a modular or incremental fashion facilitating updates as new knowledge about the network is acquired. However, it is quite difficult to build a model close enough to the structural and functional reality of the network while maintaining a high level of abstraction to make the model independent of the various engineering techniques implemented in telecommunication networks. In addition, the model built is reduced to a static image of the network and becomes rapidly obsolete when the network changes.

In order to deal with the difficulty to obtain and self-maintain an accurate model of fault propagation in large scale telecommunication networks, we propose in this paper a generic 3-layered self-reconfigurable probabilistic model for self-diagnosis of telecommunication networks. The model integrates two fields: a decision field and an artificial learning field. The decision field is based on Bayesian probabilistic reasoning [21] in order to deal with uncertainties of fault propagation process. The artificial learning [3] field brings self-reconfiguration capabilities to the generic model in order to deal with the dynamic nature of telecommunication networks. The generic model has capabilities to compute diagnosis decisions and to automatically learn changes in network topology and network behaviour.

III. A Layered Self-Reconfigurable Probabilistic Model

We focus in this section on the description and the formalism of the 3-layered generic model which can be applied to any distributed system for self-diagnosis purposes.

A. Description of the generic model

As in any distributed system, in telecommunication networks, faults typically propagate between related system components. There is also some cases where a fault can propagate only inside a system component without affecting the state of neighbor system components. We call this, local fault

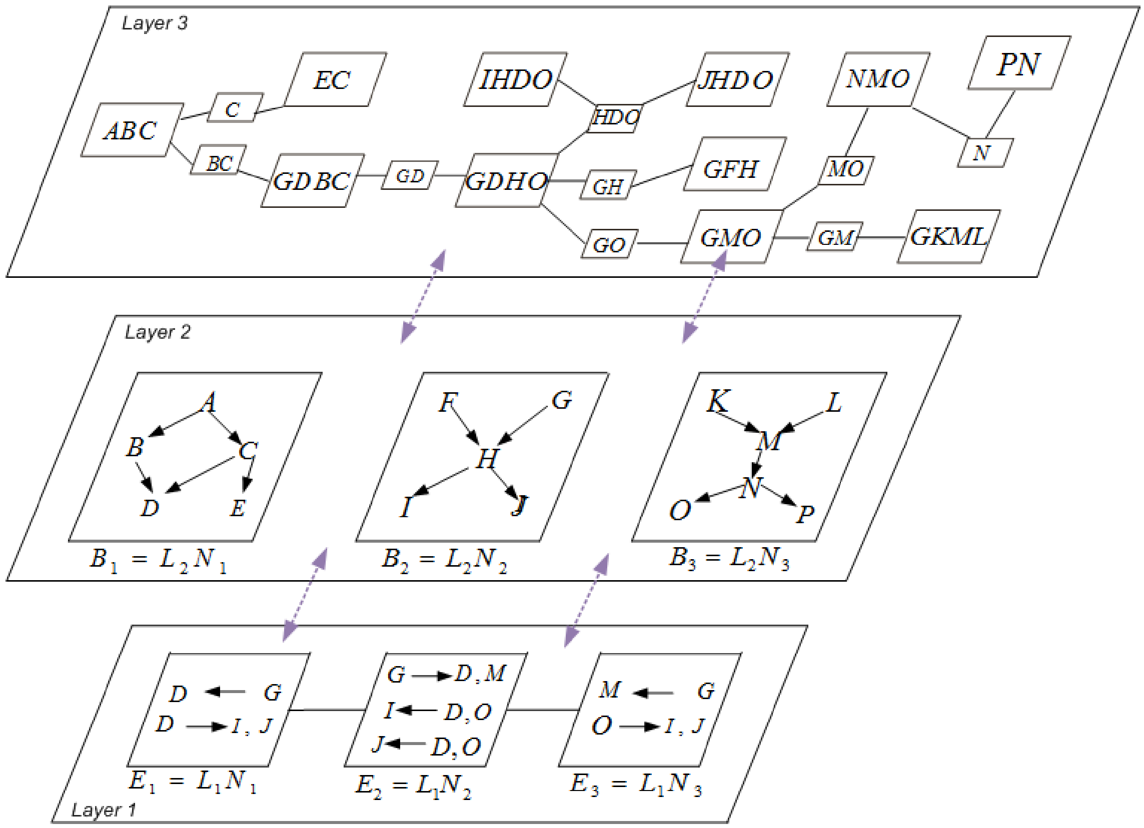

Fig. 2: A simple example of the 3-Layered Generic Model.

propagation. Obviously, distributed fault propagation is more recurrent in communication systems than their local counterpart. But keep in mind that a fault can also spread locally to a network component. A local fault propagation model is made of local dependencies which are relationships between faults (root causes), intermediate faults, counters, scalar parameters and alarms which can be observed on a network component.

The generic model that we propose clearly considers local and distributed fault propagation by separating network topology modeling and fault propagation modeling. Note that this separation brings important properties to the generic model: modularity and extensibility which facilitates self-reconfiguration. The generic model has three layers. Layer 1 models the network topology. Layer 2 models fault propagation. Layer 3 is a simplification of layer 1 and layer 2.

The layer 1 models the network topology and distributed fault propagation between related network components. Layer 1 is an undirected graph whose nodes represent network components and edges represent bidirectional links between them (see Figure 1). Any node of this layer is a Directed Acyclic Graph (DAG) which embeds distributed dependencies between a network component and its neighbors. These distributed dependencies model fault propagation between the related network components. The layer 1 of the generic model is an undirected graph of DAGs, i.e, a layer 1 node is seen as a subset of layer 2 nodes linked via distributed dependencies. In Figure 1, layer 1 represents the topology of a network of n components $E_1, E_2, ..., E_n$. The network component E_1 is linked to one neighbor E_2. The network components E_1 and E_2 are each DAGs containing distributed dependencies that carry fault propagation between them. For example, in Figure 2, each of the components E_1 and E_3 embeds distributed dependencies with its neighboor E_2. The component E_2 embeds distributed dependencies with its neighboors E_1 and E_3. In Figure 1, the roles of layer 1 agents cL_1A, rL_1A, layer 2 agents cL_2A, rL_2A and layer 3 agents cL_3A, rL_3A will be discussed later in the paper.

The layer 2 is a set of DAGs. Each DAG models local fault propagation inside a network component, i.e. a layer 1 node. In Figure 1, a layer 2 DAG B_i models local fault propagation on network component E_i. The DAGs of layer 2 are interconnected via layer 1 nodes. In order to deal with uncertainties of fault propagation phenomenon, each DAG can be transformed into a Bayesian network by quantifying strength of dependencies with Conditional Probability Distributions (CPD). A CPD quantifies the degree of influence that a subset of nodes of a DAG has on their common successor node. In Figure 1, Layer 2 contains n belief networks interconnected via Layer 1. The belief network B_i models local fault propagation inside the network component E_i. We call Layer 2 node a variable of any belief network.

Note that the union of DAGs of layer 2 and DAG of each layer 1 node gives one large DAG which models fault propagation in the entire network. But the separation between network topology modeling and network behavior (fault propagation) modeling breaks down this large DAG in many interconnected parts. So the design of layer 1 and layer 2 implements this separation and brings us the modularity and extensibility properties necessary to provide easy self-reconfiguration capabilities to the generic model.

The distributed nature of a telecommunication network introduces mutual dependencies between related network components. This means that distributed undirected loops are unavoidable between some layer 2 nodes. In addition, undirected loops may also appear in local fault propagation inside a network component. Note that, carrying out a message passing inference algorithm like the sum-product algorithm [21] at layer 2 will fail due to local and distributed undirected loops. In order to deal with local and distributed undirected loops between layer 2 nodes, we have built a third layer upon layer 2. The layer 3 is the junction tree representation [15], [16], [26] of layer 1 and layer 2.

Note that layer 1 and layer 2 are useful for learning field of the generic model since they provide modularity and extensibility properties useful for easy self-reconfiguration. Layer 3 is useful for decision field of the generic model since inference can be easily done in this layer regardless of network topology complexity at layer 1 and network behavior complexity at layer 2.

B. Formalism of the generic model

We formalize in this subsection the three layers of the generic model. We first define a local dependency and a distributed dependency. Let B_i and $B_{j \neq i}$ be two distinct layer 2 Directed Acyclic Graphs (DAGs). A local dependency on B_i is a tuple (u, v) such that u and v belong to the set of nodes of B_i. A distributed dependency between B_i and $B_{j \neq i}$ is a tuple (u, v) such that u belongs to the set of nodes of B_i and v belongs to the set of nodes of B_j. For example, in Figure 2, (B, D) is a local dependency which is part of the local fault propagation model on network component E_1. (G, D) is a distributed dependency between the related network components E_1 and E_2.

Layer 1 denoted by L_1Graph is an undirected graph of DAGs, $L_1Graph = (L_1Nodes, L_1Edges)$. L_1Nodes is the set of n layer 1 nodes or network components considered, $L_1Nodes = \{L_1N_1, L_1N_2, ..., L_1N_n\}$. L_1Edges is the set of bidirectional links between layer 1 nodes.

Each layer 1 node L_1N_i is a DAG defined as follows: $L_1N_i = (L_1V_i, L_1E_i)$. L_1V_i is the subset of layer 2 nodes embedded in layer 1 node L_1N_i. Each layer 2 node belonging to L_1V_i is part to at least one distributed dependency. L_1E_i is the set of distributed dependencies between L_1N_i and its $k_i \geq 1$ layer 1 neigboor nodes. L_1V_i is defined as follows:

$$L_1V_i = N_i \bigcup \left(\cup_{j=1}^{k_i} N_j^i \right) \qquad (1)$$

where, $N_i \subset L_2V_i, N_j^i \subset L_2V_j, j \in \{1, ..., n\} \setminus \{i\}$ such that $\forall u \in N_i, \exists v \in N_j^i | u \in pa(v) \vee v \in pa(u)$. We suppose that layer 1 node L_1N_i has k_i neighboors. L_2V_i is the subset of layer 2 nodes belonging to layer 2 DAG L_2N_i above layer 1 node L_1N_i (see Figure 1). The layer 2 DAG L_2N_i models local fault propagation on layer 1 node L_1N_i. For example, in Figure 2, $L_1N_2 = (L_1V_2, L_1E_2)$ such that $L_1V_2 = N_2 \bigcup N_1^2 \cup N_3^2$, where the subset $N_2 = \{G, I, J\}$, $N_1^2 = \{D\}$ and $N_3^2 = \{M, O\}$. L_1E_2 is the set of distributed dependencies between the component L_1N_2 and its neighboors L_1N_1 and L_1N_3. $L_1E_2 = \{(G, D), (G, M), (D, I), (O, I), (D, J), (O, J)\}$

Layer 2 is a set of n DAGs. Each DAG $L_2N_i = (L_2V_i, L_2E_i)$ models local fault propagation on network component L_1N_i. L_2E_i is the set of local dependencies between layer 2 nodes belonging to L_2V_i.

Layer 3 is the junction tree representation of the large bayesian network G obtained by combining layer 1 and layer 2 as follows: $G = \bigcup_{i=1}^{n} [L_2N_i \cup L_1N_i]$. Layer 3 is constructed in three steps. The moralization [17] of G and the triangulation [17] of the obtained moralized graph lead to a chordal graph or clique graph. The moralization consist to disorient edges of the graph G and adding a disoriented edge between each couple of parents of each node of G. The triangulation consist to add an edge to every cycle of the moralized graph whose length exceeds 3. The maximal weight spanning tree of the obtained clique graph is guaranteed to be a junction tree. The weight is the size of the intersection between adjacent cliques, i.e, the number of layer 2 nodes shared by adjacent cliques. We call layer 3 node, a clique C_i of the junction tree. As every junction tree, layer 3 satisfies the running intersection property which ensures that, the intersection $C_i \cap C_j$ is a subset of every clique and separator on the path between two cliques C_i and C_j. For example, in Figure 2, $G = C_{GDBC} \cap C_{GKML}$: G belongs to the clique C_{GMO} and the separators S_{GO} and S_{GM} which form the path between the cliques C_{GDBC} and C_{GKML}.

C. Diagnosis computations of the generic model

The layer 3 is an equivalent of layer 1 and layer 2. This means that layer 3 is sufficient to compute diagnosis decisions using for example the well known HUGIN exact inference algorithm [15] on a junction tree. In Figure 1, layer 3 nodes C_i, C_j, S_{ij} are respectively the cliques C_i and C_j of the junction tree and their common separator S_{ij}. A layer 3 node is a compound variable of some layer 2 nodes. For example, in Figure 2, layer 3 node ABC is a compound variable of Layer 2 nodes A, B, and C. Layer 3 is initialized by associating a potential to each layer 3 node. At initialization, a clique C_i has the potential ϕ_{C_i} and a separator S_{ij} has the potential $\phi_{S_{ij}}$ as follows:

$$\phi_{C_i} = \prod_{X \in Layer2, X \in C_i, pa(X) \subset C_i \vee pa(X) = \emptyset} P(X|pa(X)) \qquad (2)$$

And

$$\phi_{S_{ij}} = 1 \qquad (3)$$

We note $pa(X)$, the parent set of layer 2 node X. The potential ϕ_{C_i} of a layer 3 node C_i represents the joint conditional probability of layer 2 nodes that compose it (see equation 2). For example, in Figure 2: $\phi_{C_{ABC}} = P(B|A).P(C|A).P(A)$, $\phi_{C_{GDBC}} = P(D|B, C, G).P(G)$ and $\phi_{S_{BC}} = 1$.

The diagnosis decisions computed at layer 3 are based on evidences propagation on a junction tree. Assume C_i, C_j to be neighbooring layer 3 nodes with their common separator S_{ij} (see Figure 1). The potential ϕ_{C_i} is updated when some layer 2 nodes belonging to clique C_i are observed. This evidence propagates from clique C_i to clique C_j towards their separator S_{ij} as follows: updating the potential of the separator S_{ij} by the marginalization of equation 4, and updating the potential of the clique C_j by the product of equation 5. The notation $C_i \setminus S_{ij}$ reprensents the layer 2 nodes of the clique C_i which does not belong to the separator S_{ij}.

$$\phi^*_{S_{ij}} = \sum_{C_i \setminus S_{ij}} \phi^*_{C_i} \tag{4}$$

$$\phi^*_{C_j} = \phi_{C_j} \frac{\phi^*_{S_{ij}}}{\phi_{S_{ij}}} \tag{5}$$

We say that the clique C_j absorbs evidences from C_i [15] or that the clique C_i brings evidences to C_j. Note that, update operations of clique potentials and separator potentials are done in two recursive stages. The first stage called collect is initiated by collecting evidences from observed nodes to the root nodes. The second stage is initiated by distributing evidences from the root nodes to leave nodes. Collection of evidences to a clique C_i is done by collecting evidences to all the children of C_i followed by absorption of evidences from each child. Similarly, distribution of evidences from a clique amounts to bring evidences to each child followed by distribution of evidences from the child.

Note that, the update operations of potentials of layer 3 nodes are made by an agent called computing Layer 3 Agent cL_3A. The cL_3A updates the potential of each layer 3 node when it receives some evidences (observed layer 2 nodes) from its counterpart of layer 2 called cL_2A. When cL_2A receives updated layer 3 node potentials, it computes the marginals (beliefs) of layer 2 nodes. These two agents may communicate using a simple mechanism like shared memory through an interface between layer 2 and layer 3 that we call L2-L3 Interface (see Figure 3).

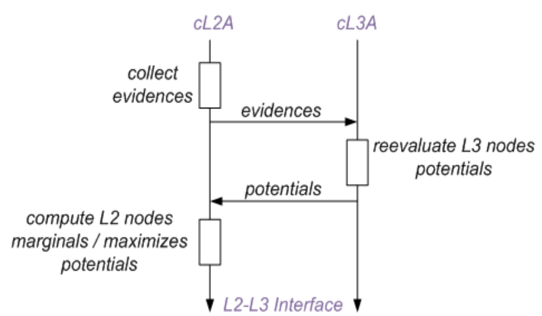

Fig. 3: Communication between layer 2 and layer 3 for beliefs updating of layer 2 nodes.

The marginal $P(X|e)$ of a layer 2 node X consistent with evidences e, is computed from the updated potential $\phi^*_{C_i}$ of any layer 3 node C_i such that $X \in C_i$: $P(X|e) = \sum_{C_i \setminus X} \phi^*_{C_i}$. Note that, after updating the potential of all layer 3 nodes, the intersection property ensures coherence between updated potentials, i.e, the marginal of a layer 2 node X is the same regardless of the clique C_i on which the marginalization operation is made. The most probable state of the layer 2 node X that is consistent with evidences, is one that has the highest probability.

Note that, there is an alternative approach to find the diagnosis without performing summation operations on updated potentials. With this approach, the diagnosis r^*, is computed from the most probable explanation w^*, of evidences as follows: $w^* = \bigcup_{C \in Layer3} w^*_C$ such that $\phi_C(w^*_C) = \max_{w_C} \phi_C(w_C)$, where w_C is a configuration of layer 2 nodes belonging to layer 3 node C and w^*_C maximizes the potential ϕ_C of C. The diagnosis r^* is the most probable configuration of layer 2 root nodes defined by $r^* = w^* \setminus i^*$, where i^* is the most probable configuration of non root layer 2 nodes consistent with evidences.

IV. Self-Reconfiguration capabilities of the generic model

In this subsection, we illustrate the potential of the generic model to self-reconfigure in order to track changes in network topology and network behavior (how faults propagate). Each of the three layers of the generic model has a reconfiguration agent which communicates by shared memory with its counterpart of the adjacent layers.

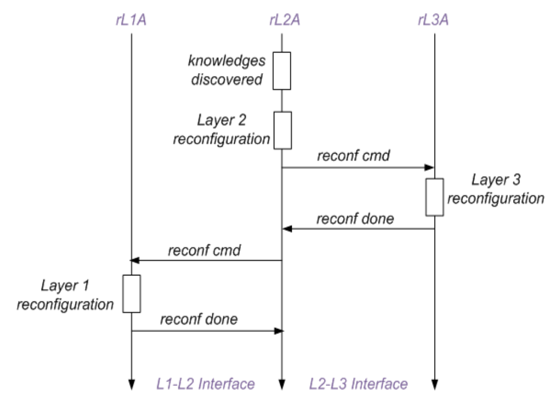

Fig. 4: Self-reconfiguration process initiated at layer 2.

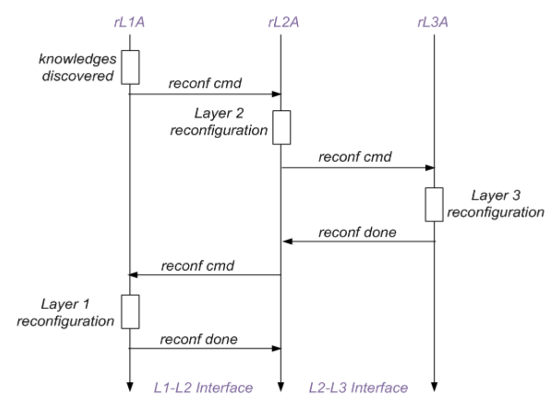

Fig. 5: Self-Reconfiguration process initiated at layer 1.

The reconfiguration Layer 2 Agent rL_2A is an artificial learning [3] system which implements knowledge discovery and data mining algorithms from a tremendous amount of network generated data. This agent must have the ability to discover new local and distributed dependencies, new states of layer 2 nodes to consider, new network statistical knowledges which require a new estimation of conditional probabilities of some layer 2 nodes. When the rL_2A discovers some new important knowledge about network behavior, it reconfigures the layer 2 of the model and sends a reconfiguration command (*reconf cmd*) to the reconfiguration Layer 3 Agent rL_3A and to the reconfiguration Layer 1 Agent rL_1A (see Figure 4). The message *reconf done* is an acknowledgement of the command *reconf cmd*.

Note that rL_2A will send a reconfiguration command to rL_1A only if the new knowledge learned concerns distributed dependencies. Indeed, local dependencies are unknown at layer 1. Note also that it is the type of the new knowledge learned at layer 2 that will permit rL_3A, after receiving a reconfiguration command from rL_2A, to know if it should reconfigure or rebuild the layer 3. For example, if the conditional probability distribution of a layer 2 node changes, the rL_3A needs only to raise an event notifying the computing Layer 3 Agent cL_3A, to recompute the initial potential of all layer 3 nodes which this layer 2 node belongs to. If new local and/or distributed dependencies are learned, the rL_3A must rebuild the layer 3.

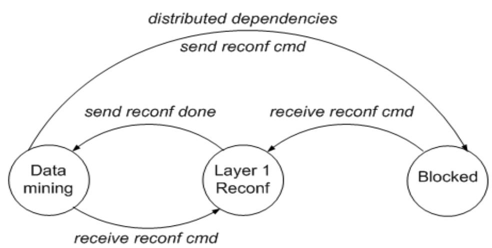

Fig. 6: State machine of the layer 1 reconfiguration agent.

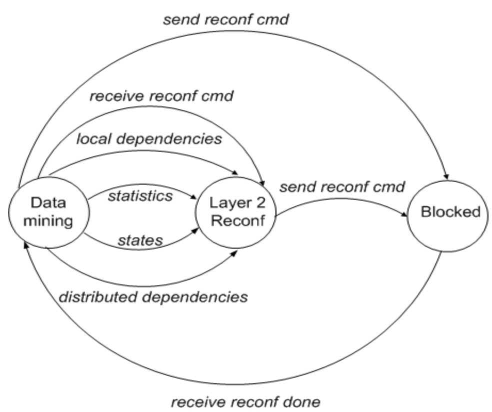

Fig. 7: State machine of the layer 2 reconfiguration agent.

The function of the reconfiguration Layer 1 Agent rL_1A is to track network topology changes like adding or removal of a network component. The rL_1A can be an automatic network topology discovery system [2], [24] or a system with an administration interface. This interface allows experts to remove a network component from the model or to specify and add a new network component to the model. The specification of a new network component includes: the name of this component, its neighbor components which already exist in the model, its local dependencies (local fault propagation model) and the distributed dependencies with each neighbor.

Note that any change in network topology requires the reconfiguration of the three layers of the model. The Figure 5 illustrates this reconfiguration process initiated by the layer 1 reconfiguration agent. The dynamic behaviour of the rL_1A is modelled by a 3-states machine (see Figure 6). Initially, the model is at equilibrium, i.e, there is no running reconfiguration process. So the rL_1A is in *Data mining* state, i.e, it waits for specification of a new network component by a network expert, it searchs for new knowledges about network topology from data collected by carried out active measures on network. When, the rL_1A discovers new knowledges about network topology, e.g, distributed dependencies, it does not reconfigure the layer 1 immediately. It sends a *reconf cmd* to the rL_2A and transits to the *Blocked* state. In *Blocked* state, the rL_1A waits for the completion of upper layers reconfiguration before entering in *Layer 1 Reconf* state. The rL_1A transits from *Blocked* state to *Layer 1 Reconf* state when it receives a *reconf cmd* from rL_2A. In the case where the rL_2A discovers new distributed dependencies, the rL_1A transits from *Data mining* to the *Layer 1 Reconf* state when it receives a *reconf cmd* from rL_2A, and returns to the *Data mining* state when it completes the layer 1 reconfiguration. The completion of the layer 1 reconfiguration yields a new equlibrium state of the model.

Layer 1 reconfiguration adds or removes new layer 1 node (network component) to/from the model. For adding a node for example, the new layer 1 node embeds distributed dependencies with its layer 1 neighboor nodes which already exist in the model. The rL_1A sends a *reconf cmd* to rL_2A (see Figure 5), waits for the rL_2A completes the layer 2 reconfiguration and adds the distributed dependencies to the neighboor nodes of the new layer 1 node. To reconfigure the layer 2, the rL_2A builds a layer 2 Bayesian network which models local fault propagation on the new network component. The rL_2A notifies the computing Layer 2 Agent cL_2A to recompute the conditional probability distribution of each layer 2 nodes having new distributed dependencies learned or specified at layer 1. Remark that only the reconfiguration agent of layer 2 takes a reconfiguration decision based on knowledges learned from network generated data. The other reconfiguration agents (rL_1A and rL_3A) wait for a *reconf cmd* from rL_2A before starting a reconfiguration process. Therefore, the rL_2A is the coordinator of our multi-agent based self-reconfigurable model.

The Figure 7 specifies the dynamic behaviour of the layer 2 reconfiguration agent using a 3-states machine. Initially, the rL_2A is in *Data mining* state, i.e, it searchs for new knowledges about network behaviour from tremendous amount of data generated by the network components. When the rL_2A learns new local dependencies, new distributed dependencies, new states of a layer 2 node to consider or new important statistical knowledges, it transits to the *Layer 2 Reconf* state in order to integrate the new knowledges learned to the layer 2 of the generic model. The rL_2A should trigger the reconfiguration of adjacents layer by sending a reconfiguration command to the reconfiguration agent of these layers. The *Blocked* state implements synchronisation between two successive reconfiguration processes, i.e, the rL_2A waits for the running reconfiguration process of the model completes before triggering another process. Doing so, the completeness and consistency of the model remain guaranteed after the integration of new learned knowledges to the generic model.

V. Application of the Generic Layered Model to GPON-FTTH self-diagnosis

We show in this section how the generic model previously proposed can be applied to perform self-diagnosis of FTTH (Fiber To The Home) access networks based on GPON (Gigabit capable Passive Optical Network) [19], [18]. For now, we used prior knowledges acquired from ITU-T standards [19], [18] of the GPON-FTTH network to build a 3-layered model of this network. We plan to transform this GPON-FTTH network model into an autonomous system by implementing self-reconfiguration capabilities of the generic model. The Figure 8 depicts the architecture of the GPON-FTTH network.

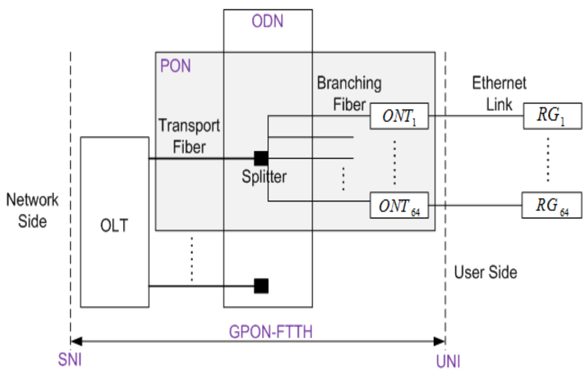

Fig. 8: GPON-FTTH network architecture.

The GPON-FTTH access network has two main network components. The Optical Line Termination (*OLT*) is located on operator side. The Optical Network Termination (*ONT*) is located on customer side. The *OLT* and *ONT* are connected through an Optical Distribution Network (*ODN*). The *ODN* is the optical infrastructure made of fibers and passive components like splitters. A Passive Optical network (*PON*) is a point-to-multipoint link inside the *ODN*. A *PON* has a tree-like topology which connects an *OLT* with a maximum of 64 *ONTs* (see Figure 8). Each *ONT* is connected to a *RG* (Residential Gateway) via an Ethernet link.

Since there is no interaction between *PONs* in *ODN*, and all *PONs* have the same behavior, we modeled one single *PON*. This model can be replicated to any *PON* of a GPON-FTTH access network. All *ONTs* connected to the same *PON* temporally share the upstream optical channel of the *PON*. The downstream channel of the *PON* is a secured broadcasting channel. The composition of Figure 9 and Figure 10 forms the application of the generic model for modeling the topology and behavior of a *PON* of the GPON-FTTH access network. The obtained model has two layer 1 nodes (*OLT*, *ONT*) and some layer 2 nodes are vectors in order to consider the tree-like topology of a *PON*.

A. Upstream modeling

The Figure 9 depicts the layer 1 *OLT* node and its layer 2 local dependencies. Distributed dependencies embedded in layer 1 *OLT* node are also depicted. We have three types of layer 2 nodes in Figure 9: faults or root causes, intermediate faults and alarms. The root causes are highlighted in Figure 9.

The transport optical fiber of the *PON* denoted by $Fiber_T(OK, AT, BR)$ can take three states. The state *OK* means that there is no transmissions anomaly on this fiber. The states *AT* and *BR* mean respectively that fiber experiences high attenuation or that fiber is broken. The temperature of *OLT* denoted by T^c_{OLT}, is a continous variable that we discretize. The power supply of *OLT* denoted by Alt_{OLT} which can be faulty or not.

The node $FaultyONT$ denotes an *ONT* which transmits upstream signal outside of its granted time slot, which may conflict with data sent by other *ONTs* on the *PON* and cause data disruption for a random set of *ONTs*, making the *PON* unusable. A *FaultyONT* can cause a Drift of Windows DOW. *OLT* raises a $DOW[i]$ alarm when an $ONT[i]$ transmits signal beyond the slot time allocated to it. See ITU-T G984.3 [19], [18] for more details.

The Software Version $SWV[i]$ alarm means that there is an incompatibility between the Image Operating System (IOS) of $ONT[i]$ and those of the *OLT*. The node $ONTConfMis$ (*ONT* Configuration Mistake) denotes a configuration error during *ONT* provisioning.

The *OLT* transmitted power Tx_{OLT} is regulated by the bias current I_{OLT}. This leads to the local dependency $I_{OLT} \longrightarrow Tx_{OLT}$. The *OLT* received power $Rx_{OLT}[i]$ from an ONT_i depends of the *OLT* voltage V_{OLT} and the state of the transport fiber $Fiber_T$.

The *OLT* received power of ONT_i also depends of the state of the branching fiber denoted by $FiberDB[i]$ and the transmitted power of this ONT_i denoted by $Tx_{ONT}[i]$. Note two local dependencies ($Fiber_T \longrightarrow Rx_{OLT}$, $V_{OLT} \longrightarrow Rx_{OLT}$) and two distributed dependencies ($FiberDB \longrightarrow Rx_{OLT}$, $Tx_{ONT} \longrightarrow Rx_{OLT}$). Remark in Figure 9 that the distributed dependencies are part of edges of the layer 1 *OLT* node which is a DAG as designed in the generic model.

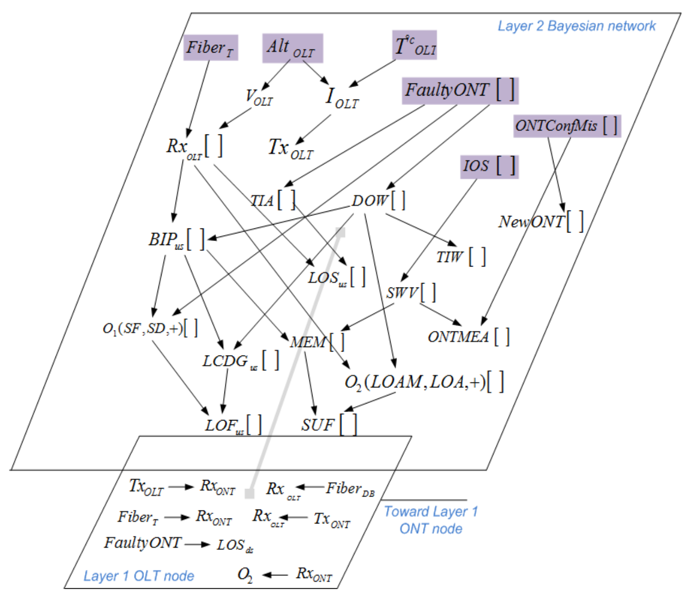

Fig. 9: Faults and alarms propagation raised by an *OLT*.

The Bit Interleaving Parity denoted by $BIPus[i]$ is computed from the Bit Error Rate *BER* of an upstream data transmission between an ONT_i and *OLT*. A poor upstream signal reception can cause bit errors leading to the local dependency $Rx_{OLT} \longrightarrow BIPus$. Upstream transmission bit errors impact the quality of signal received by *OLT* which may raised some alarms related to signal quality like SD (Signal Degraded), SF (Signal Fail), $LCGD$ (Loss of GEM Channel Delineation) and MEM (Message Error Message). See ITU-T G984.3 [19] recommendation for more details.

B. Downstream modeling

The Figure 10 depicts the layer 1 *ONT* node and its layer 2 local dependencies. The root causes $Alt_{ONT}[i]$, $T^c_{ONT}[i]$ and $FiberDB[i]$ respectively denote the power supply, the temperature and the branching fiber of $ONT[i]$.

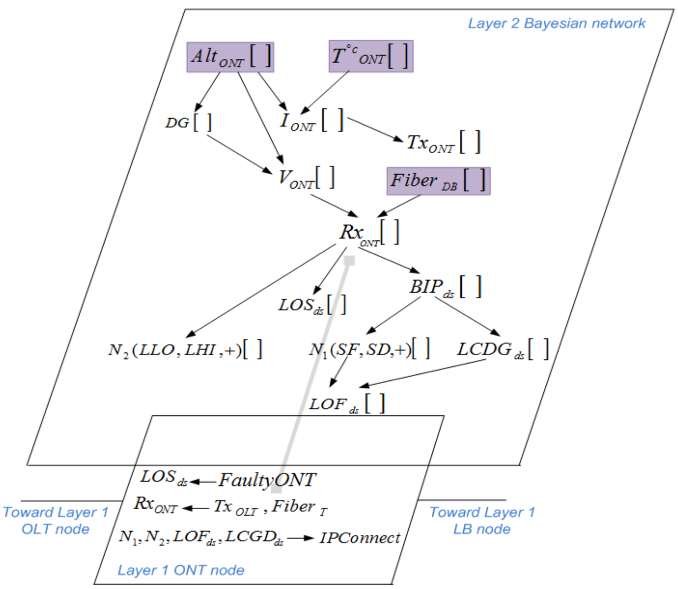

Fig. 10: Faults and alarms propagation raised by an *ONT*.

When the received power $Rx_{ONT}[i]$ of an ONT_i is less than a preconfigured minimum threshold, this *ONT* raises the (Level Low) $LLO[i]$ alarm. The (Level High) $LHI[i]$ alarm is raised by ONT_i when $Rx_{ONT}[i]$ is greater than a preconfigured maximum threshold. For simplicity and because LLO and LHI alarms can not be observed simultaneously, we have considered them as the states of the layer 2 node called $N_2(LLO, LHi, +)$. The state denoted by $+$ means that there is no LLO or LHI alarm observed. The received power $Rx_{ONT}[i]$ depends of the voltage $V_{ONT}[i]$, the state of the branching fiber $FiberDB[i]$, the state of the transport fiber $Fiber_T$ of the *PON* and the transmitted power Tx_{OLT} of *OLT*. Note the local dependencies $V_{ONT} \longrightarrow Rx_{ONT}$, $FiberDB \longrightarrow Rx_{ONT}$ and the distributed dependencies $Tx_{OLT} \longrightarrow Rx_{ONT}$, $Fiber_T \longrightarrow Rx_{ONT}$.

Now, suppose we need to extend this model by adding another layer 1 node: a Residential Gateway *RG* node for example (see Figure 8). A *RG* is a home network component that provides services to customers. It is not a GPON-FTTH network component, but adding it to the model will bring to the model the ability to correlate faults and alarms of the GPON-FTTH network with malfunctions of customer services. For example, in Figure 10, we can make such correlation with the distributed dependency $N_1, N_2, LOF_{ds}, LCGD_{ds} \longrightarrow IPConnect$, where $IPConnect$ denotes the Internet access service provided by a *RG*. To add the layer 1 *RG* node, we will need only to specify and quantify uncertainties of its local dependencies and distributed dependencies with the layer 1 *ONT* node, which already exists in the model.

VI. Experimental results of the GPON-FTTH network self-diagnosis

We present and analyse in this section the network fault diagnosis results carried out by our application and implementation of the generic model to perform active self-diagnosis of GPON-FTTH access network. The experiments are performed on a GPON-FTTH testbed on which we have considered a *PON* with two *ONTs* named ONT_1 and ONT_2. This experimental testbed allows us to emulate faults and to collect alarms raised by the GPON-FTTH network components. We also read, if availaible, the values of counters BIP_{us}, BIP_{ds} and scalar parameters $Rx_{OLT}, Rx_{ONT}, Tx_{OLT}, Tx_{ONT}$, voltages V_{OLT}, V_{ONT}, bias current I_{OLT}, I_{ONT}, temperatures T^c_{OLT}, T^c_{ONT} of *OLT* and of the two *ONTs* connected to the *PON* considered. Each table from Table I to Table VII shows the beliefs of Layer 2 root cause nodes computed by our Python implementation of the decision field of the generic model based on evidences observed and decribed in the title of each table.

We start the evaluation of the designed probabilistic fault propagation model for GPON-FTTH self-diagnosis by ckecking if inference on this model returns no fault when no alarm is observed on the *PON* and when counters and scalar parameters of *OLT* and of the two *ONTs* are nominal. The Table I presents the result of this test and shows that the positive state in bold, i.e, the good working state of each layer 2 root cause node is the most probable state.

TABLE I: No Fault: no observed alarm, counters and scalar parameters are known and are nominal for both ONT_1 and ONT_2.

Faults	States	Beliefs
$Fiber_T$	[**OK**, AT, BR]	[**0.99**, 6.e-03, 7.e-12]
$AltOLT$	[**OK**, DF]	[**0.99**, 4.e-07]
$FaultyONT_1$	[$+fto$, **-fto**]	[0.007, **0.993**]
$FaultyONT_2$	[$+fto$, **-fto**]	[0.014, **0.986**]
IOS_1	[**OK**, DF]	[**0.980**, 0.020]
IOS_2	[**OK**, DF]	[**0.980**, 0.020]
$ONTConfMis_1$	[$+ocm$, **-ocm**]	[0.002, **0.998**]
$ONTConfMis_2$	[$+ocm$, **-ocm**]	[0.002, **0.998**]
$FiberDB_1$	[**OK**, AT, BR]	[**0.91**, 8.e-02, 9.e-07]
$FiberDB_2$	[**OK**, AT, BR]	[**0.91**, 8.2-02, 9.e-07]
$AltONT_1$	[**OK**, DF]	[**0.99**, 1.e-07]
$AltONT_2$	[**OK**, DF]	[**0.99**, 1.e-07]

The Table II shows that the ONT_1 is diagnosed to be faulty when it loses upstream and downstream communication with *OLT* although the optical signal between them is not degraded (no alarm related to signal quality is observed). Note the missing values of counters and scalar parameters of ONT_1 since the GPON-FTTH network management system has no way to get these values from MIB (Management Information Base) of ONT_1. These missing values useful to compute the most probable diagnosis are inferred by the probabilistic model by using its conditional probability distributions. The loss of upstream and downstream communication between *OLT* and ONT_1 may also be due to the cut of the branching fiber of ONT_1. But this is a very rare event in reality since infrastructures of telecommunication operators always protect optical fibers from possible cuts. That is why the most probable explanation of a bidirectional loss of communication is a faulty *ONT*. However, other observations in addition to $LOSus_1$ and $LOSds_1$ alarms may change this explanation.

TABLE II: Loss of communication between *OLT* and ONT_1 with no alarm related to signal quality: alarms $LOSds_1, LOSus_1$ are observed, counters and scalars are unknown for ONT_1 but known and nominal for ONT_2.

Faults	States	Beliefs
$Fiber_T$	$[OK, AT, BR]$	[0.99, 6.e-03, 1.e-11]
$FaultyONT_1$	[**+fto**, $-fto$]	[**0.998**, 0.002]
$FaultyONT_2$	$[+fto, -fto]$	[0.014, 0.986]
$FiberDB_1$	$[OK, AT, BR]$	[0.91, 8.e-02, 1.e-06]
$FiberDB_2$	$[OK, AT, BR]$	[0.91, 8.e-02, 9.e-07]

TABLE III: Loss of communication between *OLT* and ONT_1 with alarms related to signal quality: alarms $LOSds_1, LOSus_1$, $LCGDus_1, LCGDds_1$, $SDus_1, SDds_1$ are observed, counters and scalars are unknown for ONT_1 but known and nominal for ONT_2.

Faults	States	Beliefs
$Fiber_T$	$[OK, AT, BR]$	[0.94, 5.e-02, 1.e-7]
$FaultyONT_1$	$[+fto, -fto]$	[0.422, 0.578]
$FaultyONT_2$	$[+fto, -fto]$	[0.014, 0.986]
$FiberDB_1$	[OK, **AT**, BR]	[0.409, **0.577**, 0.014]
$FiberDB_2$	$[OK, AT, BR]$	[0.92, 7.e-02, 1.e-06]

TABLE IV: Attenuation of the branching fiber of ONT_1 with known counters and scalar parameters: alarms $SDds_1, SDus_1$ are observed, variation of counters $BIPus_1$ upstream and $BIPds_1$ downstream, low level of upstream received power $RxOLT_1$ and downstream received power $RxONT_1$. Counters and scalar parameters of ONT_2 are known and are nominal.

Faults	States	Beliefs
$Fiber_T$	$[OK, AT, BR]$	[0.91, 8.e-02, 1.e-11]
$FaultyONT_1$	$[+fto, -fto]$	[1.e-06, 0.999]
$FaultyONT_2$	$[+fto, -fto]$	[0.014, 0.986]
$FiberDB_1$	[OK, **AT**, BR]	[7.e-02, **0.92**, 2.e-06]
$FiberDB_2$	$[OK, AT, BR]$	[0.92, 7.e-02, 1.e-06]

TABLE V: Attenuation of the branching fiber of ONT_1 with unknown counters and scalar parameters of ONT_1: alarms $SDds_1, SDus_1$ are observed, counters and scalar parameters of ONT_2 are known and nominal.

Faults	States	Beliefs
$Fiber_T$	$[OK, AT, BR]$	[0.93, 6.e-02, 1.e-09]
$FaultyONT_1$	$[+fto, -fto]$	[0.082, 0.918]
$FaultyONT_2$	$[+fto, -fto]$	[0.014, 0.986]
$FiberDB_1$	[OK, **AT**, BR]	[2.e-01, **0.76**, 1.e-04]
$FiberDB_2$	$[OK, AT, BR]$	[0.92, 7.e-02, 1.e-06]

In Table III, the observation of alarms notifying optical signal degradation, brings additional information to the model and changes the decision of the inference algorithm. The ONT_1 is no longer diagnosed to be *Faulty* but its branching fiber experiences important attenuation leading to communication loss with *OLT*. Remark in Table III that, contrary to the Table II, the belief that ONT_1 is faulty decreases in favor of increasing the belief that the branching fiber of ONT_1 experiences important attenuation.

In Table IV, the most probable diagnosis is the attenuation of the branching fiber of ONT_1 when SD (Signal Degraded) alarms are raised for ONT_1 with its counters and scalar parameters known and non-nominal (upstream and downstream *BIP* variation, low level upstream and downstream receive power). If for some reasons, the management system of GPON-FTTH network fails to get the values of counters and scalars of ONT_1, the model is always able to take the best decision although in this case, the belief in the branching fiber attenuation of ONT_1 decreases slightly (from 0.92 to 0.76, see Table V).

TABLE VI: Attenuation of the branching fiber of ONT_1 and ONT_2 with known counters and scalars of ONT_1 and ONT_2: alarms $SDus_1, SDds_1, SDus_2, SDds_2$ are observed, variation of $BIPus, BIPds$ for both $ONTs$, low level of upstream and downstream received power for both $ONTs$.

Faults	States	Beliefs
$Fiber_T$	[OK, **AT**, BR]	[0.41, **0.58**, 1.e-09]
$FaultyONT_1$	$[+fto, -fto]$	[1.e-04, 0.999]
$FaultyONT_2$	$[+fto, -fto]$	[5.e-04, 0.999]
$FiberDB_1$	[OK, **AT**, BR]	[0.49, **0.50**, 1.e-05]
$FiberDB_2$	[OK, **AT**, BR]	[0.49, **0.50**, 1.e-05]

Now, if we observe signal degradation between OLT and ONT_1, and between OLT and ONT_2 (see Table VI), the probabilistic model more likely believes that it is the transport fiber of the *PON* which experiences attenuation rather than each of the two branching fibers. Remark in Table VI that the probability of the transport fiber attenuation is 0.58. The probability of attenuation of each branching fiber is 0.50. The difference between these two probabilities is not very significant. Therefore, we may tend to believe that the model has trouble to take the best decision between the transport fiber attenuation or branching fibers attenuation. But, if we compute the joint probability of the two branching fibers attenuation ($0.5 \times 0.5 = 0.25$), we remark that the difference between this joint probability and the probability of the transport fiber attenuation is now very important. This result enforces the belief that it is the transport fiber of the *PON* which experiences attenuation. Note that the joint probability of the two branching fibers is the product of the probabilities of the branching fibers since they are independent. In fact the Markov Blanket $MB(FiberDB_i)$ of the variable $FiberDB_i$ is defined by the union of the following sets: the parent set PA_i of $FiberDB_i$, the children set CH_i of $FiberDB_i$ and the set OP_i of other parents of children of $FiberDB_i$. $MB_i = PA_i \cup CH_i \cup OP_i$, where $PA_i = \emptyset$ (the empty set) since $FiberDB_i$ is a layer 2 root cause node. The set $CH_i = \{Rx_{OLT_i}, Rx_{ONT_i}\}$ and the set $OP_i = \{Tx_{ONT_i}, V_{ONT_i}, V_{OLT}, Tx_{OLT}, Fiber_T\}$. The branching fibers $FiberDB_1$ and $FiberDB_2$ are independent each other because: $FiberDB_1 \notin MB(FiberDB_2)$ and $FiberDB_2 \notin MB(FiberDB_1)$.

The Table VII illustrates the case when a downstream loss of communication between OLT and ONT_1 is observed, i.e, the alarm $LOSds_1$ is raised and the GPON-FTTH network

TABLE VII: Loss downstream communication between *OLT* and ONT_1: $LOSds_1$ alarm is observed, counters and scalar parameters of ONT_1 are unknown, but known and nominal for ONT_2.

Faults	States	Beliefs
$Fiber_T$	$[OK, AT, BR]$	[0.92, 7.e-02, 3.e-11]
$FaultyONT_1$	$[+fto, -fto]$	[0.094, 0.906]
$FaultyONT_2$	$[+fto, -fto]$	[0.014, 0.986]
$FiberDB_1$	$[OK,$ **AT**$, BR]$	[8.e-02, **0.91**, 4.e-06]
$FiberDB_2$	$[OK, AT, BR]$	[0.92, 7.e-02, 1.e-06]

management system does not read the values of counters and scalar parameters of ONT_1. Note that, this is a one way loss of communication. Contrary to a bidirectional loss of communication as illustrated in Table II, ONT_1 is not diagnosed to be faulty. The most probable diagnosis computed by the model is the branching fiber downstream attenuation of ONT_1, i.e, only the downstream optical channel between OLT and ONT_1 experiences attenuation.

VII. Conclusion

We have presented in this paper a generic 3-layered probabilistic model of fault diagnosis of telecommunication networks. The generic model embeds two fields: a decision field and an artificial learning field. The generic model has capabilities to compute diagnosis decisions and to automatically learn changes in network topology and network behaviour.

We have shown how this generic model can be applied to design a probabilistic model-based approach of fault diagnosis of GPON-FTTH access network. We have presented and analyzed some network fault diagnosis results carried out by running a Python implementation of the decision field of the generic model. The results obtained are very consistent with experiments carried out on a GPON-FTTH network testbed.

We plan to transform this GPON-FTTH network faults diagnosis model into an autonomous system by implementing self-reconfiguration capabilities of the generic model. Another perspective is to bring more intelligence to the reconfiguration agent of layer 3 by designing a novel paradigm allowing it to take into account new dependencies learned to the lower layers without rebuilding all the layer 3. Doing so, scalability and reconfiguration efficiency of the generic model will increase.

References

[1] J. R. A. Goel and P. Sadayappan. Towards a 'neural' architecture for abductive reasoning. *IEEE International Conference on Neural Networks*, pages 681–688, 1998.

[2] M. N. Beacon. A hierarchical network topology monitoring system based in IP multicast. *Services Management in Intelligent Networks, no. 1960 in Lecture Notes in Computer Science*, pages 169–180, 2000.

[3] A. Cornuéjols and L. Miclet. *Apprentissage Artificiel, Concepts et Algorithmes*. EYROLLES, 2013.

[4] R. Gardner and D. Harle. Alarm correlation and network fault resolution using the kohonen self-organising map. *Global Telecommunications Conference (GLOBECOM 1997)*, pages 1398–1402, 1997.

[5] B. Gruschke. Integrated event management: Event correlation using dependency graphs. *A.S. Sethi (Ed.), Ninth International. Workshop on Distributed Systems: Operations and Management, University of Delaware, Newark, DE*, 87:130–141, October 1998.

[6] C. Hounkonnou. *Active self-diagnosis in telecommunication networks*. PhD Thesis, European University of Brittany, University of Rennes 1, INRIA, ISTIC, French, 2013.

[7] G. Jakobson and M. Weissman. Real-time telecommunication network management: extending event correlation with temporal constraints. In *Proceedings of the fourth international symposium on Integrated network management IV*, pages 290–301, 1995.

[8] G. Jakobson and M. Weissmani. Alarm correlation. *IEEE Network*, 7(6):52–59, 1993.

[9] J. Jordaan and M. Paterokl. Event correlation in heterogeneous networks using the osi management framework. *H.G. Hegering, Y. Yemini (Eds.), Integrated Network Management III, North-Holland, Amsterdam*, 36:683–695, 1993.

[10] S. C. K. Houck and A. Finkel. Towards a practical alarm correlation system. *A.S. Sethi, F. Faure-Vincent, Y. Raynaud (Eds.), Integrated Network Management IV, Chapman and Hal, London*, 86:226–237, 1995.

[11] S. Kätker. A modeling framework for integrated distributed systems fault management. *C. Popien (Ed.), Proc. IFIP/IEEE Internat. Conference on Distributed Platforms, Dresden, Germany*, pages 187–198, 1995.

[12] S. Kätker and K. Geihs. A generic model for fault isolation in integrated management systems. *Journal of Network and Systems Management*, 5(2):109–130, 1997.

[13] I. Katzela and M. Schwartz. Schemes for fault identification in communication networks. *IEEE Transactions on Networking*, 3(6), 1995.

[14] L. Lewis. A case-based reasoning approach to the resolution of faults in communication networks. *Integrated Network Management*, pages 671–682, 1993.

[15] A. L. Madsen and F. V. Jensen. Lazy propagation: A junction tree inference algorithm based on lazy evaluation. *Artificial Intelligence*, 113:203–245, 1999.

[16] M. Miyazawa and K. Nishimura. Scalable root cause analysis assisted by classified alarm information model based algorithm. *7th International Conference on Network and Service Management*, pages 1–4, 2011.

[17] P. Naim, P.-H. W., P. Leray, O. Pourret, and A. Becker. *Réseaux Bayésiens*. EYROLLES, 2008.

[18] T. S. S. of ITU. *G.977.1 Recommendation*. ITU-T, 2003.

[19] T. S. S. of ITU. *G.984.3 Recommendation*. ITU-T, 2008.

[20] L. PAU. Survey of expert systems for fault detection, test generation and maintenance. *Expert Systems*, 3:100–110, April 1986.

[21] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, 1988.

[22] J. C. R.J. Patton and T. Siew. Fault diagnosis in nonlinear dynamic systems via neural networks. *International Conference on Control*, 2:1346–1351, 1994.

[23] M. P. S. Kätker. Fault isolation and event correlation for integrated fault management. *A. Lazar, R. Sarauo, R. Stadler (Eds.), Integrated Network Management V, Chapman and Hall, London*, 60:583–596, 1997.

[24] D. C. S. Ramanathan and S. Neal. Auto-discovery capabilities for service management: An isp case study. *Journal of Network and Systems Management*, 8(4):457–482, 2000.

[25] W. T. Scherer and C. White. A survey of expert systems for equipment maintenance and diagnostics. *Fault detection and reliability: knowledge based and other approaches. Pergamon Press*, pages 3–18, 1987.

[26] D. S. Stefen Laurizen. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society, Series B*, 50(2):157–224, 1988.

[27] M. Steinder and A. S. Sethi. A survey of fault localization techniques in computer networks. *Science of Computer Programming*, 53:165–194, January 2004.

[28] M. Steinder and A. S. Sethil. Multi-layer fault localization using probabilistic inference in bipartite dependency graph. *University of Delaware, Newark, DE, Technical Report*, 2001.

[29] S. A. Yemini, S. Kliger, E. Mozes, Y. Yemini, and D. Ohsie. High speed and robust event correlation. *IEEE Communications Magazine*, 34(5):82–90, April 1996.